THE DERIVE - NEWSLETTER #32

ISSN 1990-7079

THE BULLETIN OF THE

DERIVE

USER GROUP + TI 92

Contents:

revised version 2014 **December 1998**

[1] DERIVE für den Mathematikunterricht, W. Köpf

[2] Historische Aspekte im Mathematikunterricht, M. Kronfellner

[3] Einführung in den TI-89, Teil 1 und Teil 2, B. Kutzler, bk teachware

[4] Computergestützer Physikunterricht, Experimente zur Mechanik – Auswertung mit dem CAS des TI-92, K-H. Keunecke, Texas Instruments

[5] Introduction a la géometrie avec la TI-92, J.-J. Dahan, ellipses, ISBN 2-7298-9877-8

[6] Faire des Mathématiques au lycée avec des calculatrices symboliques, Luc Trouche, IREM des Montpellier, 34095 Montpellier Cedex 05, email: irem@math.univ-montp2.fr

---

TEX Praxis, Raymond Séroul und Silvio Levy, Birkhäuser, ISBN 3-7643-2823-1

TEX - das geniale Satzsystem Donald Knuths - setzt sich auch bei nicht ganz professionellen Schreibern und Vielschreibern durch. Mit der aktuellen Version von Scientific Workplace gibt es nun auch eine Windows- Implementierung von TEX in Kombination MAPLE.

Nach der englischen Übersetzung liegt nun dieses französische Standardwerk über das Arbeiten mit TEX auch in einer sorgfaltig an den deutschen Sprachgebrauch angepassten Fassung vor. Das Buch richtet sich sowohl an den Einsteiger, wie auch an den fortgeschrittenen Anwender.
Die Kapitel können selektiv gelesen werden, viele Beispiele sind in den Text eingestreut und verführen den Lesenden zum Probieren. Insgesamt 16 Kapitel verteilen sich auf mehr als 400 Seiten. Die Kapitelüberschriften reichen vom Zeichensatz von TEX über Seitenlayout zu Tabellen und Tabellieren. Dem wichtigen Formelsetzen sind fast 40 Seiten gewidmet. Ein ausführliches Glossar und Wörterbuch runden das empfehlenswerte Buch ab.

Symbolic Rewriting Techniques, Manual Bronstein a.o., Ed., Birkhäuser, ISBN 3-7643-5901-3

This volume 15 of the series "Progress in Computer Science and Applied Logic" contains invited and contributed papers to the *Symbolic Rewriting Techniques* workshop, which was held in Ascona, 1995. Symbolic rewriting techniques are methods for deriving consequences from systems of equations, and are of great use when investigating the structure of the solutions.

Most of the contributions deal with Gröbner bases e.g. The Computation of Gröbner Bases Using an Alternative Algorithm *(JApeT),* Gröbner Fans and Projective Schemes *(D.MalT),* etc. In total 14 contributions can be found within 288 pages, each of them with an extended reference list. I must confess that it is not an easy to read book and it seems not to be an introduction into this field of mathematical research, but it might be of high value for people who have some experience in this subject matter.

Interesting Web Sites:

http://www.can.nl/
http://math.osu.edu/research/math-links
http://www.nctm.org/ National Council of Teachers of Mathematics

Liebe DERIVE- und TI-Freunde,

Wie schnell doch so ein Jahr vergeht! Besonders am Jahresende, wenn die Abende lange sind, wird mir das so richtig bewusst.

Ich hoffe, dass es für Sie ein gutes Jahr war. Von unserer Seite kann ich das glücklicherweise behaupten: es war sehr arbeitsam aber auch bereichernd in vieler Hinsicht (unsere Enkelin Kimberly, die viele von Ihnen kennen, hat vor einigen Tagen eine Schwester bekommen, Yvonne Caroline.)

Im DNL#29 habe ich von einem geplanten EU-Projekt berichtet. Das Projekt ist bewilligt worden und im Jänner gibt es das erste Expertentreffen.

Ein weiterer Höhepunkt war die Konferenz in Gettysburg, die dank der Bemühungen von Pat und Carl Leinbach und ihrem Team ein Erfolg in vieler Hinsicht geworden sind. Vielen Dank, die Tage in Pennsylvania werden wir nicht vergessen.

Ich darf Sie bei dieser Gelegenheit auf die Konferenz 1999 in Hagenberg in Oberösterreich hinweisen. Wir hoffen, dass von diesem Treffen ähnliche Impulse ausgehen, wie von den ersten, bereits legendären Krems-Konferenzen.

Im Jahr 1999 erwarten wir DERIVE 5. Ich durfte schon ein wenig damit probieren: viele unserer Wünsche an DERIVE sind darin verwirklicht worden.

Viele interessante Artikel liegen noch für den DNL bereit Bitte haben Sie Geduld. Es ist immer eine schwierige Aufgabe, den DNL zusammenzustellen. So vieles wäre zu veröffentlichen: da gibt es z.B. einen spannenden Beitrag zu Gruppen von Richard Schorn, eine ganz ungewöhnliche DERIVE-Anwendung, und vieles andere mehr. Ich möchte aber besonders die vielen Lehrer unter Ihnen bitten, erfolgreiche Unterrichtseinheiten zur Verfügung zu stellen.

Da die T192-Anwendungen fast ausnahmslos auch auf dem T189 laufen und umgekehrt, sind auch Ideen zum Einsatz dieses neuen Gerätes willkommen. Falls Sie irgendwelche Wünsche haben, dann melden Sie sich bitte. Es könnte durchaus sein, dass wir Ihnen helfen können.

Beachten Sie bitte meine neue e-mail Anschrift (S.2). Es bleibt mir nur noch, Ihnen allen ein frohes Weihnachtsfest und ein gesundes, friedliches und erfolgreiches Jahr 1999 zu wünschen. Uns wünschen wir, dass Sie wie bisher weiter zur DUG halten und uns gewogen bleiben.

Josef Böhm

Dear DERIVE and TI-friends,

How fast a year is passing by - especially at the end of the year, when the evenings are so long, I become aware of that.

I hope that it has been a good year for you. Fortunately my wife and myself can say so: there has been plenty of work but also a lot of sunshine and happiness (our granddaughter - many of you know her - got a sister a couple of days ago, Yvonne Caroline.)

In DNL#29 I reported about a planned European Union project. This project has been approved and we will have the first meeting of experts in January.

Another c1imax was the Gettysburg Conference, which became a success in many respects thanks to Pat and Carl Leinbach's and their teams efforts. Many thanks to you, we will never forget the great days in Pennsylvania.

On this occasion I' d like to remind you on the Conference 99 in Hagenberg, Upper Austria. We hope that this meeting will give an impetus similar to the legendary Krems Conferences.

In 1999 we look forward to *DERIVE 5*. I was allowed to try it a bit: many of our wishes have been realized.

Many interesting contributions are ready to be published in the DNL. Please be patient. Believe it or not, it is hard work to compile each DNL. So many articles are lying on or under my desk: e.g. Richard Schorn contributed a very unusual *DERIVE* application in Group Theory and many others. I' d like to ask the teachers among you to make successful teaching units available to our community.

As TI92-applications also work on the TI89 nearly without exception and vice versa, we welcome ideas realized on this new device. If you have any special wishes, feel free to ask. It seems very likely that we can help.

Please notice my new e-mail address (page 2). Finally we want to send you our very best Seasons Greetings. We have the wish for us that you will stick to the DUG as you have done up to now and stay kindly disposed to us,

Josef Böhm

The *DERIVE-NEWSLETTER* is the Bulletin of the *DERIVE 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* as well as to create a group to discuss the possibilities of new methodical and didactical manners in teaching mathematics.
We have established a section dealing with the use of the *TI*-92 and we try to combine these modern technologies.

