THE DERIVE - NEWSLETTER #49

THE BULLETIN OF THE DERIVE USER GROUP + CAS-TI

Contents:

March 2003

New publications

[1] Mathematikunterricht mit Parametern in der Sekundarstufe 1, *Eberhard Lehmann*, Schroedel

[2] Derive 5 im Mathematikunterricht der Sek-1, *Hans-Jürgen Kayser*, bk-teachware SR-35

There are a lot of new publications which can be downloaded from the ACDCA-website.

http://www.acdca.ac.at/material/neu.htm

You can find ACDCA- and T^3-materials as well. Most of the papers are in German, but we are busy to add more and more English papers, too.

New materials include among others:

Beispielsammlung – Collection of Problems (eine ungeheure Menge von Aufgaben, die dem Technologie gestützten M-Unterricht angepasst sind. Alle Aufgaben mit Durcharbeitung mit verschiedenen Technologien / *a rich resource on new technology oriented problems + solutions*)

Vektorrechung mit einem CAS (2 Skripten von Thomas Himmelbauer, bzw. Jürgen Geißelbrecht)

Software for simulating random experiments (Günter Razenberger)

Beschreibende Statistik und Explorative Datenanalyse (Markus Paul)

FAQ (Frequently Asked Questions - Tipps und Tricks im Umgang mit dem TI-92/Voyage)

Finanzmathematik 1 und 2 (für die TIs von Josef Böhm)

Fächerübergreifende Anwendungen der Winkelfunktionen (Tania Koller)

Dear DUG-members,
please inform us about new publications on the use of *DERIVE* and/or the CAS-TIs. We also appreciate all information about interesting websites.

Some websites:

`http://education.ti.com/us/global/freederivetutorials.html`
Download *DERIVE* Tutorials (video clips produced by Terence Etchells!!)

`http://www.harderweb.de/malte/logix.html`
Download a free TI-89/92+/Voyage 200 Application LOGIX for simulating circuits.

`http://shop.bk-teachware.com`
This is **the** address for *DERIVE* - & TI-related books. There is also a rich resource of additional software and inspiring maths books.

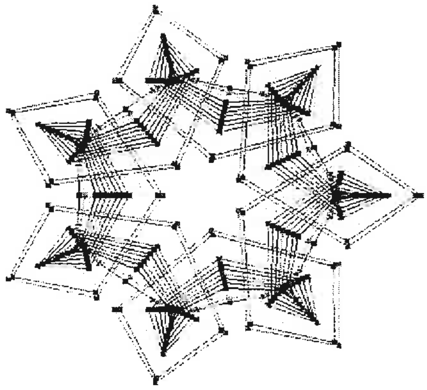
`http://www.tibs.at/wessenberg/`
A fine website of a Tyrolean colleague (Innsbruck) full of valueable links (in German)

Download all *DNL-DERIVE*- and TI-files (+ the "Moon"-file) from

`http://www.acdca.ac.at/t3/dergroup/index.htm`
`http://www.bk-teachware.com/main.asp?session=375059`

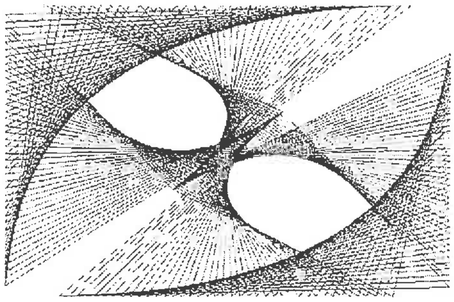
Dear DUG Members,
welcome to *DNL#49* in 2003. Are you realizing that with the next Newsletter we will celebrate the "Golden *DNL*"? 50 issues from 1991 until now, from the puritanic DOS version 1.06 to the glamorous *DERIVE* 5.06, from the first *DERIVE* Conference in Krems 1992 to VISIT-ME 2002 in Vienna, from the simple pocket calculator to the Voyage 200. And we all were witness of this process and many of us much more, because they very actively have been involved in the development and improvement of soft- and hardware and - that seems to be very important -also in spreading and propagating a meaningful use of these powerful tools in maths education.

The "Golden *DNL*" will present the "Silver Titbits #25". Johann Wiesenbauer asked for a break this time and promised to present a special "Titbits #25" in *DNL#50*.

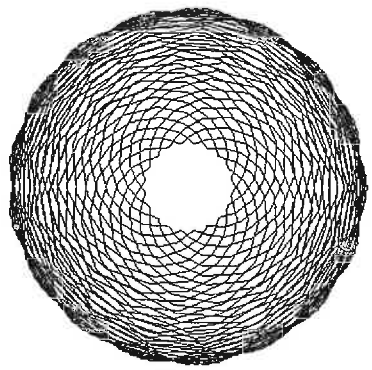


As you can see on the front page, I slightly changed the "+ TI-92" into "+ CAS-TI" incorporating all symbolic TI-devices in this subtitle as TI-89/92/92+ and Voyage 200.

As you can also see on the front page we have so many various contributions. I would like point at the extended User Forum with a bundle of interesting hints, tipps and mails. Out of one of the questions came the "Glider" which could help to close a gap in the *DERIVE* performance. In one of the next *DNLs* I´ll go back to an earlier contribution made by Eugenio Roanes and will bring "programmed colours" on the screen.



We have started with a new column in this Newsletter, initiated by our Spanish friend Enric Puig. We will have a "*Challenge for Programmers*". Its main intention is to enforce *DERIVE* programming presenting not too difficult problems. But I do expect that also TI-programmers might feel challenged. This new column will not replace the *ACDC*-column, but they both will alternate. We all hope that many of you will contribute either in tackling the problems or in providing new challenges.



With my best regards
Josef

Josef

Important: Two days after giving the master copy of *DNL#48* to the printing shop I received the full contribution "With the Sine on the Track of the Moon". Elvira Malitte, Wilfried Herget and Karin Richter did their best to provide a professional translation of their German article, what you can see reading their paper page 7. I can understand that you all will prefer having the complete article translated by the authors. What I can do is the following: **You can download the complete original article from our website (see Information Page).**
Sorry for being too hasty, but before Christmas it is always very busy at the print shop.

The *DERIVE-NEWSLETTER* is the Bulletin of the *DERIVE* & CAS-*TI User Group*. It is published at least four times a year with a contents of 44 pages minimum. The goals of the *DNL* are to enable the exchange of experiences made with *DERIVE* and the *TI*-89/92/*Voyage* 200 as well as to create a group to discuss the possibilities of new methodical and didactical manners in teaching mathematics.

As many of the *DERIVE* Users are also using the CAS-*TIs* the *DNL* tries to combine the applications of these modern technologies.

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

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

Next issue: June 2003
Deadline 15 May 2003

Preview: Contributions waiting to be published

Finite continued fractions Stefan Welke, GER
Kaprekar´s "Self numbers", Richard Schorn, GER
Some simulations of Random Experiments, Josef Böhm, AUT
Wonderful World of Pedal Curves, Josef Böhm
Another Task for End Examination, Josef Lechner, AUT
Tools for 3D-Problems, Peter Lüke-Rosendahl, GER
Penrose Inverse of a Matrix, Karsten Schmidt, GER
ANOVA with DERIVE/TI, MacDonald. R. Phillips, USA
Hill-Encription, Josef Böhm
CAD-Design with DERIVE and the TI, Josef Böhm
Sierpinski-Tetrahedrons and Octahedrons, Heinz-Rainer Geyer, GER
Ways to Write with TI and DERIVE, Milton Lesmes-Acosta
Avoiding Convolution and Transforming Methods, M. Lesmes-Acosta
The "Joseph-Game", Rüdeger Baumann, GER
2D- & 3D-Visualization of Moebius Transformations, Tim Comar, USA
The Random Number Generator of the TI-92, Benno Grabinger, GER
and
R. Setif, FRA; J. Vermeylen, BEL; C. Leinbach, USA; G. Aue, GER; Koller, AUT, K.-H. Keunecke, GER, ...and others

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

Stepwise Programming – Output of intermediate results

Neal Oden noden@erols.com

Three questions that probably betray my novice status:

1) Is there any *DERIVE* command similar to Mathematica's DROP that causes *DERIVE* to forget old definitions? So far, the easiest way that I have found is to save the things I want to remember in a .MTH file, then exit and re-run *DERIVE*, to cause it to forget everything else.

2) In the commands below, is there a way to get Derive to print out intermediate values of x?

```
#1:    x := 0
#2:    LOOP(IF(x = 5, RETURN x), x := x + 1)
```

3) Can you suggest a place for good documentation of procedural programming in *DERIVE*? The online help files are not really that helpful, the *DERIVE* manual says little, and advising users to learn procedural programming by looking at utility files doesn't really do it for me.

Aleksey D. Tetyorko chib@MEGASTYLE.COM

As I see it, you try to debug?
My suggestions:

a) You can use a global variable:

```
#1:   PROG(l:=[],x:=0,LOOP(l:=APPEND(l, [x]),IF(x=5,RETURN x),x:=x+1),l)
#2:   l=
```

b) You can change `LOOP` to `ITERATES` with strange arguments (ask me by mail)

c) Try not to use equality in `IF`, maybe, `IF(x>=5,...` will be better.
Aleksey

Terence Etchells T.A.Etchells@LIVJM.AC.UK

Hi Neal (and anyone else who reads this),

1) There is no general command that clears all variable and function definitions from your active Algebra Window, although it would be a good feature. An efficient work around is to remove from the Algebra Window any definitions you want to remove (keeping the definitons you want to keep, obviously). Ensuring that you are in the Algebra Window, i.e. not in the entry bar, select all the the expressions using Select All from the Edit menu or pressing Ctrl A. Copy the expressions using Ctrl C or the Copy icon or the copy command in the Edit menu. Now open a new Algebra Window (using the File menu or clicking on the New Algebra Window icon) and paste the expressions into the new Algebra Window, (again using Ctrl V, or paste from the Edit menu or the Paste Icon). All the unwanted definitions will not be present in this new Algebra Window. Return to the old Algebra Window and close it without saving it.

Once you've done this process a few times it's not as complicated as it may seem in print.

2) *DERIVE* works (presently) as a Turing machine, in that it perfoms all actions before it returns an output. I believe that this may be addressed in the future. By the way a more efficient way to increment a variable by one is to use `x:+1` instead of `x:=x+1`.

3) I have a website `http://www.cms.livjm.ac.uk/deriveprogramming` that introduces the fundamentals of *DERIVE* programming.

Hope this helps,

Terence

DNL

Dear all,

concerning question (1) I recommend following Terence's advice from above. I have one additional comment: In many cases the fact that you can "hide" earlier definitions turns out to be an advantage.

Question (2):

I found the following "trick??". As you did one has to make the respective variable(s) "global". Take program `stepwise(n)` as an example. Edit `stepwise(5)` and simplify this command again and again (pressing the "="-button in the Edit line) which results in the output as shown below. Unlike Aleksey's realisation this program works really stepwise. i.e. you call and see one intermediate value after the other.

```
#1:   [x := 0, subtotal := 0]

      stepwise(n) :=
        Prog
          Loop
            x :+ 1
#2:         subtotal := subtotal + x^2
            If x > n
               RETURN "done"
            RETURN [x, subtotal]

#3:   stepwise(5)

#4:   [1, 1]

#5:   [2, 5]

#6:   [3, 14]

#7:   [4, 30]

#8:   [5, 55]

#9:   done
```

```
#10:  [x := 0, subtotal := 0]

      steploop(n) :=
        Loop
          x :+ 1
#11:      subtotal := subtotal + x^2
          If x > n
             RETURN "done"
          RETURN [x, subtotal]

#12:  steploop(5) = [1, 1]

#13:  steploop(5) = [2, 5]

#14:  steploop(5) = [3, 14]

#15:  steploop(5) = [4, 30]

#16:  steploop(5) = [5, 55]

#17:  steploop(5) = done
```

Don't forget to reset the variables after performing one or more runs of the program, and for a restart, of course. As you can see, a `loop()` behaves like a complete program.

Quick response to question (3)

I share your opinion about the very (too) rare programming resources.

Beside Terence's Website the *DERIVE* Newsletter is one of the very few resources on programming in DERIVE. There is also a book on Programming in *DERIVE 5*, but it is available only in German.

Comparing Coefficients

Rainer Wonisch

Hi, Josef

How can I force *DERIVE* to perform a comparison of coefficients. One colleague asked me today. Take the following example:

$$x^2 - 4x + 5 = x^2 + px + q$$

should return values for p and q.

DNL

Dear Rainer,

I found the following solution, assuming working with polynomials only and further assuming that they are of same degree. Maybe that some *DERIVER*s will find a more elegant approach.

```
coeff(poly, v_) := VECTOR(POLY_COEFF(poly, v_, i), i, 0, POLY_DEGREE(poly, v_))
```

$$\text{compcoeff(expr1, expr2, v\_)} := \left[\text{VECTOR}(v\_^{i}, i, 0, \text{POLY\_DEGREE(expr1, v\_)}), \text{coeff(expr1, v\_)}, \text{coeff(expr2, v\_)}\right]`$$

$$\text{compcoeff}(x^2 - 4\cdot x + 5, x^2 + p\cdot x + q, x) = \begin{bmatrix} 1 & 5 & q \\ x & -4 & p \\ x^2 & 1 & 1 \end{bmatrix}$$

$$\text{compcoeff}(3\cdot a\cdot x^3\cdot y^3 + a\cdot x^2\cdot y - 3\cdot x^2 + 10\cdot x - 1, -x^3 + x^2 - 2\cdot b\cdot x^2 + a\cdot x\cdot y - y^3, x)$$

$$\begin{bmatrix} 1 & -1 & -y^3 \\ x & 10 & a\cdot y \\ x^2 & a\cdot y - 3 & 1 - 2\cdot b \\ x^3 & 3\cdot a\cdot y^3 & -1 \end{bmatrix}$$

$$\text{compcoeff}(3\cdot a\cdot x^3\cdot y^3 + a\cdot x^2\cdot y - 3\cdot x^2 + 10\cdot x - 1, -x^3 + x^2 - 2\cdot b\cdot x^2 + a\cdot x\cdot y - y^3, y)$$

$$\begin{bmatrix} 1 & -3\cdot x^2 + 10\cdot x - 1 & x^2\cdot(1 - 2\cdot b) - x^3 \\ y & a\cdot x^2 & a\cdot x \\ y^2 & 0 & 0 \\ y^3 & 3\cdot a\cdot x^3 & -1 \end{bmatrix}$$

Repeating

Robert Hunn rwhunn@WORLDNET.ATT.NET

I am using the Derive 5.05 version. I enter the rational 14/15 and then approximate, giving of course the value 0.9333333333; but when I simplify the resulting rational is 9333333333/10000000000. That is to be expected. However, is there a function or procedure that could be programmed to convert back to the original rational if the decimal includes the repeating part?

Jim FitzSimons

Approximate with fewer digits. #9: a := 0.9333333333 #10: b := APPROX(a, 6) Simp(#10') #11: b := 0.933333 #12: $b = \frac{14}{15}$ (My DERIVE 5.06 doesn´t require fewer digits, but it still needs the b:= approx(...). Taking fewer digits for the output does not result in 14/15. Josef)	$\frac{14}{15}$ 0.9333333333 $\frac{9333333333}{10000000000}$ $(b := \text{APPROX}(0.9333333333, 6)) = \frac{14}{15}$ $(b := \text{APPROX}(0.9333333333)) = \frac{14}{15}$

Johann Wiesenbauer j.wiesenbauer@tuwien.ac.at

There is still another solution though not quite as elementary as Jim's.

You could compute the first few convergents of the number at issue and take the very first convergent that approximates to the same number:

```
CONVERGENTS(0.9333333333, 5)
```

$$\left[0,\ 1,\ \frac{13}{14},\ \frac{14}{15},\ \frac{1866666661}{1999999994},\ \frac{3733333336}{4000000003}\right]$$

```
[0, 1, 0.9285714285, 0.9333333333, 0.9333333333, 0.9333333333]
```

As you can see, this yields 14/15 in your example indeed! I leave it to you to write a program that does all the necessary steps automatically.

Cheers, Johann

3D Plots of solids of revolution

Matthieu Gouin woopee77@yahoo.com

I have a question about *DERIVE* 5. How can I plot a volume of revolution around the Y-axis when I have a function defined in termsof x, such as $y = f(x)$

I found how to plot the volume of the same function around the X-axis by using :
`[x, f(x)·COS(t), f(x)·SIN(t)]` and plot this "vector" with 3-D plot.

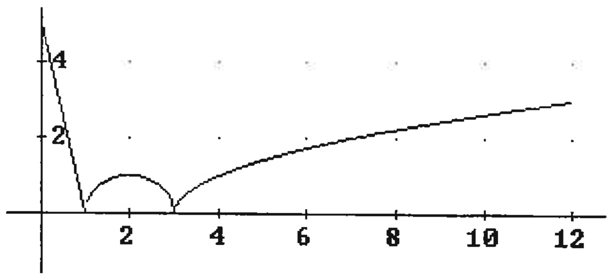
How can I do the same around the Y-axis ?

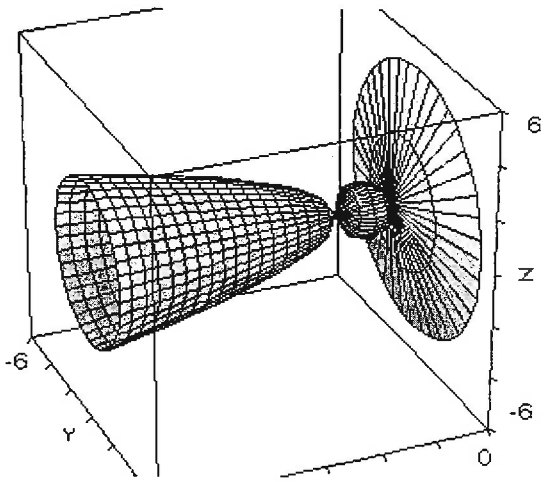
Vladimir Bondarenko vvb@mail.strace.net

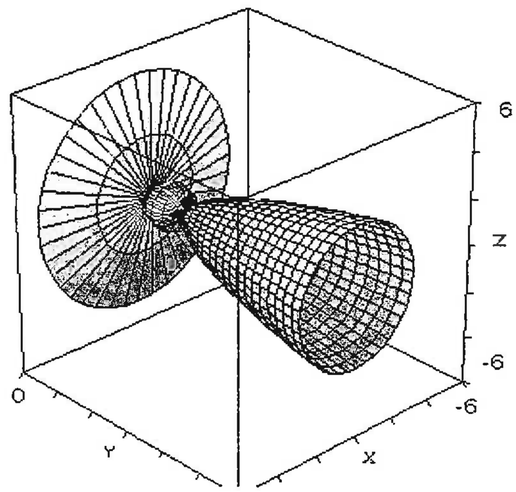
Please use `[f(s)·COS(t), s, f(s)·SIN(t)]` to change the direction.

Best wishes,

```
f(x) :=
  If x < 0
     ?
     If x ≤ 1
        5 - 5·x
        If x ≤ 3
           √(1 - (x - 2)^2)
           If x ≤ 12
              √(x - 3)
              ?
```







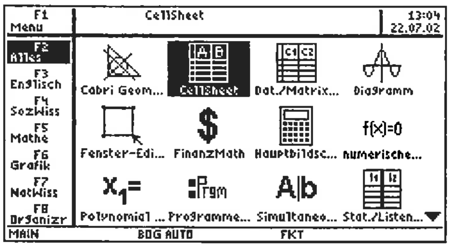
to be continued on page 21

With the Sine on the Track of the Moon (2)[1]

Wilfried Herget, Elvira Malitte, Karin Richter, Halle (Saale)

Evaluation of worksheets 1 and 2 using *CellSheet* on Voyage 200

CellSheet is a „mini" spread-sheet system developed as flash application for both the graphics calculator TI-83+ and the symbolic algebra calculators TI-89 and TI-92+ or its successor Voyage 200, respectively. The opposite illustration shows the selection of the CellSheet application from a variety of applications offered as standard feature for the Voyage 200.



