THE DERIVE - NEWSLETTER #29

THE BULLETIN OF THE DERIVE USER GROUP + TI 92

Contents:

revised version 2012 **March 1998**

[1] **Realitätsnaher Mathematikunterricht mit *DERIVE***, H.-J. Henn, (Dümmlerbuch 4565), Dümmler Verlag Bonn, ISBN 3-427-45651-X.

Arbeitsblätter zur Mathematisierung realer Probleme: Vom Regenbogen übers Auto und Autobahntrassen zu wirtschaftlichen Problemen.

Working sheets for modelling real life problems: the rainbow, some problems in connection with cars, how to plot a road, economy and money problems.

[2] **Empfehlungen für den Mathematikunterricht an Gymnasien**, Niedersächsisches Kultusministerium, Referat Presse und Öffentlichkeitsarbeit, Schilfgraben 12, 30159 Hannover

Leitlinien zur Themenauswahl und Unterrichtsgestaltung, Zu Veränderungen des Unterrichts, 5 exemplarische Unterrichtsbeispiele (durchgängig für den Einsatz des *TI*-92).

***Keith Eames** produced a booklet on DERIVE for Windows on Algebra which he uses for his year 12 students. If you are interested then please contact me or him. You can find his email address in this DNL.*

***Wilhelm Weiskirch** and **Heiko Knechtel** are asking for partners in the frame work of an EU-Comenius project. They both are teachers at a Gymnasium in Niedersachsen, Germany, and they want to exchange experiences and to develop supporting materials for upper secondary level teaching (special emphasis will be given on the TI-92). If you are interested then please contact w.weiskirch@t-online.de or HKnechtel@aol.com.*

Interesting WEB sites:

All websites are valid in 2012!

http://www.maa.org/mathland/mathtrek_2_2_98.html (Ivar Peterson's MathTrek)

http://sprott.physics.wisc.edu/pickover/ Both web sites are recommended by Jerry Glynn

http://metric.ma.ic.ac.uk/came/ The home page of the IC-CAME (International Council for Computer Algebra in Mathematics Education)

http://sunsite.utk.edu/math_archives/.http/contests/ Mathematics Contests, Competitions, and Problem Sets

http://ticalc.org A very informative site for TI-92 users

http://www.t3ww.org/ The T^3 site (**T**eachers **T**eaching with **T**echnology)

Internationales Symposium zur Didaktik der Mathematik

Thema: ***Mathematische Bildung und neue Technologien***

Universität Klagenfurt, 28.Sept. - 2.Okt. 1998

Information, Application and Submission of papers:

http://www.uni-klu.ac.at/groups/math/didaktik/symp98/symp98.htm

or: Universität Klagenfurt,Inst.f.Mathematik, Abteilung f. Didaktik der Mathematik,

Universitätsstr. 65, A-9020 Klagenfurt, Tel.: ++43 (0)463 2700 429

Liebe DERIVE und TI-Freunde,

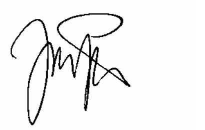
Wieder ist ein Newsletter fast fertig. Es fehlt nur noch der "Letter". Trotz der Arbeit, die es macht alles in eine gemeinsame Form zu gießen ist es immer wieder eine Freude, eine Ausgabe zusammenzustellen da Ihre vielen Beiträge das weite Spektrum der *DERIVE* und *TI*-92 Anwendungen zeigen. Es ist immer eine Herausforderung, die Beiträge für einen Newsletter zu sammeln, so dass alle in 46 Seiten Platz findet und auch ein wenig zusammenpaßt.

Wolfgang. Pröpper hat eine Kolumne für „Anfänger“ versprochen , muß sich aber für diesen DNL wegen allzuvieler Verpflichtungen (Vorträge, Kurse) entschuldigen. Ich glaube aber, dass Peter Mitics Überlegungen zum Programmieren mit DERIVE gerade auch für nicht so geübte DERIVE-Anwender von Nutzen sein können. Peter hat zwar im Titel "DERIVE 3.x", sein Aufsatz hat aber nach meiner Meinung auch in Zeiten von DfW 4.09 und DfD 4.x nichts von Aktualität eingebüßt.

Ich mache Sie gerne auch hier auf die Suche von Willi Weiskirch und Heiko Knechtel nach Partnern für ein EU-COMENIUS Projekt aufmerksam. Ich darf Ihnen aber auch gleichzeitig berichten, dass wir Ende Februar ein Projekt "Moderne Technologien im Mathematikunterricht" bei der EU eingereicht haben. Dem Pädagogischen Institut Niederösterreich als Koordinierende Einrichtung haben sich das ZKL Münster (Deutschland), das CTM an der Uni Plymouth (England), die Uni Göteborg (Schweden), das Freudenthal Institut (Niederlande) und die Katholische Uni Leuven (Belgien) als Partner angeschlossen. Ich werde Sie über die Entwicklung weiter informieren - wenn das Projekt bewilligt wird.

Abschließend möchte ich Sie auf die beigelegten Programme für die Gettysburg Conference und für die Münsteraner Pfingsttagung hinweisen.

Mit vielen Grüßen



A brief History of the DERIVE User Group

- From 1991 until December 1997

  1000 pages - without the front pages - containing:

  184 contributions from
  90 different contributors from
  21 countries

- We had

  302 questions, answers, messages, comments,... in the User Forum

- until DNL#28 we have put

  133 DERIVE- and TI-related books on our bookshelf.

Dear DERIVE and TI-friends

And so another Newsletter is nearly ready to go to print. Despite the work involved in finding a common shape for all the various contributions each time, I have again enjoyed compiling an issue because your many contributions are show such a wide field of DERIVE and TI-92 applications. It is always a challenge to gather articles in a way so that they will fill 46 pages and that they more or less fittogether.

Wolfgang Pröpper has promised to establish a "Beginner´s Column" but he has to apologize for this issue because of too many other engagements (lectures, courses). I think that Peter Mitic´s comments on "Program-ming with DERIVE" might be useful even for not so experienced DERIVE users. Although Peter´s article has "DERIVE 3.x" in its title in my opinion it has not lost anything of its topicality even in times of DfW 4.09 and DfD 4.x.

I´d like to point out to you W.Weiskirch´s und H.Knechtel´s request for partners for an EU-COMENIUS-project. I am pleased to report that we have applied for a project "Modern Technologies Teaching Mathematics". With the Pedagogical Institute of Lower Austria as the coordinating institution the following institutions have joined as project partners: ZKL Münster (Germany), CTM University Plymouth (England), University of Gothenburg (Sweden), Freudenthal Institute (Netherlands), and the Catholic University of Leuven (Belgium). I´ll keep you informed about the progress of the project
(- if we will get the approval.)

Finally I would like to call your attention to the enclosed programs and registration forms for the Gettysburg Conference and for the TI-92 meeting in Münster.

With my best regards

Falls Sie Ihre Mitgliedschaft noch nicht erneuert haben sollten, dann tun Sie das bitte so bald wie möglich. Herzlichen Dank.

If you have not renewed your membership for 1998 then please do that as soon as possible. Many thanks.

The *DERIVE-NEWSLETTER* is the Bulletin of the *DERIVE User Group*. It is published at least four times a year with a content of 40 pages minimum. The goals of the *DNL* are to enable the exchange of experiences made with *DERIVE* as well as to create a group to discuss the possibilities of new methodical and didactical manners in teaching mathematics.
We have established a section dealing with the use of the *TI*-92 and we try to combine these modern technologies.

Editor: Mag. Josef Böhm
A-3042 Würmla
D´Lust 1
Austria
Phone/FAX: 43-(0)660 31 36 365
email: 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 *D-N-L*. The more contributions you will send, the more lively and richer in contents the *DERIVE Newsletter* will be.

Next issue: June 1998
Deadline: 15 May 1998

Preview: Contributions for the next issues

Experimenting with GRAM & SCHMIDT, Schonefeld, USA
Extracting Logic Propositions from Numerical Data, Etchells, UK
3D-Geometry, Reichel, AUT
Linear Programming, Various Projections, Word Problems, Böhm, AUT
A Utility file for complex dynamic systems, Lechner, AUT
Examples for Statistics, Roeloffs, NL
Fractals and other Graphics, Koth, AUT
Implicit Multivalue Bivariate Function 3D Plots, Biryukov, RUS
Riemann, a package for the TI-92, Böhm & Pröpper, AUT/GER
Parallel Curves, Wunderling, GER
Quaternion Algebra, Sirota, RUS
Concentric Curve Shading, Speck, NZL
Drawing in the Plane with *DERIVE*, Mata & Torres, ESP
Information Technologies in Geometry, Rakov & Gorokh, UKR
Parametric 3D-Plots with DERIVE and 3DV, Welke, GER
Various Training Programs for Students on the TI-92, Böhm, AUT
A Critical Comment on the "Delayed Assignation" := =, Kümmel, GER

and
Setif, FRA; Vermeylen, BEL; Leinbach, USA; Weth, GER; Kirmse, GER
Aue, GER; Koller, AUT; Mitic, UK; Tortosa, ESP; Schorn, GER;
Santonja, ESP; Dorner, USA and and and ....

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

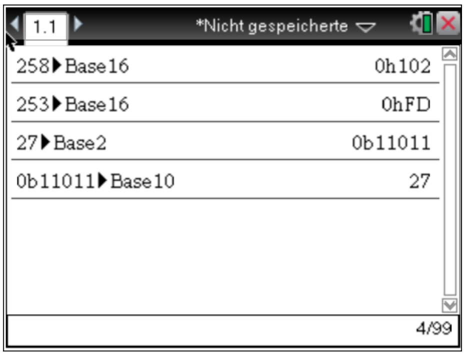
HeinzRainer Geyer, St.Katharinen, Germany http://members.aol.com/gymgutenbg
HeinzRainer.Geyer@t-online.de

I produced another small tool, because I wanted my class to perform conversions using various number bases. Unfortunately there are no functions HEX(a), OCT(a), DUAL(a) etc., which will convert a decimal number into another base. (I must admit, that I did not search very long). My results are no numbers, but lists of digits. Therefore it might be not very easy to continue with calculations. But it is just the beginning. Any suggestions are appreciated and welcome.

```
#1:   BASES.MTH

#2:   CaseMode := Sensitive

#3:   REM(number, base) := MOD(number, base)

      SELCT(r) := [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F]
#4:                                                       r + 1

#5:   SEQU(number, base) := ITERATES(FLOOR(x, base), x, number)

      AUX(number, base) := VECTOR(SELCT(REM((SEQU(number, base)) , base)), k, 1,
#6:                                                             k

        DIMENSION(SEQU(number, base)) - 2)

#7:   CONV(a, x_) := REVERSE_VECTOR(AUX(a, x_))

#8:   DUAL(a) := CONV(a, 2)

#9:   [DUAL(a) := CONV(a, 2), QUAD(a) := CONV(a, 4), PENT(a) := CONV(a, 5)]

#10:  [OCT(a) := CONV(a, 8), DEZ(a) := CONV(a, 10), HEX(a) := CONV(a, 16)]

#11:  ------------------ T E S T S ---------------------

#12:  AUX(258, 16) = [2, 0, 1]

#13:  HEX(258) = [1, 0, 2]

#14:  HEX(253) = [F, D]

#15:  OCT(27) = [3, 3]

#16:  OCT(512) = [1, 0, 0, 0]

#17:  DUAL(27) = [1, 1, 0, 1, 1]

#18:  CONV(1199, 12) = [8, 3, B]
```

As you can see there are some functions implemented in TI-Nspire. 0h and 0b are the initials for hexadecimal and dual numbers.

There is no function for converting into any other number system.



This is a DERIVE 6 program for converting decimal numbers into any other system. The results are strings, of course. It should be a nice problem for students to program the conversion back to decimal numbers, Josef

```
conv(x, b, r) :=
  Prog
    r := ""
    Loop
      If x < b
         RETURN APPEND(IF(MOD(x, b) < 10, STRING(MOD(x, b)),
                CODES_TO_NAME(55 + MOD(x, b))), r)
      r := APPEND(IF(MOD(x, b) < 10, STRING(MOD(x, b)),
                CODES_TO_NAME(55 + MOD(x, b))), r)
      x := FLOOR(x, b)
```

```
[dual(x) := conv(x, 2), oct(x) := conv(x, 8), hex(x) := conv(x, 16)]

hex(258) = 102

hex(253) = FD

oct(512) = 1000

dual(27) = 11011

conv(1199, 12) = 83B

conv(1199, 20) = 2JJ
```

Stefan Welke, Bonn, Germany Spwelke@aol.com

A remark on John Alexiou's suggestion in DNL #25 concerning substitutions with the ITERATE function. Alexiou mentions that his (very elegant) use of ITERATE improves the practice with the LIM-function insofar as it applies to any size and shape of vectors and matrices (especially) to number and symbolic calculations. I want to point out a special problem with symbolic substitutions which arises with simultaneous substitutions of several variables.

I start with the definition of two functions:

SUBST1.MTH

```
subst_alex(expr_, s_) := ITERATE(expr_, LHS(s_), RHS(s_), 1)

subst_lim(expr_, var_, s_) :=   lim    expr_
                              var_→s_
```

We now want to substitute in the expression $y - x^2$ the variable x → a and y → b!"

```
                  2                     2
subst_alex(y - x , [x = a, y = b]) = b - a

                 2                      2
subst_lim(y - x , [x, y], [a, b]) = b - a
```

We recognize that both functions are doing the same in this situation. The situation changes if the variable which should be substituted appears in the expression which substitutes the variable in question. Consider the following example:

$$\text{subst\_alex}(y - x^2, [x = x + y, y = x - y]) = - x^2 + x - y$$

$$\text{subst\_lim}(y - x^2, [x, y], [x + y, x - y]) = - 4\cdot x^2 + x\cdot(4\cdot y + 1) - y^2 - y$$

We do not have the same result. The correct result should be:

$$(x - y) - (x + y)^2 = - x^2 + x\cdot(1 - 2\cdot y) - y^2 - y$$

So neither SUBST_ALEX nor SUBST_LIM wins. Notice that both vesions work well if only one variable is substituted:

$$\text{subst\_alex}(y - x^2, x = x + y) = - x^2$$

$$\text{subst\_lim}(y - x^2, x, x + y) = - x^2 - 2\cdot x\cdot y - y^2 + y$$

Conclusion: Simultaneous substitution works in both cases correctly only if the substituted variable does not appear in the expression which replaces this variable.

DNL: *Stefan Welke has submitted a contribution overcoming this difficulty to appear in the next DNL.*

DERIVE 6 behaves as expected when applying its SUBST-command:

$$\text{SUBST}(y - x^2, [x, y], [x + y, x - y]) = - x^2 + x\cdot(1 - 2\cdot y) - y^2 - y$$

$$\text{SUBST}(y - x^2, x, x + y) = - x^2 - 2\cdot x\cdot y - y^2 + y$$

The WITH-Operator of TI-Nspire refuses performing the substitution. It needs auxiliary variables:

$y-x^2|x=a \text{ and } y=b$ $\quad b-a^2$

$y-x^2|x=x+y \text{ and } y=x-y$

"Error: Invalid constraint"

$a-b^2|a=x+y \text{ and } b=x-y$

$-x^2+x\cdot(2\cdot y+1)-y^2+y$

3/99

A.White, New York, USA **AWHITE@sbu.edu**

Sirs, I am trying to calculate the double integral

$$\int_{2}^{\infty}\int_{0.2\pi}^{0.3\pi} 5pe^{-ps}\,dp\,ds \quad \text{or equivalently} \quad \int_{0.2\pi}^{0.3\pi}\int_{2}^{\infty} 5pe^{-ps}\,ds\,dp.$$

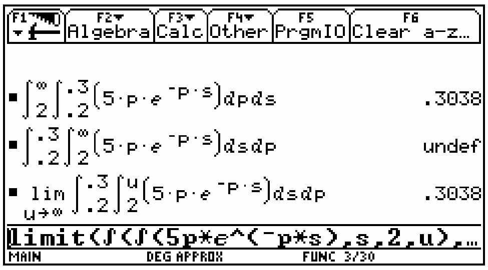
Doing these by hand yields approximatley 0.3038. Using *DERIVE* on a PC yields approximately 0.3038. However, using Derive on a TI-92 yields

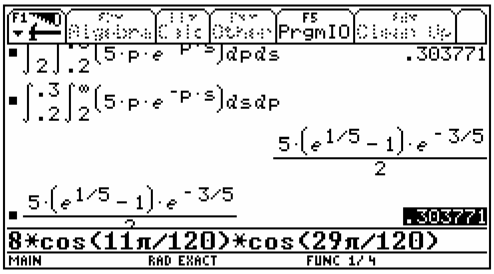
`INT(INT(5*p*#e^(-p*s),s,2,inf),p,0.2,0.3)` as a mess and

`INT(INT(5*p*#e^(-p*s),p,0.2,0.3),s,2,inf)` as undef.

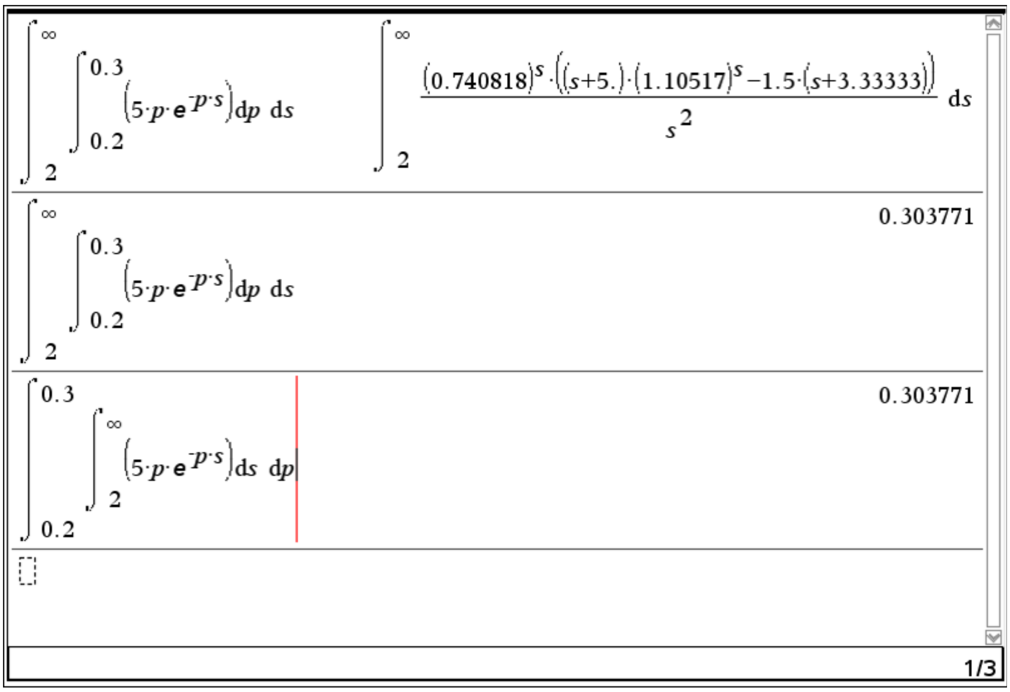
Any suggestions on how to handle these on a Ti-92?
Thank you. Al

DNL: *Compare your input with the input on the TI-edit line.Both "INT" and "INF" are DERIVE - Syntax. Use the* [∞] *and the Integral key* ([2nd] [7]). *Take care of* [e^x], *don´t enter an ordinary "e".*





The left screen is from a TI-92 (OS 2.07 of the V200 behaves the same). The right screen is from a Voage 200, OS 3.10. See below how TI-Nspire copes with this problem:



DERIVE 6:

$$\int_2^{\infty}\int_{0.2}^{0.3} 5\cdot p\cdot e^{-p\cdot s}\,dp\,ds = \frac{5\cdot e^{-2/5}}{2} - \frac{5\cdot e^{-3/5}}{2}$$

$$\int_2^{\infty}\int_{0.2}^{0.3} 5\cdot p\cdot e^{-p\cdot s}\,dp\,ds = 0.3037710248$$

$$\int_{0.2}^{0.3}\int_2^{\infty} 5\cdot p\cdot e^{-p\cdot s}\,ds\,dp = \frac{5\cdot e^{-3/5}\cdot(e^{1/5}-1)}{2}$$

$$\int_{0.2}^{0.3}\int_2^{\infty} 5\cdot p\cdot e^{-p\cdot s}\,ds\,dp = 0.3037710248$$

Ivano Moschetti, Italy **imoschet@novanet.it**

Hallo, I´m a beginner in DERIVE (and English). I´m interested in algebraic expressions and other, that seems DERIVE bugs. What is the reason of this singular problem?

For example:

- $x+1+\frac{1}{x-2} = 3+\frac{1}{x-2}$; DERIVE solve [x = 2] !?! Why??
- $\frac{\ln x - 1}{\ln x}$ in graphic plot for x = 0, y isn´t 1 !?! Why??

Write to me other examples and reason. (My English is terrible, sorry)

Ivano

George F. Feissner, USA **feissnerg@snycorva.cortland**

Moschetti´s problem is an interesting one! When one solves the equation

x + 1 + 1/(x – 2) = 3 + 1/(x – 2)

using either the SOLVE command or the SOLVE function in DERIVE, one gets the spurious solution x = 2.

I got the same result with

x^2 + 1/(x – 2) = 2 + x + 1/(x – 2).

Is DERIVE simplifying the equation by "cancelling" the 1/(x-2) from each side?