Editor: Mag. Josef Böhm
A-3042 Würmla
D´Lust 1
Austria
Phone: 43-(0)660 3136 365
email: nojo.boehm@pgv.at

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

Next issue: March 1999
Deadline: 15 February 1999

Preview: Contributions for the next issues

3D-Geometry, Reichel, AUT
Graphic Integration, Linear Programming, Various Projections a. o., Böhm, AUT
A Utility file for Complex Dynamic Systems, Lechner, AUT
Examples for Statistics, Roeloffs, NL
INTEG() – a Package for the *TI*s, Pröpper & Böhm, GER & AUT
Fractals and other Graphics, Koth, AUT
Implicit Multivalue Bivariate Function 3D Plots, Biryukov, RUS
Quaternion Algebra, Sirota, RUS
Various Training Programs for Students on the TI-92, Böhm, AUT
A Critical Comment on the "Delayed Assignation" := =, Kümmel, GER
Sand Dunes, River Meander and Elastica, The lighter side ...., Halprin, AUS
Type checking, finite continued fractions, ..., Welke, GER
On the Resolution of the Linear Differential Equation ......, Candel, ESP
Share Holders' Considerations using a CAS, Böhm, AUT
Kaprekar's "Self Numbers", Finite Groups, Schorn, GER
Graphic solution of Linear Programming Problems in the *TI*, Kirmse, GER
TI-versions of Spigot Algorithms for π and *e*, Witthinrich, GER
Problems from Physics, Magiera, POL

and
Setif, FRA; Vermeylen, BEL; Leinbach, USA , Koller, AUT; Meagher, AUT; Wiesenbauer, AUT; Aue, GER; Speck, NZL, and and and ....

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

<u>Volker Loose, Germany</u>

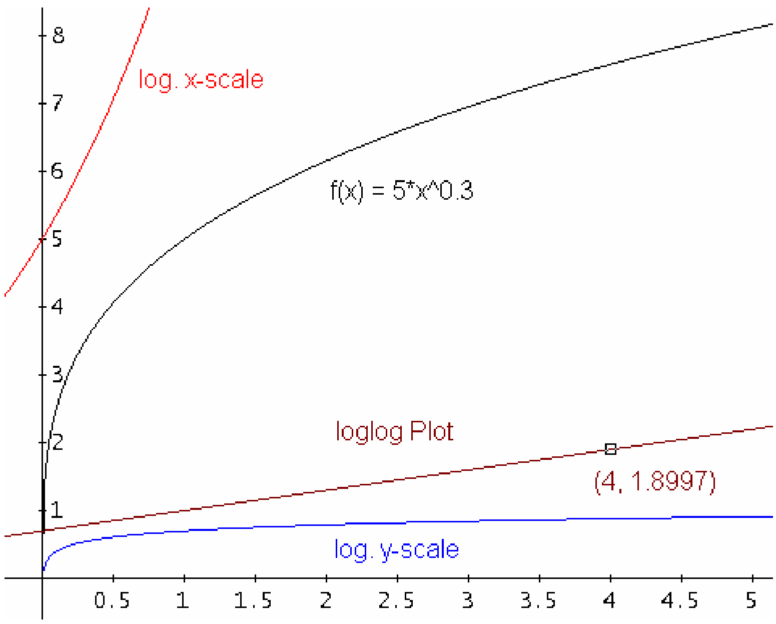
Hi everyone, is it possible to have axes with logarithmic scale in *DERIVE*?

<u>Peter van den Sanden, Netherlands</u>

There are no logarithmic scales in *DERIVE*. However, you can create logarithmic scales (using any base *a*) with a few substitutions which can be declared by functions:

`f(x) :=`	⇒ the function you want to plot
`f_log(x, a) := f(a^x)`	⇒ logarithmic *x*-scale
`f_log_y(x, a) := LOG(f(x), a)`	⇒ logarithmic *y*-scale
`f_log_y(x, a) := LOG(f(x), a)`	
`f_log_xy(x, a) := LOG(f(a^x), a)`	⇒ logarithmic *x*- and *y*-scale

```
              0.3
f(x) := 5·x

f_log(x, 10)

f_log_y(x, 10)

f_log_xy(x, 10)

                   1/5
f(10000) = 50·10

APPROX(LOG(f(10000), 10)) = 1.898970004
```



Michael Johann, Germany

I am using DfW 4.08. How can I make it to get only the real solution for $x^3 = -1$? I tried it by defining the range of x and Algebra Status – in vain.

DNL: There were several answers in the *DERIVE*-news pointing at the right choice of the branch for evaluating $(-1)^{1/3}$. A nice answer was given by Terence Etchells, Liverpool:

Hi Michael,

Hmm, I'm intrigued! Why do you only want the real solution(s)? If you have a vector of solutions (e.g. [x = 1, x = -1, x = 1 + *i*, ...]) and you wish to pick out the real ones, then a function REAL_SOLVE does the trick:

This request is obsolete for the recent version of *DERIVE*. Terence's trick is no more necessary and it doesn't work because the output of SOLVE has changed. Using *DERIVE* 6 it is easy to control the domain of the solutions.

$$\text{SOLVE}(x^3 = -1) = (x = -1)$$

$$\text{SOLVE}(x^3 = -1, x) = \left(x = \frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} \vee x = \frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} \vee x = -1\right)$$

$$\text{SOLVE}(x^6 - 1 = 0) = (x = -1 \vee x = 1)$$

$$\text{SOLVE}(x^6 - 1 = 0, x)$$

$$x = -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} \vee x = -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} \vee x = \frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} \vee x = \frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} \vee x = -1 \vee x = 1$$

$$\text{SOLUTIONS}(x^3 = -1, x, \text{Real}) = [-1]$$

$$\text{SOLUTIONS}(x^3 = -1, x) = \left[-1, \frac{1}{2} + \frac{\sqrt{3}\cdot i}{2}, \frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}\right]$$

$$\text{SOLUTIONS}(x^6 = 1, x, \text{Real}) = [1, -1]$$

$$\text{SOLUTIONS}(x^6 = 1, x) = \left[1, -1, \frac{1}{2} + \frac{\sqrt{3}\cdot i}{2}, \frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}, -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2}, -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}\right]$$

DNL: Another FAQ (frequently asked question) appeared in the *DERIVE-News* followed by a very extended answer given by Johann Wiesenbauer, which seems to be interesting enough to be presented for you all:

I was wondering if anyone knows a command for *DERIVE* that is similar to EVSUB from *MuMath* or SUBS *in MAPLE*. I want to evaluate $f(x) = x^3 + 2x + 1$ when $x = 5$ without using Manage Substitute. Can anyone help?

Blain Priddle

Some could. See Johann's answer on the next page. Although we do have now the SUBST-command which is very powerful I don't hesitate to reprint Johann's proposals, Josef.

#1: $f(x) := x^3 + 2\cdot x + 1$

#2: $f(5) = 136$

Note: Unlike the other solutions this one cannot be used within programs!

#3: $\lim_{x\to 5} (x^3 + 2\cdot x + 1) = 136$

#4: $\sum_{x=5}^{5} (x^3 + 2\cdot x + 1) = 136$

#5: $\prod_{x=5}^{5} (x^3 + 2\cdot x + 1) = 136$

#6: $(\text{VECTOR}(x^3 + 2\cdot x + 1, x, 5, 5))_1 = 136$

#7: $\text{TAYLOR}(x^3 + 2\cdot x + 1, x, 5, 0) = 136$

#8: $\text{ITERATE}(x^3 + 2\cdot x + 1, x, 5, 1) = 136$

In terms of simplicity this survey should rather be complete (anybody out there who proves me wrong?) apart from slight variations such as

#9: $\sum(x^3 + 2\cdot x + 1, x, [5]) = 136$

#10: $(\text{ITERATES}(x^3 + 2\cdot x + 1, x, 5, 1))_2 = 136$

I think the most appropriate solutions in this case should be #2 or #3. Whereas in more general situations I have a strong preference for #8. Let's wait what *DERIVE* 5 has in store for us.