Essential properties of a spread-sheet program were realized in Cell-Sheet

- The cells can contain the following types: numbers, formulas, terms, strings, functions.
- In the formulas one can use absolute and relative cell-references as well as a great number of pre-defined functions.
- It is easy to perform usual manipulations like
  selecting cells or domains, respectively, copying, inserting, deleting,
  inserting or deleting rows or columns,
  importing or exporting matrices, lists, variables,
  opening or storing files,
  sorting selected domains
  ...
- Different statistical calculations are predefined.
- Representation of numerical data by means of different diagram types.
- ...

Thus CellSheet offers a significant extension of the possibilities given by the TI-83+ STAT-menu as well as of the DATA-MATRIX- editor of the symbolic calculator.

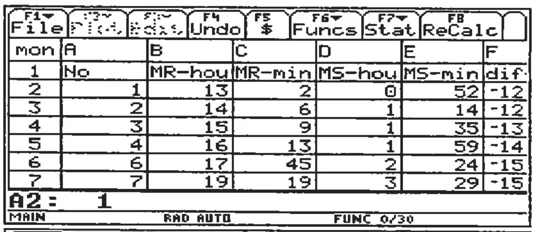
Data and formula input in CellSheet

When entering data into CellSheet one can take over all gaps of the initial table in a similar way. Every cell of the worksheet can be reached at any time.

Our intention is a graphical visualization and therefore we will drop also here the input of incomplete records.

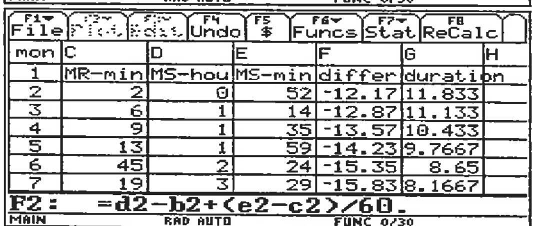
Entering the time into a cell will lead to an error message. Only integers and numbers with decimal point are admissible as numerical values. This is why here the data input is organised quite similarly to the corresponding input into a graphics calculator.

The illustrations on the right hand side (below) show the input of the first data and formulas as well as the values calculated by them. The formula input is done by entry into a cell. The usual manipulations of marking, selecting a domain, and inserting lead to an efficient formula input in the whole domain. In doing so one completes the relative cell references. Computing the formula in column F yields a number with decimal point and not a fraction, since the decimal number 60 was used as quotient.



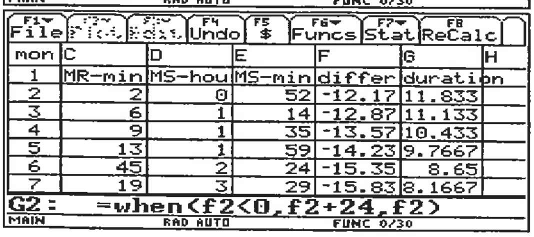
mon	A	B	C	D	E	F
1	No	MR-hou	MR-min	MS-hou	MS-min	dif·
2	1	13	2	0	52	-12
3	2	14	6	1	14	-12
4	3	15	9	1	35	-13
5	4	16	13	1	59	-14
6	6	17	45	2	24	-15
7	7	19	19	3	29	-15

A2: 1



mon	C	D	E	F	G	H
1	MR-min	MS-hou	MS-min	differ	duration	
2	2	0	52	-12.17	11.833	
3	6	1	14	-12.87	11.133	
4	9	1	35	-13.57	10.433	
5	13	1	59	-14.23	9.7667	
6	45	2	24	-15.35	8.65	
7	19	3	29	-15.83	8.1667	

F2: =d2-b2+(e2-c2)/60.

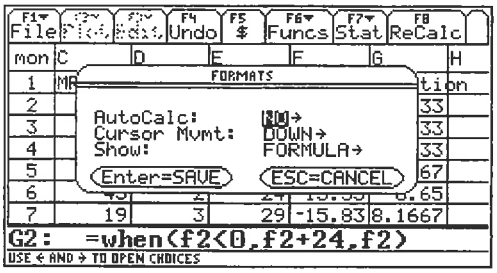


mon	C	D	E	F	G	H
1	MR-min	MS-hou	MS-min	differ	duration	
2	2	0	52	-12.17	11.833	
3	6	1	14	-12.87	11.133	
4	9	1	35	-13.57	10.433	
5	13	1	59	-14.23	9.7667	
6	45	2	24	-15.35	8.65	
7	19	3	29	-15.83	8.1667	

G2: =when(f2<0,f2+24,f2)

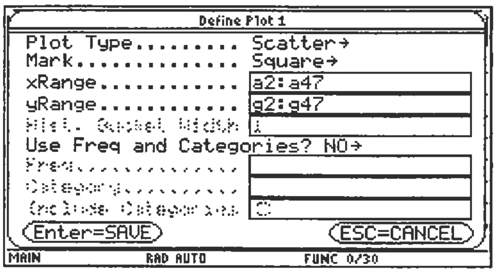
[1] See information on page 1

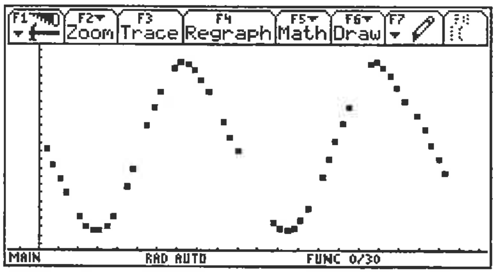
We don't want to waste too much time with the data input and formula calculation in CellSheet and therefore we must take care to switch off the continuous automatic recaclulation within this program. This setting can be done in the F1-menu under Format, see the opposite illustration.



Graphical data representation

No column variables are given here for the graphical representtion, but it is possible to select subdomains of the entered data. Here we first used all existing data records and defined a corresponding plot. This yields the illustration on the right hand side.





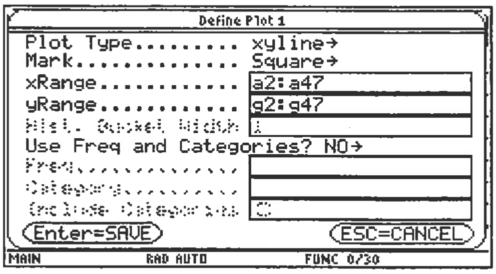
Does it make sense to connect the data points to obtain a curve?

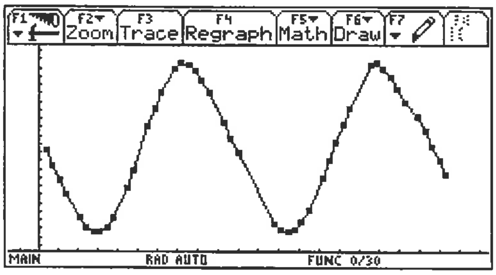
There are two points of view and both have to be discussed:

➢ *In the real context* we have measured values that are connected – with respect to the functional relationship considered here – with discrete natural numbers (i.e. the numbers of the day) for the domain of definition. Therefore the domain of definition is *discrete.*

➢ If, on the other hand, the main emphasis is on the mathematical quality of the functional relationship then it does make sense to investigate the sine function for a *continuous* domain of definition. Finally, when interpreting the sine curve fitted to the measured values, we have again to return to the discretization, that is, we have to consider only integer values of the domain of definition.

The following setting yields a curve with the data points connected:





Which sine curve has this shape? Let's try to give a stepwise solution to this question by means of different approaches.

Which Sine fits best?

Worksheet 3

1. Try first functions with

$$y(x) = a \cdot \sin(x) + d \, .$$

What's the meaning of a here? And what is the meaning of d?

What is the maximal duration of moonlight, what is the minimal one? What is the meaning of this for the value of a? What's the meaning for d?

2. Now try functions with

$$y(x) = a \cdot \sin(b \cdot x) + d \, .$$

What's the meaning of b here?

3. Are you satisfied with the fitting? If necessary, make further changes to the function

4. How can you fill in in a meaningful way data missing in the initial table on the duration of moonlight? Among others, the duration of moonlight is missing for July 5 and for July 8-Aug 2.

Which duration of moonlight can be predicted for Sep 30?

Date	Day-Number	Duration of moonlight
05.07.	5	
30.07.	30	
31.07.	31	
01.08.	32	
02.08.	33	
30.09.		

Evaluation of worksheet 3 using TI-83+

We begin with the formulation $y(x) = a\ \sin(x) + d$

Consideration concerning *a*:

The ***amplitude*** can be estimated quite simply from the difference between the greatest and the least *y*-value (see list D)

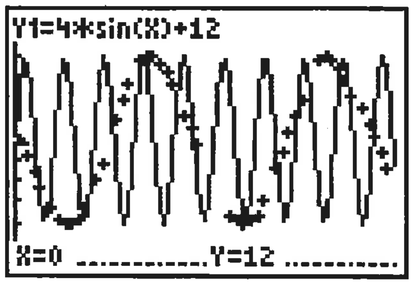
$a \to (y_{max} - y_{min})/2 \to (16 - 8)/2 = 4.$

Consideration concerning *d*:

Suitable translation along the *y*-axis: $d \to y_{min} + a \to 8 + 4 = 12.$

Hint: In the WINDOW-setting we magnify the *y*-domain here a little bit so that some room is left in the graphics window for giving the corresponding function equation.

This yields the first trial for a sine function as fitting as possible:



Our first sine-proposal $Y_1 = 4\ \sin(x) + 12$ oscillates "too fast". In the represented domain it manages to reach 10 oscillations, whereas the data crosses manages only two.

Correction: We extend our previous formulation

$y(x) = a\ \sin(x) + d$ to the more general statement $y(x) = a\ \sin(b\ x) + d.$

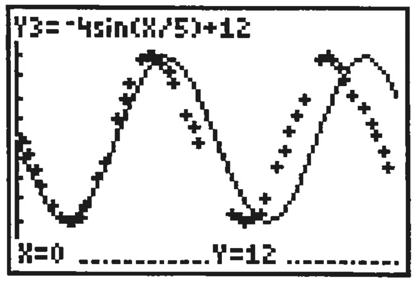
Consideration concerning *b*: Fitting the ***period length.***

Proposal $b \to 1/5$:

Concerning the frequency this yields a clearly better approximation of the sine curve to the data crosses.

However: The oscillation of the data points first goes downwards and then upwards. The sine function behaves conversely. – What is to be done?

A possible idea of solution is ***mirroring*** – which is done by changing the sign of *a* in the function term.

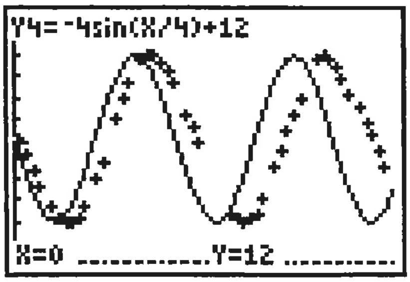


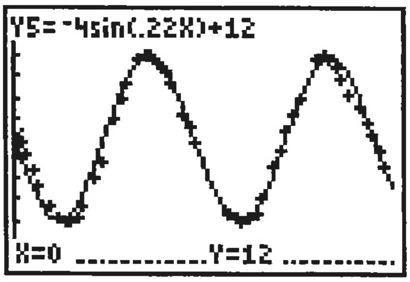
There still remains to be done some ***fine tuning*** with the value of *b*, in order to regulate the period length more exactly:

The last function-proposal obviously yields a quite good approximation to the data points.

Therefore we will use further on

$Y_5 = -4\ \sin(0.22\ x) + 12.$

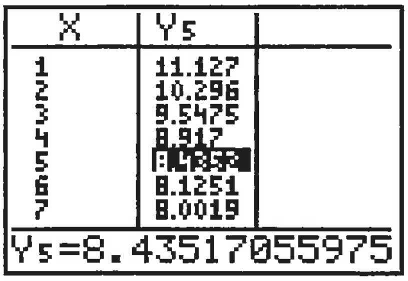


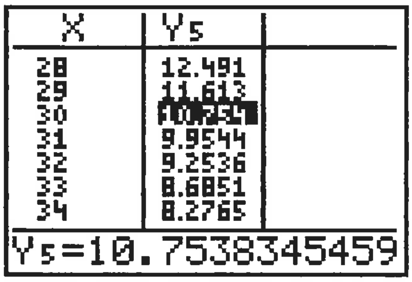


How can we in a meaningful way fill in data missing in the table on the duration of moonlight?

For this we can use the value table of Y_5.

The first gap in our data list occurs on the 5th day. Moreover e.g. the duration of moonlight is missing for the days from July 30 through Aug 2, that is for the days with the numbers 30 through 33.





From the value table we conclude that the duration of moonlight on July 5 is approximately 8 hours and 25 min, on July 30 it is 10 hours and 45 min, on July 31 it's just under 10 hours (some 3 minutes less) etc.

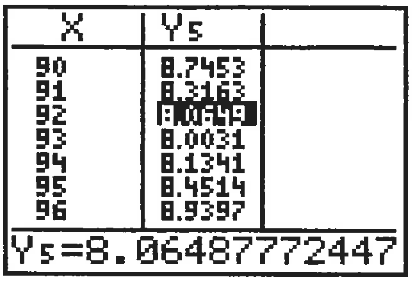
Which duration of moonlight can be predicted for Sep 30?

Evaluating the value table or the graph can again prove helpful here, but it is also possible to call the function at a specific place.

By our definitions Sep 30 has number 92.

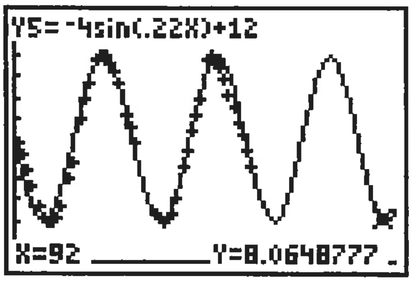
1. Approach

We may look up the corresponding place in the value table or …



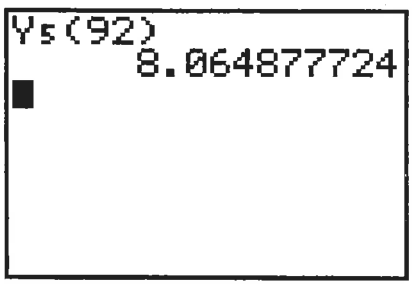
2. Approach

use TRACE in the graphics window with an x-domain extended (over WINDOW) or…



3. Approach

call the relevant function value directly in the main display.

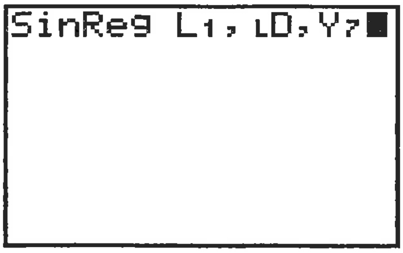


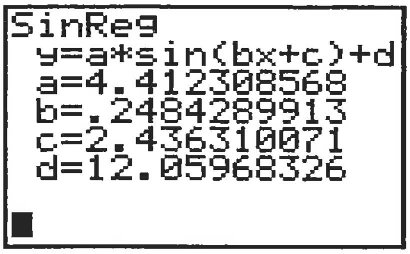
Result: On Sep 30 we expect a duration of moonlight of more than 8 hours.

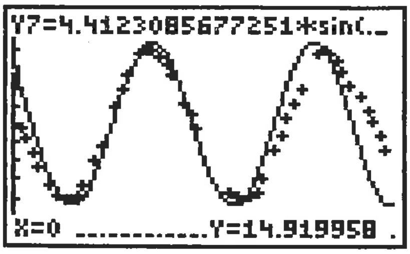
Curve fittings offered by TI-83+

To get the solution we could also use the possibilities offered by the TI-83+ for the direct curve fitting (regression analysis).

The menu STAT CALC offers, among others, the possibility of determining a sine-regression by means of `SinReg`. To do so one has to enter there the list names of the x-values and y-values (here L_1 and D) as well as to determine the Y-variable in which the result of the regression is to be stored (here Y_7), see the left-hand side of the following illustration. Then the parameter values suggested by the calculator will be displayed and the graph of Y_7 can be drawn.

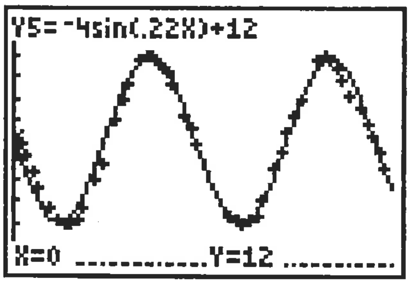






It is an obvious choice to compare the graph determined just now by the sine regression SinReg of the graphics calculator with our previous proposal $Y_5 = -4\sin(0.22x) + 12$.

The comparison is impressing and it is a little bit disappointing for all pure calculator fans: the superiority of our curve proposal in comparison to the SinReg-solution given by the calculator should be clearly visible – in particular due to the obviously better fitting for higher numbers of the day.



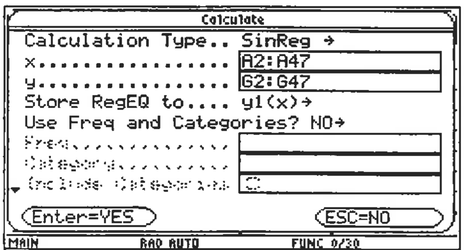
Curve fittings offered by Voyage 200

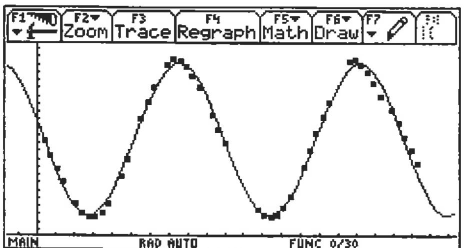
The sine regression in the Stat-menu of CellSheet offers new possibilities for experiments. For the regression analysis one does not necessarily use all data of the corresponding columns, but upon request only subdomains. It can be interesting to use different data domains for a regression analysis and at the same time to examine the following question:

- Has the extent of the used data any effect on the result of the sine regression?
- We choose different domains for the x-values, but always the same number of value pairs. Does the sign regression yield different results?
- We choose different domains for the x-values with same interval length each, which is obtained from the difference between the greatest and the least x-values. What do you see from the results?
- How to interpret the results?

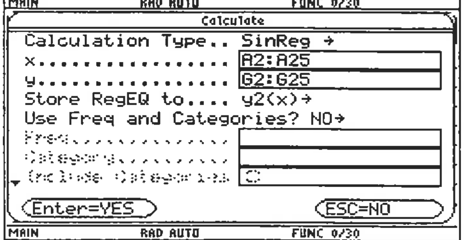
Here is a choice from the different investigations:

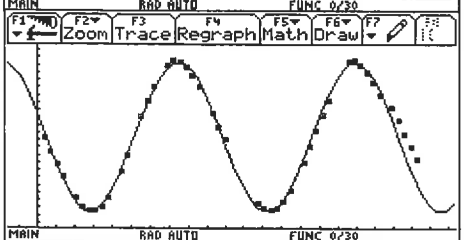
First all data are used and the result of the regression is stored in y1(x). The graphical visualization shows a good curve fitting.



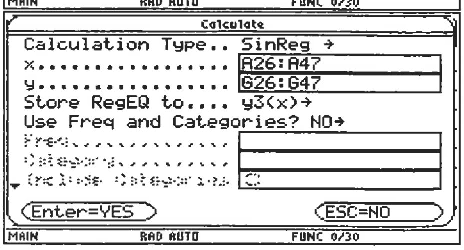


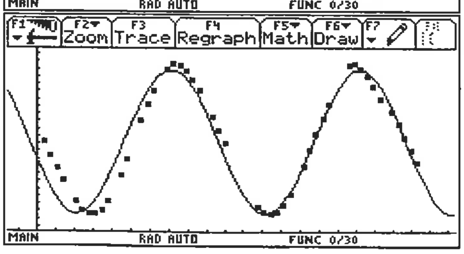
Using the first half of the corresponding x-interval also yields a satisfying result. The curve fitting in the used interval is better than in the adjacent domain.