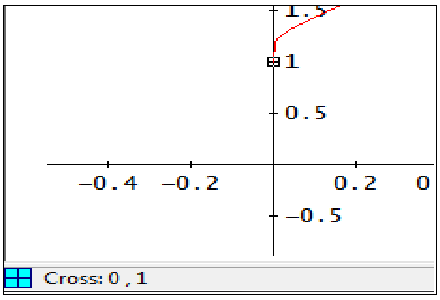
Al Rich, Honolulu, USA swh@aloha.com

As x approaches 2, the left and right side of the above equations approach the same value. Since in DERIVE, 1/0 is well defined as ± ∞, substituting x = 2 in the above equations yields equal infinite values on both sides of the equation. Thus, in DERIVE´s eyes at least, x = 2 is a valid solution.
If you are unhappy with this situation, I recommend substituting solutions returned by DERIVE back into the original equation. If the result is infinite values on the left and right sides of the equation, you are free to disregard the solution as spurious. - Aloha, Albert D. Rich, Applied Logician

DERIVE 6's answers:

$$\text{SOLVE}\left(x + 1 + \frac{1}{x - 2} = 3 + \frac{1}{x - 2},\ x\right) = (x = 2)$$

$$\text{SUBST}\left(\frac{\text{LN}(x) - 1}{\text{LN}(x)},\ x,\ 0\right) = 1$$



Alfonso Poblacion, Valladolid, Spain alfonso@gauss.mat.eup.uva.es

Problems with FOURIER

Dear Derivers,

Using DERIVE 3.0 I have found the following problem:

I want to obtain The Fourier expansion of ABS(SIN(x)) in [-π, π] . Well, I loaded INT_APPS.MTH and then simplified `FOURIER(abs(sin(x)),x,-π,π,2)`. This gave me $\frac{2}{\pi} - \frac{5\cos(2x)}{3\pi}$. If you plot this function together with the initial one, you can see at first sight that there is something that is strange in the behaviour of the second function. Then I simplified `FOURIER(abs(sin(x)),x,-π,π,4)` and obtained $\frac{2}{\pi} - \frac{5\cos(2x)}{3\pi} + \frac{\cos(3x)}{4\pi} - \frac{2\cos(4x)}{15\pi}$. Well, if you plot this you can see that this is not an expected FOURIER expansion.

Then I tried to find the coefficients from its definition with `INT(abs(sin(x))cos(nx),x,-π,π)/π` declaring n as a positive integer. Then you find $\frac{2\ (-1)^n}{\pi(1-n^2)}$. If you build the expansion with this result, this does not work either. (Not only seeing the plots, this gave me a first clue to suspect that something went wrong. By this time, I did the calculations myself by hand).

Knowing the behaviour of the Fourier expansion, then I tried a third attempt. As the function is even, the coefficients can be obtained from `2/π*int(abs(sin(x))cos(nx),x,0,π)`. Then, finally you will have the right answer: $\frac{2((-1)^n + 1)}{\pi(1-n^2)}$.

The problem is that DERIVE uses limits to evaluate definite integrals. If you try `1/π*INT(sin(x)cos(nx))` or `1/π*INT(abs(sin(x))cos(nx))`, you will find a right antiderivative, but when it is evaluating in π^+ or π^-, the program fails.

For example, with a2,

lim(antiderivative,x, π-) - lim(antiderivative,x,- π+) = -10/(6π), that is not correct.

Does this happen in later versions? If so, I think that this must be advertised.

The same problem occurs with integrals like int(abs(sin(x))cos(2x),x,0,π). DERIVE simplifies it with ?, but between 0 and 4, for example, you have a right answer.

Any comment will be appreciated. Thanks.

Leonardo Volpi, Firenze, Italy **VOLPI_LEONARDO.FI@mailbox.enel.it**

Dear Derivers,

In the last letter from Alfonso (29/8/97) , I have found a few interesting things that I would like to discuss with you.

I have had same troubles (overall in teaching) with similar problems I will try to explain with an example:

Suppose to have the following function

y = ABS(SIN(x))*COS(2*x)

and we want to compute the definite integral from -π to π. The function is continuous in this interval, and then integrable. DERIVE gives the following result:

INT(ABS(SIN(x))*COS(2*x),x,-π,π)=-2/3

That is wrong! Let's have a better look to the given function. We can redefine y in the following (didactic) form:

$$y(x)=\begin{cases} -\sin(x)\cos(2x) & \text{for } -\pi \le x < 0 \\ 0 & \text{for } x = 0 \\ \sin(x)\cos(2x) & \text{for } 0 < x \le \pi \end{cases}$$

Now we can compute its antiderivative function dy/dx = I(x):

DERIVE gives: INT(SIN(x)*COS(2*x),x) = COS(x)/2-COS(3*x)/6

So the function I, unless two arbitrary constant C1 and C2, is:

$$I(x,C1,C2)=\begin{cases} -\dfrac{\cos(x)}{2}+\dfrac{\cos(3x)}{6}+C1 & \text{for } -\pi \le x < 0 \\ 0 & \text{for } x = 0 \\ \dfrac{\cos(x)}{2}-\dfrac{\cos(3x)}{6}+C2 & \text{for } 0 < x \le \pi \end{cases}$$

We can make it continuous by a right choice of C1 and C2:

COS(0)/2 + COS(3*0)/6 + C1 = 0 , thus C1 = 1/3
COS(0)/2 - COS(3*0)/6 + C2 = 0 , thus C2= -1/3

Finally, we have:

$$I(x)=\begin{cases} -\dfrac{\cos(x)}{2}+\dfrac{\cos(3x)}{6}+\dfrac{1}{3} & \text{for } -\pi \le x < 0 \\ 0 & \text{for } x = 0 \\ \dfrac{\cos(x)}{2}-\dfrac{\cos(3x)}{6}-\dfrac{1}{3} & \text{for } 0 < x \le \pi \end{cases}$$

The function I(x) is continuous and derivable on interval - π <= x <= π and, it is: dy/dx = I(x)
So I(x) is the antiderivative function of y(x) on all interval [- π, π].
The Integral Fundamental Theorem gives:

INT(y(x),x,-π,π) = I(π) -I(-π) = -2/3 - (+2/3) = -4/3 that, of course, is correct.

Now let´s see how DERIVE works. Unfortunately, DERIVE can not integrate functions defined as above.
DERIVE uses, instead, same special functions as ABS(x), FLOOR(x), SIGN(x) etc. that makes the definition functions easier and more compact, but, on the other hand, hides same important conceptual problems.
In fact DERIVE gives the following antiderivative function:

```
INT(ABS(SIN(x))*COS(2x),x) = SIGN(SIN(x))*(COS(x)/2-COS(3x)/6)
```

We should believe that the function above is the antiderivative of y(x) on our interval. Well, we should get wrong! In fact the function SIGN(SIN(x))*(COS(x)/2-COS(3*x)/6) is not continuous (and , thus, not derivable) in x=0, π ,- π, We easy, can see it by the plot , as well, from the following:

```
SIGN(SIN(π))*(COS(π)/2-COS(3*π)/6)="+-"1/3
SIGN(SIN(0))*(COS(0)/2-COS(3*(0))/6)="+-"1/3
SIGN(SIN(-π))*(COS(-π)/2-COS(3*(-π))/6)="+-"1/3
```

DERIVE uses limits to overcome the ambiguity of sign at the end of interval (and this is correct, I think) but can do nothing for the "internal jumps" as in 0. In fact if we name:
DERIVE compute the following:

```
LIM(SIGN(SIN(x))*(COS(x)/2-COS(3x)/6),x,π,-1)=-1/3
LIM(SIGN(SIN(x))*(COS(x)/2-COS(3x)/6),x,-π,1)=1/3
INT(ABS(SIN(x))*COS(2x),x,-π, π)=-1/3 - (+1/3) = - 2/3
```

I am not sure, but we should aspect wrong results when the antiderivative function makes "jumps" into integration interval. If it happens we must divide the original interval into sub-intervals (as we usually do by hand) and integrate separately. In our example, we have:

```
INT(ABS(SIN(x))COS(2x),x,-π,π)=

INT(ABS(SIN(x))COS(2x),x,-π,0)+INT(ABS(SIN(x))COS(2x),x,0,π)=-2/3-
2/3=-4/3
```

that, finally, is correct.

--

There is another point: The function `SIGN(SIN(x))(COS(x)/2-COS(3x)/6)` returned by DERIVE is an antiderivative of same function in $-\pi < x < \pi$?

The answer is yes if we introduce the generalized function delta(x) of Dirac:

`INT(delta(x),x,-inf,inf) = 1,` delta(x) = 0 for x< 0 and x > 0

With this statement, we can write:

`d/dx(SIGN(SIN(x))(COS(x)/2-COS(3x)/6)) = ABS(SIN(x))COS(2x)+ 2/3 delta(x)` , for $-\pi < x < \pi$

Integrating each member, for $-\pi < x < \pi$,we have:

```
d/dx(SIGN(SIN(x))(COS(x)/2-COS(3x)/6)) =
      LIM(SIGN(SIN(x))(COS(x)/2-COS(3x)/6),x,π,-1) -
      LIM(SIGN(SIN(x))(COS(x)/2-COS(3x)/6),x,-π,1) = -2/3
```

```
INT(2/3 delta(x),x,-π, π) = INT(2/3 delta(x),x,-∞,∞) = 2/3
INT(ABS(SIN(x))COS(2x),x,-π,π) = I
```

So, rearranging we have: -2/3 = 2/3 + I , thus I = -4/3 and that is correct

The problem has been solved in the meanwhile as you can see below in the DERIVE 6 expression:

$$\int_{-\pi}^{\pi} |\mathrm{SIN}(x)|\cdot\mathrm{COS}(2\cdot x)\,dx = -\frac{4}{3}$$

--

Finally, another aspect of built-in function SIGN(x) that makes confusion. For x=0 , it returns: SIGN(0)="+-"1 . I hear some of my students saying:

"the sign of zero is plus or minus one..!"

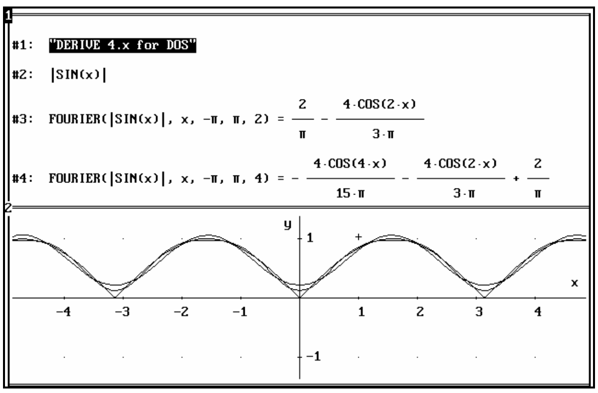
And I have to scratch this silly idea by repeating continuously that "for each x there must be one and only one.etc., etc."

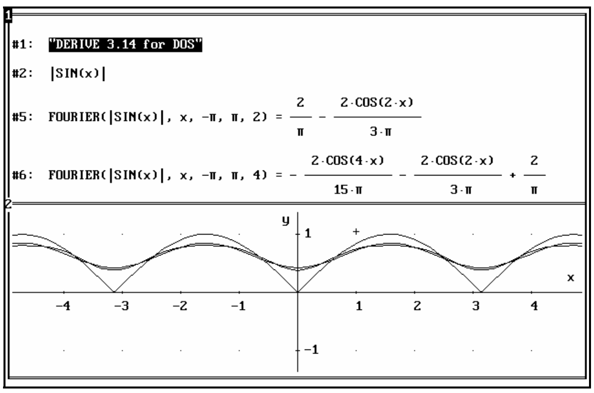
--

Hope hearing other comment. Thanks . Hi, everyone.

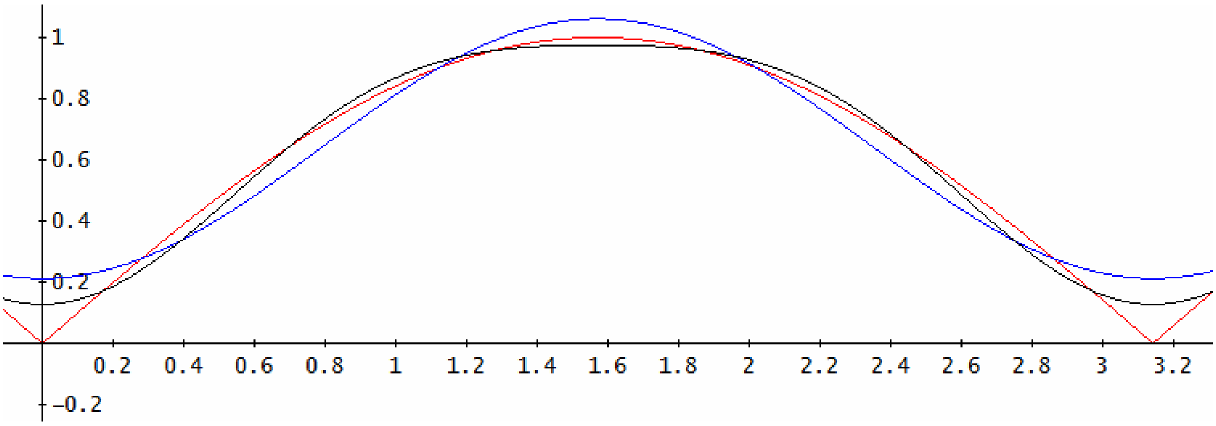
Leonardo

Compare the DERIVE screen shots: DERIVE 4.x delivers the correct FOURIER expansion. Josef





The respective DERIVE 6 screen shot is given on the next page.



(Sometimes you can follow an evolution of ideas in the Derive-News-Group. I find it useful for me - and I hope for you, too - to collect all the messages dealing with one problem and put them together. The following story is a good example of the fact that a question with four or five lines might bring a mathematical discussion into motion. Now you are invited to follow

The History of Henry Granholm's Equation

Tuesday, 20. January 1998, 16:28:44 **henry.granholm@helsinki.fi**

The equation $4^x + 7 = 2^{2-x}$ has one real solution (approximately – 0.868).

If we solve the equation algebraically we get six complex solutions, but no real solution. There is nothing wrong in the complex solutions. We find all of them, and the real solution, if we use substitution y = 2^x. (The equation is equivalent with $(2^x)^3 + 7*2^x = 4$, and the equation $y^3 + 7y = 4$ has one solution in R^+.)

This problem was found in DfW 4.07 and 4.08. Can DfD find all solutions? Are there any other similar bugs in DERIVE? - Henry

Tuesday, 20. January 1998, 18:40:19 **jeff@friedmann.com**

I have run the equation on my HP 200 based DfD and gotten SEVEN solutions as indicated (not just 6 complex ones), the first one given reduces to – 0.867943 on approximating (DERIVE solves the equation nicely without any numerical simplifying). - Jeffrey F.Friedman

Wednesday, 21. January 1998, 18:05:41 **ja72@prism.gatech.edu**

From SOLVE(4^x+7 = 2^(2-x),x) with DfW 4.0 *(and with DfD 4.06, too. Josef)*
we get:

```
         [        x = -0.867943        ]
         [ x = 1.43397 - 2.41271·î      ]
         [ x = 1.43397 + 6.65200·î      ]
#9:      [ x = 1.43397 - 11.4774·î      ]          Approx(#8)
         [ x = 1.43397 + 2.41271·î      ]
         [ x = 1.43397 - 6.65200·î      ]
         [ x = 1.43397 + 11.4774·î      ]
```

no BUG here ...... - john

Wednesday, 21. January 1998, 17:49:51 **henry.granholm@helsinki.fi**

feissnerg@snycorva.cortland.edu wrote

> What exactly did you enter and what commands did you give?

E.g. Solve - Algebraically on th eq. in question, Simplify-Basic on SOLVE(4^x + 7 = 2^(2 – x),x). It´s no difference. I get the exact solution on the complex roots, but nothing else. The result is the same if I try to solve $(2^x)^3 + 7*2^x = 4$.

I don´t get any error messages; nor any hints that there could be solutions DERIVE cannot find.

After some testing (about a dozen equations) it seems as if the problem was connected with the equation $y^3+7y=4$. The same problem seems to occur always when the equation has one positive irrational, and two complex solutions. There is no problem if the positive solution is rational, or, if the equation we get after the substitution has three real solutions. And there is no problem if the equation is of degree less than 3.

Other problematic equations are e.g. $-9^x + 3^{5-x} - 2 = 0$

$2^{-x} - 2^{2x} - 4 = 0$

$2^{3x} - 2^x = 7$

Wednesday, 21. January 1998, 21:36:00 **math@rhombus,be**

With DERIVE for DOS 3.1 the equation is solved symbolically into all 7 solutions, but the message "solutions not verified" appears! Apparently this bug creeped in from DfW 4.07 on. - Jan Vermeylen

Wednesday, 21. January 1998, 23:07:33 **Joe.H.Frisbee@usahq.unitedspacealliance.com**

A great deal more about Granholm´s equation, $4^x + 7 = 2^{2-x}$, may be learned by separating the equation into its real and imaginary parts. In particular, letting the variable x be redefined as u + î v, both u and v real, permits equations for the two parts to be obtained and plotted separately.

```
RE:   2^(2u)*COS(2*v*LN(2))-2^(2-u)*COS(v*LN(2))+7 = 0
IM:   2^(2u)*SIN(2*v*LN(2))+2^(2-u)*SIN(v*LN(2)) = 0
```

The intersections of the two sets of solutions are the ones of interest for the original equation. It turns out that there are many more of these solutions than the numbers referred to in the previous messages. In fact, it appears that there is an infinite number of solutions. This is with the principal branch only and can easily be seen in the complex polar representation of y as well.

It seems that the bigger question for any version of DERIVE is how it limits the number of solutions. By the way, I have used a DOS 3.03 and a WINDOWS 4.04 and have gotten from 3 to 9 solutions depending on which form of x is used. All of my solutions are consistent with those previously reported. - Joe.H.Frisbee, Houston

DNL: *What a wonderful email address!*

Thursday, 22. January 1998, 09:17:15 **swh@aloha.com**

When solving equations it is possible that spurious solutions may be introduced (for example, when both sides of an equation are squared). Therefore, after finding all candidate solutions, DERIVE back substitutes them back into the original equation, and only returns those that actually satisfy the equation.

Depending on the complexity of the solution and the equation, it can be very difficult to simplify the difference of the two sides of an equation to zero (this is called the zero-recognition problem in the computer algebra business). In your problem verifying the real solution to the equation requires the denesting of a cube-root of a sum of square-roots.

A major redesign of the denesting algorithm implemented in version 4.07 of DERIVE significantly expanded the class of expressions that could be denested. However, a few expressions formerly denestable were no longer.

Unfortunately, the verification of the real solution of your equation results in one of these expressions.

The problem will be resolved by version 4.09 of DERIVE. A DERIVE for Windows version 4.09 update will be available for downloading from www.derive.com as soon as it is released.

We appreciate your bringing this problem to our attention so it could be resolved.

>Can Derive for DOS find all solutions?

Since any given version of DERIVE for Windows and DERIVE for DOS use the same math engine, they will both return the same results for a math problem.

>Are there any other similar bugs in Derive?

The zero-recognition has been proven to be insoluable in general. Thus, there will always be problems that any computer algebra system, including DERIVE, cannot do. - Aloha, Albert D. Rich

(DERIVE 4.09 shows seven solutions, indeed. Josef)

Thursday, 22. January 1998, 10:57:20 **J.Wiesenbauer@tuwien.ac.at**

To All, as the flood of e-mails keeps pouring in concerning Henry Granholm's equation

```
4^x+7 = 2^(2-x)          (*)
```

I strongly feel that I should also make a contribution to it. As Henry has already pointed out one could solve that equation by substituting y = 2^x and solving the resulting equation

```
y^2+7 = 4/y             (**)
```

Subsequently for each of the three different solutions y = y1,y2,y3 of the latter equation the equation

```
2^x = y                 (***)
```

must be solved. If y1 is the unique real solution of (**), then

```
x1=LN((SQRT(1353)/9 + 2)^(1/3) - (SQRT(1353)/9 - 2)^(1/3))/LN(2)
```

is the corresponding unique real solution of (*). Surprisingly enough, after substituting x = x1 in (*) you will get the result 'wrong', i.e. according to Al Rich DERIVE fails to solve the corresponding zero-recognition problem.

This accounts for the disappearance of the real solution. But what about the complex solutions? After all, the corresponding zero-recognition problem is a lot more complicated for them and again you can show by resubstituting the solutions x = x2 and x = x3 that DERIVE fails to solve it. This is what I think might be a plausible explanation: In contrast to the real solution DERIVE doesn't even try to verify them!

Strangely enough, there is a far more serious bug in connection with this problem that has been overlooked by all contributors (with the possible exception of Joe Frisbee). Which one? Well, obviously the exact solution of (***) w.r.t. the variable x should be

```
x = log(y,2)+2k*pi*#i/ln(2),
```

where k is an arbitrary integer. For some reason DERIVE ignores all solutions except for the one with k = 0, if log(y,2) is real, and k = –1,0,1 otherwise. Why?
I haven't the faintest idea! Only God and Al Rich know the answer... At any rate, this accounts for the 6 complex solutions in the case above. (It goes without saying that I am using the current version 4.08 of DfW.)

Let me conclude by pointing out that I enjoy problems of this kind a lot (thanks to Henry!) and I would like to see more of them here! - Cheers, Johann

Friday, 23. January 1998 00:06:02 **Joe.H.Frisbee@usahq.unitedspacealliance.com**

Following Johann Wiesenbauer's lead, I have formalized what I believe to be a complete and exact solution description. There are two families of solutions, one in the left half plane (LHP) and the other in the right half plane (RHP). The previously presented real solution (~ -0.8679) is a member of the LHP family and the reported complex solutions are in the RHP family. The LHP family is expressed rather simply and it is odd that DERIVE does not give additional solutions in this family. Using the complex representation, X = U + V î , the two families of solutions (provided I have not mistyped) are:

U_LHP = (2*LN((SQRT(451) + 6*SQRT(3))^(1/3) – (SQRT(451) – 6*SQRT(3))^(1/3)) –
– LN(3))/(2*LN(2))

V_LHP = (2*N*PI)/LN(2)

U_RHP = (LN((559 – 12*SQRT(1353))^(1/3) + (559 + 12*SQRT(1353))^(1/3) + 7) –
– LN(3))/(2*LN(2))

V_RHP = PI*(2*N + 1)/(2*LN(2)) + (ASIN(1 – 3*U_RHP)*(-1)^N)/LN(2)

Notes: 1) In these equations N is an integer on (-INF,+INF).
2) The real component of X is constant in each family.
3) The arcsine function presents the solution on (0,PI/2).
4) N = 0 yields the single real solution which is in the LHP family.
5) U_LHP is the same as Johann's X1 solution of (*).