There was another advice from Nurit Zehavi, Israel, which might be interesting even for other occasions:

#11: subs([exp], [numbers]) := VECTOR(APPEND([x], [exp]), x, APPEND([x], [numbers]))

#12: $\text{subs}\left(\begin{bmatrix} x^3 + 2\cdot x + 1 \end{bmatrix}, [5]\right) = \begin{bmatrix} x & x^3 + 2\cdot x + 1 \\ 5 & 136 \end{bmatrix}$

#13: $\text{subs}([2\cdot x, 20\cdot x, 200\cdot x], [4, 7, 25]) = \begin{bmatrix} x & 2\cdot x & 20\cdot x & 200\cdot x \\ 4 & 8 & 80 & 800 \\ 7 & 14 & 140 & 1400 \\ 25 & 50 & 500 & 5000 \end{bmatrix}$

Now we can use the SUBST-command not only for substituting one variable. We can substitute a list of variables – and we can substitute for subexpressions:

$$\text{SUBST}(x^3 + 2\cdot x + 1,\ x,\ 5) = 136$$

$$\text{SUBST}(x^3\cdot y^3 + 2\cdot x\cdot y + 1,\ [x,\ y],\ [5,\ 3]) = 3406$$

$$\text{SUBST}\left(x^2 + \frac{3}{x^2 + 2},\ x^2,\ a\right) = \frac{3}{a + 2} + a$$

But unfortunately this does not hold for all cases. As you can see, I was able to substitute for x^2 but not for $(x+1)$ or $(x+1)^2$.

$$\text{SUBST}\left((x + 1)^2 + \frac{3}{(x + 1)^2 + 2},\ x + 1,\ a\right) = \frac{3}{x^2 + 2\cdot x + 3} + a^2$$

$$\text{SUBST}\left((x + 1)^2 + \frac{3}{(x + 1)^2 + 2},\ (x + 1)^2,\ a\right) = \frac{3}{x^2 + 2\cdot x + 3} + a$$

$$\lim_{x + 1\to a}\left((x + 1)^2 + \frac{3}{(x + 1)^2 + 2}\right) = \lim_{x + 1\to a}\left(\frac{3}{x^2 + 2\cdot x + 3} + (x + 1)^2\right)$$

$$\lim_{x\to a - 1}\left((x + 1)^2 + \frac{3}{(x + 1)^2 + 2}\right) = \frac{3}{a^2 + 2} + a^2$$

If you can do without the headline you can take the TABLE-command instead of Nurit's function from 1999.

$$\text{TABLE}([2\cdot x,\ 20\cdot x,\ 200\cdot x],\ x,\ [4,\ 7,\ 25]) = \begin{bmatrix} 4 & 8 & 80 & 800 \\ 7 & 14 & 140 & 1400 \\ 25 & 50 & 500 & 5000 \end{bmatrix}$$

DNL: The next topic – Piecewise defined functions – appeared after a question posed by Allan S. Hugo. He wanted to define and then to plot a piecewise defined function like:

$$y = f(x) = \begin{cases} x & 0 \le x < 1 \\ x^2 & 1 \le x < 3 \\ 2x + 3 & x \ge 3 \\ \text{undef.} & \text{else} \end{cases}$$

I will summarize the very fundamental discussion which followed:

<u>Andrej Jakobcic, Slovenia</u>

Try this:

```
f(x):=IF(x<0, ?,IF(x<1, x, IF(x<3, x^2, 2x + 3)))
```

<u>Johann Wiesenbauer, Vienna</u>

Of course, this is the appropriate way of solving this problem, though purist might prefer this version of f(x):

```
f(x):=IF(x≥0, IF(x<1, x, IF(x<3, x^2, 2x + 3)))
```

<u>Ralph Freese, Honolulu</u>

You could also use *DERIVE*'s CHI-function. `CHI(a,x,b)` is 1 for x between a and b and 0 otherwise (assuming $a < b$). So graphing

```
f(x):=CHI(0,x,1)*x + CHI(1,x,3)*x^2 + CHI(3,x,inf)*(2x+3)
```

also does what you want.

<u>Johann Wiesenbauer, Vienna</u>

What looks very elegant at the first glimpse (most people don't even know of the existence of the built-in CHI-function!) it is wrong at the second. The function

```
f(x):=CHI(0,x,1)*x + CHI(1,x,3)*x^2 + CHI(3,x,inf)*(2x+3)
```

yields 0 for negative values of x, but should be undefined in this region. I would have liked this definition, but I'm afraid, at least one IF is necessary e.g.

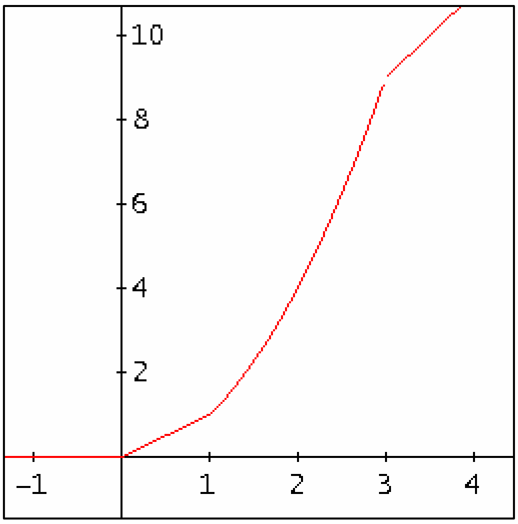
```
f(x):=IF(x ≥ 0, CHI(0,x,1)*x + CHI(1,x,3)*x^2 + CHI(3,x,inf)*(2x+3))
```

Due to the definition of `CHI(a,x,b)` for $x = a$ and $x = b$, which is weird in my opinion, f(1), and f(3) yield a question mark instead of the correct values. (By the way, the same also goes for the version of Ralph Freese and my proposal for a correction.) Moreover, f(0) is undefined. Like it or lump it, the run off the mill solution using nested IF-statements (essentially due to Andrej Jakobic) seems to be the only correct one.

$$f(x) := \chi(0, x, 1)\cdot x + \chi(1, x, 3)\cdot x^2 + \chi(3, x, \infty)\cdot(2\cdot x + 3)$$

TABLE(f(x), x, -1, 5) =

-1	0
0	0
1	?
2	4
3	?
4	11
5	13



(You can observe the gap for x = 3! Take care to edit the Chi-function in uppercase CHI, Josef.)

David Stenenga, Honolulu

The function `CHI(a,x,b)` may not be that familiar to most *DERIVE* users but it is quite useful for certain types of problems. In our lab manual (see http://www.math.hawaii.edu/CalcLabBook/[*]), Ralph Freese and I have a nice example in connection with spline approximations.

The idea is that you are given a ($n \times 2$)-matrix which represents the *xy*-coordinates of some sampling of function values. We assume that the *x*-values are strictly increasing: $x_1 < x_2 < x_3 <\ldots < x_n$ but the *y*-values are arbitrary. The idea with spline functions is to define a piecewise polynomial function f(x), which is a polynomial f(x,k) on the interval $x_k < x < x_{k+1}$. Adjoining polynomials are subject to certain conditions such as continuity and perhaps some smoothness.
Using nested IF-statements would be impractical unless the number of points is pretty small. Instead, we do the following (file PIECEWISE.MTH):

#1: `[F(x, k) :=, data :=]`

#2: $$\sum_{k=1}^{\mathrm{DIMENSION(data)}-1} F(x,k)\cdot\chi(\mathrm{data}_{k,1},\ x,\ \mathrm{data}_{k+1,1})$$

#3: `Example: Piecewise linear interpolation:`

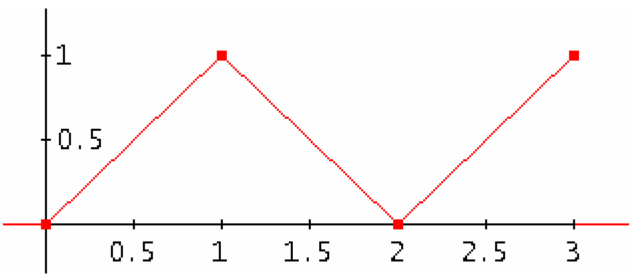
#4: $$F(x,k) := \frac{\mathrm{data}_{k+1,2} - \mathrm{data}_{k,2}}{\mathrm{data}_{k+1,1} - \mathrm{data}_{k,1}}\cdot(x - \mathrm{data}_{k,1}) + \mathrm{data}_{k,2}$$

#5: `Sample problem for piecewise linear interpolation:`

#6: $$\mathrm{data} := \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 0 \\ 3 & 1 \end{bmatrix}$$

#7: `Simplify #2`

#8: $$\frac{(2 - x)\cdot\mathrm{SIGN}(x - 3)}{2} + |x - 2| - |x - 1| + \frac{|x|}{2}$$



Of course, this technique will give a zero value for points outside the interval [x_1, x_n] but then this can be corrected as noted by Johann.