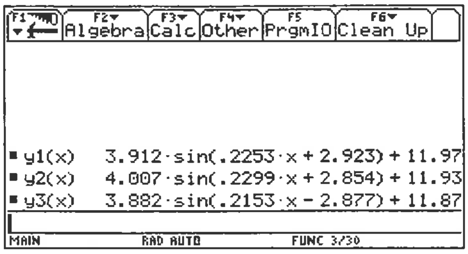


If the remaining value pairs are evaluated we obtain a particularly good curve fitting in the domain.





The above regression equations were stored in the variables y1(x), y2(x), and y3(x), respectively. The illustration on the right hand side (below) shows a summary of the function terms in the main display of the calculator.



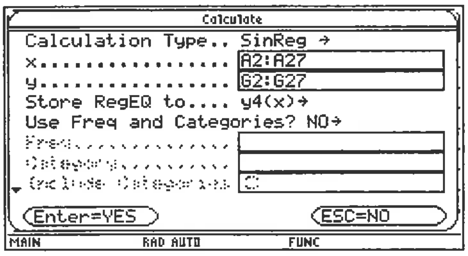
Small changes – big effect

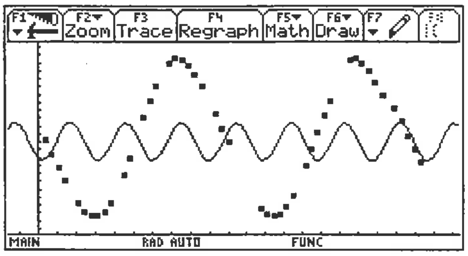
Small changes in the used domain can nevertheless lead to surprising results. The following illustrations show the respective domains chosen. Moreover we see the results of SinReg.

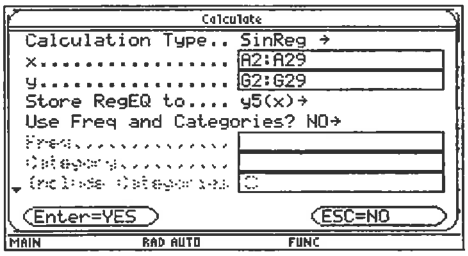
The first example gives a graphical visualization of the calculated function together with the data points. It is obvious that we don't succeed in getting a curve fitting.

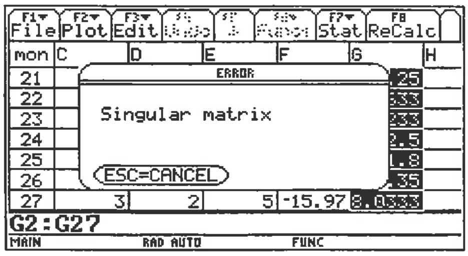
In the second example the computation is interrupted by the remark "singular matrix". This indicates the complex numerical problems to be managed.

We dispense with the graphical visualization of the third example. It is sufficient to consider the parameters *a* and *b*.

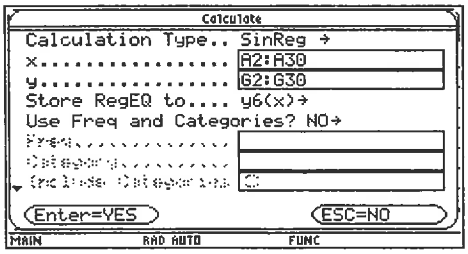


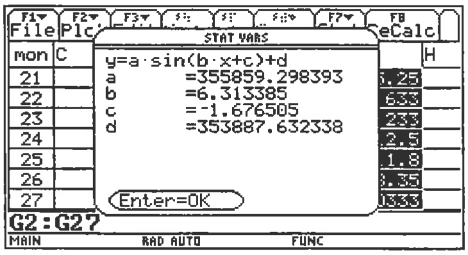


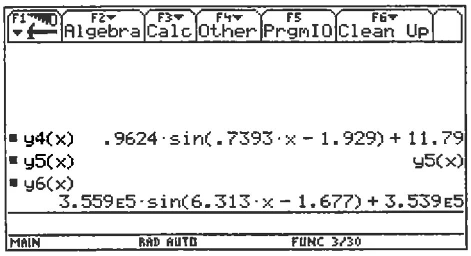




Here are the function terms derived from SinReg and associated with these three examples:







It is surely desirable that a calculator always gives correct results. However, these examples can also support a critical handling of calculator programs. In these examples the incorrectness of the results is obvious – fortunately.

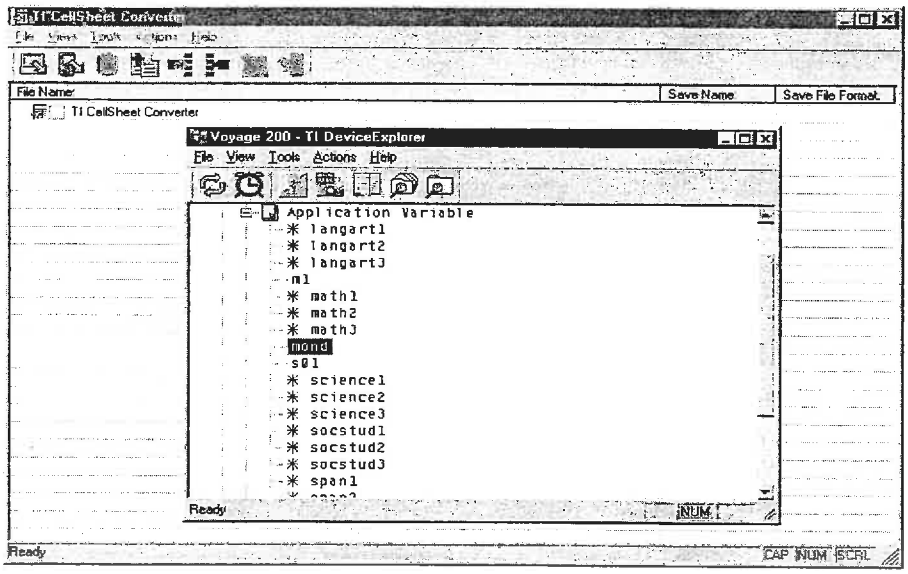
The solution of extensive systems of equations is a task hiding behind the regression analysis – a very difficult problem not only for small-sized calculators but sometimes also for much more powerful computer algebra systems and may lead also here to errors.

Data exchange between CellSheet und Excel

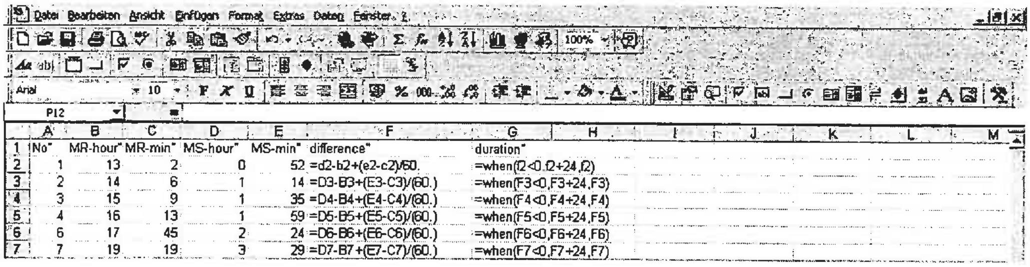
The „mini" spread-sheet program CellSheet can come up to some expectations with respect to a spread-sheet program, but not all wishes will be fulfilled. A data exchange with suitable PC-software may be helpful in getting supplements in different areas e.g. concerning the

- scope of work
- graphical visualization
- interaction
- ...

Using the software *TI CellSheet Converter,* a simple exchange of worksheets is possible between CellSheet, Excel and other spread-sheet programs via PC an Macintosh. The following illustration was chosen as data source in the *TI CellSheet Converter* of Voyage 200 and if the cable connection between PC and symbolic calculator works then all interesting data are at our disposal for transfer to the PC.



We select the file existing on the Voyage 200 and copy this file to the PC. After converting into Excel format and storing, the associated file can be opened in Excel. Here is a detail from the corresponding file:

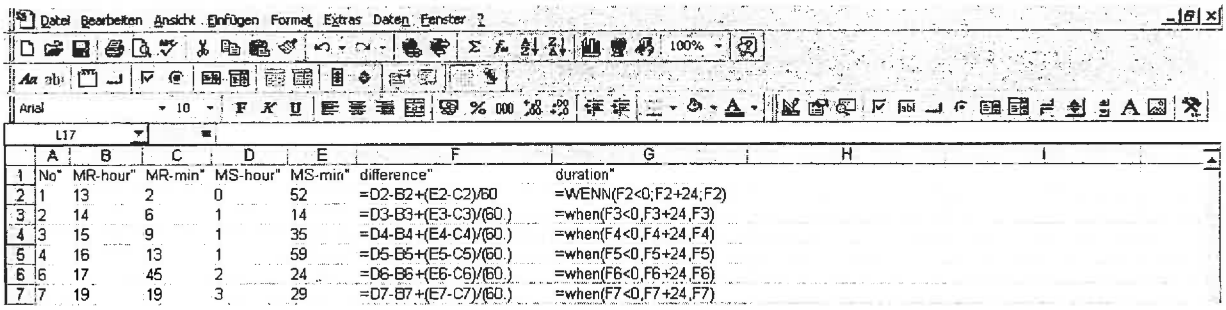


	A	B	C	D	E	F	G
1	No"	MR-hour"	MR-min"	MS-hour"	MS-min"	difference"	duration"
2	1	13	2	0	52	=d2-b2+(e2-c2)/60.	=when(f2<0,f2+24,f2)
3	2	14	6	1	14	=D3-B3+(E3-C3)/(60.)	=when(F3<0,F3+24,F3)
4	3	15	9	1	35	=D4-B4+(E4-C4)/(60.)	=when(F4<0,F4+24,F4)
5	4	16	13	1	59	=D5-B5+(E5-C5)/(60.)	=when(F5<0,F5+24,F5)
6	6	17	45	2	24	=D6-B6+(E6-C6)/(60.)	=when(F6<0,F6+24,F6)
7	7	19	19	3	29	=D7-B7+(E7-C7)/(60.)	=when(F7<0,F7+24,F7)

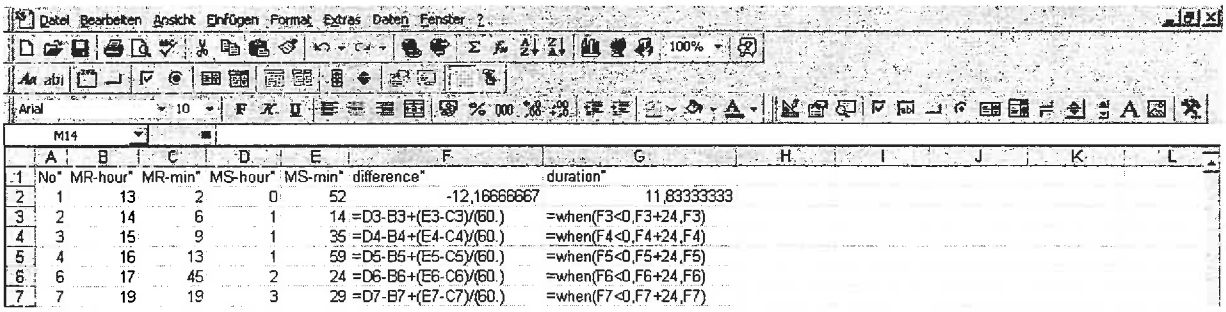
The formulas in the columns F and G will not be calculated but rather handled as text. Some fittings must still be performed manually.

In column F the decimal point appears. Deleting the decimal point is easily done here and this leads to a correct calculation of the formula in Excel. This example shows, however, that all cells containing in CellSheet numbers with decimal point will lead to problems when converted to Excel. Unfortunately the decimal point in the number representation of CellSheet will not be converted into the point of the number representation of Excel.

The designations of the predefined functions and associated separators are not „yet“ fitted during conversion. It is necessary to replace condition “when” (CellSheet) by “WENN” (Excel) and the two points by semicolons.



	A	B	C	D	E	F	G
1	No"	MR-hour"	MR-min"	MS-hour"	MS-min"	difference"	duration"
2	1	13	2	0	52	=D2-B2+(E2-C2)/60	=WENN(F2<0;F2+24;F2)
3	2	14	6	1	14	=D3-B3+(E3-C3)/(60.)	=when(F3<0,F3+24,F3)
4	3	15	9	1	35	=D4-B4+(E4-C4)/(60.)	=when(F4<0,F4+24,F4)
5	4	16	13	1	59	=D5-B5+(E5-C5)/(60.)	=when(F5<0,F5+24,F5)
6	6	17	45	2	24	=D6-B6+(E6-C6)/(60.)	=when(F6<0,F6+24,F6)
7	7	19	19	3	29	=D7-B7+(E7-C7)/(60.)	=when(F7<0,F7+24,F7)

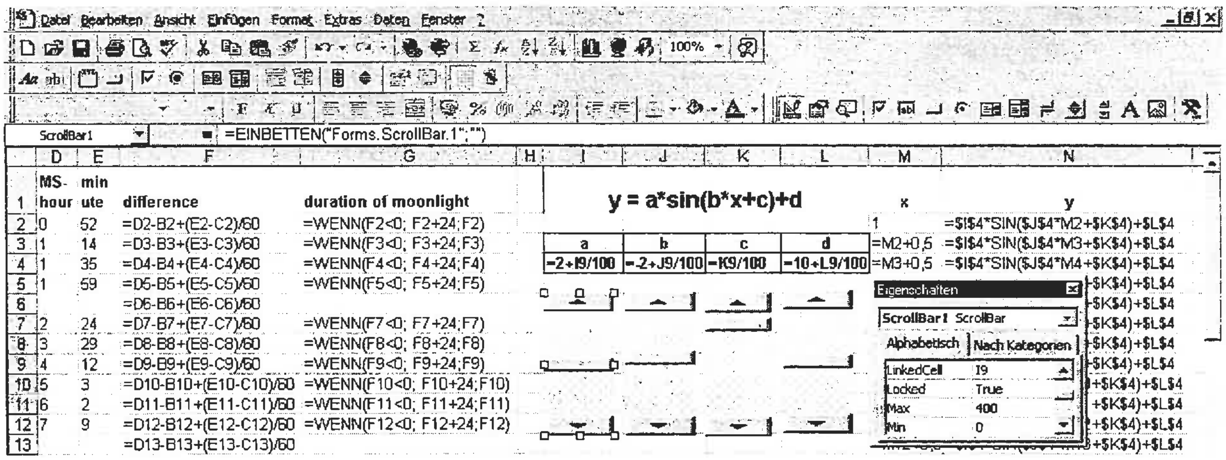


	A	B	C	D	E	F	G
1	No"	MR-hour"	MR-min"	MS-hour"	MS-min"	difference"	duration"
2	1	13	2	0	52	-12,16666667	11,83333333
3	2	14	6	1	14	=D3-B3+(E3-C3)/(60.)	=when(F3<0,F3+24,F3)
4	3	15	9	1	35	=D4-B4+(E4-C4)/(60.)	=when(F4<0,F4+24,F4)
5	4	16	13	1	59	=D5-B5+(E5-C5)/(60.)	=when(F5<0,F5+24,F5)
6	6	17	45	2	24	=D6-B6+(E6-C6)/(60.)	=when(F6<0,F6+24,F6)
7	7	19	19	3	29	=D7-B7+(E7-C7)/(60.)	=when(F7<0,F7+24,F7)

This solves the problems that arose here. By the usual manipulations Copy, Select and (Complete/Fill in) we quickly succeed completing the fitting in all cells. The time consuming work task of a spreadsheet program – i.e. the data input – could be used on the PC without “rework”.

Control elements for dynamic curve fitting in Excel

There are further possibilities to visualize a stepwise curve fitting. By means of the control element toolbox it is possible to define elevators for varying the parameters. The following illustration shows a “scrolling” of the parameter *a* over 401 values (see properties of the ScrollBar – Max 400, Min 0) in the interval [2;6] (see formula in cell I4 with respect to I9 – “LinkedCell” of the ScrollBar). A similar statement holds true for the parameter variation of *b*, *c*, and *d* (see the formulas J4, K4, and L4).

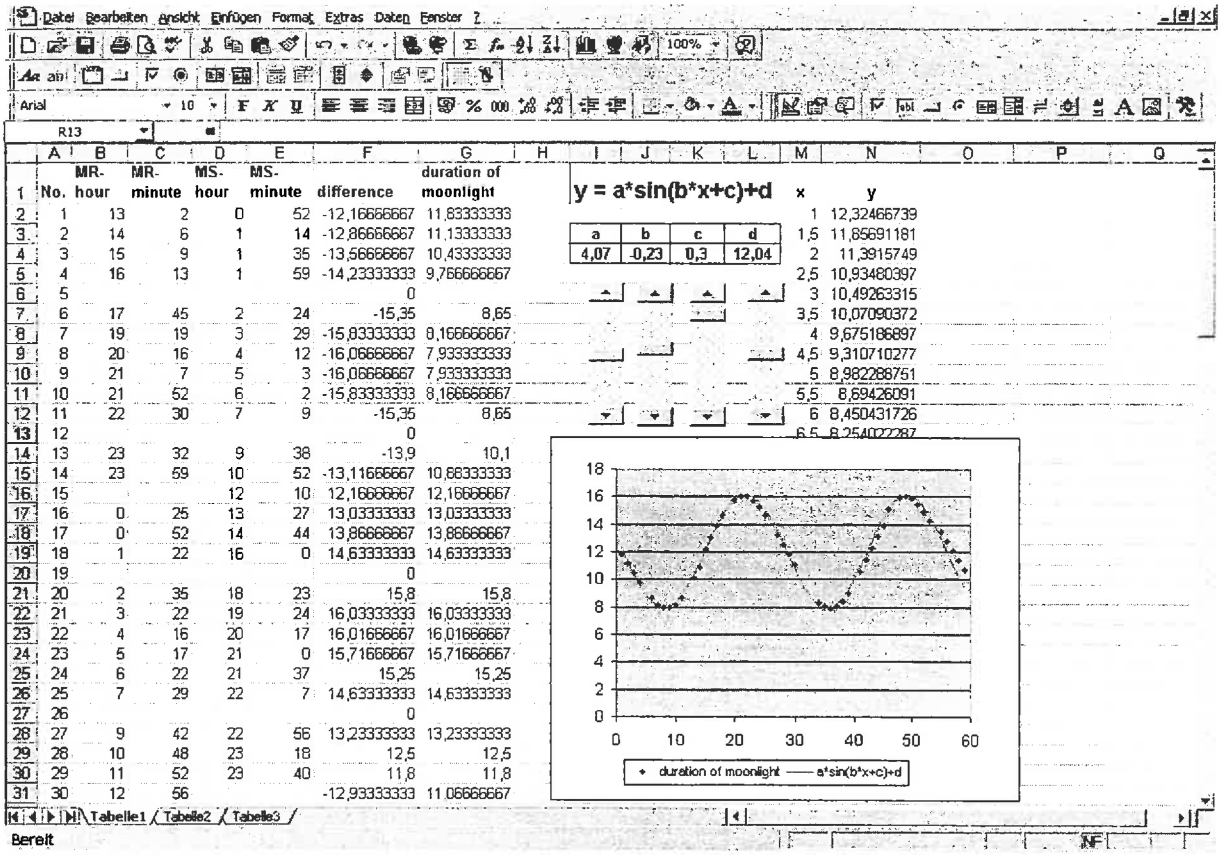


	D	E	F	G
1	MS-hour	minute	difference	duration of moonlight
2	0	52	=D2-B2+(E2-C2)/60	=WENN(F2<0; F2+24;F2)
3	1	14	=D3-B3+(E3-C3)/60	=WENN(F3<0; F3+24;F3)
4	1	35	=D4-B4+(E4-C4)/60	=WENN(F4<0; F4+24;F4)
5	1	59	=D5-B5+(E5-C5)/60	=WENN(F5<0; F5+24;F5)
6			=D6-B6+(E6-C6)/60	
7	2	24	=D7-B7+(E7-C7)/60	=WENN(F7<0; F7+24;F7)
8	3	29	=D8-B8+(E8-C8)/60	=WENN(F8<0; F8+24;F8)
9	4	12	=D9-B9+(E9-C9)/60	=WENN(F9<0; F9+24;F9)
10	5	3	=D10-B10+(E10-C10)/60	=WENN(F10<0; F10+24;F10)
11	6	2	=D11-B11+(E11-C11)/60	=WENN(F11<0; F11+24;F11)
12	7	9	=D12-B12+(E12-C12)/60	=WENN(F12<0; F12+24;F12)
13			=D13-B13+(E13-C13)/60	

	I	J	K	L
3	a	b	c	d
4	=2+I9/100	=-2+J9/100	=K9/100	=10+L9/100

	M	N
1	x	y
2	1	=$I$4*SIN($J$4*M2+$K$4)+$L$4
3	=M2+0,5	=$I$4*SIN($J$4*M3+$K$4)+$L$4
4	=M3+0,5	=$I$4*SIN($J$4*M4+$K$4)+$L$4
5		+$K$4)+$L$4
6		+$K$4)+$L$4
7		+$K$4)+$L$4
8		+$K$4)+$L$4
9		+$K$4)+$L$4
10		+$K$4)+$L$4
11		+$K$4)+$L$4
12		+$K$4)+$L$4
13		+$K$4)+$L$4

The graphical visualization in the diagram and the value table in columns M and N reflect the dynamically changing parameter settings.