I would like to echo Johann's comment concerning interest and challenge in such problems. As a user of DERIVE, I seldom concern myself with its inner workings. Problems like this should help to keep us all aware of the complexities of mathematical problems and the software capabilities required to handle them. Also, I would like to say that we have not really been talking about a bug in DERIVE (apologies to Soft Warehouse) but more of a level of sophistication. I am continually pleased and amazed at what DERIVE can do. It is an exceptionally useful tool for applied mathematical analysis, as well as for education in which many of you are involved.

Friday, 23. January 1998 17:50:30 **J.Wiesenbauer@tuwien.ac.at**

To All,

I apologize to all who are sick and tired of getting e-mails concerning Henry Granholm's notorious equation

$$4^x + 7 = 2^{2-x}$$

but after reading Joe Frisbee's last e-mail I have a terribly guilty conscience and I simply want to get rid of it.

It appears that there is something wrong with Joe's solution of the equation above. The thing is that his second series U_RHP + #i*V_RHP doesn't make sense since V_RHP, which is supposed to be real, yields complex values due to the fact that the argument of the arcsine function is out of the range [-1,1]. Since he is referring to my "lead", this leads to the most embarassing conclusion that there must be something wrong with it.

To make up for it, I have written a DERIVE-routine for equations that can be transformed by a substitution of the form y=a^x (assuming that x is the main variable) into a solvable one:

```
EXPSOLVE(u,x,a) := VECTOR(x = LOG(y_,a) + 2·PI·@·#i/LN(a),y_,
                       RHS(SOLVE(LIM(u,LOG(y_,a)), y_)))
```

Check e.g.

```
EXPSOLVE(4^x + 7 = 2^(2 - x), x, 2)
```

as well as Henry's "other problematic equations":

```
EXPSOLVE(- 9^x + 3^(5 - x) - 2 = 0, x, 3)
EXPSOLVE(2^(3x) - 2^x = 7, x, 2)
EXPSOLVE(2^(-x) - 2^(2x) - 4 = 0, x, 2)
```

(Note: If you want to see the approximate solutions after simplifying the expressions above you should press the "Approximate"-icon twice. This will lead to a better representation!)

But the real value of the routine above should be that you can use it as a "tutorial", as it were, on solving such equations manually by studying the code!

For the rest, I agree wholeheartedly with Joe's opinion that the term "bug" might be inappropriate in the case in question. Frankly, I didn't expect DERIVE (or any other CAS for that matter) to solve such equations correctly...
Ah, I feel much better now! Apologies again... - Cheers, Johann

Friday, 23. January 1998 19:53:46 **Joe.H.Frisbee@usahq.unitedspacealliance**

To All,

Well I warned you there could be typing errors. Johann you are correct that the V_RHP is in error. The arcsine term should read ASIN(2^(1–3*U_RHP)) which will satisfy the [-1,1] requirement. With this correction the right half plane V is:

V_RHP = PI*(2*N + 1)/(2*LN(2)) + (ASIN(2^(1 – 3*U_RHP))*(-1)^N)/LN(2)

Hopefully this will put everything in a complete and errorless form. Now if I could only find some software that compares my intent to my action I'd really be happy!
Regards from: Joe Frisbee

In the following you can see how DERIVE 6.10 performs now solving Granholm's Equation. The solutions in Approximate Mode are very strage!

```
Precision := Approximate

Notation := Decimal
```

$$\left[\text{SOLUTIONS}\left(4^x + 7 = 2^{2-x}, x\right)\right]' = \begin{bmatrix} -0.8679410104 \\ \infty \\ \infty + 9.064720283\cdot i \\ \infty - 9.064720283\cdot i \\ \infty + 13.59708042\cdot i \\ \infty + 4.532360141\cdot i \\ \infty - 4.532360141\cdot i \end{bmatrix}$$

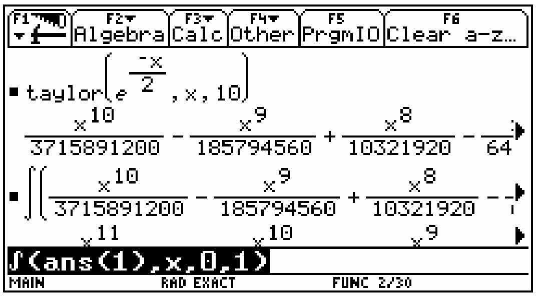
Now I am working in Exact Mode. The SOLUTIONS matrix is the result of approximating the exact results – which are looking very "exotic"

```
Precision := Exact

Notation := Rational
```

$$\left[\text{SOLUTIONS}\left(4^x + 7 = 2^{2-x}, x\right)\right]'$$

$$\begin{bmatrix} -0.8679410104 \\ 1.433970505 + 2.412717018\cdot i \\ 1.433970505 - 6.652003264\cdot i \\ 1.433970505 + 11.47743730\cdot i \\ 1.433970505 - 2.412717018\cdot i \\ 1.433970505 + 6.652003264\cdot i \\ 1.433970505 - 11.47743730\cdot i \\ -0.8679410104 + 9.064720283\cdot i \\ -0.8679410104 - 9.064720283\cdot i \end{bmatrix}$$

Josef Lechner, Viehdorf, Austria josef.lechner@online.edvg.co.at



A bug on the TI-92??? At least a strange behaviour!!

the indefinite integral: no problem, but:

the definite integral: **busy, busy, busy**, ...

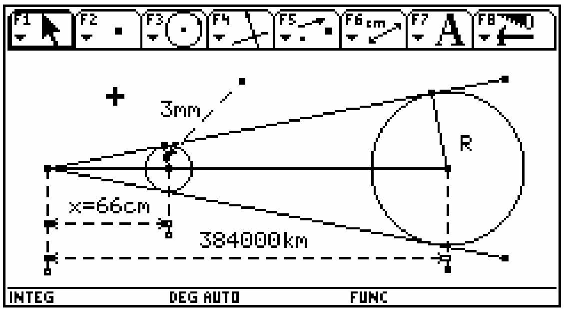
See more on page 43.

Betrachtung des relativen Fehlers bei einer Standardaufgabe zum zweiten Strahlensatz mit Hilfe des TI-92

Inspection of the relative Error in a Standard Problem to the 2nd Theorem of Rays using the TI-92

Jörg Mäß, Bonn, Germany

Eine bekannte Aufgabe zum 2.Strahlensatz lautet:



"Eine Erbse von 6mm Durchmesser verdeckt gerade den Vollmond, wenn man sie 66cm vom Auge entfernt hält. Berechne einen Schätzwert für den Mondradius." (LS 9, Klett Verlag, S 145).

"A pea with 6mm diameter hides the full moon if you hold it at a distance of 66cm from your eye. Find an estimate for the moon´s radius."

Aus dem zweiten Strahlensatz erhält man einen Schätzwert von ca 1745.5 km. Vergleicht man diesen Wert mit einer Lexikon-Angabe für den Mondradius von 1738 km (Meyers Großes Taschenlexikon, Mannheim 1987), so errechnet man einen relativen Fehler von

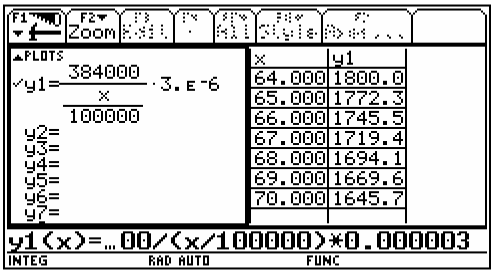
$$\frac{|1745.5 - 1738|}{1738} \cdot 100\% \approx 0.43\%.$$

The 2nd Theorem of Rays leads to an estimate of 1745.5 km. In a lexicon we find 1738 km for the moon´s radius, so we have a relative error of 0.43%. The statement "66 cm" could puzzle you. There is somebody checking if the pea is really hiding the moon and simultaneously trying to measure the pea´s distance from his eye. So the question appears how the estimates´ errors would change if somebody would (mistakenly) believe that the pea is hiding the moon at 65 cm or at 68 cm. We define together with the students a "Moon Estimation Function" y1(x) with x the distance eye - pea. The table on the TI-92 shows that this function is monotonous - as expected - and that the error depends on over- or underestimation of the pea´s distance.

Die Angabe "66 cm" könnte stutzig machen: Da versucht jemand zu erkennen, ob eine Erbse nun wirklich gerade den Mond verdeckt und unternimmt gleichzeitig Meßversuche mit einem Zollstock vor seinem Auge. Es liegt also die Frage nahe, wie sich die Fehler der Schätzwerte verändern, wenn man (irrtümlich?) glaubt, dass die Erbse den Mond schon bei 65 cm oder bei erst 68 cm verdeckt. Dazu kann man mit den Schülern eine "Mondschätzfunktion" definieren, die einem Augenabstand x [in cm] einen Schätzwert für den Mondradius y1(x) [in km] zuordnet:

$$y1(x) = \frac{384000}{\frac{x}{100000}} \cdot 0.000003 = \frac{115200}{x}$$

Ein Blick auf die Wertetabelle des *TI*-92 zeigt das erwartete Monotonieverhalten und dass es für den absoluten Fehler bedeutsam ist, ob man den Augenabstand über- oder unterschätzt.



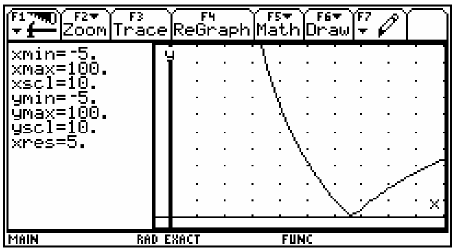
Die "Fehlerprozentfunktion" kann man mit den Schülern unter Rückgriff auf y1 entwickeln:

$$y2(x) = \frac{|y1(x) - 1738|}{1738} \cdot 100$$

Der Graph dieser Funktion ist interessant:

We produce an "Error Procent Function" using y1 and study its interesting graph. We can find some connections between mathematical facts and reality, eg
– Can you explain the asymmetric behaviour? Does it follow your experience?
– How can you interpret a point ($x_C|y_C$) presented in the "Trace"-mode?
– What is the solution of y1(x) < 5? How can you find the solution graphically using the TI-92?

Auch hier sind Verknüpfungen von mathematischen Sachverhalten mit der Wirklichkeit gegeben, beispielsweise:

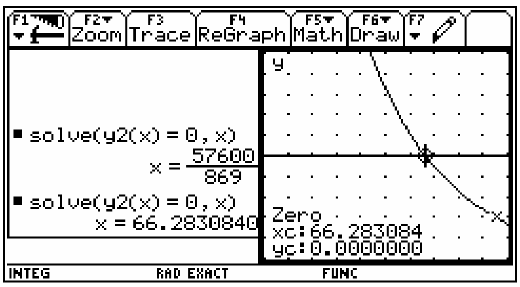


– Was bedeutet die Asymetrie? Folgt sie aus der Anschauung?
– Wie kann man einen Punkt ($x_C|y_C$), der im "Trace" Modus angezeigt wird, interpretieren?
– Was gibt die Lösungsmenge von y2(x) < 5 an? Wie kann man sie mit Hilfe des *TI*-92 ermitteln?

Schließlich kann man den *TI*-92 das Minimum der Funktion numerisch ausrechnen lassen. Es ergibt sich, unabhängig vom Untersuchungsintervall etwa T(66,28308|0,00000653). Wie ist dieser Punkt inhaltlich zu deuten? Ist er wirklich ein Tiefpunkt? Eigentlich müßte doch - anschaulich gedacht - das Minimum 0 sein. Das sieht man auch auf einer anderen Darstellungsebene, nämlich anhand der Formel:

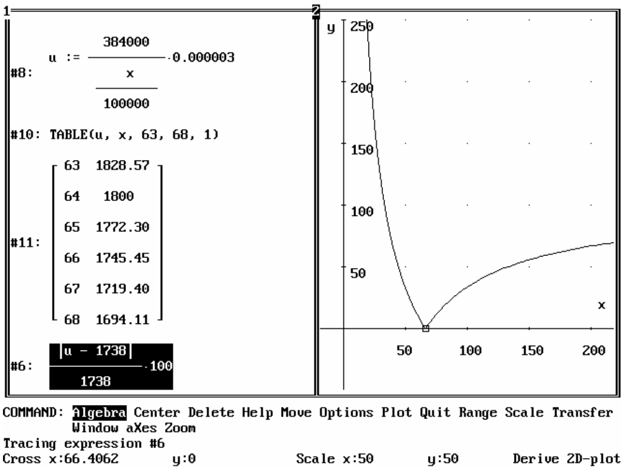
$$\frac{|y1(x) - 1738|}{1738} \cdot 100 = 0 \quad \Leftrightarrow \quad y1(x) = 1738 \quad \Leftrightarrow \quad (\text{"Solve"–Befehl}) \ x = \frac{57600}{869}$$

Die Schüler erkennen hier nicht nur die Fehlbarkeit des Rechners, sondern machen die weitgehende Erfahrung, dass man doch ein exaktes Ergebnis erhalten kann, wenn man dem Computer die richtigen Fragen stellt.



(Finding the zero(s) of y2(x) without the abs-function could be another approach. Josef)

Finally the TI-92 gives the numerical value of the function´s minimum (66.28308|0.00000653). Can you explain that point? Is that really a mimimum? It should have the value 0!! Using an equation we find our consideration confirmed. We obtain an exact solution. So the students do not only recognize the fallibility of the machine but also the fact that it might be necessary to put the question in the appropriate form.



You can do that all using *DERIVE*, too.

Produce your own TABLE-function (tblset included):

I´d like to add another question to the students:

Can you explain the endbehaviour of y2(x)?

The DERIVE Table function:

```
TABLE(u_,x_,x1_,x2_,dx):=VECTOR([x_,u_],x_,x1_,x2_,dx)
```

EXPLOITING NEW FEATURES IN DERIVE 3: MULTIPLE DECISIONS AND WHOLE STRUCTURE PROGRAMMING

Peter Mitic, Southampton, UK

ABSTRACT

Two important features were introduced in *DERIVE,* version 3, which simplify and extend programming considerable. The first is the provision for a multiple branch decision construct. The second is a new form of the `VECTOR` construct. The latter allows a whole structure programming paradigm, and both can simplify programming considerably and produce faster code. An example from data analysis is used to illustrate the effectiveness of these constructs.

INTRODUCTION

Programming in *DERIVE* is often limited by a lack of programming structures which are present in many other procedural programming languages and symbolic computation packages. We suggested the addition of several features to the *DERIVE* language, and, following some discussion with Soft Warehouse (Al Rich 1994), two specific suggestions have now been adopted in version 3. Specifically, we requested a multiple decision construct, similar to CASE in Pascal, and a loop construct such as FOR in C. The equivalents of CASE and FOR are now possible, albeit indirectly.

We show how these constructs can simplify some programming tasks considerably. Two examples are presented to illustrate them. The first is to generate a list of random numbers which does not have a uniform distribution. This process has a clear educational advantage in that it teaches a specific technique for transforming uniformly distributed random numbers in a simple way. The second is to produce a histogram of the data produced. Both are set in the context of simulating random events, such as throwing a die. *DERIVE* is also compared with Minitab in this respect.

SIMULATION OF RANDOM EXPERIMENTS

A random experiment is a trial for which there is a well defined set of outcomes, but the result of any particular trial cannot be determined in advance. Examples are throwing a die, tossing a coin, noting the manufacturer of the next vehicle which appears at a particular point in the road, or measuring the height of an object. A random variable is associated with each experiment. The internal DERIVE function we use for simulating random events is `RANDOM`, which may be adapted to simulate discrete or continuous events. `RANDOM(z) + 1` generates a random integer between 1 and z inclusive and `RANDOM(1)` generates a random seal number between 0 and 1. Both are uniform distributions. The following session shows how a sequence of 10 dice throws and a sequence of 10 random numbers in the range (0,1) may be simulated.

```
#1:   RANDOM(0) = 1543547412

#2:   die6 := VECTOR(RANDOM(6) + 1, i, 1, 10)

#3:   die6 = [5, 2, 4, 2, 1, 1, 5, 2, 5, 3]
```

This may be adapted to generate random numbers in the range (a,b) by replacing `RANDOM(1)` by `a + (b–a)*RANDOM(1)`.

Random number generation by *DERIVE* is undoubtely slower than the random number generation by a compiled or dedicated package, but is convenient to use this facility in conjunction with other algebraic facilities, and to avoid tedious import and export of data.

```
#4:   uniform01 := VECTOR(RANDOM(1), i, 1, 10)

#5:   [uniform01]'

       ┌ 0.1720337480 ┐
       │ 0.3028220951 │
       │ 0.3251353608 │
       │ 0.7732377516 │
#6:    │ 0.1898334924 │
       │ 0.4121080849 │
       │ 0.7332578500 │
       │ 0.3535702487 │
       │ 0.6601174869 │
       └ 0.9991258988 ┘
```

Don´t forget to start each session with simplifying `RANDOM(0)` to make your random numbers really "random". Otherwise you would get the same sequence of random numbers each session.

THE WHOLE STRUCTURE AND RECONSTRUCTION PROGRAMMING PARADIGMS FOR TRANSFORMING DATA

Given a list, suppose that we wish to transform every element in the list in the same way. For example, we may wish to square each element. The pseudocode for this process is as follows.

```
Procedure Transform(v)
     For i:= 1 to n
          Transform element i of vector v
     EndFor
EndProcedure
```

Transforming every element in a vector was a tedious process in *DERIVE*, version 2, and involved isolating each element in the vector by deconstruction, transforming it, and then reconstructing the vector. A template for this reconstruction programming paradigm is:

```
VECTOR(F(ELEMENT(v,i_)),i_,1,DIMENSION(v)),
```

where v is the vector to be transformed, and F() is a transformation function such as F(x):=x^2.

The improvement we suggested is to introduce a new programming paradigm which avoids deconstruction, transformation and reconstruction. In this paradigm, the vector to be transformed and the transformation function are merely combined in a single function. The syntax which has been implemented is **VECTOR(F(x),x,v).** This has two principal advantages. First, it greatly simplifies programming syntax. Secondly, it works at least three times as fast as reconstruction.

Given that the numbers generated by *DERIVE´*s RANDOM function are uniformly distributed, we can transform a list of such numbers such that the resulting numbers have a non-uniform distribution, In the case where the distribution function, *F(x)*, for a random variable, *Y*, which has a non-uniform distribution, is one to one, it is easy to transform observations from a uniform distribution described by a random variable *X* so that they have the distribution function *F(x)*. A proof of this result may be found in, for example, (Tocher 1963) and (Mitrani 1982).

In the example below we generate random numbers which are exponentially distributed with probability density function $f(X) = 2\,e^{-2X}$ ($X \geq 0$). The required transformation is $Y = \dfrac{\ln(1-X)}{-2}$.

The reconstruction paradigm is illustrated in expressions #10 - #12, below, and the whole structure programming paradigm is illustrated in expressions #14 - #17.

#7: $F(x) := \dfrac{\text{LOG}(1 - x)}{-2}$

#8: uniform03 := VECTOR(RANDOM(1), i, 1, 5)

#9: uniform03 := [0.0376575, 0.83266, 0.357145, 0.807229, 0.454272]

#10: expran3 := VECTOR(F(uniform03_i), i, 1, DIM(uniform03))

#11: expran3

#12: [0.0191924, 0.893863, 0.220918, 0.823126, 0.302817]

```
#13:  or

#14:  expran33 := VECTOR(F(x), x, uniform03)

#15:  expran33

#16:  [0.01919247973, 0.8938640492, 0.2209184065, 0.8231283389, 0.3028175581]

#17:  [0.01919247973, 0.8938640492, 0.2209184065, 0.8231283389, 0.3028175581]

#18:  uniform05 := VECTOR(RANDOM(1), i, 1, 5)

      expran := VECTOR(F(uniform05 ), i, 1, DIM(uniform05))
#19:                              i

#20:  expran

#21:  [0.308353, 0.268722, 0.292794, 0.768173, 0.461253]

#22:  expran1 := VECTOR(F(x), x, uniform05)

#23:  expran1

#24:  [0.422584, 0.604105, 0.509711, 0.277994, 1.12106]

#25:  [0.662413, 0.69277, 0.285336, 0.101508, 0.387406]

#26:  NotationDigits := 6
```

Do you note the difference: we have the same random numbers from #12 to #17 (because `uniform03` was first evaluated and the generated numbers then used several times. `uniform05` (#18) was not evaluated and then each call of `uniform05` in other expressions (here expran1) result in a new sequence of random numbers. Josef

The compactness of the new programming construct is readily apparent. The new VECTOR construct works between three and six times as fast as the deconstruction-transformation-reconstruction technique because considering each element in a list and transforming it is 'hard wired' in the DERIVE.EXE file.

More importantly, this is the first application in *DERIVE* of the concept that it is better to treat an object as a whole rather than consider individual elements in the list. This concept has been recognised for some time in *Maple* (with the function *map*) ans is a major design feature of *Mathematica* (with the function *Map*). This is, however, only a partial solution to the 'loop' problem, since it is now possible to use the new VECTOR construct to implement a loop in *DERIVE* if and only if all the elements within the scope of the loop are contained in a single list.

Any suitable package can, of course, be used to generate random numbers from a given (non-uniform) distribution. Minitab, for example, has a wide range of options for common distributions, and is only necessary to choose one and generate as much data as is required. Minitab is also usually faster than *DERIVE*. However, this does not emphasise the theoretical considerations behind generating the random number distributions concerned, and we wished to stress the relevant concepts for later work.

THE MULTIPLE DECISION CONSTRUCT AND FREQUENCY COUNTS

It proved to be very difficult to produce grouped frequency counts for a vector of random data, as generated above, despite the simplicity of the problem. In a simple example involving discrete data, given a sequence of throws of a 6-sided die, we wish to count the number of ones, twos, threes etc. For continuous data, we might wish to assign random numbers in the range (0,1) to classes (termed *buckets*) 0 - 0.25, 0.25 - 0.5, 0.5 - 0.75 and 0.75 - 1. This was such a problem in *DERIVE*, version 2, that for 100 random numbers or fewer it was preferable to produce grouped frequency counts by hand. In order to program this, we would wish to implement pseudo-code such as following, in which dice throws are counted.

```
Procedure CountVector(v)
      For i:= 1 to 6
            Initialise Count(i) to zero
      EndFor
      For i:= 1 to Elements_in_vector(v)
            Case Elements_in_vector(v) is
            1: Increment Count(1) by 1
            2: Increment Count(2) by 1
            3: Increment Count(3) by 1
            4: Increment Count(4) by 1
            5: Increment Count(5) by 1
            6: Increment Count(6) by 1
            EndCase
      EndFor
EndProcedure
```

The problem is that it was not possible to implement a construct similar to the CASE statement in Pascal directly. In addition, *DERIVE* does not support external memory variables such as `Count` in the pseudo-cose above. Consequently, such memory variables must be maintained internally to a function. A multiple IF statement can be written, but it rapidly becomes very unwieldy and difficult to alter. This makes programming difficult and the result is slow to run. One implementation for version 2 is given in Appendix 1, which is a listing of a suitable utility file, COUNT.MTH. This contains a function `POS(e,v)`, which locates the position of an element e in a vector v. This function is an important function in list processing because it can be used extensively for pattern matching.