[*] The website is still active. You can find a file containing some DERIVE-tools at
http://math.hawaii.edu/wordpress/category/undergrad-posts/computing-posts/derive-posts/

Johann Wiesenbauer, Vienna

Hi all,

In general, the function `CHI(a,x,b)` is certainly very useful, but as we have seen its current definition for $x= a$ and $x = b$ may cause problems, because its values are ambiguous, viz. 1/2 ± 1/2. Thus strictly speaking it's not even a function!

For this reason I'd like to suggest a more general *DERIVE*-function `CHI(a,x,b,c,d)` (hoping that Albert Rich is all ears) that is both a "genuine" function and also one that seems to live up to all (?) expectations:

1. `CHI(a,x,b,c,d)` coincides with the current `CHI(a,x,b)` for $a < x < b$.
   Additionally, its values for $x = a$ and $x = b$ are c and d, respectively. In particular, this includes the indicator functions for the intervals $[a,b]$, $(a,b]$, $[a,b)$, (a,b) as well as the case $c = d = ½$, which plays an important role on some occasions, e.g. when dealing with Fourier series.

2. `CHI(a,x,b,c)` is `CHI(a,x,b,c,1-c)`.This includes the indicator functions for the intervals $(a,b]$ and $[a,b)$ and again the case $c = 1/2$. These versions of `CHI` can be used to construct piecewise defined functions such as in David's example, but the results will be "genuine" functions, i.e. they are uniquely defined for all x. When using nonlinear functions as in example

   ```
   f(x):=CHI(0,x,1)*x + CHI(1,x,3)*x^2 + CHI(3,x,inf)*(2x+3)
   ```

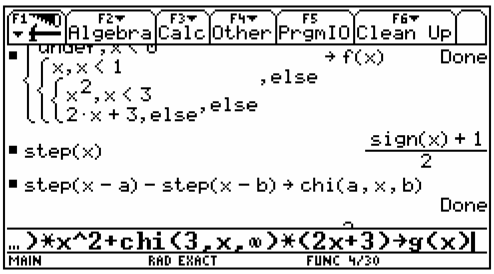
   this fundamental requirement is usually not fulfilled if x belongs to the set $[x_1, x_2, ..., x_n]$. (It must be said though that for x_1 or x_n- or even both – an adjustment of the function value may be necessary.

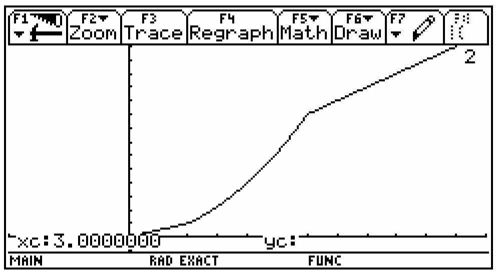
3. `CHI(a,x,b)` is defined according to its current definition in order to achieve total compatibility.

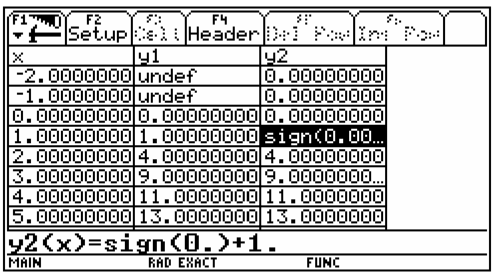
In my opinion, this modification of `CHI` is worth considering. What does the *DERIVE* community think of it?

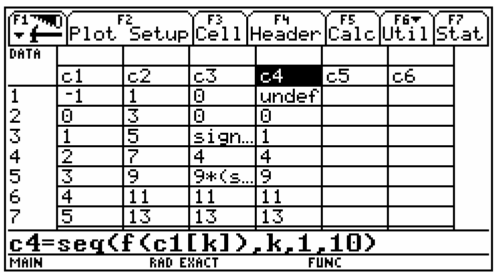
Cheers, Johann.

It is interesting how the TIs are treating piecewise functions. In some respects they behave similar – and this is not really surprising but Nspire has a special feature.









The Voyage 200 (and TI-92) does not know the CHI-statement. But following a contribution of our Canadian experts (DNL#91) it is easy to implement. Instead of IF in *DERIVE* we have to use the when-construct. When investigating the table we can observe the same deficiencies of the CHI-defined function. Problems will appear when generating a value table in the Data-Matrix Editor. In column c3 I defined c3 = g(c1) – which works but f(c1) in column c2 does not work correctly (f(1) = 1!!). So one cannot apply a when-generated function on a whole column. What to do? You can follow the construction of c4: define again a sequence addressing each single cell in column c1.

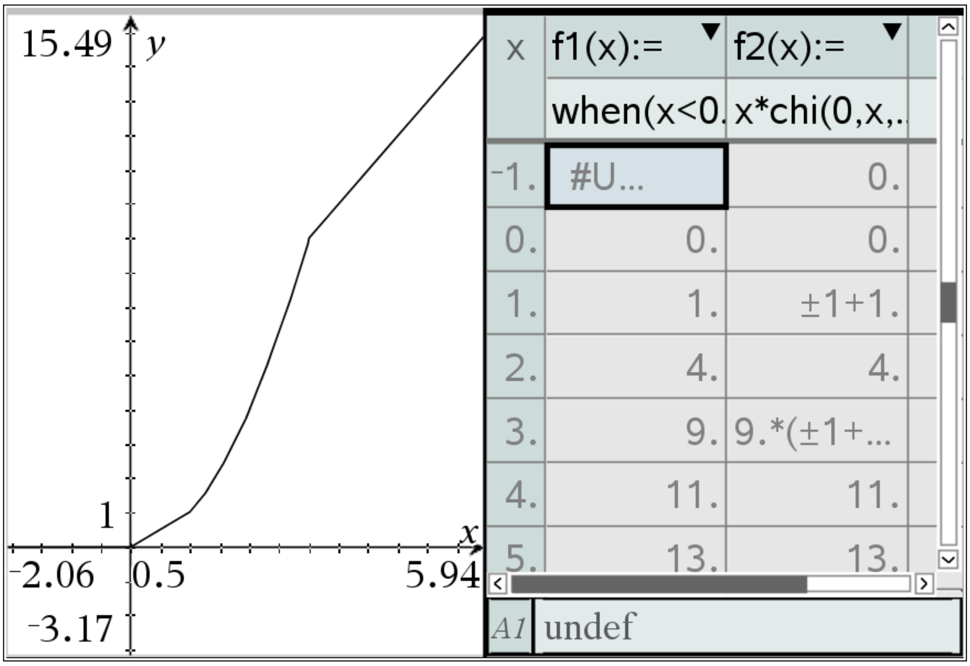
Turning to TI-Nspire we can do the same as with the Voyage 200 – and we face the same problems.