	No.	MR-hour	MR-minute	MS-hour	MS-minute	difference	duration of moonlight
2	1	13	2	0	52	-12,16666667	11,83333333
3	2	14	6	1	14	-12,86666667	11,13333333
4	3	15	9	1	35	-13,56666667	10,43333333
5	4	16	13	1	59	-14,23333333	9,766666667
6	5					0	
7	6	17	45	2	24	-15,35	8,65
8	7	19	19	3	29	-15,83333333	8,166666667
9	8	20	16	4	12	-16,06666667	7,933333333
10	9	21	7	5	3	-16,06666667	7,933333333
11	10	21	52	6	2	-15,83333333	8,166666667
12	11	22	30	7	9	-15,35	8,65
13	12					0	
14	13	23	32	9	38	-13,9	10,1
15	14	23	59	10	52	-13,11666667	10,88333333
16	15			12	10	12,16666667	12,16666667
17	16	0	25	13	27	13,03333333	13,03333333
18	17	0	52	14	44	13,86666667	13,86666667
19	18	1	22	16	0	14,63333333	14,63333333
20	19					0	
21	20	2	35	18	23	15,8	15,8
22	21	3	22	19	24	16,03333333	16,03333333
23	22	4	16	20	17	16,01666667	16,01666667
24	23	5	17	21	0	15,71666667	15,71666667
25	24	6	22	21	37	15,25	15,25
26	25	7	29	22	7	14,63333333	14,63333333
27	26					0	
28	27	9	42	22	56	13,23333333	13,23333333
29	28	10	48	23	18	12,5	12,5
30	29	11	52	23	40	11,8	11,8
31	30	12	56			-12,93333333	11,06666667

y = a*sin(b*x+c)+d

a	b	c	d
4,07	-0,23	0,3	12,04

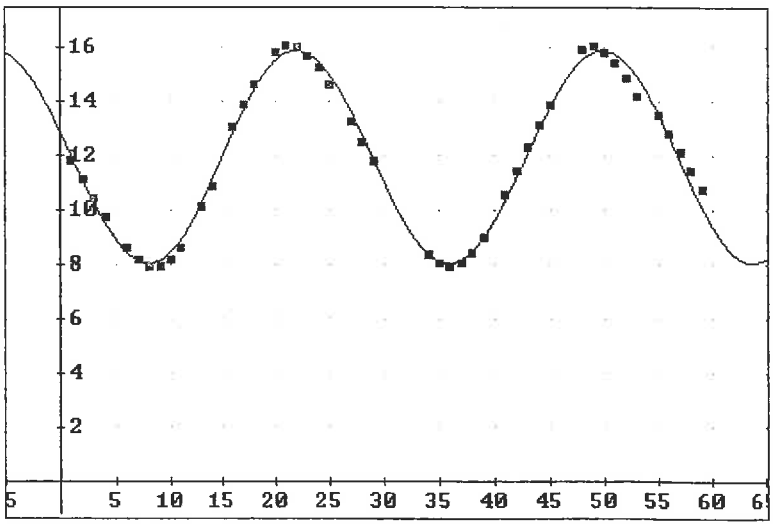
x	y
1	12,32466739
1,5	11,85691181
2	11,3915749
2,5	10,93480397
3	10,49263315
3,5	10,07090372
4	9,675186897
4,5	9,310710277
5	8,982288751
5,5	8,69426091
6	8,450431726
6,5	8,254022287

Suggested reading:

W. Herget, E. Malitte, K. Richter [2002]: Der Mond ist aufgegangen ... In: W. Herget, E. Lehmann (Hrsg.): Exponential- und Winkelfunktionen. Neue Materialien für den Mathematikunterricht mit dem TI-83 / -89 / -92 in der Sekundarstufe I. Hannover: Schroedel Verlag.

It is not difficult to transfer the data from Excel to DERIVE too (DNL#41). In DNL#45 one can find the trigonometric regression `sinreg(data_table)`. *Compare with the TI- and Excel results. Josef*

```
sinreg([days, duration]`)

[ SinReg Line:  y = 3.911550787·SIN(0.2252584765·x + 2.922878099) + 11.96865130 ]
[    SSE:                              3.126026507                                ]
```



PUTZER'S METHOD FOR THE CALCULATION OF $e^{\mathbf{A}t}$

Francisco M. Fernández, Cequinor (Conicet), Facultad de Ciencias Exactas,
Universidad Nacional de La Plata, Calle 47 y 115, Casilla de Correo 962,
1900 La Plata, Argentina
e-mail: framfer@isis.unlp.edu.ar

Systems of coupled first-order ordinary differential equations with constant coefficients appear in many branches of physics and chemistry. It is custommary to write them in compact matrix form as

$$\frac{d\mathbf{X}(t)}{dt} = \mathbf{AX}(t) \tag{1}$$

where $\mathbf{A}$ is an $n \times n$ matrix and $\mathbf{X}(t)$ is a column matrix with n elements that depend on the independent variable t. Since the elements of $\mathbf{A}$ do not depend on t one can express the solution in matrix form as

$$\mathbf{X}(t) = e^{\mathbf{A}t}\mathbf{X}(0) \tag{2}$$

where $\mathbf{X}(0)$ is a column matrix representing the initial conditions. It is clear that the problem of solving the system of equations (1) reduces to the calculation of the $n \times n$ matrix $e^{\mathbf{A}t}$. One way of doing it is to find a similarity transformation of the matrix $\mathbf{A}$ into its diagonal form $\boldsymbol{\lambda}$: $\boldsymbol{\lambda} = \mathbf{U}^{-1}\mathbf{AU}$, and then write $e^{\mathbf{A}t} = \mathbf{U}e^{\boldsymbol{\lambda}t}\mathbf{U}^{-1}$. The calculation of the diagonal matrix $e^{\boldsymbol{\lambda}t}$ is straightforward because its elements are given by $\left(e^{\boldsymbol{\lambda}t}\right)_{ij} = \exp(\lambda_i t)\delta_{ij}$, where $\boldsymbol{\lambda}_{ij} = \lambda_i \delta_{ij}$ are the elements of $\boldsymbol{\lambda}$.

In Derive Newsletter **40**, page 7 Cardia Lopes and Pinto proposed this method to solve kinetic equations. Although this procedure is suitable for an interactive pedagogical demonstration of Derive capability of handling matrices, it may not be the most straightforward algorithm for automation. In what follows we propose an alternative approach.

Putzer's method [1] provides a convenient algorithm for the systematic calculation of $e^{\mathbf{A}t}$, where $\mathbf{A}$ is an $n \times n$ matrix with eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$. For this purpose we need the matrix polynomials

$$\mathbf{P}_0(\mathbf{A}) = \mathbf{I}, \quad \mathbf{P}_k(\mathbf{A}) = \prod_{m=1}^{k}(\mathbf{A} - \lambda_m \mathbf{I}), \quad k = 1,2,\ldots,n \tag{3}$$

and the scalar coefficients $r_j(t)$, $j = 1,2,\ldots,n$, given recursively by the differential equations

$$r_1'(t) = \lambda_1 r_1(t), \quad r_1(0) = 1$$

$$r_{k+1}'(t) = \lambda_{k+1} r_{k+1}(t) + r_k(t), \quad r_{k+1}(0) = 0, \quad k = 1,2,\ldots,n-1. \tag{4}$$

It can be proved that the exponential matrix is given by

$$e^{\mathbf{A}t} = \sum_{k=0}^{n-1} r_{k+1}(t)\mathbf{P}_k(\mathbf{A}). \tag{5}$$

It is clear that the calculation just outlined reduces to obtaining the eigenvalues of **A**, solving straightforward differential equations, and performing simple matrix multiplications. Derive easily carries out all such operations; its only limitation is that it does not give us the repeated eigenvalues of the matrix **A**, or the repeated roots of its characteristic polynomial

$$|\mathbf{A} - \lambda\mathbf{I}| = p(\lambda) = a_0 + a_1\lambda + \ldots + a_n\lambda^n \tag{6}$$

which are needed for the application of Putzer's method. The simplest way of determining the multiplicity of a given eigenvalue is through its substitution into the derivatives of the characteristic polynomial. For example, if the multiplicity of λ_j is k then λ_j is also a root of the first $k-1$ derivatives of the characteristic polynomial.

Once we have $e^{\mathbf{A}t}$, we can solve the more general problem:

$$\frac{d\mathbf{X}(t)}{dt} = \mathbf{AX}(t) + \mathbf{B}(t), \qquad (7)$$

where $\mathbf{B}(t)$ is an $n \times 1$ column matrix. The well-known result

$$\mathbf{X}(t) = e^{\mathbf{A}t}\mathbf{X}(0) + \int_0^t e^{\mathbf{A}(t-s)}\mathbf{B}(s)\, ds \qquad (8)$$

reduces to equation (2) when $\mathbf{B}(t) = \mathbf{0}$.

In what follows we outline a suite of program functions necessary to obtain $e^{\mathbf{A}t}$. The program function `roots` determines the degree `n2` of an input polynomial, finds its roots and place them in the vector `v1`. If `n1=n2-DIM(v1)` is nonzero, then the program finds out the `n1` missing repeated zeros. To this end, the program constructs a vector `vectdif` with the first `n1` derivatives of the characteristic polynomial, substitutes the roots, one by one, into it, and produces a new vector `v2` with all the roots, including the repeated ones.

```
roots(poly,v1,n1,n2,v2,vectdif,i,j,i1):=prog(
                          n2:=poly_degree(poly),
                          v1:=solutions(poly,x),
                          n1:=n2-dim(v1),
                          if(n1=0,return v1),
                          vectdif:=vector(dif(poly,x,i1),i1,1,n1),
                          v2:=[],
                          i:=1,
                          loop(
                               v2:=append(v2,[v1 sub i]),
                               j:=1,
                               loop(
                                    if(subst(vectdif sub j,x,v1 sub i)=0,
                                       v2:=append(v2,[v1 sub i])),
                                    j:+1,
                                    if (j>n1,exit)
                                   ),
                                i:+1,
                                if(i>dim(v1) or dim(v2)=n2,return v2)
                              )
                          )
```

The program appears in the Algebra Window as shown at the right hand side.

You can edit the program with your text editor as shown above and then copy and paste it into the Expression Edit Line of *DERIVE*.

Among the downloadable files you can find putzer.txt, which contains all programs in a structured form, ready for transferring to *DERIVE* (Josef).

```
roots(poly, v1, n1, n2, v2, vectdif, i, j, i1) :=
  Prog
    n2 := POLY_DEGREE(poly)
    v1 := SOLUTIONS(poly, x)
    n1 := n2 - DIM(v1)
    If n1 = 0
       RETURN v1
    vectdif := VECTOR(DIF(poly, x, i1), i1, 1, n1)
    v2 := []
    i := 1
    Loop
      v2 := APPEND(v2, [v1↓i])
      j := 1
      Loop
        If SUBST(vectdif↓j, x, v1↓i) = 0
           v2 := APPEND(v2, [v1↓i])
        j :+ 1
        If j > n1 exit
      i :+ 1
      If i > DIM(v1) ∨ DIM(v2) = n2
         RETURN v2
```

Given a vector of eigenvalues lam the program function `polr` constructs the functions $r_j(t)$ according to equations (4) above and returns them in a vector `r`.

```
polr(lam,j,r,t):=prog(
                    j:=1,
                    r:=[exp(lam sub 1*t)],
                    loop(
                         j:+1,
                         if(j>dim(lam),return r),
                         r:=append(r,[exp(lam sub j*t)*int(r sub(j-1)*
                                exp(-lam sub j*t),t,0,t)])
                        )
                   )
```

The program function `polp` constructs the polynomials $\mathbf{P}_j(\mathbf{A})$ from the vector λ of eigenvalues of the matrix a = **A** according to equation (3).

```
polp(a,lambda,p,j):=prog(
                         p:=[1],
                         j:=1,
                         loop(
                              p:=append(p,[p sub j*(a-lambda sub j)]),
                              j:+1,
                              if(j>dim(a)-1,return p)
                             )
                        )
```

Finally, the program function `eat` constructs the exponential $e^{\mathbf{A}t}$ according to Putzer's algorithm, calling the program functions just commented.

```
eat(a,poly,lambda,r,p):=prog(
                  poly:=charpoly(a,x),
                  lambda:=roots(poly),
                  r:=polr(lambda),
                  p:=polp(a,lambda),
                  sum(r sub(k+1)*p sub(k+1),k,0,dim(lambda)-1)
                        )
```

```
eat(a, poly, λ, r, p) :=
  Prog
    poly := CHARPOLY(a, x)
    λ := roots(poly)
    r := polr(λ)
    p := polp(a, λ)
    Σ(r↓(k + 1)·p↓(k + 1), k, 0, DIM(λ) - 1)
```

As an example of the application of Putzer's method embodied in the suite of program functions just developed, we first consider a simple mechanical problem: the equations of motion for the harmonic oscillator:

$$\frac{d}{dt}\begin{pmatrix} x(t) \\ p(t) \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{m} \\ -m\omega^2 & 0 \end{pmatrix}\begin{pmatrix} x(t) \\ p(t) \end{pmatrix} \tag{9}$$

where , m, ω, $x(t)$ and $p(t)$ are the mass, frequency, coordinate and momentum, respectively. Straightforward application of `eat` to the square matrix with the option `Trigonometry := Expand` gives us the well known result:

```
Trigonometry := Expand
```

$$mm := \begin{bmatrix} 0 & \frac{1}{m} \\ -m\cdot\omega^2 & 0 \end{bmatrix} \tag{10}$$

$$eat(mm)\cdot\begin{bmatrix} x \\ p \end{bmatrix} = \begin{bmatrix} x\cdot COS(\omega\cdot t) + \frac{p\cdot SIN(\omega\cdot t)}{m\cdot\omega} \\ p\cdot COS(\omega\cdot t) - m\cdot\omega\cdot x\cdot SIN(\omega\cdot t) \end{bmatrix} = \begin{pmatrix} x(t) \\ p(t) \end{pmatrix}$$

In chemical kinetics one commonly finds equations of the form (1), where **X**(t) is a column matrix of concentrations of n chemical species. As a particular example we consider the chemical reactions described by Cardia Lopes and Pinto in *DERIVE* Newsletter **40**, page 7 which result in the matrices

$$\begin{aligned} \frac{dC_A}{dt} &= -2C_A + C_B \\ \frac{dC_B}{dt} &= 2C_A - 1.8C_B \\ \frac{dC_C}{dt} &= 0.2C_B \\ \frac{dC_D}{dt} &= 0.6C_B \end{aligned} \quad \rightarrow \quad \mathbf{A} = \begin{pmatrix} -2 & 1 & 0 & 0 \\ 2 & -1.8 & 0 & 0 \\ 0 & 0.2 & 0 & 0 \\ 0 & 0.6 & 0 & 0 \end{pmatrix}, \quad \mathbf{X}(0) = \begin{pmatrix} 50 \\ 5 \\ 0 \\ 0 \end{pmatrix}. \qquad (11)$$

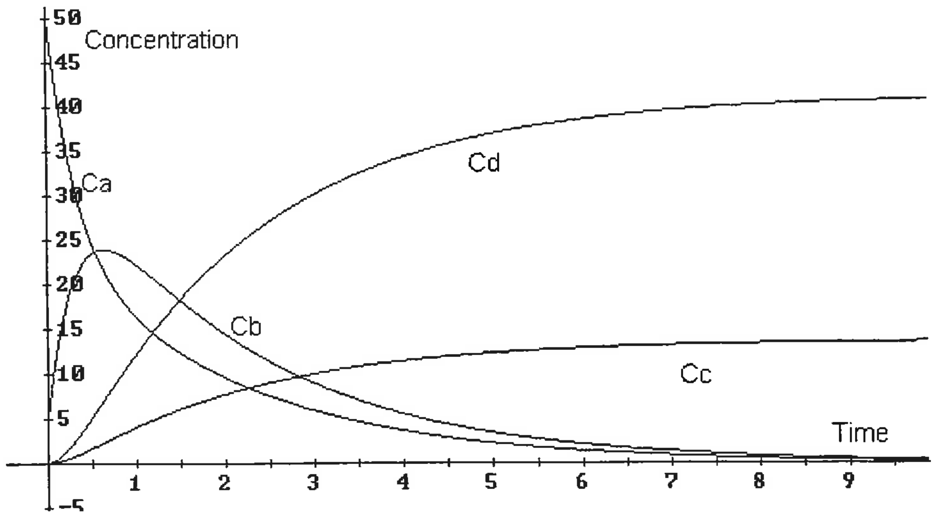
The reader should refer to that Newsletter for more details of the chemical process. The suite of programs described aboved gives us exact expressions for the time-evolution of the concentrations as shown below:

```
ma:=[-2,1,0,0;2,-1.8,0,0;0,0.2,0,0;0,0.6,0,0]
c0:=[50;5;0;0]
eat(ma) . c0
```

Part of the simplified result and the approximated result

$$\left[\begin{array}{l} 25\cdot \hat{e}^{t\cdot(\sqrt{201}/10\ -\ 19/10)} + \\ \hat{e}^{t\cdot(\sqrt{201}/10\ -\ 19/10)}\cdot\left(\frac{5\cdot\sqrt{201}}{2} + \frac{5}{2}\right) + \\ -\hat{e}^{t\cdot(\sqrt{201}/10\ -\ 19/10)}\cdot\left(\frac{5\cdot\sqrt{201}}{8} + \frac{55}{8}\right) + \hat{e}^{-} \\ -\hat{e}^{t\cdot(\sqrt{201}/10\ -\ 19/10)}\cdot\left(\frac{15\cdot\sqrt{201}}{8} + \frac{165}{8}\right) + \hat{e}^{-} \end{array}\right] \qquad \left[\begin{array}{l} 25\cdot\hat{e}^{-0.482\cdot t} + 25\cdot\hat{e}^{-3.31\cdot t} \\ 37.9\cdot\hat{e}^{-0.482\cdot t} - 32.9\cdot\hat{e}^{-3.31\cdot t} \\ -15.7\cdot\hat{e}^{-0.482\cdot t} + 1.98\cdot\hat{e}^{-3.31\cdot t} + 13.7 \\ -47.2\cdot\hat{e}^{-0.482\cdot t} + 5.95\cdot\hat{e}^{-3.31\cdot t} + 41.2 \end{array}\right]$$

Here, *ca*, *cb*, *cc*, and *cd* (the components of the resulting vector) are the concentrations of species a, b, c, and d, respectively, described by Cardia Lopes and Pinto. *DERIVE* also produces the following figure that shows the temporal change of those concentrations:



We believe that Putzer's method provides an elegant algorithm for solving systems of coupled ordinary first-order differential equations with constant coefficients that is suitable for the application of computer algebra systems. The suite of program functions outlined above may facilitate the treatment of several problems of pedagogical and technical interest. The author of this article does not claim to be an expert programmer and is certainly aware that there is ample room for the improvement of those program functions.