In the session below, we simulate 10000 throws of a die. The bucket boundaries are stated as [1.5, 2.5, 3.5, 4.5, 5.5] and the vector of random numbers can be counted directly. We also generate 10000 uniformly distributed random numbers in the range (0,1) and classify them into buckets 0-0.25, 0.25 - 0.5, 0.5 - 0.75 and 0.75 - 1. The vector `buckets` is redefined. The boundaries 0 1 and 1 are implicit in that a number which is less than 0.25 necessarily correspons to the bucket 0-0.25 and a number which is greater than 0.75 necessarily corresponds to the bucket 0.75-1. The function `POSITIONS` must be called in conjunction with `COUNT`.

You have to preload the Utility file COUNT.MTH (Appendix 1).

(The sample with 10000 throws was simulated by me using DfD 3.14XM and DfD 4. The calculation times shown in the comments are from versions 3.14 and 4, and DERIVE 6, resepctively. Josef)

```
#1:   LOAD(D:\DOKUS\DNLs\DNL98\MTH29\COUNT.MTH)

#2:   buckets1 := [1.5, 2.5, 3.5, 4.5, 5.5]

#3:   COUNT(VECTOR(RANDOM(6) + 1, i, 1, 10000), buckets1)

#4:   [1678, 1630, 1691, 1671, 1683, 1647]
```

27.5 sec DfD 3.14, 34.8 sec DfD 4, 1.17 sec DERIVE 6

```
#5:   buckets2 := [0.25, 0.5, 0.75]

#6:   COUNT(POSITIONS(VECTOR(RANDOM(1), i, 10000), buckets2), buckets2)

#7:   [2512, 2515, 2511, 2462]
```

In *DERIVE,* version 3, it is now possible to write a multiple IF statement in a compact form, although the CountVector pseudo-code still cannot be implemented because memory variables outside procedures are still not supported. The functions POSITIONS and COUNT, listed in Appendix 1, can be combined in the following definition.

POSITIONS1() then simplifies to the same type of vector as expression #10, above. POSITIONS1 is a less flexible expression than the combination of POSITIONS and COUNT, because it is hard to alter the bucket boundaries by merely stating new ones. However, this amendment permits generation and counting of random numbers in about one fifth the time that was taken previously. It is therefore worthwhile.

```
#8:   POSITIONS1(v) := Σ([IF(x < 1.5), IF(1.5 ≤ x < 2.5), IF(2.5 ≤ x < 3.5),
        IF(3.5 ≤ x < 4.5), IF(4.5 ≤ x < 5.5), IF(5.5 ≤ x)], x, v)

#9:   POSITIONS1(VECTOR(RANDOM(6) + 1, i, 1, 10000))

#10:  [1682, 1708, 1707, 1628, 1629, 1646]
```

12.8 sec DfD 3.14XM, 13.4 sec DfD 4, 1.75 sec DERIVE 6

In order to increase the usefulness of POSITIONS1, it is necessary to note that construct such as IF(alpha ≤ x ≤ beta) occur repeatedly in POSITIONS1. This string is prepended and appended by terms IF(x<alpha) and IF(beta<=x) respectively. In the following function, the bucket boundaries are held in a vector *b* and VECTOR is used to implement the repeated string of IF constructs.

$$\#11:\quad \text{POSITIONS2}(v,\ b) := \Sigma\left(\left[\text{IF}(x < b_1),\ \text{VECTOR}(\text{IF}(b_i \le x < b_{i+1}),\ i,\ 1,\ \text{DIM}(b) - 1),\ \text{IF}(b_{\text{DIM}(b)} < x)\right],\ x,\ v\right)$$

The result is a nested vector of the form [e_1, [e_2, e_3, ... e_{n-1}], e_n], which is less useful than the unnested form [e_1, e_2, e_3, ... e_{n-1}, e_n]. This nested form may be 'flattened' by applying the function `FLATTEN`, which removes the inner brackets by appending the third and prepending the first element of v to its second element.

```
#12:  POSITIONS2(VECTOR(RANDOM(6) + 1, i, 1, 10000), buckets1)

#13:  [1731, [1686, 1652, 1620, 1666], 1645]

#14:  FLATTEN(v) := APPEND([v ], APPEND(v , [v ]))
                             1           2    3

#15:  FLATTEN([1731, [1686, 1652, 1620, 1666], 1645])

#16:  [1731, 1686, 1652, 1620, 1666, 1645]
```

Combining `POSITIONS2` and `FLATTEN`, we can define `POSITION3` as below:

```
#17:  POSITIONS3(v, b) := FLATTEN(POSITIONS2(v, b))

#18:  POSITIONS3(VECTOR(RANDOM(6) + 1, i, 1, 10000), buckets1)

#19:  [1636, 1625, 1707, 1709, 1642, 1681]
```

22.8 sec DfD 3.14XM, 35.1 sec DfD 4, 2.95 sec DERIVE 6

The price for using the new function `POSITIONS3` is that it takes about twice as long to execute as the less flexible `POSITIONS1`.

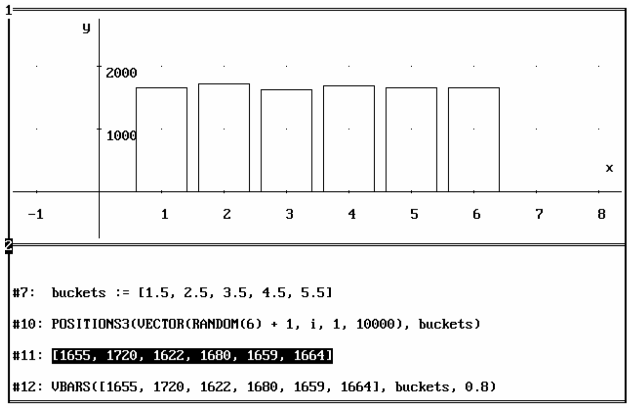
ILLUSTRATIONS FOR FREQUENCY DISTRIBUTIONS

In order to proceed with further work on data analysis, we required a means to graph a histogram of data obtained as above. The code for producing a rudimentary histogram is discussed here for the completeness.

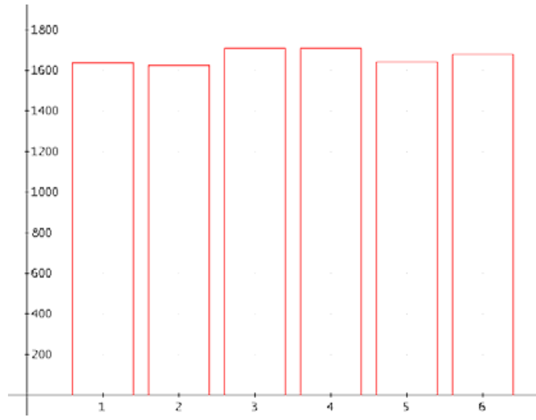
Many packages, including the majority of dedicated statistics packages and spreadsheets, and some symbolic manipulators, have built-in functions for graphing and producing histograms. *DERIVE* does not, and specific functions have to be coded. The result can be slow and inflexible in that non-equal interval widths are harder to code. However, the equal interval widths option is standard in many other packages, and anything more sophisticated is not.

Functions for producing a histogram in which the interval widths are constant are contained in BARCHART.MTH, listed in Appendix 2. We illustrate the case of graphing the grouped frequency distribution of 10000 random numbers which simulate tossing a fair die 10000 times. The output (#13) obtained by simplifying
VBARS([1655,........],buckets,0.8) is not given because it is long and only serves to provide data for producing the graph.

```
#20:  LOAD(D:\DOKUS\DNLs\DNL98\MTH29\BARCHART.MTH)

#21:  VBARS([1636, 1625, 1707, 1709, 1642, 1681], buckets1, 0.8)
```



DERIVE for DOS



DERIVE 6.10

Alternatively, the problem can be solved by exporting a vector as a text file, manipulating it using an external program and then importing the result. We have used the same method (Mitic and Thomas 1995) to remove unwanted symbols from a vector. This was succesful, but at the expense of a non-negligible Pascal programming overhead. In this case, it would only be worth pursuing if the time gained from using a compiled language is greater than the time lost by creating the export text file and using it elsewhere. *DERIVE* does have the advantage that it provides an integrated environment in which simulation, graphing and algebraic manipulation can all be done.

CONCLUSION

Two new advances to the *DERIVE* programming language have been identified and examples of their efficacy have been presented. Although they are not a complete solution to the loop construct and to the multiple decision construct which we originally envisaged, they can be applied with reasonable ease, and adapted to more general cases. We intend to use the simulation and illustration activities presented here to illustrate some concepts in probability and statistical models in a later paper.

Using *DERIVE* is not without its problems and in some cases, computations are better done by hand, usually because this is quicker. This is clearly not practical with a large amount of data, so the need to solve problems by programming becomes important. Random number generation, classification of data and graphing can all be performed faster and with simpler programming by other packages. Significant improvements in version 3 serve to mitigate. The integrated symbolic manipulation environment offered by *DERIVE* can outweigh its speed and programming disadvantage when non-numerical tasks are involved.

REFERENCES

Mitic, P and Thomas, P. (1995) Modelling the Effects of a Firebreak with DERIVE. International DERIVE Journal #2, July, Ellis-Horwood

Mitrani, L (1982) Simulation techniques for discrete event systems, Cambridge University Press

Tocher, K.D. (1963) The Art of Simulation, English University Press

Rich, A.D (1994) Private Communication with Soft Warehouse

APPENDIX 1

Utility file COUNT.MTH

These functions may be used to produce a frequency count for elements in the vector *v*, given bucket boundaries in the vector *b*.

`INCV(v,p)` increments the p^{th} element of *v*. Next, we define functions to find the position of an element, *e*, of *v*. `POS(e,v)` produces an integer in the range [1,*n*]. If *e* is less than the first element of *v*, `POS(e,v)` returns 1. If *e* is greater than the last element of *v*, `POS(e,v)` returns *n*+1. Otherwise `POS(e,v)` calls `MIDPOS(e,v)` which returns an integer in the range [2,n], which indicates the bucket to which it belongs.

`POSITIONS(v,b)` analyses each element in the vector *v* in this way.

`INIT_COUNT(v)` initialises the count for each bucket to zero and `COUNT(v,b)` maintains a tally of elements in vector *v* based on the class boundaries vector, *b*.

```
INCV(v,p):=REPLACE_ELEMENT(ELEMENT(v,p)+1,v,p)

MIDPOS(e,v):=MAX(ITERATES(IF(ELEMENT(v,i_)>=e,i_,i_+1),i_,1,DIMENSION(v)))

POS(e,v):=IF(e<ELEMENT(v,1),1,IF(e>ELEMENT(v,DIMENSION(v)),
          DIMENSION(v)+1,MIDPOS(e,v)))

POSITIONS(v,b):=VECTOR(POS(ELEMENT(v,u_),b),u_,1,DIMENSION(v))

INIT_COUNT(v):=ITERATE(APPEND(w_,[0]),w_,[],DIMENSION(v)+1)

COUNT(v,b):=ELEMENT(ITERATE([1+ELEMENT(c_,1),INCV(ELEMENT(c_,2),
           ELEMENT(v,ELEMENT(c_,1)))],c_,[1,INIT_COUNT(b)],DIMENSION(v)),2)
```

APPENDIX 2

Utility file BARCHART.MTH

The midpoints, left hand- and right hand end points of the bars to be produced are calculated by the functions `X_MID(b)`,`X_END1(b)` and `X_END2(b)`. These are combined in the vector `X_ORDS(b)`. `RECT(a,b,h,w)` builds a rectangle of height *h*, and width *w* with (*a,b*) as centre point of its base. The function VBARS(ct,b,w) produces a vector of rectangles, using data in a vector *ct*, which can then be plotted. It is necessary to set the following plot options to obtain reasonable looking bars: Plot Color Cycling off (Color Auto No in DfD) and Points Connected and Size Small (in the State submenu in DfD).

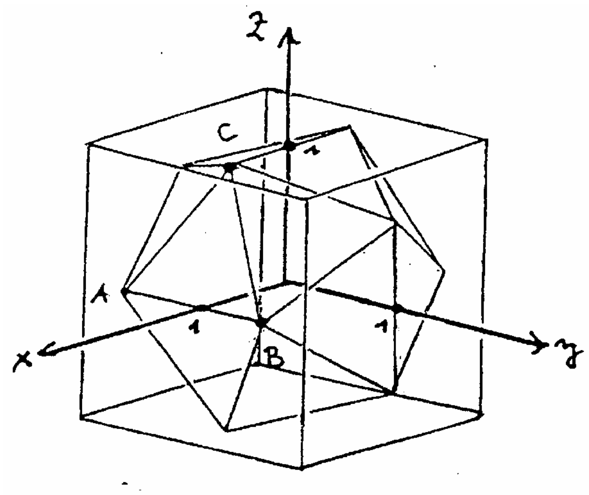
```
X_MIDS(b):=VECTOR((b SUB (i+1)+b SUB i)/2,i,1,DIMENSION(b)-1)

X_END1(b):=(-b SUB 2+3*b SUB 1)/2

X_END2(b):=(-b SUB (DIMENSION(b)-1)+3*b SUB DIMENSION(b))/2

X_ORDS(b):=APPEND(APPEND([X_END1(b)],X_MIDS(b)),[X_END2(b)])

RECT(a,b,h,w):=[[a-w/2,b],[a-w/2,b+h],[a+w/2,b+h],[a+w/2,b]]

VBARS(ct,b,w):=VECTOR(RECT((X_ORDS(b)) SUB i,0,ct SUB i,w),i,1,
               DIMENSION(X_ORDS(b)))
```

Darstellung von Polyedern - Platonische und archimedische Körper im Mittelpunkt

Hubert Weller, Lahnau, Germany

Die platonischen Körper - *The Platonic Solids*

Zur Bestimmung der Koordinaten der Eckpunkte werden auch die Dualitäten der Körper (z.B. Würfel <==> Oktaeder) ausgenutzt. Interessant ist die Berechnung der Koordinaten der Eckpunkte von Ikosaeder und Dodekaeder.



For computing the vertices we use the duality of the solids (eg Cube ⇔ Octahedron). Of special interest is the calculation of the vertices of an icosahedron and a dodecahedron.

Icosahedron:
A := [1,-s, 0], B := [1,s,0] *and* C := [s,0,1] *with*
$\overline{AB} = \overline{AC} = \overline{BC} = 2s$ *gives* $s = \frac{\sqrt{5}}{2} - \frac{1}{2}$.

Der Ikosaeder läßt sich aus einem Würfel aufbauen,wobei wir das Koordinatensystem in die Mitte des Würfels legen. Mit dem Ansatz A := [1, -s, 0], B := [1, s, 0] und C := [s, 0, 1] läßt sich die Zahl s berechnen, da alle Kanten die Länge 2s haben.

IKO_DODE.MTH

```
#1:   The Icosahedron in the cube; the cube:

#2:   [w1 := [1, -1, -1], w2 := [1, 1, -1], w3 := [-1, 1, -1], w4 := [-1, -1, -1]]

#3:   [w5 := [1, -1, 1], w6 := [1, 1, 1], w7 := [-1, 1, 1], w8 := [-1, -1, 1]]

#4:   cube := [w1, w2, w3, w4, w1, w5, w6, w2, w6, w7, w3, w7, w8, w4, w8, w5]

              ┌ -0.5  -0.25 ┐
#5:   proj := │  1     0    │
              └  0     1    ┘

#6:   cube•proj

#7:   The vertices of the Ikosaeder - The vertices of the Icosahedron

#8:   [a := [1, -s, 0], b := [1, s, 0], c := [s, 0, 1], d := [0, 1, s], e := [0, 1, -s]]

#9:   and so on .......
```

#10: Calculation of s with |ac| = 2s

#11: $(a - c)\cdot(a - c) = (2\cdot s)^2$

#12: $\left[s = \frac{\sqrt{5}}{2} - \frac{1}{2},\ s = -\frac{\sqrt{5}}{2} - \frac{1}{2}\right]$

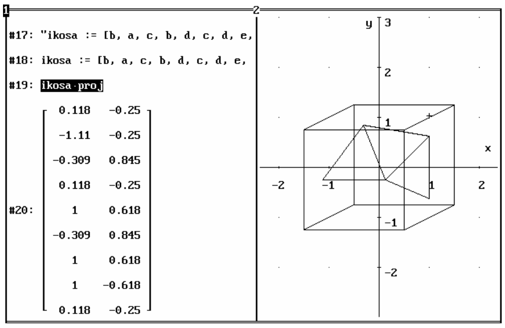
#13: Only the positive solution makes sense

#14: $s := \frac{\sqrt{5}}{2} - \frac{1}{2}$

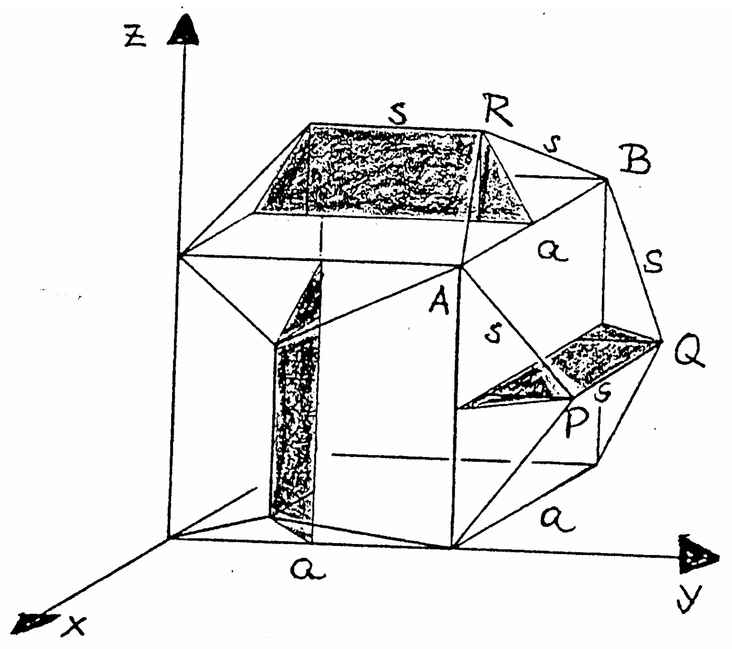
#15: ikosa = [b, a, c, b, d, c, d, e, b, ...]

#16: ikosa := [b, a, c, b, d, c, d, e, b]

#17: ikosa·proj



Auch der Dodekaeder läßt sich aus einem Würfel aufbauen, indem auf jede Würfelfläche ein Walmdach mit der Höhe h aufgesetzt wird.



The dodecahedron can be constructed from a cube attaching an Italian roof with height h on each side face. Let the length of the cube edges a = 2 and the length of all the dodecahedron´s edges s. So you can easily find values for h and s.

Die Kantenlänge des Dodekaeders sei s. Die Kantenlänge des Würfels sei a = 2. Damit werden zunächst die Eckpunkte des Dodekaeders definiert:

A := [0 ,2 ,2] , B := [-2 ,2, 2] , P := [- 0.5(2 – s), 2 + h, 1], Q := [-1 – 0.5s, 2 + h, 1]
und R := [-1, 1 + 0.5s, 2 + h] usw...

Da alle Kanten des Fünfecks die Länge s (auch AP) und alle Diagonalen die Länge 2 haben (auch AQ), können die Werte für s und h berechnet werden.

IKO_DODE.MTH (cont.)

#18: Der Dodekaeder um den Würfel – The Dodecahedron around the cube

#19: Das Walmdachdodekaeder – The hip-roof-dodecahedron

#20: [c1 := [0, 2, 2], c2 := [-2, 2, 2]]

#21: [p := [- 0.5·(2 - s), 2 + h, 1], q := [-1 - 0.5·s, 2 + h, 1], r := [-1, 1 + 0.5·s, 2 + h]]

#22: and so on ....

#23: Calculation of s and h; |c1p| = s and |c1q| = 2

#24: $(c1 - p)\cdot(c1 - p) = s^2 = \left(h^2 + \frac{s^2 - 4\cdot s + 8}{4} = s^2\right)$

#25: $(c1 - q)\cdot(c1 - q) = 2^2 = \left(h^2 + \frac{s^2 + 4\cdot s + 8}{4} = 4\right)$

#26: $\text{SOLUTIONS}\left(h^2 + \frac{s^2 - 4\cdot s + 8}{4} = s^2, h\right) = \left[\frac{\sqrt{(3\cdot s^2 + 4\cdot s - 8)}}{2}, -\frac{\sqrt{(3\cdot s^2 + 4\cdot s - 8)}}{2}\right]$

#27: $\text{SOLUTIONS}\left(h^2 + \frac{s^2 + 4\cdot s + 8}{4} = 4, h\right) = \left[\frac{\sqrt{(- s^2 - 4\cdot s + 8)}}{2}, -\frac{\sqrt{(- s^2 - 4\cdot s + 8)}}{2}\right]$

#28: Only the positive solutions make sense!