$$f1(x):=\text{when}\left(x<0,\text{undef},\text{when}\left(x<1,x,\text{when}\left(x<3,x^2,2\cdot x+3\right)\right)\right) \quad \textit{Done}$$

$$step(x):=\frac{\text{sign}(x)+1}{2} \quad \textit{Done}$$

$$chi(x):=step(x-b)-step(x-b) \quad \textit{Done}$$

$$f2(x):=x\cdot chi(0,x,1)+x^2\cdot chi(1,x,3)+(2\cdot x+3)\cdot chi(3,x,\infty) \quad \textit{Done}$$



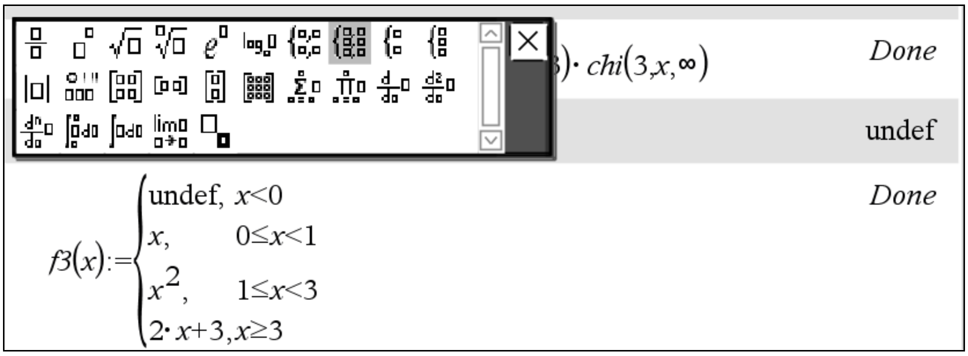
The left half of the screen shows the function value tables. Corresponding to the Data-Matrix Editor we want to generate a value table in the Lists & Spreadsheet Application. There we find that it is not possible to apply the "seq-trick" from above (see col. E next page). What we can do is presented in col. D: calculate the value in *D1* and then copy down the cell in the column. This works!!

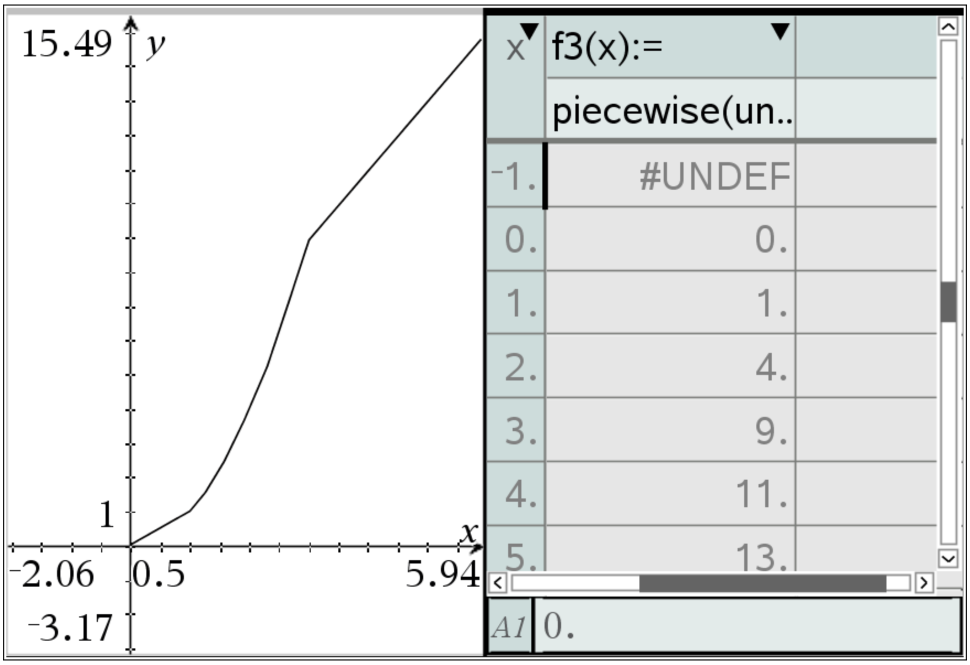
But we have a better solution with TI-Nspire: use the tablet for generating a piecewise defined function. This yields a perfect definition which meets all our requirements.

	A	B	C	D	E
=		=f1(a[])	=f2(a[])		=seq(f1(a[k]),
1	-1	1	0	#UNDEF	piecewise(u...
2	0	3	0	0	piecewise(u...
3	1	5	±1+1	1	piecewise(u...
4	2	7	4	4	piecewise(u...
5	3	9	9*(±1+1)	9	piecewise(u...
6	4	11	11	11	piecewise(u...
7	5	13	13	13	piecewise(u...

D1 =f1(a[1])

Use the tablet for defining the piecewise defined function:





Josef Böhm, Würmla

Working with the *TI-92* in my first form I made the following experience: I prepared an example for a written test:

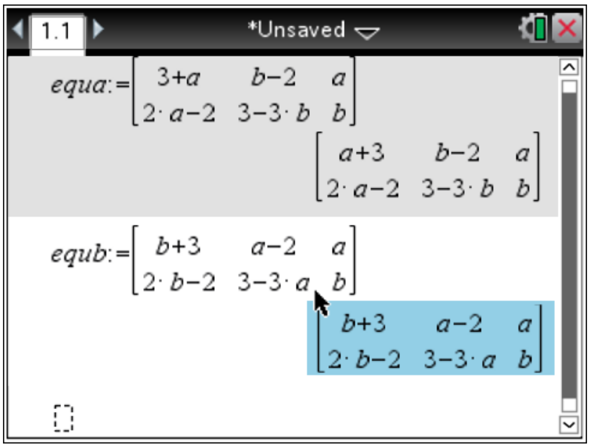
A linear system is given: $$\begin{aligned} a\,x + b\,y &= a + 2y - 3x \\ 2a\,x - 3b\,y &= b - 3y + 2x \end{aligned}$$

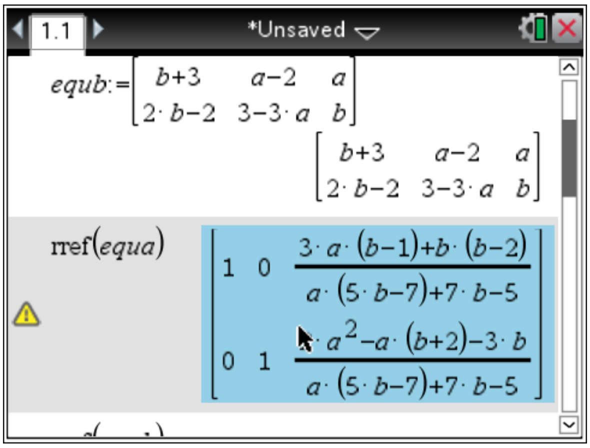
a) Solve this system for x and y.

b) Find at least one pair (a,b) such that the system has no resolution.

My idea was to check if the students were able to bring the system in a form to pick out the coefficients for writing them into the respective matrix. Then they should apply `rref()` and interpret the solution in order to find values for a and b which let "disappear" the denominator.