REFERENCES

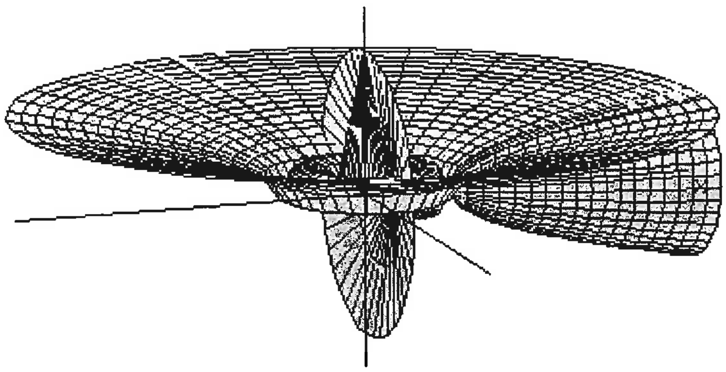
[1] T. M. Apostol, Calculus, Vol II, Second Edition, Blaisdell Publishing Company, Waltham, 1969, pp 205-208

Continued from page 6

DNL

One could understand the question so that Matthieu intends to leave the function as $y = f(x)$ and rotate this function around the y-axis generating just another body. Then use

```
[s·COS(t), s·SIN(t), f(s)]
```



In this 3D-Plot one can see both resulting volumes of revolution.

See another example for demonstrating the difference between the two kinds of hyperboloids.
I remember that pupils always had problems to imagine the two varieties of solids of revolution.
Josef

```
#5:   ff(x) := √(x^2 - 1)
```

hyperboloid of two sheets

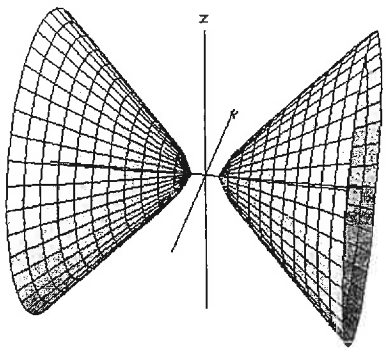
```
#6:   [s, ff(s)·COS(t), ff(s)·SIN(t)]
```

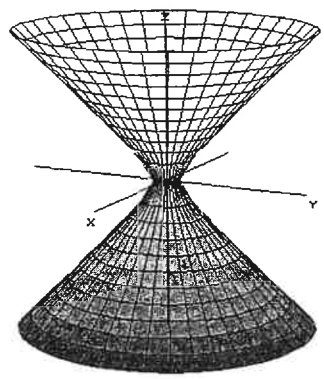
hyperboloid of one sheet

```
#7:   [s·COS(t), s·SIN(t), ff(s)]
```

complete hyperboloid of one sheet

```
#8:   [ s·COS(t)   s·SIN(t)    ff(s) ]
      [ s·COS(t)   s·SIN(t)  - ff(s) ]
```





Tanz der Wallace-Geraden
Wallace-Lines Dancing

Rüdeger Baumann, Celle, Germany

(Today is Thursday, 27 February and this contribution fits excellent to this date because today the famous Vienna Opera Ball takes place. The original contribution is in German and I try to give English summaries. You can download both a German and an English DERIVE file. Josef)

Im Unterricht *Analytische Geometrie* wurde die folgende Aufgabe gestellt:

Gegeben sei ein Dreieck ΔABC und sein Umkreis.

(a) Von einem Punkt P auf dem Umkreis sollen die Lote auf die drei Seiten des Dreiecks gezeichnet und die Lotfußpunkte bestimmt werden. Diese Lotfußpunkte stehen in einer interessanten Beziehung zueinander. In welcher?

(b) Welche Figur ergibt sich, wenn der Punkt P auf dem Umkreis wandert? Untersuchen Sie zunächst ein spezielles Dreieck und versuchen Sie danach, allgemeine Aussagen (Vermutungen, Behauptungen) aufzustellen und nach Möglichkeit zu beweisen.

Given is a triangle ΔABC and its circumcircle.

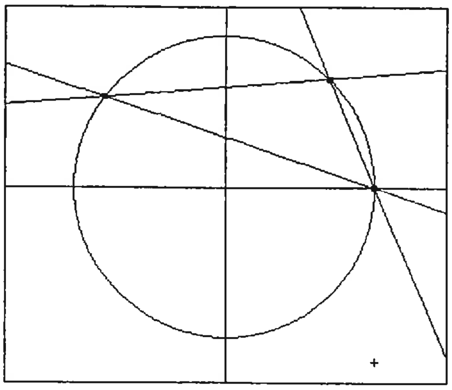
(a) Draw the perpendicular lines from a point P also lying on the circle with respect to the sides of the triangle and mark the resulting pedal points. These points form an interesting relationship. Which one?

(b) Which figure will be created when point P moves on the circle? First investigate a special triangle and then try to set up general statements (conjectures, assertions) and if possible to prove them.

Lösung der Schülerin Evelyn Pieper

(a) Ich beginne mit einem Kreis k, und zwar sei k der Einheitskreis um den Nullpunkt, und lege auf k drei Punkte A, B, C, fest. Damit ist k der Umkreis des Dreiecks ΔABC.

```
CaseMode := Sensitive
Kreis(φ) := [COS(φ), SIN(φ)]
[A := Kreis(2.5), B := Kreis(0), C := Kreis(0.8)]
Dreieck1(X, Y, Z) := [X, Y, Z, X]
g(X, Y) := X + t·(Y - X)
Dreieck2(X, Y, Z) := [g(X, Y), g(X, Z), g(Y, Z)]
Bild1 := [Kreis(φ), Dreieck1(A, B, C), Dreieck2(A, B, C)]
```



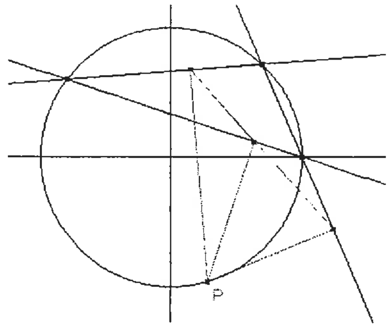
Student Evely Pieper´s solution:

(a) I start with the unit circle k with its center in the origin and fix three points A,B,C on it defining the triangle ΔABC. Hence k is the circumcircle of the triangle. P = (cos(5), sin(5)) is the point from where to draw the perpendicular lines giving the three pedal points as intersections between the sides of the triangle and the perpendicular lines. The formula for the pedal point is well known from earlier lessons (math teacher Academic Councilor Morçeau).

Der Punkt, von dem aus die Senkrechten auf die drei Seiten gebildet werden sollen, sei $P = (\cos(5), \sin(5))$. Die Formel für den Lotfußpunkt ist im Unterricht (Mathematiklehrer Akadem. Rat Morçeau) erarbeitet worden.

```
P := Kreis(5)

Lotfußpunkt(Q, X, Y) := ((X - Y)·(Q - Y)·X - (X - Y)·(Q - X)·Y) / (X - Y)^2

L1 := Lotfußpunkt(P, B, C)
L2 := Lotfußpunkt(P, A, C)
L3 := Lotfußpunkt(P, A, B)
Strecken := [[P, L1], [P, L2], [P, L3], [[L1, L2, L3]]]
Bild2 := APPEND(Bild1, Strecken)
```

$$\text{Lotfußpunkt}(Q, X, Y) := \frac{(X - Y)\cdot(Q - Y)\cdot X - (X - Y)\cdot(Q - X)\cdot Y}{(X - Y)^2}$$



Es sieht so aus, als ob die drei Lotfußpunkte auf einer Geraden liegen. Ich habe dies an mehreren Lagen von *P* überprüft; den allgemeinen Beweis möchte ich noch etwas zurückstellen.

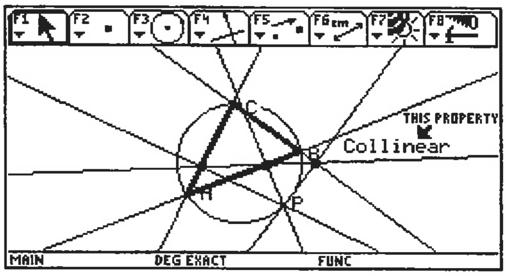
(b) Unter der Annahme, dass meine Behauptung zutrifft, definiere ich die „Lotfußpunktgerade" *LFG* als Verbindungsgerade der drei Lotfußpunkte, dann lasse ich den Punkt *P* auf dem Kreis wandern und registriere, wie sich die zugehörigen Lotfußpunktgeraden verhalten.

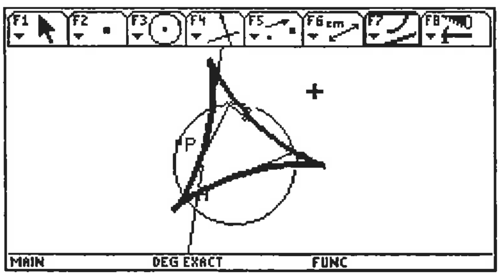
It seems to be that the pedal points L1, L2 and L3 are lying on one line. I checked this using several positions of *P*. I´ll postpone the prove at the moment.

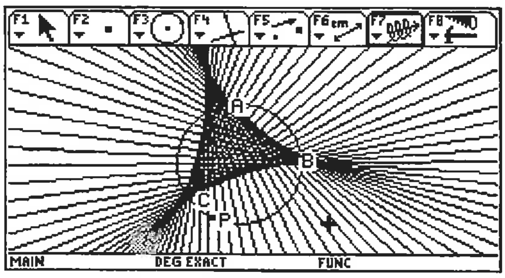
(b) Assuming that my conjecture is true I define the "pedal point line" *LFG* as the connecting line of two pedal points, then I create a family of these lines moving point *P* on the circle and inspect the behaviour of the respective *LFG*s.

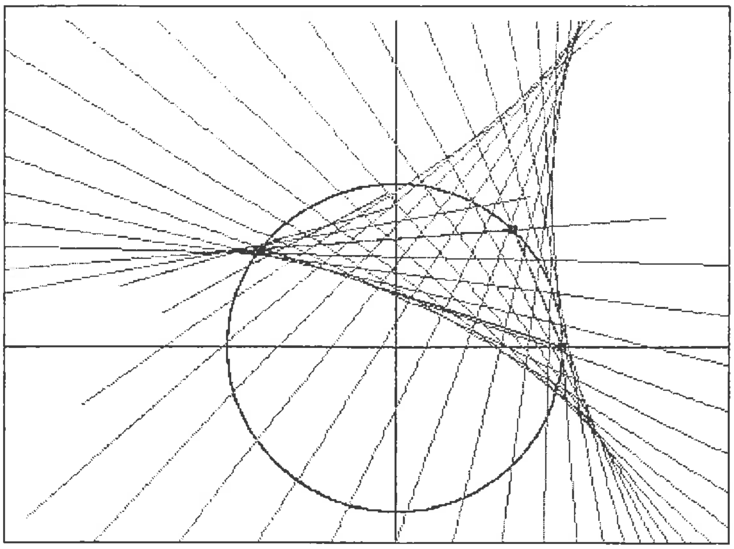
```
LFG(Q, X, Y, Z) := g(Lotfußpunkt(Q, X, Y), Lotfußpunkt(Q, X, Z))

Geradenschar(n) := VECTOR(LFG(Kreis(φ), A, B, C), φ, 0, 2·π, 2·π/n)

[Kreis(φ), Dreieck1(A, B, C), Geradenschar(30)]
```

In the meanwhile naughty pupil Pepi – Viennese form for Josef – sitting in the last row, was working on his TI and tried to prove the conjecture in his way. By the way he animated *P* and let the device find the locus (envelope) of the lines. Josef









Nun zum allgemeinen Beweis:

Ich muss von „allgemeinen Punkten *AA,BB,CC* und *PP* ausgehen. Dies versuchte ich zuerst durch die Definitionen `[AA:=Kreis(φ1), BB:=Kreis(φ2),CC:=Kreis(φ3)]` zu machen, um dann die Rechnung wie oben durchzuziehen. Aber beim Aufruf `CROSS(LL1-LL2,LL1-LL3)=` zur Prüfung auf Kollinearität versank *DERIVE* in tiefes Nachdenken, aus dem es nur ein Klick auf den Knopf *Abbrechen* erlöste.

Danach habe ich es mit kartesischen Koordinaten probiert, und zwar habe ich als *AA* den Nullpunkt angenommen und $BB = (c,0)$ auf die x-Achse gelegt. Dann wurde der Umkreis konstruiert; die Formeln für den Umkreismittelpunkt *UM* und Umkreisradius *UR* sind aus dem Unterricht bekannt.

The generalized proof: I have to start with „generalized" points *AA, BB, CC* and PP. At first I tried defining: `[AA:=Kreis(φ1), BB:=Kreis(φ2), CC:=Kreis(φ3)]` in order to perform the calculation as done before. But calling `CROSS(LL1-LL2,LL1-LL3)=` for proving collinearity *DERIVE* sank into deep thought from which it only could be released by pressing the *Abort* button.

```
[[AA := [0, 0]], [BB := [c, 0], CC := [q, h]]]
```

$$\mathrm{Lot}(v) := [-v_2, v_1]$$

$$\mathrm{UM}(X, Y, Z) := \frac{(Y^2 - Z^2)\cdot \mathrm{Lot}(X) + (Z^2 - X^2)\cdot \mathrm{Lot}(Y) + (X^2 - Y^2)\cdot \mathrm{Lot}(Z)}{2\cdot \mathrm{CROSS}(X - Y, X - Z)}$$

$$\mathrm{UR}(X, Y, Z) := \frac{|X - Y|\cdot|Y - Z|\cdot|Z - X|}{2\cdot \mathrm{CROSS}(X - Y, X - Z)}$$

Um den allgemeinen Punkt *PP* auf dem Umkreis zu bekommen, löse ich die Gleichung `Kreis2(UM(AA,BB,CC), UR(AA,BB,CC))` nach y auf und setze dann $PP = (x,y)$. Der Aufruf `CROSS(LL1-LL2,LL1-LL3)` liefert dieses Mal, wie erwartet, null. Also sind die Punkte *LL1*, *LL2*, *LL3* kollinear.

For obtaining the general point *PP* on the circumcircle I solve equation `Kreis2(UM(AA,BB,CC), UR(AA,BB,CC))` for *y* and set $PP = (x,y)$. Calling again `CROSS(LL1-LL2,LL1-LL3)` delivers now – as expected -, zero. So I could prove that *LL1, LL2, LL3* are collinear.

$$\mathrm{Kreis2}(M, r) := ([x, y] - M)^2 = r^2$$

$$PP := [x, (\mathrm{SOLUTIONS}(\mathrm{Kreis2}(\mathrm{UM}(AA, BB, CC), \mathrm{UR}(AA, BB, CC)), y))_1]$$

```
LL1 := Lotfußpunkt(PP, BB, CC)
LL2 := Lotfußpunkt(PP, AA, CC)
LL3 := Lotfußpunkt(PP, AA, BB)
CROSS(LL1 - LL2, LL1 - LL3) = 0
```

Schüler(in)fragen an die DNL-Leser:

(1) Was ist das für ein merkwürdiges Gebilde, das von den Lotfußpunktgeraden eingehüllt wird?

(2) Die Punkte *A, B, C, P* bilden ein Sehnenviereck. Man könnte nun auch *A* die Rolle von *P* spielen lassen und *P* zum Dreieckspunkt degradieren. Das liefert eine neue Lotfußpunktgerade. Dasselbe für *B* und *C*. Wie liegen diese vier Geraden? Schneiden sie sich vielleicht gar in einem Punkt?

Lehrerfrage: Wie ist der von Evelyn geführte Beweis aus didaktischer Sicht zu beurteilen? (Vergleich mit klassischen bzw. vektorgeometrischen Beweisen.) **Haben Schüler(innen), die einen solchen Beweis führen können, die zugehörige Geometrie verstanden?** Ist es aus didaktischer Sicht wünschenswert, die Schüler solche Beweisführungen zu lehren?

Evelyn´s questions for the DNL-community:

(1) What is the strange figure, envelopped by the pedal point lines?

(2) Points *A, B, C* and *P* are forming an inscribed quadrilateral. One could exchange the roles of *A* and *P*. *P* will become a point of triangle *BCP* and *A* moves along the circle. This results in a new pedal point line. Do the same with *B* and *C*. Could it be that the four lines intersect in one point?

Teacher´s question: How to comment Evelyn´s proof from a didactical point of view? (Compare with classic, eg. geometrical proofs). **Have students, who are able to perform proofs like the one presented above understood the underlying geometry?** Is it desirable from the didactical point of view to teach proofs like this?

A Bug in PADE.MTH?

Rick Nungester nungester@mail.com

`http://mathworld.wolfram.com/PadeApproximant.html` describes Pade Approximants, and *DERIVE* includes function `PADE(y,x,x0,n,d)` in file APPROX.MTH. See the website or *DERIVE* Help for more information on Pade. The website shows that

`PADE(EXP(x),x,0,3,2)=(60+36x+9x^2+x^3)/(60-24x+3x^2).`

But *DERIVE* gives

`PADE(EXP(x),x,0,3,2)=(20x^3[3/5,3/20]sub 3+3x^2+12x+20)/(x^2-8x+20).`

What is the "`[3/5,3/20] SUB 3`" notation in the *DERIVE* output, and why does it apparently equal 1/60?

While **Vladimir Bondarenko** from the Symbolic Testing Group confirmed this to be a bug in *DERIVE* **Johann Wiesenbauer** did more:

I thought myself for some time that the current PADE in Derive is bug-infested, but technically speaking it is a feature rather than a bug, because they say in the online-help referring to PADE(y, x, x0, n, d) explicitly

`"You must have either n = d or n = d-1."`

Hence, the case n = 3 and d = 2, which you tried out, is simply not allowed in *DERIVE*. (God only knows why, but that's the way it is!)

Until there is an "official" change in this regard, I recommend using my following "translation" into Derive 5.06 (with standard settings), which doesn't have that restriction as to the choice of m and n and is also more than two times faster:

$$\text{pade}(\text{EXP}(x),\ x,\ 0,\ 3,\ 2) = \frac{x^3 + 9\cdot x^2 + 36\cdot x + 60}{3\cdot(x^2 - 8\cdot x + 20)}$$

File PADE.MTH is among the downloadable files. Many thanks Johann for improving this function.

Detection of Periods –
- An Exercise in Functional Programming

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

1 - Introduction

Rüdeger Baumann gave in DNL #47 a generalization of a problem given by Milton Lesmes Acosta. It appeared that repeated iteration of a particular function which is defined on the set of natural numbers runs into a finite cycle. Rüdeger asked the reader for a "period-recognizing-routine". We shall see that a different programming style which is called functional programming will produce simpler code and a solution to the period-recognizing-problem.

Functional programming simply means: Apply functions instead of iteration or recursion. We start with a familiar example: The function

$$collatz(n) := \begin{cases} n/2 & \text{if } n \text{ is even} \\ 3n+1 & \text{if } n \text{ is odd} \end{cases}$$

is supposed but not yet proved to end in the cycle {4,2,1} for every natural number n.

```
     collatz(n) :=
       If EVEN?(n)
#1:       n/2
          3·n + 1
#2: ITERATES(collatz(k), k, 17, 20)
#3: [17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, 4, 2, 1, 4, 2, 1, 4, 2]
```

If we omit the fourth argument of the `ITERATES`-function then the iteration stops if the first number appears for a second time, i.e. if a second cycle starts.

```
#4: ITERATES(collatz(k), k, 17)
#5: [17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, 4]
```

2 - Milton's riddle #3

Let us try to define a function that (1) separates the digits of a natural number, then (2) raises the numbers obtained in this way to the k-th power, and (3) finally adds these powers.