#29: $\text{SOLUTIONS}\left(\frac{\sqrt{(3\cdot s^2 + 4\cdot s - 8)}}{2} = \frac{\sqrt{(- s^2 - 4\cdot(s - 2))}}{2}, s\right) = [\sqrt{5} - 1, - \sqrt{5} - 1]$

#30: $s := \sqrt{5} - 1$

#31: $h := \frac{\sqrt{(- s^2 - 4\cdot(s - 2))}}{2}$

#32: $h := \frac{\sqrt{5}}{2} - \frac{1}{2}$

#33: Eine Dodekaederfläche:

#34: dofl1 := [c1, p, q, c2, r, c1]

#35: dofl1·proj

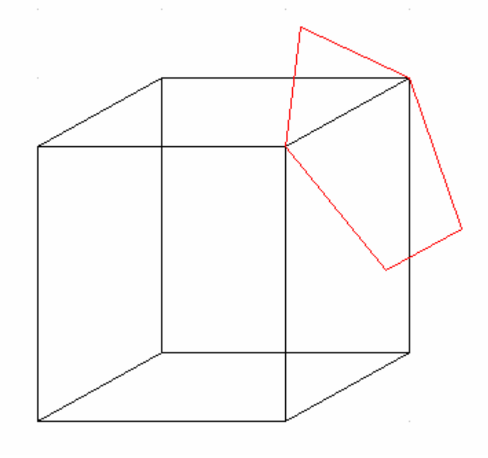
#36: [c3 := [-2, 0, 2], c4 := [0, 0, 2], c5 := [0, 2, 0]]

#37: [c6 := [-2, 2, 0], c7 := [-2, 0, 0], c8 := [0, 0, 0]]

#38: cube2 := [c1, c2, c3, c4, c1, c5, c6, c2, c6, c7, c3, c7, c8, c4, c8, c5]

#39: cube2·proj

Next page shows the plot of the cube together with one face of the dodecahedron.



Verdeckte Kanten und Determinanten *Hidden lines and Determinants*

Die bisher benutzte Darstellung ist besonders einfach, denn ein Körper wird durch eine einzige Punktfolge (d.h. durch eine Matrix) beschrieben. Allerdings hat diese Form der Darstellung auch große Nachteile, denn u.U. werden einige Kanten mehrmals gezeichnet und man sieht alle verdeckten Kanten. Deshalb haben wir uns überlegt, wie die Form der Darstellung verbessert werden kann.

Wenn sich ein **konvexer** Körper im Raum bewegt, weiß man, dass eine Fläche, die vorher sichtbar ist, in dem Moment, in dem sie „verschwindet“ (oder eine vorher nicht sichtbare Fläche „auftaucht“), ihre ORIENTIERUNG ändert. Dies ist die entscheidende Idee bei den folgenden Überlegungen.

Die DETERMINANTE einer 2x2-Matrix läßt sich deuten als Flächeninhalt des von den beiden Zeilen-vektoren aufgespannten Parallelogramms. Das Vorzeichen gibt dabei eine Information über die Orientierung, d.h. wenn der erste Zeilenvektor auf direktem Weg in den zweiten gedreht wird, dreht man bei einer positiven Determinante im mathematisch positiven Sinn, bei einer negativen Determinante im mathematisch negativen Sinn.

Z.B. ist

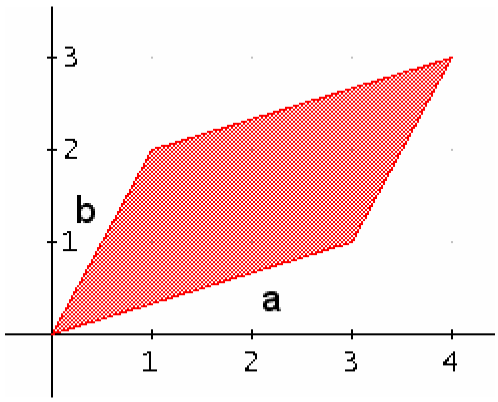
$$DET\begin{pmatrix} 3 & 1 \\ 1 & 2 \end{pmatrix} = 5$$

Das von den Vektoren $\vec{a} = [3 \quad 1]$ und $\vec{b} = [1 \quad 2]$ aufgespannte Parallelogramm hat den Flächeninhalt 5.

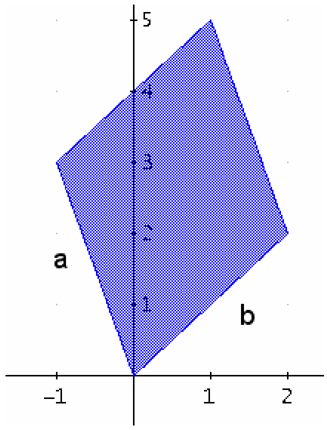
It is known that a side face of a convex solid when moved in space is changing its orientation at the moment when it is changing its visibility. This is leading to the following algorithm:

The determinant of a 2 × 2 matrix can be interpreted as the area of a parallelogram built by the two row vectors r_1 and r_2. Its sign gives information about its orientation. (= 1 if a positive rotation would move r_1 into r_2).

The area of the parallelogram built by the vectors $\vec{a} = [3 \quad 1]$ and $\vec{b} = [1 \quad 2]$ is 5.



Das von den Vektoren $\vec{a} = [-1 \quad 3]$ und $\vec{b} = [2 \quad 2]$ aufgespannte Parallelogramm hat den Flächeninhalt -8; dieser ist negativ, weil das Parallelogramm eine andere Orientierung hat. Wir benutzen das als Entscheidungskriterium dafür, ob eine Fläche gezeichnet werden soll oder nicht.



Here the parallelogram´s area is -8. We use the sign as a criterion whether a side face should be plotted or not.

Eine andere Art der Darstellung von konvexen Körpern

Another way to represent convex solids

Die Eckpunkte werden wie bisher als Zeilenvektoren beschrieben. Z.B. bei einem Würfel:

`p1:=[1,0,0], p2:=[1,1 0],` ....

Da allerdings bei jeder Fläche entschieden werden soll, ob sie gezeichnet werden soll oder nicht, stellen wir eine Fläche als Punktfolge (geschlossener Streckenzug) dar:

`fl1:=[p1,p2,p6,p5,p1 ], fl2:=[p2p3p7p6p2 ], ...`

Bei der Definition der Flächen muß darauf geachtet werden, dass die Punktfolge beim Blick von außen auf die Fläche im mathematisch positiven Sinn durchlaufen wird.

Einen Körper beschreiben wir nun als Vektor von Flächen, d.h. einen Vektor von Matrizen:

`cube:=[fl1,fl2,fl3,fl4,fl5,fl6]`

Zum Test, ob eine Fläche gezeichnet werden soll oder nicht, werden nur die ersten 3 Punkte dieser Fläche benutzt. Das Vorzeichen der Determinante derjenigen Matrix, in deren ersten Zeile die Projektion des Vektors vom 2. zum 3.Flächenpunkt und in deren zweiten Zeile die Projektion vom 2. zum 1.Flächenpunkt steht, liefert das Kriterium, ob die Fläche gezeichnet wird. (Den i-ten Punkt der k-ten Fläche eines Körpers erhält man mit dem Befehl `cube SUB k SUB i` oder über `ELEMENT(cube,k,i))`.

The points are written as row vectors. Each side face is defined by a closed sequence of points. The solid is described by a vector of side faces (see `cube:= [...]` *above). Defining the side faces (polygons) you have to take care that the sequence of points has to run through in positive direction when you look at the polygon from outside of the solid. For testing the orientation you need only the coordinates of the first three - projected - points.*

$$\text{TEST(solid, k)} := \text{DET}\left(\left[\text{solid}_{k,3} - \text{solid}_{k,2}, \text{solid}_{k,1} - \text{solid}_{k,2}\right] \cdot \text{proj}\right)$$

Eine Fläche wird gezeichnet, wenn beim Test ein positiver Wert berechnet wird, sonst wird sie nicht gezeichnet!

The side face will be plotted if the test results in a positive value!

$$\text{PIC\_2\_D(solid)} := \text{VECTOR(IF(TEST(solid, k)} > 0, \text{solid}_k \cdot \text{proj), k, 1, DIM(solid))}$$

HID_LINE1.MTH

```
#1:   Verdeckte Kanten bei konvexen Körpern - Hidden lines on convex solids
#2:   Die Determinante als orientierter Flächeninhalt wird benutzt
#3:   Die Eckpunkte des Körpers - the vertices
#4:   [p1 := [1, 0, 0], p2 := [1, 1, 0], p3 := [0, 1, 0], p4 := [0, 0, 0]]
#5:   [p5 := [1, 0, 1], p6 := [1, 1, 1], p7 := [0, 1, 1], p8 := [0, 0, 1]]
#6:   Die Flächen - dabei auf mathematisch positiven Umlaufsinn achten !!
#7:   The faces - take care of the positive rotation direction
#8:   [fl1 := [p1, p2, p6, p5, p1], fl2 := [p2, p3, p7, p6, p2], fl3 := [p3, p4, p8, p7, p3]]
#9:   [fl4 := [p4, p1, p5, p8, p4], fl5 := [p8, p5, p6, p7, p8], fl6 := [p4, p3, p2, p1, p4]]
#10:  Der Körper - the solid
#11:  cube := [fl1, fl2, fl3, fl4, fl5, fl6]
#12:  Die Projektionsmatrix - the projection matrix
```

#13: $\text{proj} := \begin{bmatrix} -0.5 & -0.25 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$

```
#14:  Der Test, ob eine Fläche gezeichnet werden soll - plotting test: plot or not plot?
```

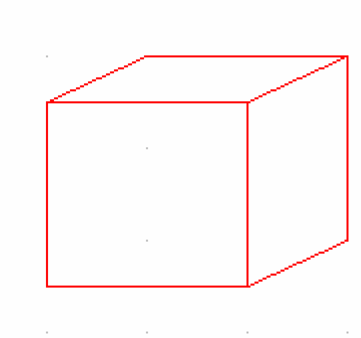
#15: $\text{TEST(solid, k)} := \text{DET}\left(\left[\text{solid}_{k,3} - \text{solid}_{k,2},\ \text{solid}_{k,1} - \text{solid}_{k,2}\right]\cdot\text{proj}\right)$

```
#16:  Das 2D-Bild - the 2D-projection
```

#17: $\text{PIC\_2\_D(solid)} := \text{VECTOR(IF(TEST(solid, k)} > 0,\ \text{solid}_k\cdot\text{proj}),\ k,\ 1,\ \text{DIM(solid))}$

```
#18:  PIC_2_D(cube)
```

Plot #18 in order to obtain an image of the cube:



Bewegungen des Körpers im Raum können mit Hilfe von Matrizen dargestellt werden, dabei werden ausschließlich die sichtbaren Flächen gezeichnet. Als Beispiel soll ein Würfel auf die Spitze gestellt werden.

Movements in space can be realized using matrices. Only the visible side faces will be plotted. As an example we represent a cube standing on its top.

#19: Wir stellen einen Würfel auf die Spitze! – The cube on its top!

#20: 45°-Drehung um die z-Achse – 45°-Rotation around z-axis

#21: $DZ(\alpha) := \begin{bmatrix} COS(\alpha) & SIN(\alpha) & 0 \\ -SIN(\alpha) & COS(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix}$

#22: $cube1 := VECTOR(cube_k \cdot DZ(45 \cdot 1°), k, 1, DIM(cube))$

#23: Rotation around the x-axis – Drehung um die x-Achse

#24: The cube's diagonal is rotated into the z-axis (around the x-axis)

#25: $DX(\alpha) := \begin{bmatrix} 1 & 0 & 0 \\ 0 & COS(\alpha) & SIN(\alpha) \\ 0 & -SIN(\alpha) & COS(\alpha) \end{bmatrix}$

#26: $\beta := ATAN\left(\frac{1}{\sqrt{2}}\right)$

#27: $cube2 := VECTOR\left(cube1_k \cdot DX\left(\frac{\pi}{2} - \beta\right), k, 1, DIM(cube1)\right)$

#28: Der Würfel steht am Kopf! – The cube is standing on its top!

#29: $ax1 := \begin{bmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ \infty & \infty & \infty \\ 0 & 1 & 0 \\ 0 & 2 & 0 \\ \infty & \infty & \infty \\ 0 & 0 & 1 \\ 0 & 0 & 3 \end{bmatrix}$

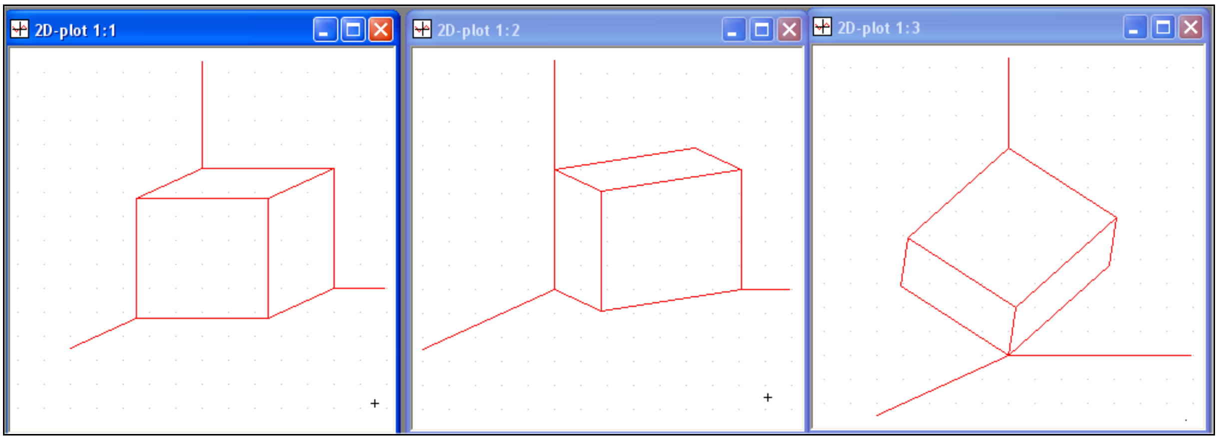
#31: $ax2 := \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ \infty & \infty & \infty \\ 0 & \sqrt{2} & 0 \\ 0 & 2 & 0 \\ \infty & \infty & \infty \\ 0 & 0 & 0 \\ 0 & 0 & 3 \end{bmatrix}$

#33: $ax3 := \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ \infty & \infty & \infty \\ 0 & 0 & 0 \\ 0 & 2 & 0 \\ \infty & \infty & \infty \\ 0 & 0 & \sqrt{3} \\ 0 & 0 & 3 \end{bmatrix}$

#30: [ax1·proj, PIC_2_D(cube)]

#32: [ax2·proj, PIC_2_D(cube1)]

#34: [ax3·proj, PIC_2_D(cube2)]



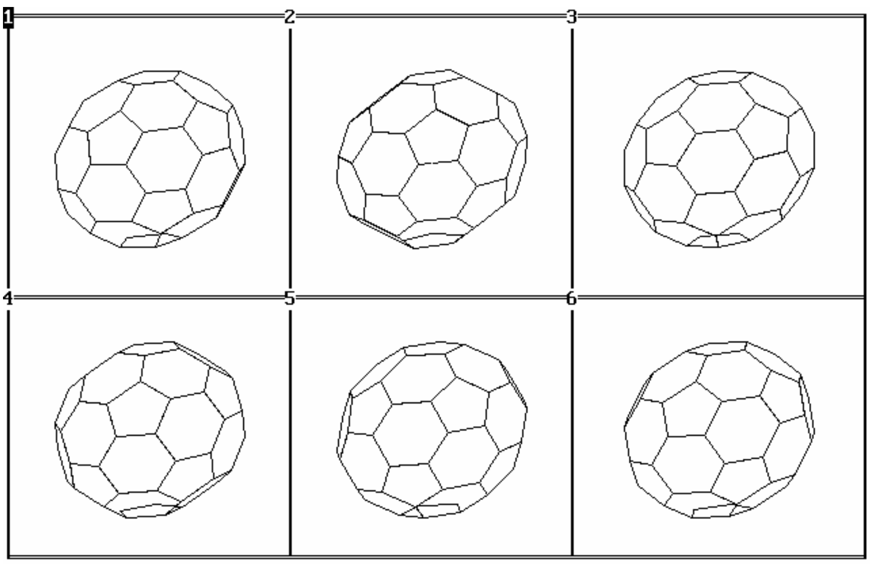
Literatur - *References*

H.-J. Elschenbroich/J.-C. Meiners, Computergraphik und Daestellende Geometrie im Unterricht der Linearen Algebra, Dümmler (Bonn) 1994

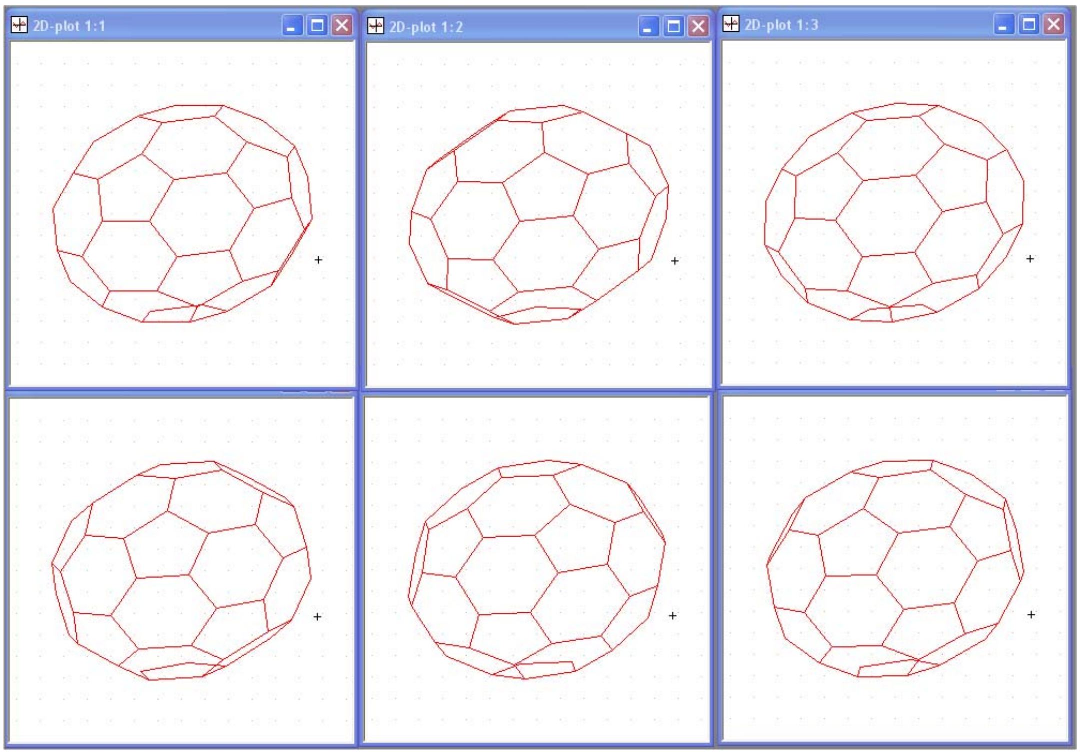
W.Kroll (Hrsg.), Neue Impulse für die Raumgeometrie, mathematik lehren Heft 77 1996
Friedrich Verlag (Velber)

H.Weller, Determinanten in einem Kurs Lineare Algebra, DdM 1 , 1979 , (S.62-72)

E.-C.Wittmann, Elementargeometrie und Wirklichkeit, Vieweg (Braunschweig) 1987



In DNL#21 Richard Schorn presented the "Bucky Ball" - a model of the C60-molecule (12 pentagons and 20 hexagons. I used the data of C60LAPHI.MTH and Richard Schorn´s sketch from page 42 to present the Bucky Ball and let it rotate round the z-axis. Josef. (BUCKY.MTH)



Dear Joseph, North Balwyn, 5 April 1996

I have various findings, which I believe both the Derive community and the mathematical community will find of interest, so I shall start to send them to you successively, this letter being the largest by way of introduction.

The first, I have called the "Cesaro Glove-Osculant", a Super-Osculant, eine besondere Schmiegungstangentenkurve, a sheath osculant. It opens a new and innovative look at osculants, and simultaneously gives one the power to generate images that closely simulate a sought-after image and that may have been too difficult to discover by other means. It was introduced by Ernesto Cesaro in his famous tome, Vorlesungen Über Natürliche Geometrie around the turn of the century.

I have opened a mathematical 'can of worms', too much for one person to develop to its full potential, so I invite fellow members of DUG or readers of the Newsletter to write to me, in English, and perhaps we can form a team for further development of this line of research.

Enclosed the paper SUPEDUPE.WPF on disk and print-out, as well as Derive files, OSCCES1n.MTH and two image files, TAUTUREn.WPG on disk.

Yours sincerely,

David Halprin.

SUPER DUPER OSCULANTS

David Halprin, North Balwyn, Australia

ABSTRACT & INTRODUCTION

As an initial analogy, consider the case of a form-fitting glove (Handschuh), [or sheath (Scheide)], for a hand (or surface) in three dimensions. Suppose that there exists a machine for manufacturing the sheath, which depends on a set of three parametric equations being fed into the computerised instructions and, if one provides these equations to define the surface, then out comes the sheath exactly form-fitting. Suppose there exists a set of surfaces, whose definition is in terms of intrinsic variables, and which cannot be transformed into a Cartesian parametric set. This means that the machine cannot produce the appropriate sheaths until and unless an algorithmic method of close approximation can be devised to transform the equations appropriately.

This paper deals with the two-dimensional counterpart to the sheath, which we can describe as a piece of bent wire, which can be laid on top of the curve in the plane, and covering it exactly. If the wire were bent by a machine, which depended on the equation of the curve being given in a pair of parametric Cartesian equations, then it would produce an exact replica of the curve. If, however, we could only represent the curve in Intrinsic variables, then we would hope to approximate the curve with a pair of parametric equations, which, although not being exactly the defined curve, would so closely approximate it as to be almost error free for most of its chosen arc. Such a curve could be described best as a very special type of osculant, as yet, not dealt with in papers and texts on osculants, since it osculates over almost all the curve, rather than some small arc centred about a nominated point, or group of points, on the curve.

Hence this paper is to show the theory and derivation of the algorithm and then to exemplify it in a routine, (in Derive or DeriveXM), which can illustrate almost any curve one desires to show.(See enclosed OSCCES1N.MTH files)

SUPER-OSCULANTS (THEIR KITH AND KIN)

An Osculant in the plane can be looked upon as one of several possible mathematical devices or tools, (some inter-related), to assist one in the investigation of a geometrical image of all, (or part), of a nominated base curve in the plane. Its usage may be:-

A) direct, in any one of at least four separate categories of osculants, (Penosculants, General Osculants, Chief-Osculants & Super-Osculants)

or

B) indirect and/or incipient, (perhaps via overt consideration of order-of-contact), and/or one or more of the ascending series of geometrical properties, (qualitative and quantitative), starting with:-

1) Gradient, [first order], then
2) Radius of curvature, [second order], and
3) Deviation (aberration) or spiralation, [third order].
4) ⇒ not to mention often limitless higher order possibilities.