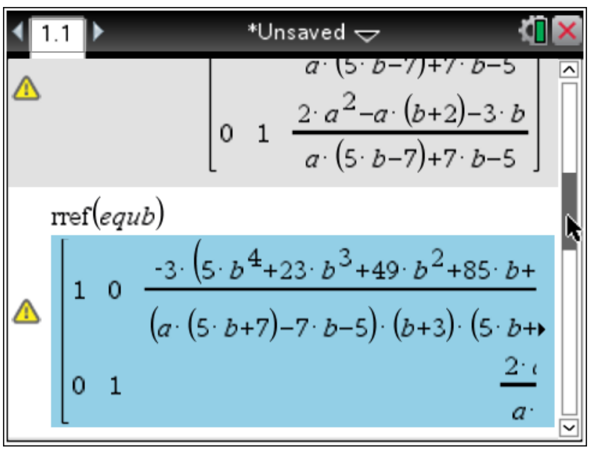
(As TI-Nspire behaves pretty the same as TI-92 and Voyage 200 as well, I present the Nspire screens, Josef)

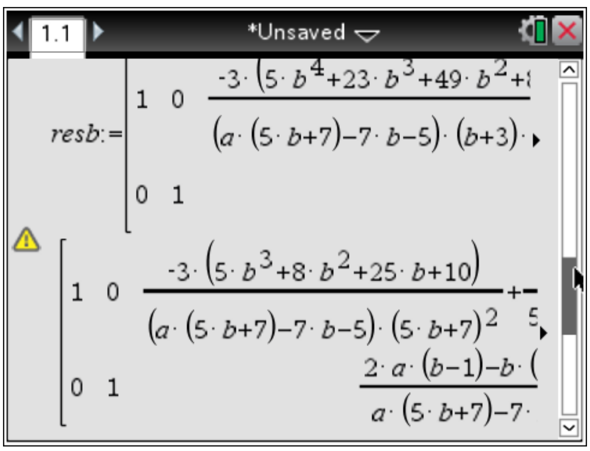




One – female – student gave a very quick response. She wrote a = -3 and b = 2 without using the denominator and explained – during the test: Look at the first row: a = -3 and b = 2 leads to a contradiction, thus the system cannot have any solution!) I was happy to have such students!

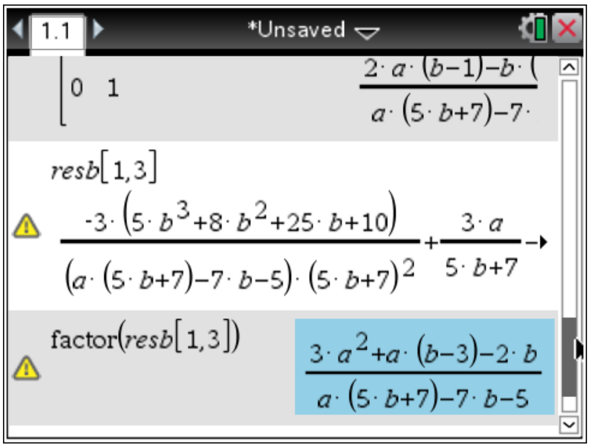
For the second group I just exchanged a and b on the left hand side in the system given above - without checking the further procedure at home. During the test I encountered after the good surprise described above a bad one, too:





The solution for *x* was a huge expression (the students could have taken the solution for *y* but they were too confused to do so). I encouraged the students to either do so or to copy the *x*-solution into their test forms – and made up my mind for the future to check even the “easiest” examples.

Back home I applied `factor` on that crazy expression for *x* and checked the same procedure with *DERIVE* using `row_reduce(equb)`, wich worked properly – as expected. So I wrote a mail to David Stoutemyer reporting my “problem” and very soon I could receive his answer.



David Stoutemyer, Honolulu

This is a nice example of how reordering variables can make a big difference: Variable “*a*” is more main than variable “*b*”, and the choice about whether or not to combine rational expressions over a common denominator depends on the relative primeness of the two denominators. In this case the common denominator form is more compact, but the CAS doesn’t know that in advance and can’t afford to try all combinations.
Aloha, David

There was another question concerning systems of linear equations which initiated a nice discussion. Several answers followed. I’ll publish two of them which might be useful for you.

Ingo Kronberger,

I’d like to get a program for *DERIVE* with that I can do a Gaussian algorithm. I’d like the program to show all steps *DERIVE* is performing. Does someone have such a program or an idea how to do one? Thanks a lot.

Johann Wiesenbauer, Vienna

Here are some ideas that might be useful in this connection. Suppose you have a system of linear equations such as

$$x + y + z = 6$$
$$2x - y + z = 3$$
$$3x + 2y - 2z = 1$$

For my solution I need two functions:

```
PIVOT(a, i, j, dir) ≔ IF(dir, a, VECTOR(IF(dir·m_ < dir·i, a↓m_, IF(m_ = i,
a↓i/a↓i↓j, a↓m_ - a↓m_↓j/a↓i↓j·a↓i)), m_, DIMENSION(a)), PIVOT(a, i, j, 1))
```

The pivot element a_{ij} of the matrix A is used to force all elements of A of it (or above it, if `dir` = -1) to zero. Moreover the row a_i is scaled in such a way that the new a_{ij} becomes 1. If the fourth parameter `dir` is omitted, it is assumed to be 1 by default.

```
NEWA(u) := a := u
```

This tiny auxiliary function is used for "self-assignments", i.e. for assignments a := u where the expression u contains a. The double assignment a:= = is supposed to do this, but it doesn't yet work in *DfW*.

Now a step-by-step solution could look like this:

#4: $a := \begin{bmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 1 & 3 \\ 3 & 2 & -2 & 1 \end{bmatrix}$

#5: $\text{NEWA(PIVOT(a, 1, 1))} = \begin{bmatrix} 1 & 1 & 1 & 6 \\ 0 & -3 & -1 & -9 \\ 0 & -1 & -5 & -17 \end{bmatrix}$

#6: $\text{NEWA(PIVOT(a, 2, 2))} = \begin{bmatrix} 1 & 1 & 1 & 6 \\ 0 & 1 & \frac{1}{3} & 3 \\ 0 & 0 & -\frac{14}{3} & -14 \end{bmatrix}$

#7: $\text{NEWA(PIVOT(a, 3, 3, -1))} = \begin{bmatrix} 1 & 1 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$

#8: $\text{NEWA(PIVOT(a, 2, 2, -1))} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$

This is just an outline of what could be done, but it should be easy to adapt the method above according to your own wishes, in particular, if you want smaller steps or other matrix operations such as swapping rows (cf. the utility-file VECTOR.MTH for this).

NEWA is still working, the double assignment was not "reanimated" in later *DERIVE*-versions. There is one fact in Johann's procedure which can be improved. it is necessary to take variable "a" for the matrix to be transformed. I changed this and made the input a little bit more comfortable, Josef.

```
#9:   m :=

      newm(mx, i, j, dir) :=
#10:    Prog
          m := PIVOT(mx, i, j, dir)
```

#11: $b := \begin{bmatrix} 4 & 1 & 1 & 6 & 20 \\ 2 & -1 & 6 & 3 & -10 \\ 3 & 2 & -2 & 8 & 0 \\ 4 & -5 & 2 & 1 & 2 \end{bmatrix}$

#12: $\text{newm(b, 1, 1)} = m := \begin{bmatrix} 1 & \frac{1}{4} & \frac{1}{4} & \frac{3}{2} & 5 \\ 0 & -\frac{3}{2} & \frac{11}{2} & 0 & -20 \\ 0 & \frac{5}{4} & -\frac{11}{4} & \frac{7}{2} & -15 \\ 0 & -6 & 1 & -5 & -18 \end{bmatrix}$

#13: newm(m, 2, 2) = m :=
$$\begin{bmatrix} 1 & \frac{1}{4} & \frac{1}{4} & \frac{3}{2} & 5 \\ 0 & 1 & -\frac{11}{3} & 0 & \frac{40}{3} \\ 0 & 0 & \frac{11}{6} & \frac{7}{2} & -\frac{95}{3} \\ 0 & 0 & -21 & -5 & 62 \end{bmatrix}$$

#14: newm(m, 3, 3) = m :=
$$\begin{bmatrix} 1 & \frac{1}{4} & \frac{1}{4} & \frac{3}{2} & 5 \\ 0 & 1 & -\frac{11}{3} & 0 & \frac{40}{3} \\ 0 & 0 & 1 & \frac{21}{11} & -\frac{190}{11} \\ 0 & 0 & 0 & \frac{386}{11} & -\frac{3308}{11} \end{bmatrix}$$

#15: newm(m, 4, 4, -1) = m :=
$$\begin{bmatrix} 1 & \frac{1}{4} & \frac{1}{4} & 0 & \frac{3446}{193} \\ 0 & 1 & -\frac{11}{3} & 0 & \frac{40}{3} \\ 0 & 0 & 1 & 0 & -\frac{176}{193} \\ 0 & 0 & 0 & 1 & -\frac{1654}{193} \end{bmatrix}$$

#16: newm(m, 3, 3, -1) = m :=
$$\begin{bmatrix} 1 & \frac{1}{4} & 0 & 0 & \frac{3490}{193} \\ 0 & 1 & 0 & 0 & \frac{1928}{193} \\ 0 & 0 & 1 & 0 & -\frac{176}{193} \\ 0 & 0 & 0 & 1 & -\frac{1654}{193} \end{bmatrix}$$

Note that the classical Gauss elimination ends at expressions #6 and #14 and the rest must be done by substitution.