The function `NAMES_TO_CODES` separates the digits and returns a list of their ASCII-codes. Subtraction of 48 in each case gives a list of the digits.

```
#6: MAP_LIST(k - 48, k, NAME_TO_CODES(123))
#7: [1, 2, 3]
```

The following function contains all three steps listed above:

```
#8:  g(n, m) := Σ(k^m , k, MAP_LIST(k - 48, k, NAME_TO_CODES(n)))
#9:  g(153, 2)
#10: 35
```

The next two functions compute the sequence of iteration from the starting number to the first repeating of a number resp. the period itself.

```
#11: milton(num, expo) := ITERATES(g(k, expo), k, num)
#12: milton(153, 2)
#13: [153, 35, 34, 25, 29, 85, 89, 145, 42, 20, 4, 16, 37, 58, 89]
#14: milton_period(n, m) := milton(FIRST(REVERSE(milton(n, m))), m)
#15: milton_period(153, 2)
#16: [89, 145, 42, 20, 4, 16, 37, 58, 89]
```

So far the period-recognition works successfully. The next example shows that Rüdeger missed the first occurrence of the number 13139 in his second example:

```
#17: milton(153, 4)
#18: [153, 707, 4802, 4368, 5729, 9603, 7938, 13139, 6725, 4338, 4514,
     1138, 4179, 9219, 13139]
```

3 - Milton's riddle #2

Let us first note that REVERSE reverts the order of digits of any positive integer.

```
#19: REVERSE(123)
#20: 321
```

Thus we can skip the time consuming recursion introduced by Rüdeger in DNL #47, p.39. Our function palindrome_process reads like this:

```
#21: palindrome_process(n, m) :=
     ITERATES(IF(k = REVERSE(k), k, k + REVERSE(k)), k, n, m)
#22: palindrome_process(19)
#23: [19, 110, 121, 121]
#24: palindrome_process(6999)
#25: [6999, 16995, 76956, 142923, 472164, 933438, 1767777, 9545448,
      17990907, 88900878, 176701866, 844809537, 1580717985, 7477888836,
      13866776583, 52434543414, 93869086839, 187737183678,
      1064118921459, 10605417036060, 16668488486661, 16668488486661]
```

We have dropped the second argument so ITERATES stops at the first repetition of a number, i.e. when the result is a palindrome number.

Our last example shows that we cannot always drop the second argument since even after half an hour of computation no palindrome appears:

```
#26:  palindrome_process(196, 30)

#27:  [196, 887, 1675, 7436, 13783, 52514, 94039, 187088, 1067869,
        10755470, 18211171, 35322452, 60744805, 111589511, 227574622,
        454050344, 897100798, 1794102596, 8746117567, 16403234045,
        70446464506, 130992928913, 450822227944, 900544455998,
        1800098901007, 8801197801088, 17602285712176, 84724043932847,
        159547977975595, 755127757721546, 1400255515443103]
```

These examples show how functional programming can achieve the same results with less programming efforts.

A letter from Heinz-Rainer –and its consequences: the "Glider".

<u>From Heinz-Rainer Geyer:</u>

Hi all, hi Josef (Böhm),

I like the sketchpad idea in last DNL 48,p 32. I tried to show moving tangents on some curves.
It all worked fine in school with Version 5.04 for example:

```
#6:   f(x):= x^2-5x+7
#7:   tang(x,a):= f'(x)*(x-a)+f(a)
#8:   tang(x, hCross)
```

But when I repeated it at home with Version 5.06 I got the message:
Derive can't plot that expression.

I simplified `tang(x,hCross)` and received:

$$\#9:\quad \left(x - \frac{69}{22}\right)\cdot \lim_{69/22\to 69/22} \mathrm{DIF}\left(\frac{559}{484}, \frac{69}{22}\right) + \frac{559}{484}$$

What about that?
What has changed between 5.04 to 5.06?
To get a result you have to simplify the function tang(x,a) first and make a new definition

$$\#14:\quad \mathrm{tang2}(x0) := x\cdot(2\cdot x0 - 5) - x0^2 + 7$$

Has anybody an idea?
Rainer

<u>Albert D. Rich's Answer</u>

Hello Rainer,
Thanks for pointing out this change in the way version 5.06 of Derive simplifies derivatives of user-defined functions. The expression simplifier was modified for version 5.06 to more efficiently pass arguments to user-defined functions. Unfortunately, this had the side-effect that you reported. For the next version of Derive I have modified the system to efficiently pass arguments and return the expected results.
Aloha,
Albert D. Rich

Josef´s Answer

Dear all (and special greetings to my friend Heinz-Rainer).

As I can see fortunately Rainer could save his PC of the daring floods in his home.
I appreciate your application of this "hidden" feature of DERIVE and I am sure that there are many other ways for a meaningful use of (hcross,vcross).

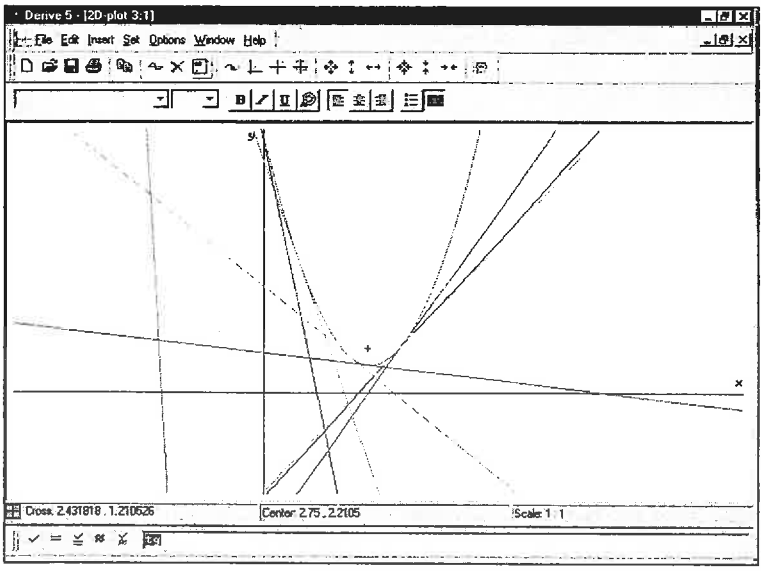
To your problem:

You can overcome the difficulty by first deriving and then evaluating the derivative. (It is not as simple as on the TI-92 - and I must admit that I prefer the more mathematical way as it is performed by DERIVE.

```
f(x) := x^2 - 5·x + 7

tang(x, a) := LIM(DIF(f(x), x), x, a)*(x - a) + f(a)

tang(x, hCross)
```

This works properly. You might prefer a shorter version:

```
tg := LIM(DIF(f(x), x), x, hCross)·(x - hCross) + f(hCross)
```

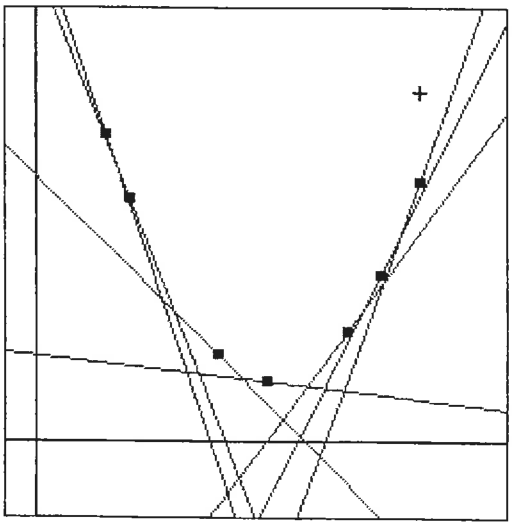


Each call of `tg` in the 2D Plot Window returns a tangent of function f(x) defined above. You only have to move the cursor cross in any position and the horizontal coordinate of the cross places the respective tangent.

As you can see `tg` is activated - now I move the cursor (mouse or arrow-keys) and then press the plot button.

The tangent should appear.
The next function plots the osculating point together with the tangent:

```
tgpt := [tg, [hCross, f(hCross)]]
```



(You don´t need simplifying this, just plot!)
returns not only the tangent but also the osculating point (set point size LARGE). You could go on and add the slope-triangle, the respective point to find point by point the first derivative as a locus....

I attach the dfw-file containing the graphs.

Once more many thanks for pointing at this interesting possibility which helps again visualizing mathematics.

Best regards
Josef

Johann Wiesenbauer´s comments:

Hi Josef,

Let me just add some remarks of my own on Rainer's problem. Note that DERIVE won't accept the third line unless you open the 2D-plot window and set the mouse cursor somewhere!

```
f(x) := x^2 - 5x + 7
tang(x, a) := f'(a)(x - a) + f(a)
tang(x, hCross)
```

This will not work though, as there will be the notorious message "Sorry, the highlighted expression cannot be plotted!" But there is a simple and straight-forward remedy, which is: "Simplify the second line!". Obviously, Rainer's pupils did this at school and he forgot (?) to do this at home.
In the same way another more serious problem can be fixed. Let's define the following function

```
f1(x):=dif(fx,x)
```

I know this is not necessary as all derivatives f'(x),f''(x), etc. are available in this form anyway. (Believe it or not, before reading Rainer's mail I had not been aware of this fact!) Anyway,...If we try to find the value of f1(x) at x = 2 we get

```
f1(2) = DIF(1, 2)
```

It should be clear what went wrong here: DERIVE substitutes x = 2 before (!) forming the first derivative. (Sadly enough, this deficiency was introduced in the most recent version 5.06!) Again, simplifying the definition of f1(x) before using it will help. (This should be done in this case anyway and that's why only novices will ever encounter this error!) On the other hand, the simplification of f'(2) doesn't cause any problems at all, which is really strange!

Hope these remarks helped to throw more light upon the matter.

Cheers,
Johann

Again Heinz-Rainer:

Hi Johann and Josef,

thanks for your remarks:

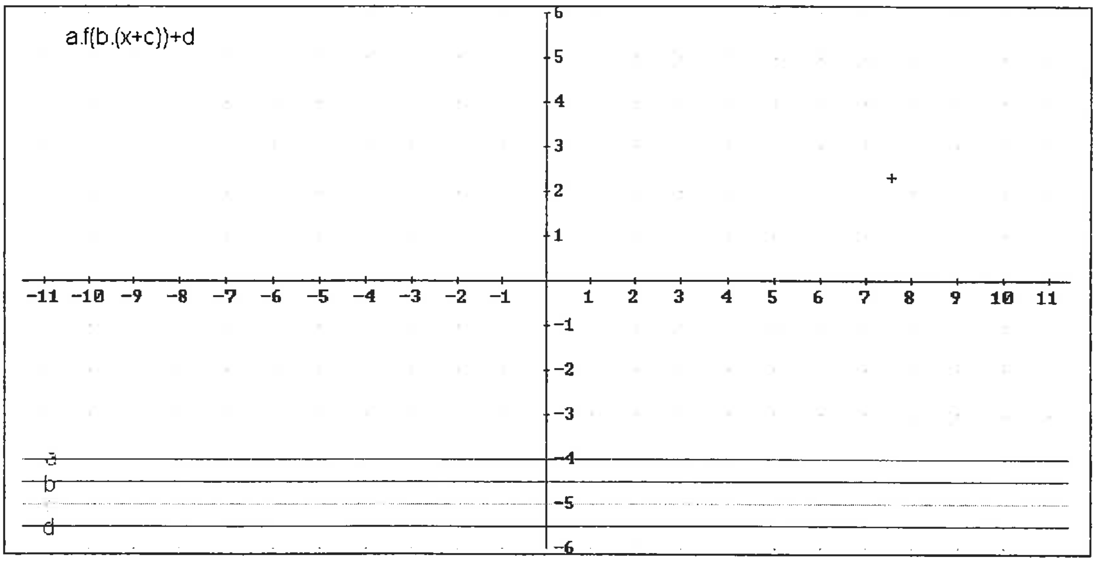
Of course one should have an open 2D-plot to get cursor (cross) coordinates. But fact is, that in v5.04 you don´t have to simplify first but in v5.06 - I skipped V5.05 -, so I can't tell when this behaviour changed.
AND WHY? (Ask Theresa and Albert)
Johann made the problem clear: by substituting `a = hCross` BEFORE simplifying you get the result I copied in line #9.
Ok, Johann is right again when he mentioned that simplifcation BEFORE using should be done anyway. BUT in didactic matters in the beginning I won´t my students have a look on the simplified form. Later on when they are used to work with DERIVATES its ok.

Let me mention another visualisation aspect:
you can have the function-plot invisible in the background color and even the cursor invisible while in trace mode you plot the tangents and ask the students for the type of the function. But don´t forget to take another plotcolor for the lines.

Josef's "Glider":

Heinz-Rainer's mail reanimated an old idea, which slept deep in my mind: Make your glide controllers in *DERIVE*: So prepare a 2D-Window with four horizontal lines at the bottom with labels a, b, c and d. My idea was to trace along the lines and take the x-values of the trace-points as parameters for transforming any funktion f(x) to $a \cdot f(b \cdot (x + c) + d$.

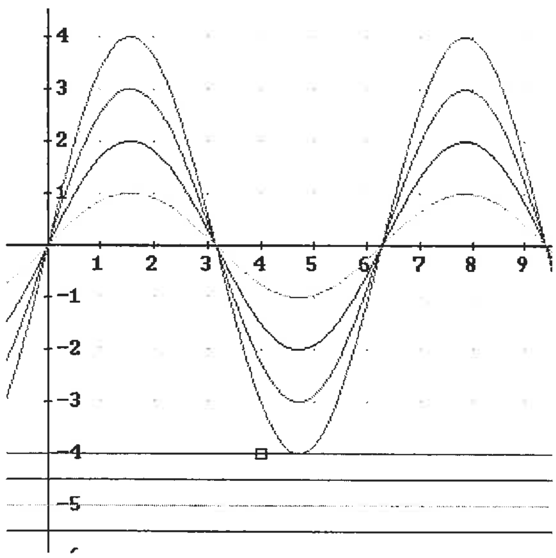
The plot window is embedded into the Algebra Window




```
#1:   [y = -5.5, y = -5, y = -4.5, y = -4]
#2:   [a := 0, b := 0, c := 0, d := 0]
      glider(u) :=
        Prog
          If vCross = -5.5
             d := hCross
          If vCross = -5
#3:          c := hCross
          If vCross = -4.5
             b := hCross
          If vCross = -4
             a := hCross
          a·SUBST(u, x, b·(x + c)) + d
```

The parameter variables a – d must be defined as global variables. Their basic values are 0, and can be changed if necessary.

Double click on the embedded prepared 2D-Window, edit or activate glider(sin(x)) and plot. Switch to the Trace Mode and "hike" on the colored horizontal lines. Each new PLOT shows the influence of the changed respective parameter.



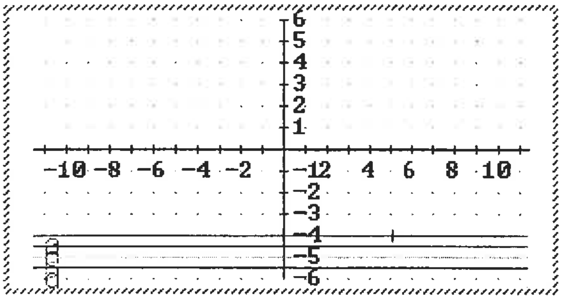
In order to investigate another function one has to
1) edit `glider(.....)` and

2) if necessary initialise parameters a,b,c,d because as global variables they would remain the same.

The figure shows the `glider(sin(x))`, with intial values a = b = 1 and c = d = 0. Then I moved the trace box along the a-line from 1 to 4.

I was not completely satisfied, because I wanted to have a more general glider for parameter variation.

```
#8:    [f(x, a, b, c, d) :=, a := 0, b := 0, c := 0, d := 0]

       gglider(dummy) :=
         Prog
           If vCross = -5.5
              d := hCross
           If vCross = -5
#9:           c := hCross
           If vCross = -4.5
              b := hCross
           If vCross = -4
              a := hCross
           RETURN f(x, a, b, c, d)
```



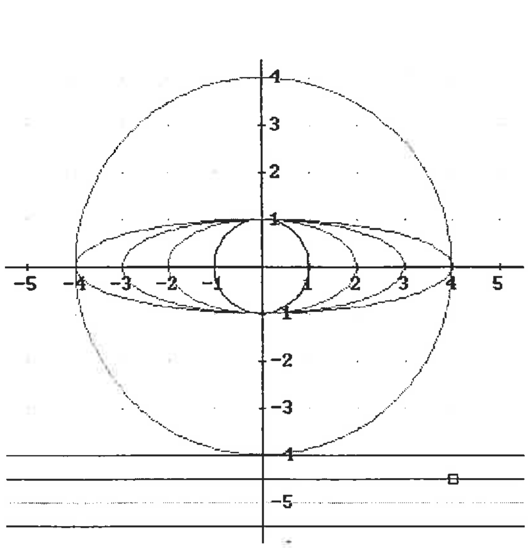
```
#10:   [f(x, a, b) := x^2/a^2 + y^2/b^2 = 1, a := 1, b := 1]
```

Now let me investigate the influence of a and b on the form of the ellipse: I activate the provided embedded 2D-Plot Window, write `gglider()` into the edit line and plot .... a circle. Then I trace along the a-line from 1 to 4 and plot one ellipse after the other one, observing the respective change of the form. In position $a = 4$ I move the cursor down, place it on the b-line and plot now the ellipse with $a = b = 4$, receiving again a circle .....

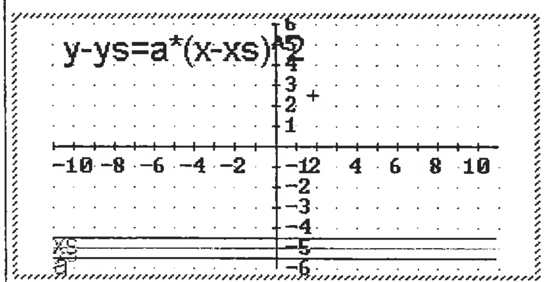
Today I received a mail from my colleague Gerti Feder, Vienna. She wrote:

Dear colleague,
I was very glad about your glider. It fits just now perfect into my teaching concepts! But I adapted it for my 2nd forms. It is one for glider for linear functions and another one for quadratic functions.

Wonderful, that is exact the DNL´s intention: not only to provide ready made tools, but also to inspire for own ideas. Bravo, Gerti Feder.



```
#1:    [y = -5.5, y = -5, y = -4.5]

#2:    [xs := 0, ys := 0, a := 0]

       glider(u) :=
         Prog
           If vCross = -5.5
              a := hCross
#3:           If vCross = -5
                 ys := hCross
                 If vCross = -4.5
                    xs := hCross
           SUBST(u, x, a·(x - xs)^2 + ys)
```




```
#4:    [xs := 0, ys := 0, a := 0]

#5:    glider(x)
```

A Symbolic TVM-Solver for the Symbolic TIs

Josef Böhm

The `FINANCE`-Application `TVM-Solver` for the TIs is a very valueable tool for solving financial mathematics problems. This application was developed for the TI-83 calculators and unfortunately it was not adjusted for the CAS-capabilities. So we face some unnecessary deficiencies:

- It is not possible to enter fractions for PpY (number of payments per year), which is necessary if the payment periods are greater than one year,
- It is not possible to enter ∞ for the number of payments for calculating an everlasting annuity,
- For applying a discount rate instead of an interest rate it is necessary (and nasty) to transform the discount rate into the equivalent interest rate,
- The TVM-Solver automatically equals the number of payments per year PpY and the number of compounding periods per year CpY and one has to adjust the values,
- It is impossible to enter a function or variable at any entry of the table.