A plane curve may be investigated in many different ways, to elucidate its various geometric properties. One way draws comparison of the base curve with other curves, (such as osculants), by comparing features. viz:-

FIRST ORDER

The gradient of the base curve at a nominated point is the gradient of the osculating straight line, (the tangent), at that point.

SECOND ORDER

The radius of curvature of the base curve at a nominated point is the radius of the osculating circle.

THIRD ORDER

The deviation (aberration) of a base curve at a nominated point is the deviation of an osculating parabola.

The spiralation of a base curve at a nominated point is the spiralation of an osculating logarithmic spiral.

In all, (but one), of the above-cited cases we have used, as nominated osculants, those curves, (called "Chief-Osculants"), which have the chosen quality as a constant, namely:

a straight line has a constant gradient,
a circle has a constant radius, and
a logarithmic spiral has both constant deviation and constant spiralation.

N.B. A parabola does NOT have constant deviation, despite some authors having errone-ously stated an alleged constancy.

The chief-osculants of higher orders than 3 are difficult to plot, especially from the fifth order onwards, hence the particular geometric image associated with each is nigh impossible to envisage, yet a mathematician can deem it to be so visualised for the purpose of using it as a referential device, (both qualitatively and quantitatively), when describing some new base curve. There are, however, better ways to gain advantage from the osculant concept, namely by using:-

A) Some specially selected general osculants of a chosen order of contact, [requiring integration, (either producing a closed form resulting equation, or an approximated plot by numerical integration techniques)], or

B) Some specially devised super-osculant, which requires no integration at all, but will require differentiation of the intrinsic equation of the base curve up to the nth. order to produce an osculant with an (n-1)th. order-of-contact. This is often so much like the base curve, between the selected plotting limits, that, to all intents and purposes, it may well be looked upon as the base curve's image, morphologically speaking, and thereby one has obtained a geometric image of a curve, that otherwise may be almost impossible to plot or at least most difficult to plot accurately.

Now to demonstrate the techniques for A) and B) above.

A) There is a recursive relationship that explicitly defines all such Chief-Osculants.

RECURSIVE RELATIONSHIP

With respect to those Intrinsic Qualities of the Chief-Osculants, numbered above with the numerals only, in brackets:-

a) When 'n' is odd:-

13.1 $$Q_n = \frac{dQ_{n-1}}{ds} = Q'_{n-1}$$

b) When 'n' is even:-

13.2 $$Q_n = \frac{1 + (Q'_{n-2})^2}{Q''_{n-2}} = \frac{1 + Q_{n-1}^2}{Q'_{n-1}}$$

Where $Q' = \frac{dQ}{ds}$ & $Q'' = \frac{d^2Q}{ds^2}$

where:- $Q_1 = \frac{dy}{dx} = \tan\phi,\ Q_2 = \rho,\ Q_3 = \rho',\ Q_4 = \frac{1 + (\rho')^2}{\rho''}$

N.B. Dimensional analysis of the odd-orders shows pure number, while that of the even-orders shows length, L.

Avoidance of ambiguity is the `Essence of the Contract'. Hence, we now definitely and verifiably have an infallible way to investigate any order-of-contact.

The establishment of these Chief-Osculants in differential form, (as illustrated in the Table at end of this paper), does not necessarily mean that we can EASILY identify either the Curves or the Geometric Images, so portrayed, since the Calculus involved for some of these probably involves indefinite integrals, which cannot be represented in terms of known functions, but can be drawn by evaluating by means of numerical approximation, or on a computer monitor, with a `graph-plotter' program, such as Derive or DeriveXM, which integrates as it plots each point.

B) The Super-Osculants are derived from one or more of the six Intrinsic Equations to a particular base curve. Even though there are six possible pairs of parametric equations for such a Super-Osculant, the ultimate choice, that produces the greatest semblance to a major section of the base curve, is most likely to come from either of the Whewell Equations, since, of the three intrinsic variables, [the radius of curvature (r rho), the arclength (s) and the tangential angle (f phi)]. r being in an equation, (as it is in the two Cesaro equations and two Euler equations), is better avoided where possible, especially if it is the independent variable, since, either

1) as a parameter in a pair of equations, it cannot be expected to range between two nominated extremes as the other two variables do so readily, or
2) it often occurs in the denominator of the terms of the Taylor series and it necessarily takes a zero value at cusps and poles.

This is an overview of what Cesaro did to obtain the equations of his osculants, and then my own extensions of his ideas, which, perhaps, he did not consider, opining, perhaps, that what he did was all there was to do.

He assumed that there was no ambiguity, let alone more than two possible ways of defining order-of-contact. This assumption of his was implicit, since he never stated anything on the matter, other that attest that the order of contact was defined by the evolutes of increasing order. In other words, he states that with a second order osculant, the osculation was representative of a commonality between base curve and osculant, that they shared the same first order contact, namely gradient.

Now will follow a couple of pages, some of them full with large mathematical expressions, an interesting table and two graphs which should illustrate the power of the Super Osculants. I hesitated to publish this paper because it looked too "heavy" and I wrote back to David asking for some additional explanations and more illustrations. No answer. But in this January together with his membership renewal arrived a letter from Australia with David´s complain, why his paper has not been published until now despite the announcements on all pages #2 in many DNLs. I answered and asked once more for some additional comments. Now I received the answer very soon and I find it worthwhile to be published below. It will support you understanding my requests and his ideas as well.

Dear Joseph, North Balwyn, 10 February 1998

I received your letter today; many thanks for your prompt reply. Unfortunately I have not received your earlier letter, requesting more graphics, so I am rectifying that immediately. I was foolish to omit them since the main point of the Glove-Osculant algorithm is to demonstrate how good it is for approximating major arcs, (at least), of many common curves, as well as many curves, which have no simple closed-form algebraic expression. I have included 11 such images on the enclosed disc as well as an up-dated version of the file that achieves that end, OSCUWHE1.MTH as well as the OSCUWHE1.PRT file for your easy reading. I have inserted relevant comments about the images in those files, and I shall print them out, together with all 11 images in the accompanying letter DUGIMAGE.WPF.

Because you told me that you found nothing on Cesaro to date, I include some eulogy-type articles in Italian, written by people, who knew and respected the man and his researches into many disparate, (and sometimes untouched), areas of mathematics. Unfortunately I do not speak Italian, therefore I could only glean a smidgen of their contents. Hopefully you will gain more than me from the papers.

Your query re so many equations on pages 6-8 is very pertinent, especially if you have to retype them. I hope you have access to the WordPerfect Word Processor V 5.1 or later, since all those equations are created by the equation editor. In summary, I put in all the details so that the incredulous reader(s) won't have to take my word for it, perhaps thinking that I am using a vivid imagination, rather than an in-depth analysis of Cesaro's work with further developments by myself. Whereas he only considered this method for those equations, which I labelled Cesaro Type-1 $\rho = C(s)$ there are 5 other forms of Intrinsic Equations, and each one had to be tackled and solved by me taking many tens of hours, by hand, before I eventually was able to develop an algorithmic routine using declared functions in Derive and thereby enabling not only a valuable time-saving approach, but also no apparent limit to which high order-of-contact the glove-osculants make with the base curve. I sometimes went to the 15th order, which implied 15 successive differentiations of these declared functions in a canonical form, so that when the result was obtained, it could be used for any nominated base curve by slotting it into the general form by using the Derive Simplify Command. That is why the .MTH file has thousands of terms in the vector [X(s),Y(s)].

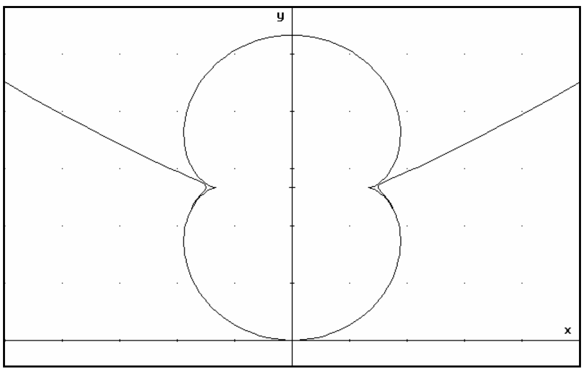
I shall also include on the disc the two .WPF files, being this letter DUG11DLH.WPF and the accompanying letter DUGIMAGE.WPF

I take the liberty of quoting from my mail of 1996, wherein I request some readers to contact me, hopefully:-

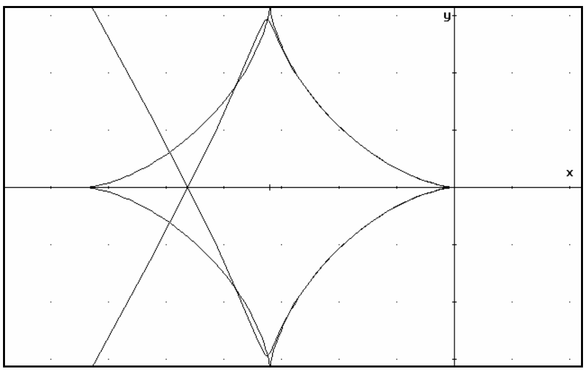
I have opened a mathematical 'can of worms', too much for one person to develop to its full potential, so I invite fellow members of DUG or readers of the Newsletter to write to me, in English, and perhaps we can form a team for further development of this line of research.

Yours sincerely,
David Halprin.

To give an impression of David´s contribution I add two screen shots of super osculants:



The *NEPHROID* and its osculant



The *ASTROID* and its osculant

(There were a lot of reactions on Johann Wiesenbauer´s column TITBITS. So we - Johann and I - decided to collect the papers in this DNL instead of having a TITBIT 13. We both assure that we are glad about the echo caused by Johann´s column and I hope that it will be a source not only of admiration for Johann´s programming skills but also for own investigations. Many thanks to all the contributors. J & J)

Lucas, Fibonacci, Pell & Company

Stefan Welke, Bonn, Germany, Spwelke@aol.com

Abstract

DERIVE´s built in matrixoperations allow a fast and unified treatment of Lucas-, Fibonacci-, Pell- and other linear recursive sequences.

1 Introduction

When I curiously examined the NUMBER.MTH - file delivered with my DfW version 4.0 I found the function PELL(n). This function defines the recursively defined sequence of Pell´s numbers: $a_1 := 0$, $a_2 := 1$ and $a_n := 2a_{n-1} + a_{n-2}$ for $n > 2$. The function´s performance is very slow. Johann Wiesenbauer suggests in TITBITS 12, DNL#28, a better and faster version generated with a function GEN_LUCAS(...). This implementation is very fast and the same function GEN_LUCAS(...) generates with different parameters the Fibonacci and other recursive sequences.

But there is still another unfied method of computing recursively defined sequences $\{a_n\}$ of the form:

$$a_1 := a,\ a_2 := b,\ a_n := \alpha\, a_{n-2} + \beta\, a_{n-1} \text{ for } n > 2. \qquad (1)$$

Consider the following matrix-vector equation, which recursively defines a sequence of vectors:

$$\begin{pmatrix} 0 & 1 \\ \alpha & \beta \end{pmatrix} \begin{pmatrix} a_{n-2} \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} a_{n-1} \\ \alpha\, a_{n-2} + \beta\, a_{n-1} \end{pmatrix}$$

It follows by induction:

$$\begin{pmatrix} 0 & 1 \\ \alpha & \beta \end{pmatrix}^{n-1} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a_n \\ a_{n+1} \end{pmatrix} \qquad (2)$$

The matrix powers applied to the starting vector (a,b) generate a sequence of pairs (a_n, a_{n+1}) of consecutive elements of the sequence $\{a_n\}$. Observe that the last equation exhibits a formal relationship between recursively defined sequences (1) and geometric sequences, because (2) reflects formally the structure of geometric sequences. The following table shows some examples:

Sequence	a	b	α	β
Fibonacci	0	1	1	1
Pell	1	2	1	2
Lucas	1	3	1	1

The characteristic polynomial of the generating matrix is

$$\det\begin{pmatrix} -\lambda & 1 \\ \alpha & \beta-\lambda \end{pmatrix} = \lambda^2 - \beta\lambda - \alpha$$

and $\lambda^2 - \beta\lambda - \alpha = 0$ is the characteristic equation of (1).

2 Implementation

A single line in *DERIVE* based on (2) defines a function R(...), which is equivalent to J.Wiesenbauers´s function GEN_LUCAS(...):

FPL.MTH

```
#1:  R(gv_,n_,sv_):=ELEMENT([[0,1],gv_]^(n_-1)*sv_,1)
#2:  FIBONACCI(n_):=R([1,1],n_,[0,1])
#3:  VECTOR(FIBONACCI(k),k,1,10)=[0,1,1,2,3,5,8,13,21,34]
#4:  PELL(n_):=R([1,2],n_,[0,1])
#5:  VECTOR(PELL(k),k,1,10)=[0,1,2,5,12,29,70,169,408,985]
#6:  LUCAS(n_):=R([1,1],n_,[1,3])
#7:  VECTOR(LUCAS(k),k,1,10)=[1,3,4,7,11,18,29,47,76,123]
```

#1 looks considerably simpler as J.Wiesenbauer´s definition of GEN_LUCAS(...) in [1]. The price for this simplicity is speed. The following table compares computational times for the functions in NUMBER.MTH and the functions #2, #4 and #6 based on the matrix method:

Function	NUMBER.MTH	Matrix method	*M.m with DfD 4*
PELL(100000)	149.5 sec	12.2 sec	*6.9 sec*
LUCAS(100000)	1.6 sec	3.6 sec	*1.9 sec*
FIBONACCI(100000)	2.2 sec	3.6 sec	*1.9 sec*

The computation times refer to a DfW 4.0 and a PENTIUM 133 with 32MB RAM. *(I added the DfD times, J.)*

FIBONACCI(...) is defined with two calls of LUCAS(...) in NUMBER.MTH, and LUCAS(...) is basically the same function as GEN_LUCAS(...) in [1]. This explains the rise from 1.6 to 2.2 seconds in the second column. There is no difference in the matrix version, because only the start vector is different, but the matrix is the same. The NUMBER.MTH version of PELL(...) uses recursion, and this fact explains the 149.5 seconds in contrast to the 12.2 seconds of the matrix method.

3 Conclusions

1. The simple and transparent matrix method is roughly half as fast as Wiesenbauer´s method, but considerably faster than the recursion. The matrix method works even in the case where the discriminant of the characteristical polynomial is zero.
2. The matrix method allows generalizations to recursive sequences of higher order. For example: Let $\{a_n\}$ be defined by $a_1 = x_1$, ... $a_4 = x_4$ and $a_n = \alpha\, a_{n-4} + \beta\, a_{n-3} + \gamma\, a_{n-2} + \delta\, a_{n-1}$ then

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \alpha & \beta & \gamma & \delta \end{pmatrix}^{n-1} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} a_n \\ a_{n+1} \\ a_{n+2} \\ a_{n+3} \end{pmatrix} \quad \text{for } n \in \mathrm{N}.$$

   It is evident how to generalize further.
3. It is possible to construct formulas for direct computation of the recursive sequences under consideration, which generalize the Binet-formula for the Fibonacci sequence. The generalized Binet-formulas are appropriate linear combinations of n-th powers of eigenvalues of the corresponding matrices [2]

References

[1] Wiesenbauer, J.: Titbits 12, DNL#28

[2] Welke, S.: Eine Verallgemeinerung der Binet-Formel für linear rekursive Folgen, to appear in **MNU**

I informed Johann about this paper and he asked me to fax it immediately to his institute (Vienna Technical University). He e-mailed an answer containing some comments and sent a Cc to Stefan Welke. Stefan expressed his thanks, considered his advice and made some changes as:

The table should read:

The following table shows some examples:

Sequence	*a*	*b*	α	β
Fibonacci	1	1	1	1
Pell	0	1	1	2
Lucas	1	3	1	1

That is the improved FPL.MTH

```
#1:  R(gv,n,sv)=([[0,1],gv]^(n-1)*sv) SUB 1
#2:  FIBONACCI(n):=R([1,1],n,[1,1])
#3   VECTOR(FIBONACCI(k),k,1,10)=[1,1,2,3,5,8,13,21,34,55]
#4:  PELL(n):=R([1,2],n,[0,1])
#5:  VECTOR(PELL(k),k,1,10)=[0,1,2,5,12,29,70,169,408,985]
#6:  LUCAS(n):=R([1,1],n,[1,3])
#7:  VECTOR(LUCAS(k),k,1,10)=[1,3,4,7,11,18,29,47,76,123]
```

Stefan writes that he - in contrary to Johann - is finding the use of both auxiliary functions and long variable names as an advantage. He prefers the shifting of the index, because he intends to present a general solution. Johanns suggestion uses an intial value of index 0, which has to be known in each case.

Find below Johann´s concluding message:

Dear Josef,

Thank you very much for sending me the interesting paper of Stefan Welke. Though due to my notorious idiosyncrasy to auxiliary functions I would have preferred the definitions

```
FIBONACCI(n) := ([[0, 1], [1, 1]]^n·[0, 1])SUB1
LUCAS(n) := ([[0, 1], [1, 1]]^n·[2, 1])SUB1
PELL(n) := ([[0, 1], [1, 2]]^n·[1, 0])SUB1
```

I don't object to his original routines either which are short and sweet as well and, above all, incredibly fast! As a matter of fact, they are more than ten times faster than former DERIVE-implementations based on the original recursion and only about two times slower than the current versions in DfW 4.08. Moreover Stefan Welke is right in saying that you don't have that restriction about the nonzero discriminant (even if this condition is usually fulfilled!)

This raises the question why I didn't take that method into consideration. The answer is quite simple: In many important applications you need the values of those series mod N for huge indices (say with 100 digits; see the example in DNL #26, p27, to see what I mean!) and here the matrix-method simply can't keep up. (You have to develop a modulo-arithmetic for matrices to avoid a memory overflow and all the performance goes to blazes!)

Anyway, concluding I would like to say that whenever didactic aspects are more important than absolute top performance Stefan Welke's routines are certainly an alternative one should consider. In particular, the people who wrote the chapter about recursive functions in the DERIVE-manual should study his paper very carefully.

Cheers, Johann

Another letter arrived - "Bits and Pieces in Number Theory" - from ***Andreas Eder****, München, Germany:*

Dear Josef,
first let me express my thanks for your good work. I just received DNL #28 and had a short look over it. It seems to be full of interesting articles once again!
Now to my little contribution: I especially like the 'Titbits from Algebra and Number Theory' by Johann Wiesenbauer, partly because I myself am very interested in Number Theory, and because I constantly learn new programming tricks by trying to understand his programs. So I discovered what he calls the ITERATE(, , , 1) trick. A very clever thing, but to be honest, I don't think it makes the programs very readable. So I thought about it and tried to encapsulate that trick in a bit of 'syntactic sugar'. This is what I have come up with:

```
LET(v,e,b):=ITERATE(b,v,e,1)
```

which should be read as: let the vars be bound to expressions in body (like the let construct in Lisp or Scheme). I think it makes programs more succinct and readable, e.g.

```
LIN_DIOPHANT(a,b,c):=LET([d_,u_],GCD_EX(a,b),IF(MOD(c,d_)=0,u_*c/d_+[b*@,-a*@]))
```

This version of LIN_DIOPHANT also introduces my version of EXTENDED_GCD, which I called GCD_EX.

```
GCD_EX(a,b):=LET([x_,g_,y_,m_],ITERATE(IF(MOD(u3_, v3_)=0,[u3_,v3_,u1_,v1_],~
[v3_, MOD(u3_,v3_),v1_, u1_-FLOOR(u3_, v3_)*v1_]),[u3_,v3_,u1_,v1_],~
[a,b,1,0]),[g_,[m_, (g_-m_*a)/b]])
```

You can see threee differences with respect to EXTENDED_GCD: first it uses LET, second it uses named variables instead of a vector and subscripts and third and most important - it does away with the components 5 and 6 of the vector in EXTENDED_GCD. These are not really needed, because if g = m*a+p*b, knowledge of g and m is sufficient to calculate p. This makes the program both shorter and more readable, and it should be slightly faster as well.

In all these, and some more cases my LET construct worked as expected. I hope it will be of some use to others, as well as my streamlined version of EXTENDED_GCD.
Yours, Andreas Eder Andreas.Eder@t-online.de

From ***Jan Vermeylen****, Kapellen, Belgium:*

Today I received DNL#28. Couldn´t stop reading it! Very interesting articles indeed. (as always, but still ....). In the 4.07 version of the NUMBER.MTH utility file you can find a new function SORT(a) which sorts the elements of a vector with different rational numbers. The function that I incorporated in my STAT.MTH file can also sort vectors with non-different irrational numbers. Check the attached file: I made two versions SORTX and SORTY (with and without frequency) - Best greetings, Jan.

(You will find both functions SORTX and SORTY in the file RESORT.MTH, Josef)

Do spread the news registered DfW users can have free upgrades from the SWH site now!

The TITBITS master´s Johann Wiesenbauer´s answer:

Sorry for the delayed answer caused by my holidays. I assure you that I'm as pleased as Punch about every feedback I get from readers of my 'Titbits'. As an avid programmer I can vividly imagine that you were very enthusiastic about your beautiful SORT-routine when it finally yielded the correct results. Therefore I hate to inform you that it has a serious flaw: Its asymptotic growth is like O(n^2), where n is the length of the list in question, which is far too slow for practical purposes. (As a matter of fact, for lists with more than 20 elements your SORT-routine becomes even slower than that of Sergey Biryukov speeded up by me to no avail in DNL #23, p39.) In other words, if you still feel like sorting with DERIVE, you are strongly advised to change to a better sorting algorithm like e.g. quicksort (here the corresponding magnitude is O(n log n) on average, i.e. its growth is almost linear!)

In the meantime, I can offer you a general SORT-routine of my own that should work in most cases. (It is longer than necessary due to two tiny DERIVE-bugs which I have described in the attached file. I am sending a carbon copy of this e-mail to Al Rich hoping that he has mercy on us.)