#17: newm(m, 2, 2, -1) = m := $\begin{bmatrix} 1 & 0 & 0 & 0 & \frac{3008}{193} \\ 0 & 1 & 0 & 0 & \frac{1928}{193} \\ 0 & 0 & 1 & 0 & -\frac{176}{193} \\ 0 & 0 & 0 & 1 & -\frac{1654}{193} \end{bmatrix}$ ROW_REDUCE(b) = $\begin{bmatrix} 1 & 0 & 0 & 0 & \frac{3008}{193} \\ 0 & 1 & 0 & 0 & \frac{1928}{193} \\ 0 & 0 & 1 & 0 & -\frac{176}{193} \\ 0 & 0 & 0 & 1 & -\frac{1654}{193} \end{bmatrix}$

This procedure offers another occasion to demonstrate that in many cases useful *DERIVE*-routines can be transferred to TI-NspireCAS.

$x:=\begin{bmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 1 & 3 \\ 3 & 2 & -2 & 1 \end{bmatrix}$ $\quad \begin{bmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 1 & 3 \\ 3 & 2 & -2 & 1 \end{bmatrix}$

$nx:=pivot(x,1,1,1)$ $\quad \begin{bmatrix} 1 & 1 & 1 & 6 \\ 0 & -3 & -1 & -9 \\ 0 & -1 & -5 & -17 \end{bmatrix}$

$nx:=pivot(nx,2,2,1)$ $\quad \begin{bmatrix} 1 & 1 & 1 & 6 \\ 0 & 1 & \frac{1}{3} & 3 \\ 0 & 0 & \frac{-14}{3} & -14 \end{bmatrix}$

$nx:=pivot(nx,3,3,-1)$ $\quad \begin{bmatrix} 1 & 1 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$

$nx:=pivot(x,1,1,1)$ $\quad \begin{bmatrix} 1 & 1 & 1 & 6 \\ 0 & -3 & -1 & -9 \\ 0 & -1 & -5 & -17 \end{bmatrix}$

$nx:=pivot(nx,2,2,1)$ $\quad \begin{bmatrix} 1 & 1 & 1 & 6 \\ 0 & 1 & \frac{1}{3} & 3 \\ 0 & 0 & \frac{-14}{3} & -14 \end{bmatrix}$

$nx:=pivot(nx,3,3,-1)$ $\quad \begin{bmatrix} 1 & 1 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$

```
"pivot" stored successfully
Define pivot(v,i,j,dir)=
Func
Local k
For k,1,dim(v)[1]
  If dir·k<dir·i Then
   v[k]:=v[k]
   ElseIf k=i Then
    v[k]:=v[k]/v[k,j]
   ElseIf dir·k>dir·i Then
    v[k]:=v[k]-v[k,j]/v[i,j]·v[i]
EndIf
EndFor
v
EndFunc
```

<u>**David Stenenga, Honolulu**</u>

As noted *DERIVE* has some nice functions doing Gauss-Jordan reduction of a matrix one step at a time using `SWAP_ELEMENTS(v,i,j)`, `SCALE_ELEMENT(v,i,s)` and `SUBTRACT_ELEMENT(v,i,j,s)` and the automatic method using `ROW_REDUCE(A,B)`.

Since I'm teaching elementary linear algebra this semester I got thinking about this process and the fact that it would be interesting to see all the steps and the elementary matrices that go with each of the steps. As a demonstration program for my students I wrote some slight modifications of these functions and came up with the following:

The new functions are `ROW_SWAP(v,k)`, `ROW_SCALE(v)`, `ROW_SUBTR(v,k)` and `NEXT_ROW(v)` Here `v` is not the matrix but instead a state vector:

`[e,a,i,j,d,flag]`

where `e` is a vector of elementary matrices, `a` is the current reduction of the matrix, (`i,j`) is the pivot position, `d` is used for computing the determinant and `flag` is set to 1 if the current column is all zeros.

For example, you start with say [[],[0,2;2,-1],1,1,1,0] and apply the various functions using F3 or F4 keys to put the last output into the next function. This would yield:

(File DS-GAUSS.MTH has been preloaded as utility file.)

#1: LOAD(D:\DOKUS\DNLs\DNL98\MTH32\DS-GAUSS.MTH)

#2: $\left[[\,],\ \begin{bmatrix}0 & 2\\ 2 & -1\end{bmatrix},\ 1,\ 1,\ 1,\ 0\right]$

#3: $\text{ROW\_SWAP}\left(\left[[\,],\ \begin{bmatrix}0 & 2\\ 2 & -1\end{bmatrix},\ 1,\ 1,\ 1,\ 0\right],\ 2\right)$

#4: $\left[\left[\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\right],\ \begin{bmatrix}2 & -1\\ 0 & 2\end{bmatrix},\ 1,\ 1,\ -1,\ 0\right]$

#5: $\text{ROW\_SCALE}\left[\left[\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\right],\ \begin{bmatrix}2 & -1\\ 0 & 2\end{bmatrix},\ 1,\ 1,\ -1,\ 0\right]$

#6: $\left[\left[\begin{bmatrix}\frac{1}{2} & 0\\ 0 & 1\end{bmatrix},\ \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\right],\ \begin{bmatrix}1 & -\frac{1}{2}\\ 0 & 2\end{bmatrix},\ 1,\ 1,\ -2,\ 0\right]$

#7: $\text{NEXT\_ROW}\left[\left[\begin{bmatrix}\frac{1}{2} & 0\\ 0 & 1\end{bmatrix},\ \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\right],\ \begin{bmatrix}1 & -\frac{1}{2}\\ 0 & 2\end{bmatrix},\ 1,\ 1,\ -2,\ 0\right]$

#8: $\left[\left[\begin{bmatrix}\frac{1}{2} & 0\\ 0 & 1\end{bmatrix},\ \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\right],\ \begin{bmatrix}1 & -\frac{1}{2}\\ 0 & 2\end{bmatrix},\ 2,\ 2,\ -2,\ 0\right]$

The automatic approach is done using `SHOW_GAUSS(A,B)`. Here is an example:

#9: $\text{SHOW\_GAUSS}\left(\begin{bmatrix}0 & 2\\ 2 & -1\end{bmatrix},\ \text{IDENTITY\_MATRIX}(2)\right)$

Note that the first column is just a counter, the second contains the elementary matrix which reduces the matrix to the one in the third column. The fourth column is the current value of the B-matrix and the last one is the determinant of the elementary matrix.

The determinant of A is just the reciprocal of the product of the fifth column entries. This is an n^3 algorithm for computing the determinant which improves the n! calculation obtained from the usual definition.