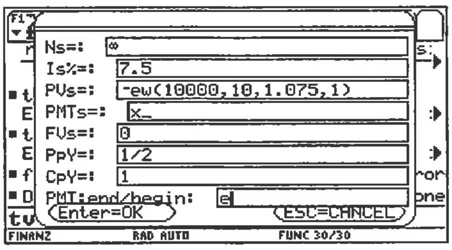
As many of you will know I was teacher at a College for Business Administration and our mathematics curriculum covers a big part of financial mathematics. I felt challenged to provide an appropriate tool for my students and colleagues which are using the CAS-TIs to overcome the deficiencies mentioned above. Several years ago I wrote a program `rente()`, which did most of the TVM's work, but has another "Layout" and presented the result on the Prgm IO screen.

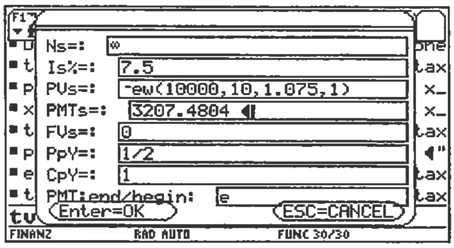
I hope that my `tvms()` – (Time-Value-Money-Symbolically) Solver overcomes all the problems.

The first example will demonstrate four of the five points mentioned above.

Example 1:
A principal of 10000€ earns i = 7.5% annually interests for 10 years. This amount is used to pay a certain annuity for "everlasting times" in two years periods. What can be paid at the end of each period?

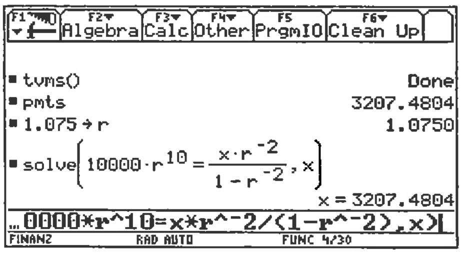
Let's assume that we have developed earlier a function `ew(..)` for finding the Future Value (= *Endwert* in German) and we will use this function in combination with the TVMS-Solver:





The unknown must be expressed as `x_`.

If necessary you could immediately proceed by first deleting the "◀"character followed by entering the new data or abort the calculation by pressing ESC. On the home screen you can recall your result (as `pmts` here).



The control calculation by traditional means:

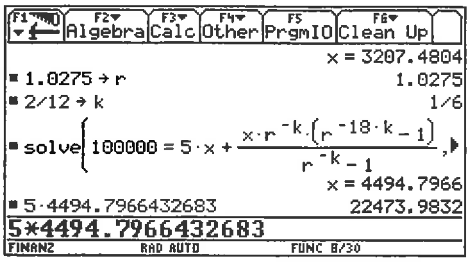
We can do more using `tvms()`. See my second example:

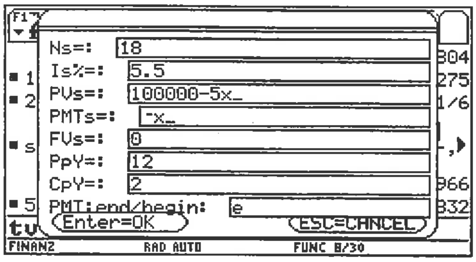
Example 2:

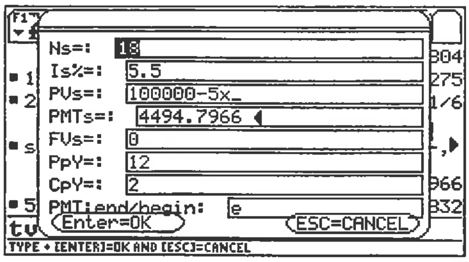
A debt of 100000 USD must be pais off by 18 monthly payments due at the end of the period and an additional payment of 5 times of the monthly payments, which must be paid immediately. What is the amount of this immediately due payment?

According to the ISMA – Conventions (International Securities Market Association) the traditional approach for solving this problem is as follows:

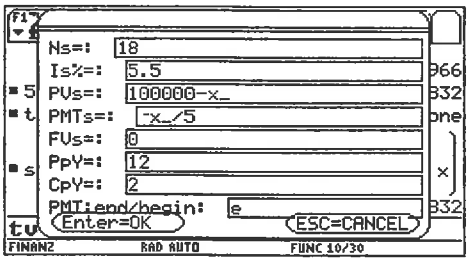
$$100000 = 5x + \frac{x \cdot r^{-1/6}\left(r^{(-1/6)\cdot 18} - 1\right)}{r^{-1/6} - 1}; \qquad r = 1 + \frac{0.055}{2} = 1.0275$$

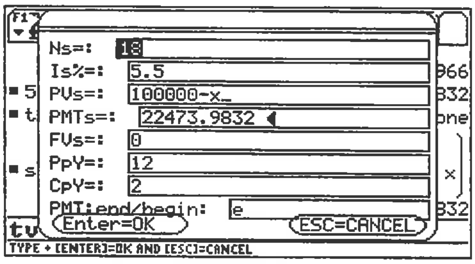




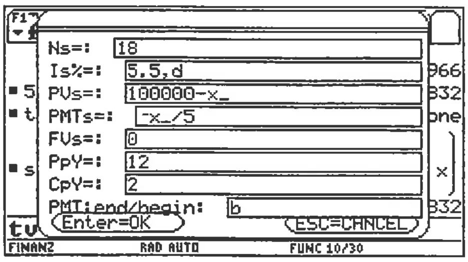


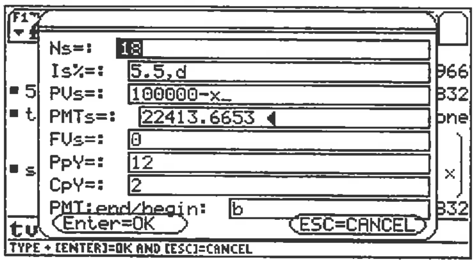
Entering the data and pressing ENTER gives the payment (*4494.80 USD*).
But one could also access directly the immediately due payment (*22473.98 USD*):





Finally I switch from an interest rate to a discount rate and from payments due at the end of a month to payments due at the begin of the monthly periods. (*The initial payment is 22413.67 USD.*)

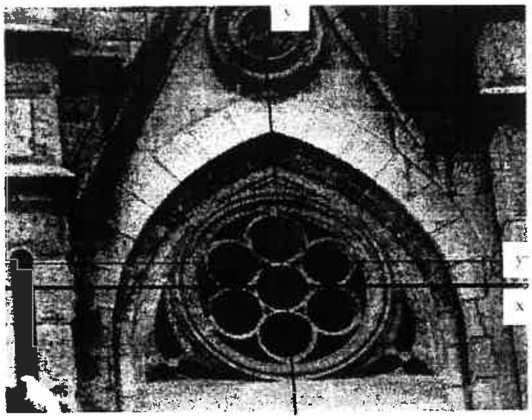


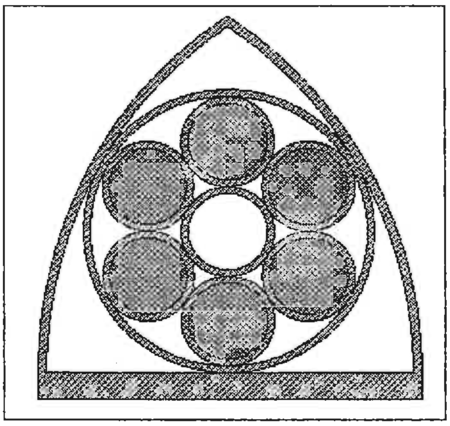


You can download two complete papers on Financial Mathematics (in German only) from the ACDCA/T³-website. One chapter of Financial Mathematics 2 deals with the "machine" behind the TVM-Solver in order to make this Black Box to a White Box for the user.

I hope to have a TVMS-Solver for *DERIVE* ready for the next *Newsletter*.

In Christmas holidays I met my friend Eberhard Lehmann in Upper Austria. We spent together Old Years´ Eve – and we talked about meaningful and motivating applications of CAS in school. He left a challenge for the New Year, a photo of a gothic window which he had taken in Mallorca. He asked me to realize the wonderful pattern with *DERIVE* [1].

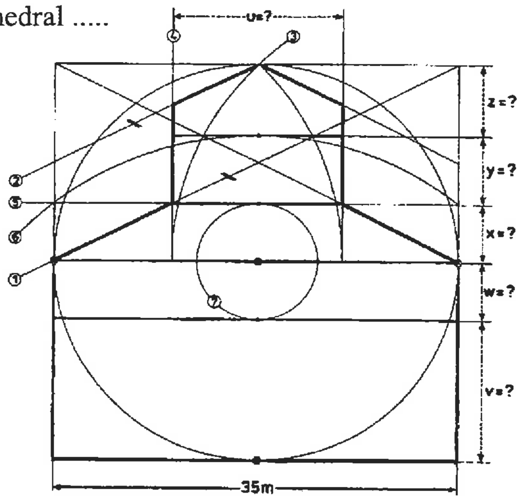


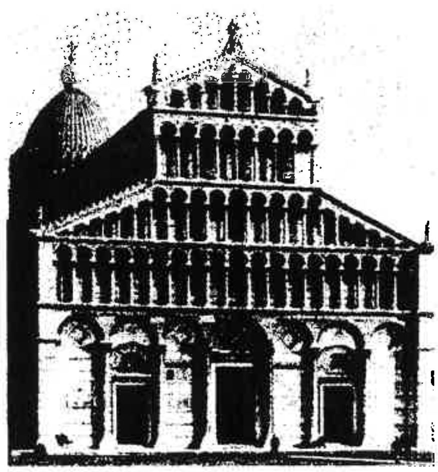


You must imagine the colors of the shadings. Send your students equipped with a camera to any churches and give them similar tasks. On the next page are some TI-screens with other patterns.

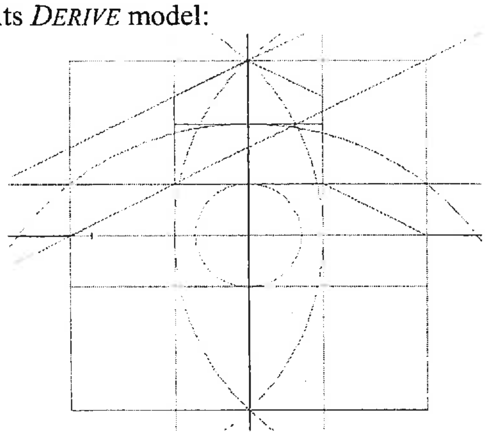
Peter Lüke-Rosendal sent another task. In a textbook he found a picture of the Cathedral of Pisa together with a sketch of its front [2]. Make a model (DERIVE, TI, any other tool ...).

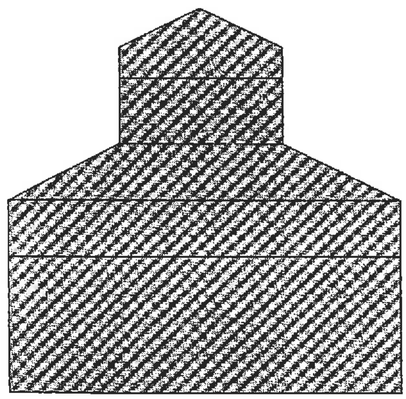
The Cathedral .....

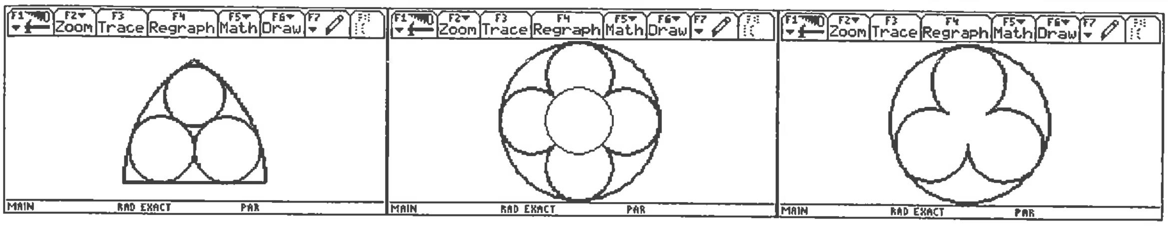




.... and its *DERIVE* model:







Some realisations of Gothic Church Windows on the TI-Graph screen.

References:

[1] *Eberhard Lehmann*, Mathematikunterricht mit Parametern, Schroedel 2002

[2] Anschauliche Geometrie" Bd. 3; Ehrenwirth Verlag

[3] *Günther Schmidt,* Gotische Maßwerkfenster im Geometrieunterricht, MU, Jg 41, Heft 3

[4] *Mabel Sykes*, Source Book of Problems for Geometry, Dale Seymour

Preparing Plot Windows in *DERIVE*

Dieter Wickmann

Dear Josef,

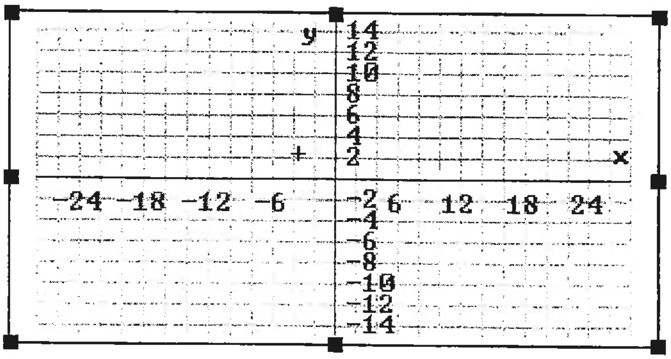
I am still working on my Bayes-package and would like to have running my programs independent on the current state of *Derive* (i.e. on the savings of the settings for the 2D Plot Window). This is not bad concerning the settings for the Algebra Window, because I can write the definitions into my mth-file (eg. `Precision := Approximate`). It seems to be that this impossible for the Plot Windows. Is there a way to define the state of a plot window, independent of the current state of the system?

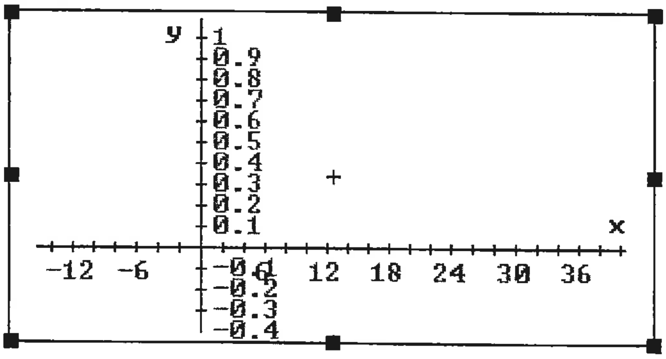
Lieber Herr Wickmann, Dear Mr. Wickmann.

your wish making the most important parameters of the plot windows "programmable" is on the top of my personal wish list (and I do hope that Theresa Shelby and Albert Rich will consider this wish for DERIVE 6. Maybe that your mail is one more step into this direction).

What I can do is offering another possiblity, which I successfully have used at occasions.

An **empty** Plot Window – containing all desired settings – is **embedded (not copied)** into the Algebra Window. A double click on this embedded window opens immediately a plot window showing all predefined settings independent of previous system settings. The files might become larger, but compressing them brings them down to a reasonable size. I attach a file plots.dfw containing two different empty 2D plot windows.





Make a try.

I hope that this is what you are expecting.

Best regards,
Josef

Interesting News from Hawaii

Hi Josef,

You recently inquired about customization of the Derive user interface. I just discovered a feature in Derive 5 and thought I'd pass it on in case you were also unaware of its existence.

Toolbar buttons may be removed/moved from the Command toolbar in the Algebra, 2D-Plot and 3D-Plot win dows by pressing the the <Alt> key (on US keyboard) and dragging them with the mouse. The same is also true for the top-level items on the menu bar. To reset the toolbars/menus back to the default, choosing to "Restore factory defaults" at Derive startup.

Among other things, this customization may be useful for those who wish to remove visible temptations from students for menu items such as "Solve". In this case, even if no toolbar button or menu item for "Solve" exists, the user may still invoke "Solve Expression" by the documented short-cut keystroke Ctrl+Shift+E or by typing it in the expression entry line.

Aloha, Theresa

On Integration

Problem:
Hi everybody ! In a math problem, I just came over this wierd answer from Derive :

INT(ê^(a·t), t) gives ê^(a·t)/a - 1/a

I did the same integral with my TI-92 Plus and it gives : ê^(a·t)/a

which I think to be OK. But what about the answer that Derive gave me ?

First answer (Ralph Freese)
Indefinite integration is only determined up to a additive constant; that is, if the derivative of g(t) is f(t) then the derivative of g(t) + c is also f(t). So both answers are correct. The Derive answer is chosen so that if one takes the limit as a -> 0, the answer is still valid.

Second Answer (Terence Etchells)
Further to Ralphs comments, to get the answer Matthieu expects we can force the variable *a* to be declared positive (i.e. not zero) through the Declare->Variable Domain option. Similary we can declare *a* to be negative.

This is a nice of example of Derive not giving us what we EXPECT, but giving us the CORRECT and CONSISTENT answer for all the variables/parameters involved.

The other good thing is that we learn a some mathematics along the way, as I would certainly have given the answer #e^(at)/a if I were doing this problem by hand (assumming the additive constant to be zero, that is).

Third Answer (Albert D. Rich)
Derive attempts to return antiderivatives that are valid for ALL values of the parameters in the integrands. In your example,

INT(ê^(a·t), t)

if the parameter *a* is 0, the integrand is 1 and the antiderivative is t.
However, the antiderivative ê^(a·t)/a is undefined at *a* = 0. The antiderivative returned by Derive (i.e. ê^(a·t)/a - 1/a) is also undefined at *a* = 0, but at least the limit as *a* approaches 0 is *t*. And since antiderivatives can differ by a constant ê^(a·t)/a and ê^(a·t)/a-1/a are both "OK", but Derive's is slightly better for the above reason.

Note that an even simpler example of this behavior by Derive is

INT(x^n, x) which simplifies to (x^(n + 1) - 1)/(n + 1)

which is correct in the limit as *n* approaches -1.

Hope this explains the reasoning behind the antiderivatives Derive returns.

Enric Puig sent a mail:

Dear Josef,
Thank you for your last mail, sorry I took so long to answer. I'm attaching a file which contains a routine to obtain the permutations of a vector of n elements taken m at a time.
Feel absolutely free to use it as you wish, after all, YOU are the Editor !
Let me simply say that when I finished this program I sent it to Johann Wiesenbauer, for comments; he gave two replies, an immediate one, which led to a little change in the auxiliary program to turn vectors to sets. This is the version you will find in the file.
The second answer came a few days later: instead of my rather mechanical approach, Johann chooses to fly high, and look at the problem from above: consequently he goes to Group Theory, and produces a fabulous program, fast as lightning (I'm still trying to understand it...).
Since this was a private exchange of e-mails, I do not feel authorized to send you his program, but I'm sure he will readily send it to you.
So, best regards, I'm sure you will decide what is best.
Enric

You can find Enric´s program in the file `permutations.dfw`. *Because I am lack of space I do not print Enric´s program but I present Johann´s solution, because he gave some interesting additional explanations on his so highly appreciated program. See now Johann´s perm():*

```
perm(v, n, k_, n_ := 2, s_ := [[1]],t_) :=
  PROG(
    LOOP(
      IF(n_ > n, exit),
      k_ := n_,
      t_ := [],
      LOOP(
        IF(k_ = 0, exit),
        t_ := APPEND(t_, VECTOR(INSERT(n_, v_, k_), v_, s_)),
        k_ :- 1),
      s_ := t_,
      n_ :+ 1),
    if(n=dimv, return s_),
    v := POWER_SET(MAP_LIST(v SUB j_, j_, {1, ..., DIM(v)}), n),
    v := VECTOR(SORT(v_), v_, v),
    SORT(APPEND(VECTOR(VECTOR(v_ SUB u_, u_, s_), v_, v))))

perm([a,b,c,d,e],5)=
```

In the first loop, I generated the symmetric group S_n, whose elements are stored in the vector s_. Then I used the built-in function POWER_SET to generate all subsets of v with n elements. This is the 3. line from below.
In the second line from below I converted those sets to lists using SORT and VECTOR. In the very last line I permuted all vectors in v using all permutations stored in s_. (Note that by far the most time-consuming operation is the sorting in the very last line. If you don't need the vectors sorted, then omit that SORT by all means!). Johann

Enric´s earlier mail included an interesting proposal for the Newsletter:

....but of course you cannot expect the experts to devote time and effort to solving problems posed by beginners who only want to be spoon-fed. I'm working right now on a program that will return a list of the permutations of m elements taken n at a time, and it's driving me crazy !! But when I succeed, (and I will !), I will submit it to the list, and I'm convinced I'm going to receive an answer.
If you could find a way to *propose a Series of Programming Challenges* which were at the same time mathematically meaningful, that would certainly encourage people like me. A sort of contest, specifically aimed at beginners, or mid-level programmers. When a few contributions have arrived, then perhaps the experts can make comments or amendments.
At any rate, DNL's make wonderful reading, and I hope you get many interesting contributions from around the world.