Please, keep me informed about your future programming efforts related to the topic above! - Cheers, Johann

RESORT.MTH

```
SORTX(v):=(ITERATE(IF(DIMENSION(x SUB 1)=0,x,[SELECT(k/=MIN(x SUB 1),k,
          x SUB 1),APPEND(x SUB 2,[MIN(x SUB 1)])]),x,[v,[]])) SUB 2

SORTY(v):=APPEND(VECTOR(VECTOR(x,i,1,DIMENSION(SELECT(k=x,k,v))),x,
          SORTX(v)))

"(Your sort-routines)"

"Comparing some DERIVE-routines for sorting of numbers"

"((c) Johann Wiesenbauer, Vienna)"

SORT0(v):=LIM(TERMS(v*VECTOR(x_^(-k_),k_,v)),x_,1)
```

(SORT0(v) is very fast, but works only if v is a list consisting of different(!) nonzero rational numbers.)

```
SORT(v):=ITERATE(IF(v_ SUB DIMENSION(v_)=0,APPEND(SELECT(u_<0,u_,v_),
         SELECT(u_=0,u_,v_),SELECT(u_>0,u_,v_)),v_),v_,VECTOR(v SUB k_,k_,
         LIM(DIF(TERMS(EXPAND(VECTOR(x_^APPROX(-v_),v_,v)*VECTOR(y_^k_,
         k_,1,DIMENSION(v)))),y_),[x_,y_],[1,1])),1)
```

You should use sort(v) if the preconditions above are not fulfilled.

```
SSORT_AUX(v,n,p):=VECTOR(IF(MOD(i+p,2)=0,IF(i=1 OR v SUB (i-1)<=v SUB i,
                  v SUB i,v SUB (i-1)),IF(i=n OR v SUB i<=v SUB (i+1),
                  v SUB i,v SUB (i+1))),i,n)

SSORT(v):=ITERATE(SSORT_AUX(SSORT_AUX(w,DIMENSION(v),0),DIMENSION(v),1),
          w,v)
```

The sort-routine by S.Biryukov speeded up by me in an earlier DNL:

```
[a:=[1,1,2,1,2,3,1,2,3,4,1,2,3,4,5], b:=[pi,SQRT(2),LOG(2),0,LOG(2)]]

SORT(a)=[1,1,1,1,1,2,2,2,2,3,3,3,4,4,5]

SSORT(a)=[1,1,1,1,1,2,2,2,2,3,3,3,4,4,5]

SORTX(a)=[1,2,3,4,5]

SORTY(a)=[1,1,1,1,1,2,2,2,2,3,3,3,4,4,5]

SORT(b)=[0,LN(2),LN(2),SQRT(2),pi]

SSORT(b)=[0,LN(2),LN(2),SQRT(2),pi]

SORTX(b)=[0,LN(2),SQRT(2),pi]

SORTY(b)=[LN(2),SQRT(2),pi]

SORTY(b)=[0,LN(2),LN(2),SQRT(2),pi]
```

There has been no difference so far, right?
But look at the following important example from number theory:

```
FAREY0(n):=(ITERATE(IF(v_ SUB 1=v_ SUB 2 AND v_ SUB 2=n,v_,[IF(v_ SUB 1=
          v_ SUB 2,1,v_ SUB 1+1),IF(v_ SUB 1=v_ SUB 2,v_ SUB 2+1,v_ SUB 2),
          IF(GCD(v_ SUB 1,v_ SUB 2)=1,APPEND(v_ SUB 3,[v_ SUB 1/v_ SUB 2]),
           v_ SUB 3)]),v_,[1,1,[]])) SUB 3
```

You should simplify each of the following lines (including the next one!!) and compare!

```
[c:==FAREY0(20),d:==FAREY0(40)]

[DIMENSION(c),DIMENSION(d)]

SORT0(c)

SORT(c)

SSORT(c)
```

```
SORTX(c)
SORTY(c)
SORT0(d)
SORT(d)
SSORT(d)
SORTX(d)
SORTY(d)
```

My times (in s on a Pentium 200 PC): 0.1,0.4,17.3,14.9,17.3 and 0.6,1.6

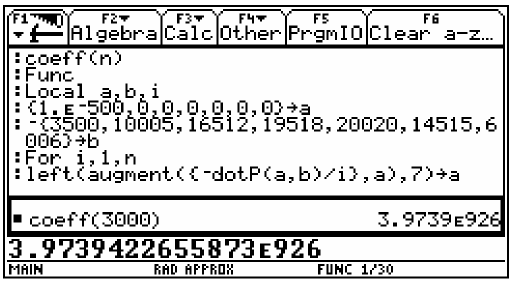
You might remember "Problem 4 of the CAS Competition (DNL#27, p25). Johann tried an approach to that challenge using the TI-92:

A *TI*-92 Approach to Problem 4

The following *TI*-92 function returns the n-th coefficient of the polynomial in question scaled down by the factor E-500. Note that the calculator must be in approximate mode!

```
coeff(n)
Func
Local a,b,i
{1.E-500,0,0,0,0,0,0}→ a
-{3500,10005,16512,19518,20020,14515,6006}→b
For i,1,n
     left(augment({dotP(a,b)/i},a),7)→a
     b+{3,5,6,6,5,3,1}→b
EndFor
Return a[1]
EndFunc
```

Computing coeff(3000) yields 3.9739422655873E926 after about 5 min, i.e. 11 significant digits of the number in question! (Note that 500 must be added to the exponential part.)



I tried the "official" solution presented on TI´s home page prob4(), consumed one set of batteries the first night running the program and started once more with a fresh set, but I interrupted after 13 hours. The I/O screen showed some progress: 12, 24, ..., 300, 600.

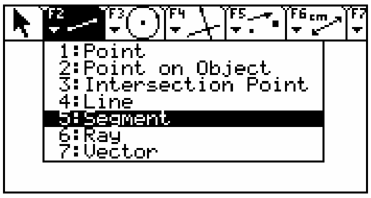
In the directory I found three lists for intermediate results containing approximately 16000, 45000 and 47000 bytes. Josef

More Exciting Trig with the TI-92

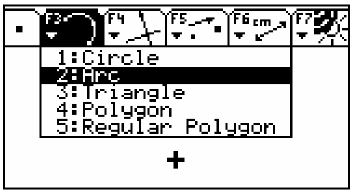
presented by Barbara Leitherer,
4th International Symposium on DERIVE and the TI-92, Saeroe, Sweden

I. Radian Measure - a FUN activity and concept

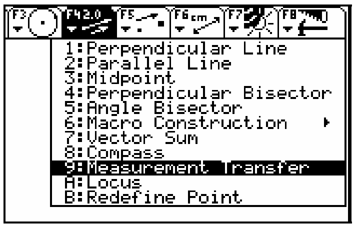
In order to the "radian measure concept", the following toolbar menus need to be accessed from the geometry viewing window:



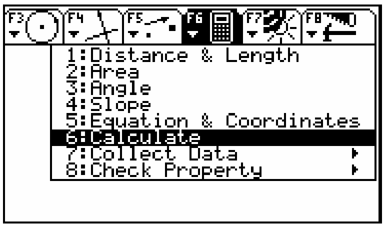
The Curves and Polygons Toolbar Menu to create circles and arcs: …



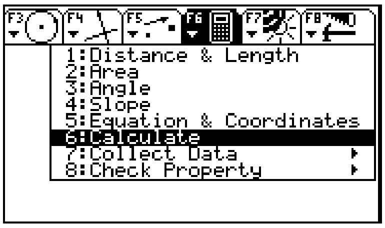
The Points and Lines Toolbar Menu to construct a radius (segment): „



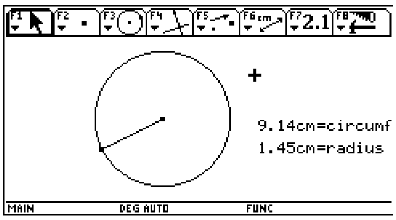
The Construction Toolbar Menu to transfer measurements: †

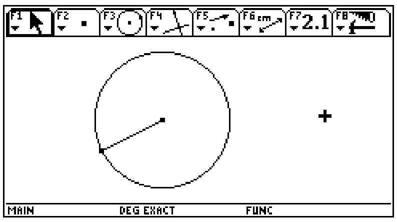


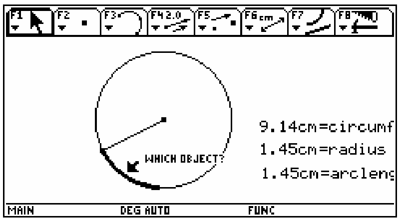
The Measurement Toolbar Menu to measure the length of an arc, a radius, a circumference, an angle and to do additional calculations: ˆ



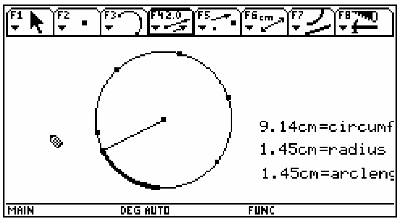
The Display Toolbar Menu to make an arc graphically stand out from the rest of the circle: ‰.

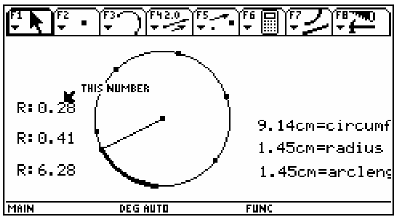


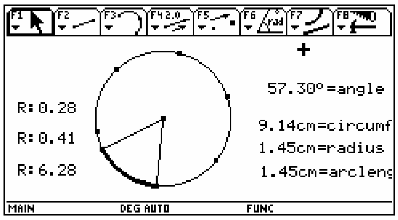




1. Open the geometry window. On one side of the viewing window construct a circle. Then construct the radius of the circle (a segment) from any point on the circle to the center point of the circle.
2. Measure the length of the radius and the circumference of the circle and label each measurement properly: right after each numerical value appears, press ¥ and the right cursor key B and type "= radius" and "=circumference". Then drag each label away from the circle area.
3. Now access the construction toolbar menu and transfer the length of the radius to the circumference which creates a new point on the circle. Pay ATTENTION to the **order in which this task has to be performed**. Select first the endpoint of the radius located on the circle, second the circle, third the measure of the radius.
4. Now overlay a minor arc from the new point constructed in step 3 and hook it up to the endpoint of the radius which is located on the circle. Display the smaller arc in thick format, label it "=arclength" and drag text and numerical value away from the circle area.







5. Repeat the procedure from step 3 and prick off a second point which is a radiuslength away from the last point constructed. Continue to overlay the circumference with more radiuses.
   How may "radiuses" will approximately fit around the total circle?
6. Use the measurement toolbar menu and divide the circumference by the length of the radius.
   What number do you get?
   What does your answer mean?
7. Now calculate the length of the remaining "little" arc. How many radians of the total circumference does it represent?
8. Now draw a radius to the point which was constructed in step 3. How big is the central angle formed by the two "radii" of the circle? Give your answer in degrees, label it "=angle" and drag it to the right above the other measurements.
9. Let´s finish this project with a little exercise:

A) Before you move on, fill the provided table with the measurements from the left and right of the circle.

	radius	circum-ference	arclength	angle	circumfer/ radius	length of the little arc	radians of the little arc
original circle							
bigger circle							
smaller circle							

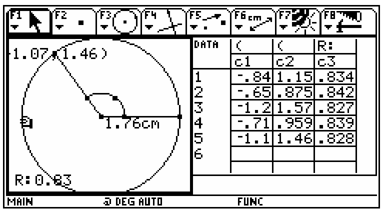
B) Drag the circle dynamically and make its radius once bigger, once smaller. Complete the missing information and answer the following question based upon your answers from the table:

Which of the performed measurements DO, which DO NOT change in this exercise?

II. Transferability to other topics

1. Defining Trigonometric Functions of any angle

eg sin θ
- ➔ create a circle and measure its radius or use the measurement toolbar menu and select "equations and coordinates"
- ➔ create a point and its coordinates on the circle
- ➔ create and measure an angle θ
- ➔ find sin θ
- ➔ find ratio y/r (either through measurement toolbar or Data/Matrix Editor)
- ➔ change radius
  again observe sin θ and y/r



2. Use for sign chart of different trigonometric functions in all four quadrants
3. OR .... for Properties of sine and cosine: cos(-*t*) = cos (*t*)
   sin(-*t*) = - sin(*t*)
4. SPACE for discussion

II. References:

1. Classroom materials I used in my Precalculus courses taught at Carroll Community College
2. TI-92 Guidebook, by Texas Instruments, 1995
3. Precalculus, Functions and Graphs, by Demana, Waits, Clemens and Foley, 3rd edition,1996.

WHAT ARE THESE PROBLEMS ABOUT?

Alfonso J. Población Sáez
E.U. Politécnica, Valladolid, España

One of the main aspects I like to tackle in this section is to propose problems for which we can use *DERIVE* to solve (although I have not to be strictly necessary to do it) in order to reach different goals: to enjoy ourselves thinking and solving them, and to find the power of *DERIVE* programming and doing calculations, and why not, pointing out new ways to improve the system where it could appear more limited. So, I hope most of you (students included) may find useful or at least entertaining the problems proposed.
This time I chose several questions with a common denominator which you also have to discover. Solutions and *DERIVE* files will appear in next newsletters.

① What is the smallest four-digit number that belongs to the residue class $\overline{13}$ Z_{17} and to the class $\overline{5}$ in Z_{12}. And the biggest?

② In a livestock fair, a farmer bought a hundred animals which costed him 200 000 pts (the peseta is the Spanish monetary unit). Hens were valuated in 100 pts each, every pig in 8 000 pts, and cows were sold at 40 000 pts each. How many animals of every sort did he buy?

③ A mason had to floor three sqared factories with a certain number of tiles. In the first factory, he put the half of the total amount of tiles. For the second one he arranged a 50% more of rows than the third one. When finished, he found that he had one tile left. What was the total amount of tiles knowing that the factories were the smallest ones that verify all conditions referred?

④ A certain three-digit number yields a quotient of 26 when divided by the sum of its digits. If the digits are reversed, the quotient is 48. What is the smallest three-digit number for which this is possible? How many are three?

⑤ Could you prove using *DERIVE* that there exists a unique magic square (excluding symmetries) of order three that uses the first nine numbers? (A magic square is a square array of numbers in which the sum of each row, column, and main diagonals is identical).

Fitting to that ACDC Contribution I can announce for the next DNL a TI-92 Program for Solving Diophantic Equations. It is a nice coincidence that it was also contributed by two Spanish DUG members, Leandro Tortosa and Javier Santa Cruz. See also Andreas Eder´s contribution an page 33 referring to J.Wiesenbauer´s TITBITS 11 drom DNL#27.

[1] **AC DC** is **A**mazing (or **A**musing **C**orner of the ***DERIVER***´s **C**uriosities) is founded by Alfonso Poblácion.

150 Years of Pi´s 250 Decimals

Tomass Romanovskis
University of Latvia, Riga, Latvia

To demonstrate the power of computers and of *DERIVE* we can use the computation of π up to thousands of decimals up to 500000. The history of competitive efforts to find as many digits of π as possible began in the middle of the last century. In 1841 W.Rutherford published the number π with 208 digits in "Philosophical Transactions", 1841, Part 11, pag. 238. At the same time the Viennese mathematician S.Straschnizky asked the 16 years old prodigy Zacharias Dase to compute π using the formula:

$$\frac{\pi}{4} = \tan^{-1}\frac{1}{2} + \tan^{-1}\frac{1}{5} + \tan^{-1}\frac{1}{8}$$

To illustrate his capabilities I´d like to mention a well known example of a multiplication. To find the product of 79532853 * 93758479 took him 54 seconds. Within two months Dase found the first 205 decimals of π and he sent his result to H.K. Schuhmacher, editor of the "Astronomischen Nachrichten". He compared Dase´s results with Rutherford´s publication. The first 152 digits were equal, but from the 153th digit they did not even have a single digit in common. Rutherford had used another formula:

$$\frac{\pi}{4} = 4\tan^{-1}\frac{1}{5} - \tan^{-1}\frac{1}{70} + \tan^{-1}\frac{1}{99}$$

In that situation Schuhmacher wrote to Dr. Clausen in Dorpat (now Tartu in Estonia) and asked him to let him have his results, because it was known that he, Clausen, had calculated π for 250 places. He had even done that twice: once using the equation:

$$\pi = 8\tan^{-1}\frac{1}{3} + 4\tan^{-1}\frac{1}{7}$$ and then using: $$\pi = 16\tan^{-1}\frac{1}{5} - 4\tan^{-1}\frac{1}{139}\ .$$

The values for $\tan^{-1}(1/3), \tan^{-1}(1/5), \tan^{-1}(1/7), \tan^{-1}(1/239)$ and for π were published in the "Astronomi-schen Nachrichten" in 1847, Nr 589, S 207 - 210. So Schuhmacher found out that the boy prodigy had calculated π accurately to 200 decimals, and Rutherford´s last 56 digits were wrong.

Today we need not publish the values for arcustangens and for π again as everybody can now repeat the process using *DERIVE*. Set Precision Approximate and Digits 250, edit Rutherford´s, Dase´s or Clausen´s formula, press one key and within fractions of a second you will get the 250 decimals places of Clausen.

What now deserves more of our admiration? Rutherford, who made human mistakes, the Dase boy,who made no mistakes, the human "super-calculator" Clausen or superquick perfect *DERIVE*? That is a matter of taste. Deep in my heart my favourites are T.Clausen and *DERIVE*. Using *DERIVE* you can go farther: by simplifying all formulae presented above in Exact Mode you will receive π exact in each case. Just try! If you approXimate asking for 250 decimals the result will appear ten times faster. I assume that π in that case is computed by a different algorithm.

DERIVE users - and not only they - should acquaint themselves with T. Clausen a bit more closely, because he was really a human super-calculator. He was born 1801 into a poor Danish family. At the age of 12 he could neither read nor write. A hobby astronomer, by name of G. Holst, gave him both some basic knowledge and the opportunity to attend school. When he was 20 he computed the planet orbits so well, that K.Schuhmacher and comet researcher V. Olbers took note of his talents. Later he published 150 scientific articles. He even solved problems which were too difficult for the great K. Gauss. But he became famous through his phantastic calculating abilities. He found errors in publications of world-renowned scientists and in logarithm tables prepared by
K. Matissen. T. Clausen showed that the Fermat number $2^{64} + 1$ is not prime but can be factorized in
274177 ∗ 67280421310721. Check it immediately with *DERIVE.*
Riga, August 27, 1997

Tomass was delegate at the FUN-Conference at Saeroe in Summer 1997. We spent some time in talking and learning about Latvia. We all remember his lecture about Kepler´s Clock and his presentation of a folk song of his home country. Tomass sent his artcle in German and I fear that my translation is not as excellent as his German paper. My best wishes to Riga, Josef

The New Supercharged *TI-92 Plus Module* with *Advanced Mathematics Software* for University Level Mathematics and Science

by Bert K. Waits and Franklin Demana
Department of Mathematics, The Ohio State University

Version 12-26-97
If you have suggestions please email them to: waitsb@math.ohio-state.edu

Texas Instruments recently announced the *TI-92 Plus Module*, a powerful upgrade for the TI-92. This new module upgrade plugs into the back of any TI-92. The original TI-92 has built-in a versatile computer symbolic algebra system, a computer interactive dynamic geometry system, five graphing utilities including 3D graphing, matrix algebra, statistics, data, table, programming, and text editor functionality. The new TI-92 module upgrade contains considerably more memory (additional RAM of 128K AND a new *User Data Archive* memory space of 384K), a new Flash ROM that is upgradable, AND new university level mathematics, science, and engineering software. The 3D graphing feature has been improved by including contour plots. Implicit graphing of relations is now possible. Any implicit, contour, or 3D graph (in any style) can be rotated ("spun") creating dynamic and exciting real-time animations and visualizations. Additional major new functionality includes numerical and exact analytic solutions of most 1st- and 2nd-order differential equations, exact solution of systems of linear and non-linear equations, advanced linear algebra, more statistics, a general numerical solver, units, and formatting improvements including removal of the circular definition error. Also assembly language programming is now possible. Some examples of the new "TI-92 Plus Module" functionality are illustrated in this paper.

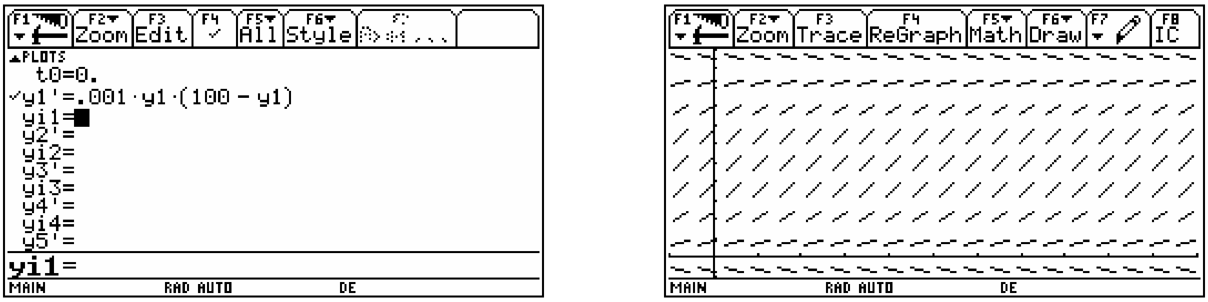
The new *flash* memory technology is a significant new feature. It makes the TI-92 electronically upgradable, adding considerable value to the product for students. With the TI-92 Plus Module, a student can download the latest software versions AND new applications direct from the TI web site (www.ti.com/calc).

Section I contains examples and Section II contains a brief "workshop."

Section I Examples

Differential Equations - Numerical Solutions
Differential equations and initial value problems can be solved using the new module.

Example I-1: A FIRST ORDER DIFFERENTIAL EQUATION
Consider the standard logistic growth model given by the first order differential equation $\frac{dP}{dt} = k * P * (C - P)$ where $P = P(t)$ is a population at time t. Let k = 0.001 and C = 100. First we visualize the differential equation with a slope field (Figure 1). The slope field display occurs as the default view if no initial conditions are specified.



[–10, 100] by [–10, 110]

Figure 1. A slope field for $\frac{dP}{dt} = 0.001P(100 - P)$.

If we specify P = 10 when t = 0, we obtain the solution to the corresponding initial value problem (Figure 2).

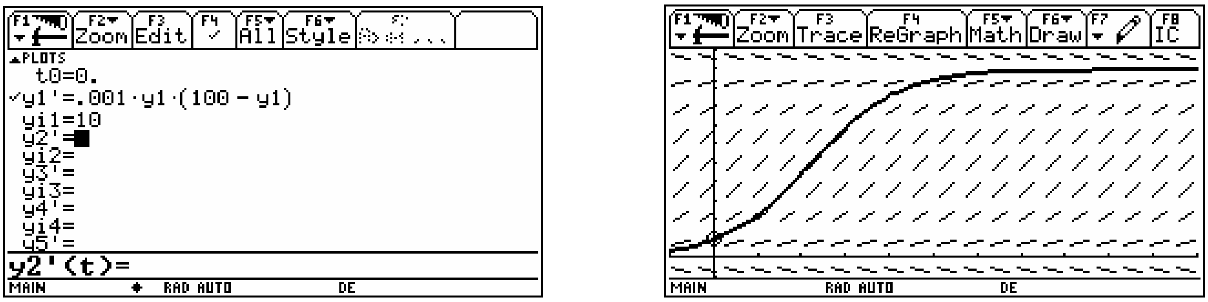


Figure 2. The solution to the initial value problem $\frac{dP}{dt} = 0.001P(100 - P),\ P(0) = 10$.

Initial condition(s) can also be selected interactively using the menu F8 [IC], or by using *lists* of initial conditions in the “Y=” editor. The table feature also provides useful numerical data as shown in Figure 3.

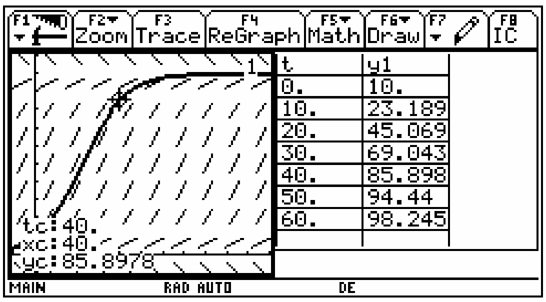


Figure 3. A graph and table of values for the solution to the initial value problem

$$\frac{dP}{dt} = 0.001P(100 - P),\ P(0) = 10.$$

Example I-2: A SECOND ORDER DIFFERENTIAL EQUATION

One standard model for the damped motion of a simple pendulum is

$\theta''(t) = -C\theta'(t) - w^2 \sin\theta(t)$ where C is a constant representing friction and air resistance and $w = \sqrt{\frac{g}{L}}$. Here L is the length of the pendulum rod and $\theta(t)$ is the angular displacement of the pendulum rod from the vertical (equilibrium position).

According to Dreyer this equation *cannot be solve analytically* (p 248, Modeling with ODE's by T.P. Dreyer). However, it is an easy task for the new TI-92 Plus Module! We solve the problem for constants $L = 2$, $C = 0.1$, and $g = 9.80665$. The solution graph (Figure 4) is shown for the initial conditions yi1= -3 and yi2 = 0. Note that we have turned on "labels" from the GRAPH FORMATS screen ([♦]] [F]).

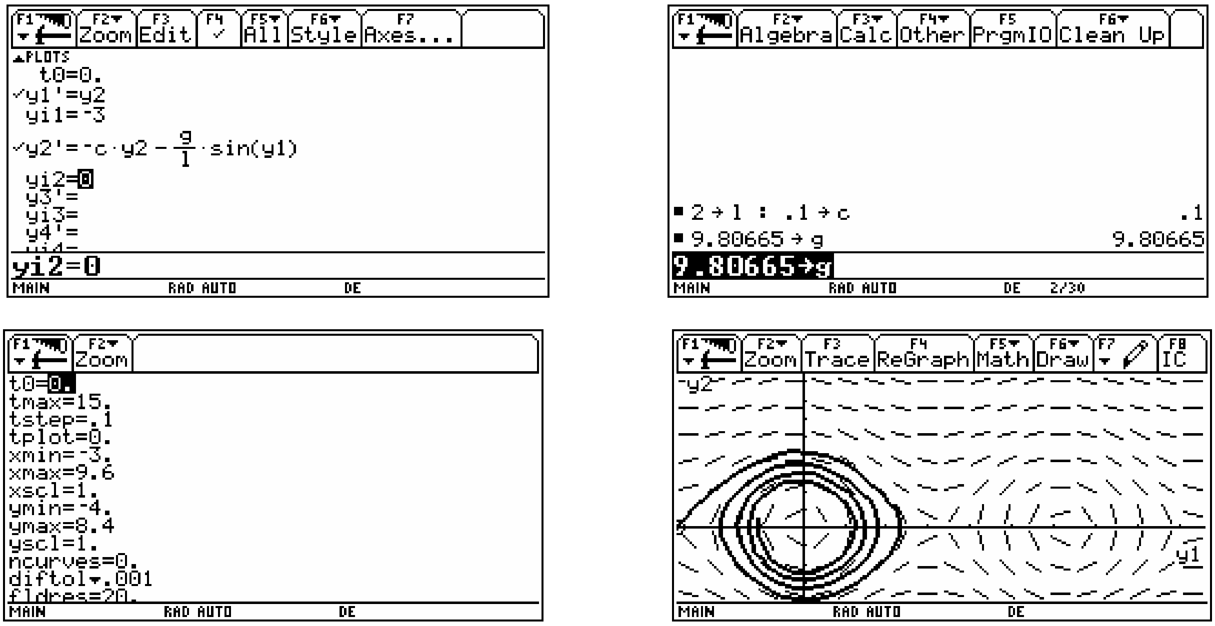


Figure 4. A solution orbit of the damped pendulum problem.

The axes settings are set using the Style menu from the "Y=" Editor. Figure 5 shows several solutions using **the interactive explore** feature (Menu F8 IC from the graphics screen). One plot was set with y1 = 0 and y2 = 5. The second was set with y1= 0 and y2 = 6 (just input these values after pressing the F8 key).

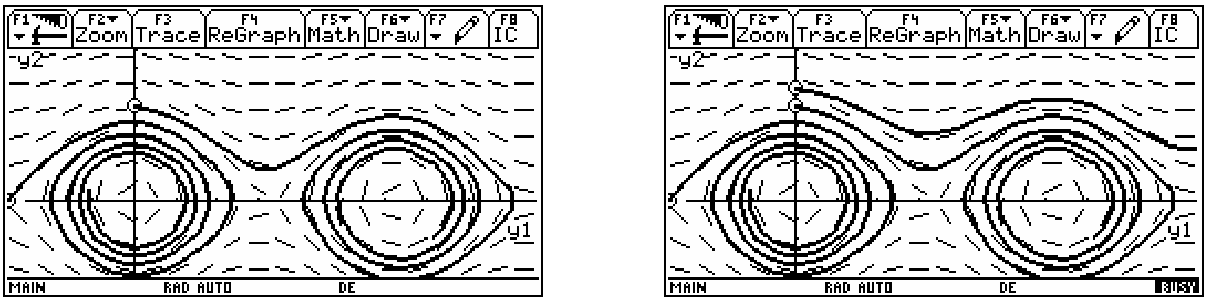


Figure 5. Additional solution orbits of the damped pendulum problem using the new *interactive initial condition* feature.

FOR FURTHER EXPLORATION
For the damped simple pendulum, discuss the singularities (center and saddle) and their stability (stable or unstable).

Differential Equations - Exact Symbolic Solutions

Example I-3: A FIRST ORDER DIFFERENTIAL EQUATION

We can use the new CSA (computer symbolic algebra) command **deSolve(** to find exact solutions to the most commonly studied 1st- and 2nd-order ordinary differential equations. The analytic solution to the logistic differential equation discussed earlier in Example I-1 is shown in Figure 6.

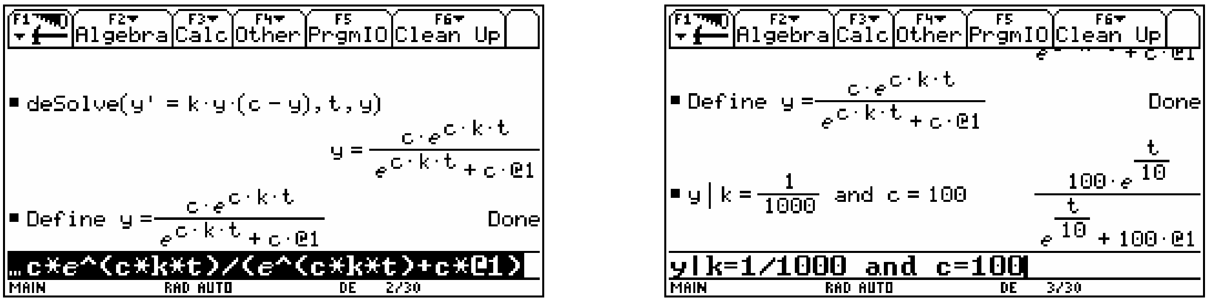
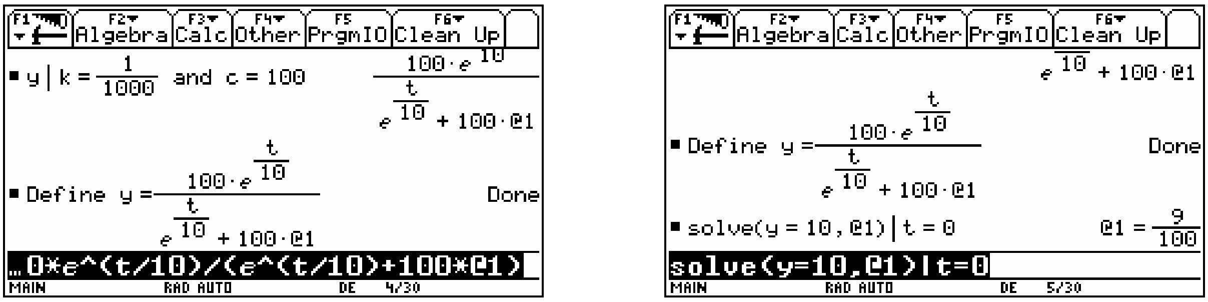


Figure 6. Using a new TI-92 Plus Module command **deSolve** to find an *exact* solution of a differential equation. Notice the general solution is found (@1 is the arbitrary constant) with ease.



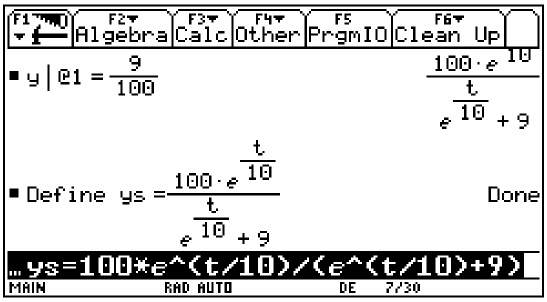
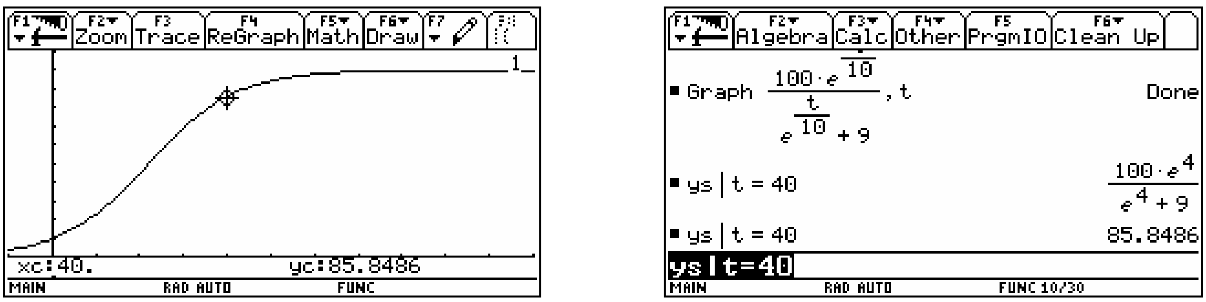


Figure 7. Determining the exact solution satisfying the initial condition P = 10 when t = 0.

In Figure 7 the regular TI-92 **solve(** command is used to find the exact solution satisfying the initial condition P = 10 when t = 0. Figure 8 shows the population graph and the value of the solution (ys, the population) at $t = 40$.



[–10, 110] by [–10, 110]

Figure 8. A graph of the exact solution and the exact solution when t = 40.

Example I-4: A CLASSIC SECOND ORDER SYSTEM OF DIFFERENTIAL EQUATIONS

Predator-Prey problems are usually expressed **as non-linear systems of coupled differential equations** like those shown in the first panel of Figure 9. The solution graphs of the two interacting populations (as functions of time t) P1(T) = y1(t) and P2(t) = y2(t)) are also shown in the first panel of Figure 6. The window is [0, 10] by [0, 25], initial conditions yi1= 2 and yi2 =5 when t = 0, and 0 ≤ t ≤ 10. The "thick" plot is y1 (we did this using [F6: STYLE] from the Y= Editor.) Note: you should turn off the slope field option from GRAPH FORMATS screen.

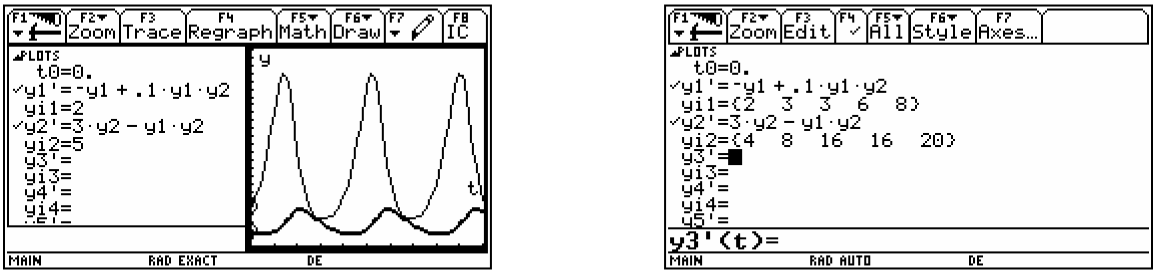


Figure 9. A graph of the exact solution and the exact solution when t = 40.

Figure 9 shows the individual population graphs as function of time t. Now we change the AXES settings to obtain perhaps the more expected phase plane portraits (the orbits). We select "X Axis: y1 and Y Axis: y2" from [F7], the AXES menu in Y=Editor application (choose 2: Custom as shown in the first panel of Figure 10). Figure 10 displays the results of graphing with initial conditions given by the two *lists* in yi1 and yi2 as shown in the second panel of Figure 9.

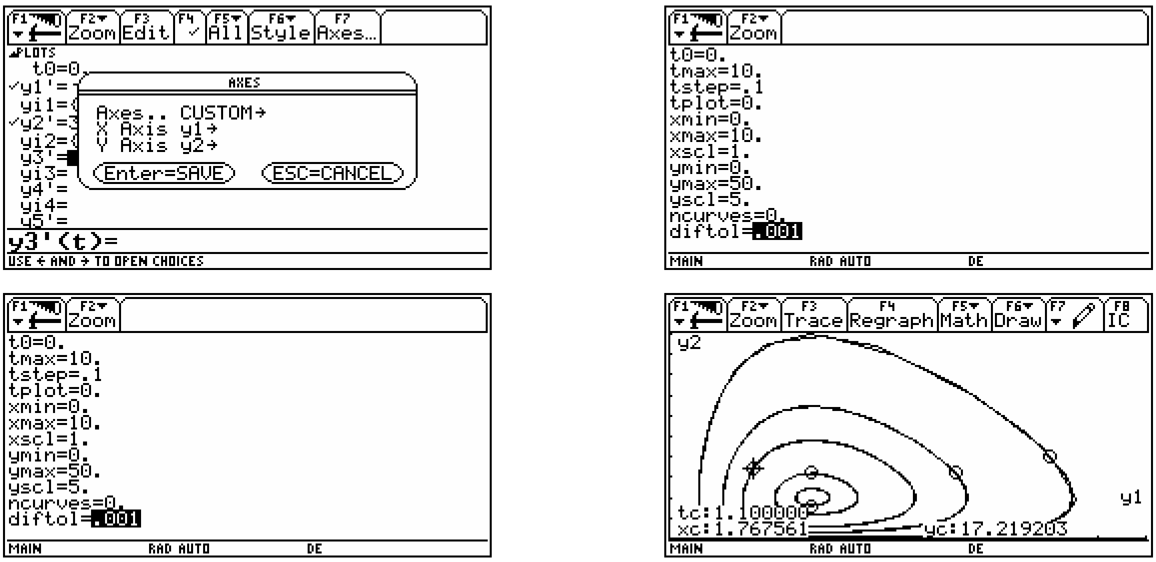


Figure 10. A graph of the exact solution and the exact solution when t = 40.

The second panel in the last row of Figure 10 shows a TRACE numerical result at time = 1.1 for one of the "orbits" (determined by what pair of initial conditions?). We have changes the ymax parameter to 50 in Figure 10.

FOR FURTHER Exploration

Approximate the **center** (*a,b*) of one of the orbits. Change the initial conditions to y1 = *a* and y2 = *b* and the F7 AXES settings back to "Time". <u>Before</u> you replot can you predict what the two popiltion graphs (as functions of time t) will look like? Graph and then *confirm* what you see using analytic paper and pencil methods of calculus.

CONTINUED....

****

This paper is 22 pages in length. You can download it (now in 2012!!) from

http://www.math.osu.edu/~waits.1/papers/ti92spam.pdf

I am sure that this information about the SPAM is interesting for DERIVE-Users who haven´t worked with a hand held tool until now. It is fascinating how many CAS-features can be implemented in such a machine. For me - and for many of us - the TI-92 and DERIVE are not separated of each other. In most cases it is possible to convert DERIVE activities into TI-92 activities. And I often successfully try to transfer TI-92 ideas onto the PC using the advantages of working with more memory and a larger screen. Josef

Additional comment (2012):

You can download a couple of papers from the website given above. They are still of importance because TI-NspireCAS is heir and successor of TI-92 SPAM, TI-92 PLUS and Voyage 200. Bert Waits and Frank Demana were important propagators of CAS in the US – and world wide. I hope that Bert can read these lines. I met him at several occasions and am keeping wonderful memories in my mind, Josef.

Keith Eames, UK **keitheames@email.msn.com**

I have a slight problem with *DERIVE* for Windows, what must the settings be in order for circles to look like circles when plotted, similarly for coordinate geometry so that squares look like squares when printed. If this is in the Derive handbook then I apoligise but I left it at work!

DNL: *Answer from Soft Warehouse:*

Automatic setting of the aspect ratio so that circles always look like circles is not part of *DERIVE* for Windows version 4.0x. We do plan address this issue in the next major release of *DERIVE* for Windows, version 5.0. In the meantime, you'll probably have the most success setting up horizontal and vertical grid values that work best for the size of the 2D plot window used. For instance on my screen, if you vertically tile one Algebra and one 2D-Plot window, then issue the Options Grids command and enter a horizontal interval of 7 and a vertical interval of 8 a circle looks pretty circular. You can also manually resize the 2D-plot window using the resizing handles to further refine the display of circles and squares.

.......Yes, we have received similar complaints. In DfW 5.0, we plan to add the capability of saving 2D & 3D plot windows to TIFF, JPEG, GIF, etc. graphic output files. So, tell your friends to hang on! - Aloha, Theresa

Gordon Evans **100302.2510@compuserve.com**

I apologise for this request, but the information I have on *DERIVE* is limited. How did you set about plotting fileds of vector functions such as:

G(x,y) = – (i y + j x)/sqrt(x^2 + y^2)   i and j being unit vectors.

Can you help please? - Regards Gordon Evans

Neil Stahl, Menasha, USA nstahl@uwc.edu

I've used the following functions with *DERIVE* for DOS. I haven't tried them with DfW but probably you can use them to find a way to plot the fields there.

It's a two-stage process. First GRID puts dots at the points, then VFIELD generates line segments for the arrows from those points. There are no heads on our arrows. In DfD one could set the parametric range for lots of curves at once. I hope that's possible in DfW. You should probably keep graph colors from cycling.

E.g. you could type, then simplify and plot:

```
GRID(-2,2,5,-2,2,5)
VFIELD([-y,x]/sqrt(x^2+y^2),-2,2,5,-2,2,5)
```

GRID generates a lattice of 2D points for a<=x<=b and c<=y<=d."

```
GRID(a,b,m,c,d,n):=VECTOR(VECTOR([x,y],x,a,b,(b-a)/m),y,c,d,(d-c)/n)
```

VFIELD can be used to plot elements of the 2D vector field v2 at points indicated as in GRID."

```
VFIELD(v2,a,b,m,c,d,n):=VECTOR(VECTOR([x,y]+t_*v2,x,a,b,
                        (b-a)/m),y,c,d,(d-c)/n)
```

DFIELD plots the direction field u (probably depending on x and/or y) at points indicated as in GRID."

```
DFIELD(u,a,b,m,c,d,n):=VECTOR(VECTOR([x,y]+t_*[1,u]/SQRT(1+u*u),
                       x,a,b,(b-a)/m),y,c,d,(d-c)/n)
```

Terence Etchells and Al Rich:

We are experience a problem with DFW on our network. We are using 3.11 on pentium class chips with 16mb RAM. When DFW loads it uses 8mb of Ram and WIN32 uses just over 4mb!! (it does not do this in Windows 95). This makes the copying and pasting of graphics into, say Word, difficult with lack of memory messages in abundance. Is this is a problem of using WIN32 and is this problem likely to occur with any 32 bit prog in Windows 3.11? As anybody else experienced this problem and is there a solution? We have tried all the suggestions on your FAQ on the web, it made no difference!

Yes. The memory management is very inflexible when DERIVE for Windows (DfW) is running under Win32s which, in turn, is running under Windows 3.1x.

Windows 3.1x is basically a 16-bit protected-mode MS-DOS program. Windows 95 is a far superior platform for running computer algebra systems, like DERIVE, because it is based on a true 32-bit operating system.

Thus, I would highly recommend that DERIVE users seriously consider converting to Windows 95, or the forthcoming Windows 98. Note that version 5 of DfW will NOT be able to run on Windows 3.1x.
- Aloha, Albert D. Rich