I found Enric´s idea great and asked some DERIVIANs for their opinion and support and fortunately I received very positive feedback including tasks for – not only for experts, who are friendly invited to show their skills, too – but for all DERIVIANs (and TI-ers, of course) who would like to face the challenges.

Although we had challenges earlier in this DNL maybe that an extra column will be an extra stimulus. I would also like to invite the TI-Users to make a try (or more of them). Johann Wiesenbauer gave the title of the column and Richard Schorn provided the first challenge.

Richard´s CHALLENGE

Instead of a very accurate description of the problem I´ll give an example:

Starting with a 4-digit number (say 3718) one generates the follwing 4-digit number by replacing the first digit by the sum of the last two digits (mod 10) and simultaneously omitting the last digit.

9371
8937
0893
2089
....
....

Investigate periodicity, take more digits, change the mod, ...

In a following mail Richard reported that he found this problem in *Praxis der Mathematik* in the sixties using the 10-digit initial number 1234567890.

Please send your solutions, findings, comments. But I´d like to invite you also to send your personal Challenge. Please keep in mind that we want to attract especially the beginners in programming and mid-level programmers.

Confidence Intervals

Josef Böhm

In the last DNLs we had some articles on statistics and I promised that Don Phillips will present a ANOVA-file for *DERIVE*. The file is in my computer and ready to be published, but I thought to cover another part of statistics before proceeding to ANOVA – the confidence interval for mean, proportion and variance.

The program `MeanConfInt(...)` refers to several auxiliary functions provided by Don Phillips (*DNL*#48). They are included in `confidence.dfw`. I print the program in its structured form because I find it very informative – even for the students – to follow the rules how to find the appropriate confidence intervals. I'd like to express my gratefulness to Don and to Fritz Tinhof. His paper on statistics was very helpful, too.

```
MeanConfInt(dat,c_lev,v:=1,sigma:=0,s:=0,n:=0,x_,e_,type):=
  prog(
    if(number?(dat)=true,
       prog(x_:=dat,
            if(sigma=0, sigma:=s)
           ),
       prog(x_:=average(dat),
            n:=dim(dat),
            if(sigma=0, sigma:=stdev(dat))
           )
      ),
    if(n>=30 or v=0,
       e_:= ["Z-Interval",inversenormal((c_lev+1)/2)*sigma/sqrt(n)],
       e_:= ["t-Interval",inverse_t((c_lev+1)/2,n-1)*sigma/sqrt(n)]
      ),
    return [e_ sub 1;append("ConfLevel = ",string(c_lev));
            append("μ = ",string(x_));append("s = ",string(sigma));
            append("μ ± ",string(e_ sub 2));
            append(string(x_-e_ sub 2)," ≤ μ ≤ ",string(x_+e_ sub 2));
            append("n = ",string(n))]
  )
```

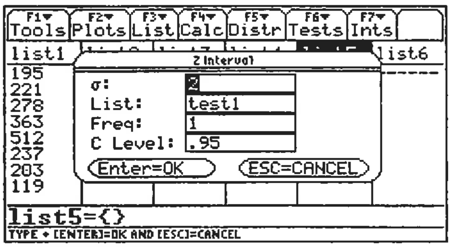
(The argument list is explained on page 42.)

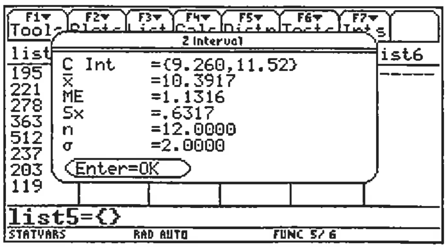
Please follow the examples.

Example 1: Small sample C.I. (Application of distribution can be forced by the 3rd parameter: 1 = t-distribution, 0 = normal distribution).

Given is sample `test1` (1st case σ of population = 2, 2nd case σ unknown)[1].

```
test1 := [9.5, 9.5, 11.2, 10.6, 9.9, 11.1, 10.9, 9.8, 10.1, 10.2, 10.9, 11]

[MeanConfInt(test1, 0.95, 0, 2), MeanConfInt(test1, 0.95)]

[[      Z-Interval          ]   [      t-Interval          ]]
[[   ConfLevel = 0.95       ]   [   ConfLevel = 0.95       ]]
[[     μ = 10.3916          ]   [     μ = 10.3916          ]]
[[       s = 2              ] , [    s = 0.631676          ]]
[[     μ ± 1.13158          ]   [    μ ± 0.401348          ]]
[[ 9.26008 ≤ μ ≤ 11.5232    ]   [ 9.99031 ≤ μ ≤ 10.7930    ]]
[[       n = 12             ]   [       n = 12             ]]
```





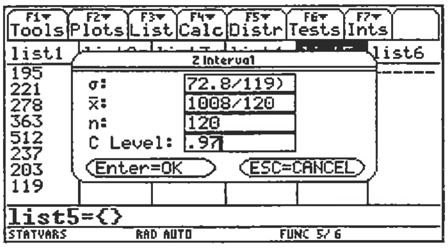
You can easily find the confidence interval on the TIs using the `Stats/List Editor` Tool.

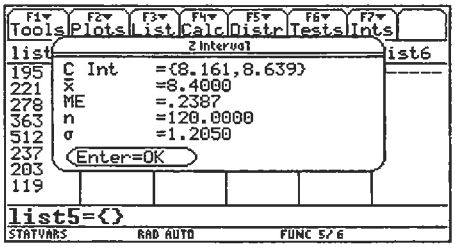
Example 2: A random sample of 120 measurements taken from a normal population gave the following data: $n = 120$, $\Sigma x = 1008$, $\Sigma(x-\mu)^2 = 172.8$. Find the 97% confidence interval [1].

As the population variance is unknown, one could take $\Sigma(x-\mu)^2/(n-1)$ for σ^2, instead of $\Sigma(x-\mu)^2/n$. Here are both possible results.

$$\text{MeanConfInt}\left(\frac{1008}{120}, 0.97, 0, 0, \sqrt{\frac{172.8}{120}}, 120\right) = \begin{bmatrix} \text{Z-Interval} \\ \text{ConfLevel} = 0.97 \\ \mu = 8.4 \\ s = 1.2 \\ \mu \pm 0.237721 \\ 8.16227 \le \mu \le 8.63772 \\ n = 120 \end{bmatrix}$$

$$\text{MeanConfInt}\left(\frac{1008}{120}, 0.97, 0, 0, \sqrt{\frac{172.8}{119}}, 120\right) = \begin{bmatrix} \text{Z-Interval} \\ \text{ConfLevel} = 0.97 \\ \mu = 8.4 \\ s = 1.20503 \\ \mu \pm 0.238718 \\ 8.16128 \le \mu \le 8.63871 \\ n = 120 \end{bmatrix}$$

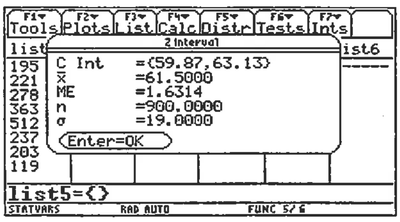




Example 3: Find the 99%-confidence interval for the mean with a given sample ($n = 900$, $\mu = 61.5$ and $s = 19$[2].

(σ of population unknown → 4[th] argument = 0!)

$$\text{MeanConfInt}(61.5, 0.99, 0, 0, 19, 900) = \begin{bmatrix} \text{Z-Interval} \\ \text{ConfLevel} = 0.99 \\ \mu = 61.5 \\ s = 19 \\ \mu \pm 1.63135 \\ 59.8686 \le \mu \le 63.1313 \\ n = 900 \end{bmatrix}$$



Example 4: `overdue` is a list of overdue amounts (in USD) for Delinquent Accounts. Give a 90%-confidence interval for the average overdue.[3]

(As $n > 30$, the program applies automatically normal distribution.)

```
overdue := [195, 221, 278, 363, 512, 237, 203, 119, 304, 368, 243, 162, 222, 221,
  193, 135, 178, 259, 141, 274, 132, 134, 236, 449, 134, 252, 180, 108, 158, 278,
  133, 275, 178, 265, 138, 365, 148, 289, 240, 190, 209, 355, 202, 146, 209, 371,
  162, 328, 82, 344, 400, 293, 222, 215, 207, 238, 160, 331, 17, 157, 142, 242, 334,
  113, 206, 232, 86, 330, 357, 219, 312, 458, 208, 229, 310, 271, 234, 227, 187, 77,
  221, 378, 194, 221, 293, 121, 244, 162, 364, 171, 289, 148, 135, 243, 310, 134,
  266, 354, 268, 280]
```

```
                                  ⎡       Z-Interval        ⎤
                                  ⎢    ConfLevel = 0.9      ⎥
                                  ⎢       μ = 233.28        ⎥
MeanConfInt(overdue, 0.9) =       ⎢       s = 90.3398       ⎥
                                  ⎢      μ ± 14.8595        ⎥
                                  ⎢ 218.420 ≤ μ ≤ 248.139   ⎥
                                  ⎣        n = 100          ⎦
```

Example 5: 225 flights of an airline are randomly chosen and the numbers of nonoccupied seats are noted. The sample mean and standard deviation are 11.6 and 4.1. Estimate the average number of unoccupied seats using a 90%-confidence interval.[3]

Not the data are given, but statistics. So we must fill in the complete parameter list:
(mean, confidence level, 0 for normal distribution, 0 for unknown population standard deviation, 4.1 for the sample standard deviation, 225 for sample size).

```
                                          ⎡       Z-Interval        ⎤
                                          ⎢    ConfLevel = 0.9      ⎥
                                          ⎢        μ = 11.6         ⎥
MeanConfInt(11.6, 0.9, 0, 0, 4.1, 225) =  ⎢        s = 4.1          ⎥
                                          ⎢      μ ± 0.449593       ⎥
                                          ⎢ 11.1504 ≤ μ ≤ 12.0495   ⎥
                                          ⎣        n = 225          ⎦
```

Visualization of the confidence interval.

We produce 50 samples ($n = 20$) of normally distributed random numbers ($\mu = 50, \sigma = 5$).
For each sample the 80%-confidence interval will be calculated.
The confidence interval should include the mean in approx 80% of experiments (~40 samples).
Then I repeat the procedure generating 100 samples ($n = 10$).

Simplify the following command and store the resulting matrix as `samples` (I reprint only a part of the matrix). Then I extract the C.I. for the first sample:

```
VECTOR(VECTOR(RANDOM_NORMAL(5, 50), k, 20), j, 50)

             ⎡ 42.9043  51.0718  48.4485  44.318   50.6375  50.3058  47.3041  53.7976
samples :=   ⎢ 45.5314  51.6722  42.65    47.4733  52.2433  45.6656  53.0238  54.644
             ⎣ 59.4154  55.6403  51.672   45.3612  53.1694  51.0945  43.9842  58.3704
```

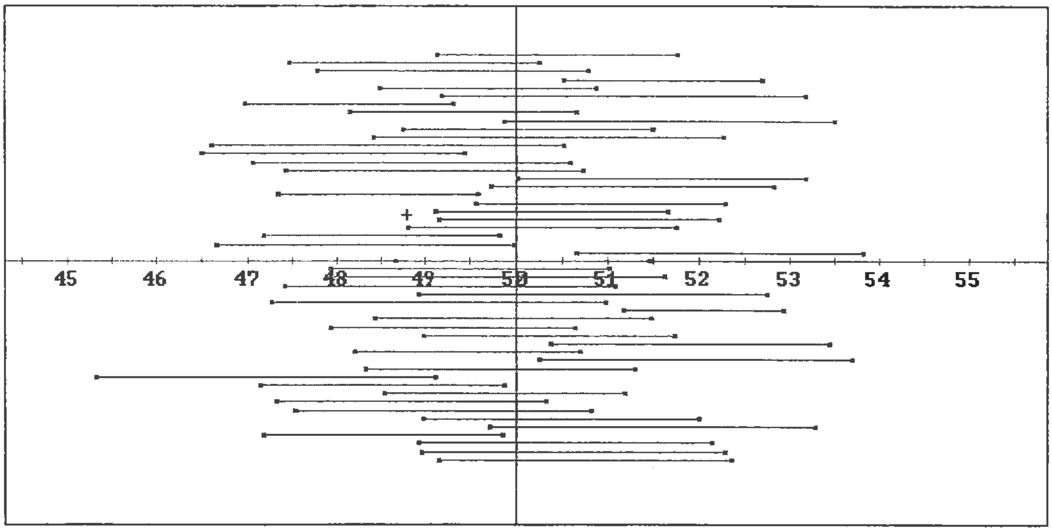
```
                                  ⎡       t-Interval        ⎤
                                  ⎢    ConfLevel = 0.8      ⎥
                                  ⎢       μ = 50.7587       ⎥
MeanConfInt(samples , 0.8) =      ⎢       s = 5.39752       ⎥
                   1              ⎢      μ ± 1.60246        ⎥
                                  ⎢ 49.1562 ≤ μ ≤ 52.3611   ⎥
                                  ⎣        n = 20           ⎦
```

For plotting purposes I change the output of the result – `m_ci_bds(..)` returns only the boundaries of the confidence interval. The VECTOR generates the list `bounds` of all intervals.

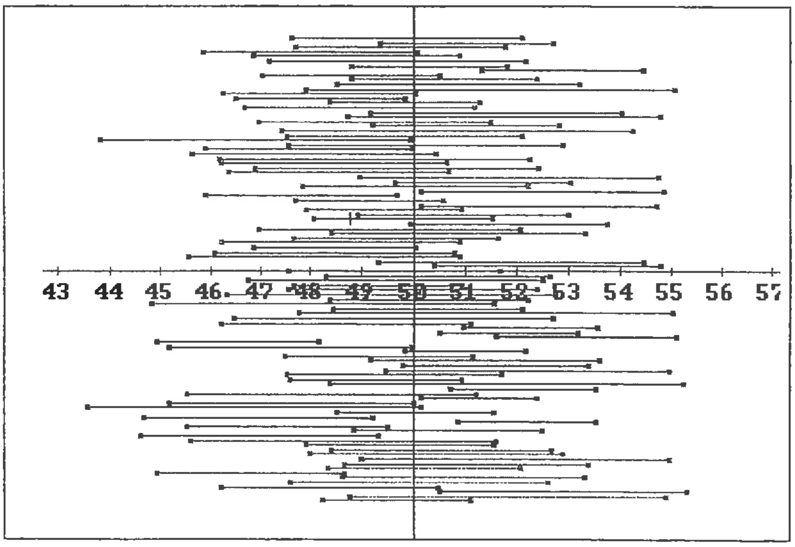
```
m_ci_bds(samples , 0.8) = [49.1562, 52.3611]
                1

bounds := VECTOR(m_ci_bds(v_, 0.8), v_, samples)

                                        ⎡ 49.1562   52.3611 ⎤
                                        ⎢ 48.9706   52.2858 ⎥
                                        ⎣ 48.9334   52.1409 ⎦

DIM(SELECT(¬ v  ≤ 50 ≤ v , v, bounds)) = 14
              1         2
```

So we find 14 confidence intervals not including the mean μ = 50. `cis` plots all confidence intervals as segments (gray) , $x = 50$ is a vertical line marking the mean (blue) and the following `select( )` selects the "bad" intervals, which should be plotted in another colour (red).

```
                ⎛⎡ bounds      -5 + 0.2·i ⎤           ⎞
                ⎜⎢       i,1              ⎥           ⎟
cis := VECTOR   ⎜⎢ bounds      -5 + 0.2·i ⎥, i, 1, 50 ⎟
                ⎝⎣       i,2              ⎦           ⎠

x = 50

SELECT(¬ v_     ≤ 50 ≤ v_    , v_, cis)
           1,1           2,1
```



Repeat the procedure with another initial vector (100 samples with $n = 10$):
`VECTOR(VECTOR(RANDOM_NORMAL(5, 50), k, 10), j, 100)`



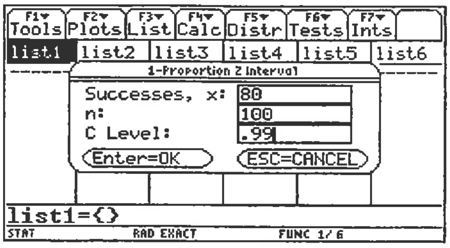
```
DIM(SELECT(v  ≤ 50 ≤ v , v, bounds2)) = 79
            1          2
```

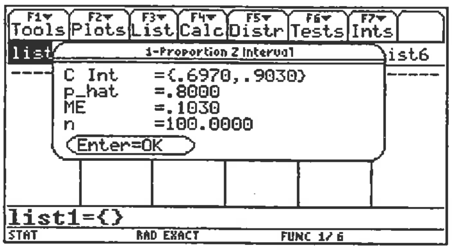
This result fits pretty well to the 80%-intervals.

Confidence intervals for the population proportion.

We use `PropConfInt(successes,n,confidence_level)`

Example 6: A new drug is tested on 100 infected mice. The drug helps in 80 cases. What is the 99%-confidence interval for the positive effect of the drug?[2]





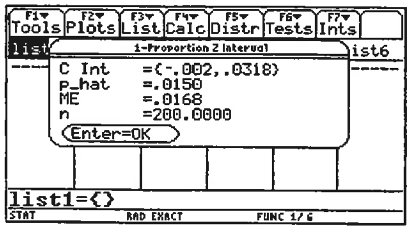

```
                           ⎡   1-Proportion Z Interval   ⎤
                           ⎢      ConfLevel = 0.99       ⎥
                           ⎢           p = 0.8           ⎥
PropConfInt(80, 100, 0.99) = ⎢          s = 0.04           ⎥
                           ⎢        p ± 0.103033         ⎥
                           ⎢ 0.696966 <= p <= 0.903033   ⎥
                           ⎣           n = 100           ⎦
```

Example 6: Among a sample of 200 US citizens 3 are violent crime victims. What is the 95% confidence interval for the proportion of crime victims among the US population?[3]

```
                          ⎡      1-Proportion Z Interval      ⎤
                          ⎢         ConfLevel = 0.95          ⎥
                          ⎢            p = 0.015              ⎥
PropConfInt(3, 200, 0.95) = ⎢         s = 0.00859505            ⎥
                          ⎢          p ± 0.0168460            ⎥
                          ⎢ -0.00184600 <= p <= 0.0318460     ⎥
                          ⎣             n = 200               ⎦
```

The procedure presented does not work, if p is likely to be near 0 or 1. Statisticians proposed an alternative method based on the Wilson point estimator of p. This procedure works well for any p even when the sample size n is very small.[3].

```
                             ⎡   1-AdjProportion Z Interval    ⎤
                             ⎢        ConfLevel = 0.95         ⎥
                             ⎢        p = 0.0245098            ⎥
AdjPropConfInt(3, 200, 0.95) = ⎢        s = 0.0108259            ⎥
                             ⎢        p ± 0.0212184            ⎥
                             ⎢ 0.00329131 <= p <= 0.0457282    ⎥
                             ⎣           n = 200               ⎦
```



Comments on this improvement, confidence interval for the variance, Don Phillips' confidence-program and the references will follow in *DNL#50*. The complete file `confidence.dfw` containing a couple of additional examples can be downloaded.