#9: $\text{SHOW\_GAUSS}\left(\begin{bmatrix} 0 & 2 \\ 2 & -1 \end{bmatrix}, \text{IDENTITY\_MATRIX}(2)\right)$

#10:
$$\begin{bmatrix}
0 & \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} & \begin{bmatrix} 0 & 2 \\ 2 & -1 \end{bmatrix} & \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} & 1 \\
1 & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & \begin{bmatrix} 2 & -1 \\ 0 & 2 \end{bmatrix} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & -1 \\
2 & \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 1 \end{bmatrix} & \begin{bmatrix} 1 & -\frac{1}{2} \\ 0 & 2 \end{bmatrix} & \begin{bmatrix} 0 & \frac{1}{2} \\ 1 & 0 \end{bmatrix} & \frac{1}{2} \\
3 & \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{2} \end{bmatrix} & \begin{bmatrix} 1 & -\frac{1}{2} \\ 0 & 1 \end{bmatrix} & \begin{bmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & 0 \end{bmatrix} & \frac{1}{2} \\
4 & \begin{bmatrix} 1 & \frac{1}{2} \\ 0 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} & \begin{bmatrix} \frac{1}{4} & \frac{1}{2} \\ \frac{1}{2} & 0 \end{bmatrix} & 1
\end{bmatrix}$$

Aloha, David.

(Another – more extended – example is included in the file DS_GAUSS.MTH, Josef)

Albert Rich, Soft warehouse, Honolulu

Hello Josef,

the following might make a good DUG Newsletter item since the DERIVE on-line help does not yet reflect this extension of the SUB operator:

Version 4 of DERIVE for WINDOWS (DfW) and DERIVE for DOS (DfD) introduced the SUB operator for extracting the elements of vectors and matrices. As explained in the DfW on-line help, you can use the SUB operator twice to extract the elements of a matrix. For example:

`[2,3,5;7,1,4;4,6,8] SUB 3 SUB 2` simplifies to the 2nd element of the 3rd row (i.e.6)

$$\begin{bmatrix} 2 & 3 & 5 \\ 7 & 1 & 4 \\ 4 & 6 & 8 \end{bmatrix}_{3,2} = 6$$

Beginning with version 4.06 of DfW and DfD, the SUB operator was given additional capabilities. If the right operand of the SUB operator was a vector of indices a vector of those elements is returned. For example,

`[2,3,5;7,1,4;4,6,8] SUB [3,2]` simplifies to a matrix consisting of the 3rd and 2nd row of the above matrix, that is

$$\begin{bmatrix} 2 & 3 & 5 \\ 7 & 1 & 4 \\ 4 & 6 & 8 \end{bmatrix}_{[3,\ 2]} = \begin{bmatrix} 4 & 6 & 8 \\ 7 & 1 & 4 \end{bmatrix}$$

Aloha, Albert.

In the on-line help of DERIVE 6 this feature of the SUB operator is explained among the "Vector Manipulation Functions", Josef.

Parametric 3d Plots with *DERIVE* and 3dV

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

DERIVE does not offer 3d plots. A triplet of *DERIVE* functions is presented to produce data of parametric curve and surface plots, which can be animated with the freeware 3dV.exe (which is among the DNL32 files).

Introduction

In the *DERIVE* User Forum, Oscar Garcia pointed out that it is possible to animate *DERIVE* 3d plots with his freeware program 3dV.exe. I immediately downloaded 3dV and was impressed by the power of this tiny 57KB program and by the simple and transparent input data format of this 3d viewer. I am citing the 3dV.doc file:

```
                    DATA FORMAT

Data is given in ASCII text files, in free format, in the following
order:

    number of points
    x, y, and z coordinates for each point (scaling is automatic)
         ...
    number of connecting lines or moves (number of items below)
    point number to draw or to move to,  and color (color=0 for
    move)
    ...

Yes, a bit obscure, but see the sample files for examples.

Capacity is limited by 10*(number of points) + 4*(moves and draws)
<= 58280.
```

A triangle with vertices (1,0,0), (0,1,0) and (0,0,1) for example yields the following 3dV input file:

```
3
1 0 0
0 1 0
0 0 1
4
1 0
2 4
3 4
1 4
```

0 is black in 16 colour mode. We are thus jumping unseen from nowhere to the first point. The connecting lines are coloured red, this is 4 in 16 colour mode. (The complete colour table is given an expression #71 in the file TO3DV.MTH.) I cannot find this obscure, this is a particular version of adjacency lists, as described in DNL#26 by Hartmut Kümmel. O. Garcia's input format has the advantage of minimizing redundancies. By the way, 3dV is a good tool for students to design wire frames of solids by hand, i.e. with an ASCII editor.

Parametric 3d plots

It is not possible, until now, to plot parametric curves and surfaces in R^3 with *DERIVE*, although *DERIVE* apparently plots a mesh of parametric lines in 3d plots. But the user has no access to this feature. One way around is to use projections of 3d plots to the plane.

3dV opens another way to display 3d objects: We first produce 3d data with *DERIVE*, we then export these data to an ASCII file and finally we display our object with 3dV. This is an improvement to exporting *DERIVE* 3d plots, which are not parametric, to ACROSPIN or 3dV.

To do the first step, we must construct functions which produce appropriate data. As *DERIVE* cannot produce data in the desired form we use lists instead. Then the triangle in the example above becomes:

```
#1: [[3],[1,0,0],[0,1,0],[0,0,1],[4],[1,0],[2,4],[3,4],[1,4]]
```

Parametric plot data in this form can easily be computed with *DERIVE*. The following *DERIVE* code prepares the definitions of the three main functions `PARAMETRIC_CURVE_DATA(...)`, `PARAMETRIC_SURFACE_DATA(...)` and `MERGE_DATA(...)`.

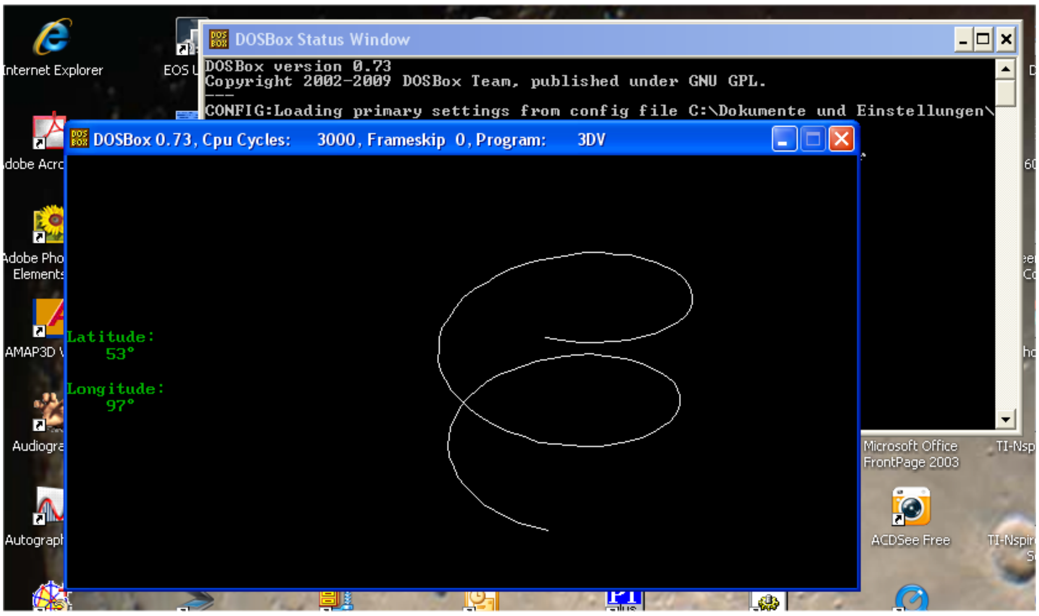
The functions for parametric curve and surface data

```
PARAMETRIC_CURVE_DATA(expr_,r_,col_):=APPEND(APPEND([[r_↓3]],
VECTOR(ITERATE(expr_,VARIABLES(expr_),k_(r_↓2-r_↓1)/(r_↓3-1),1),
k_,0,r_↓3-1)),APPEND([[r_↓3]],VECTOR([j_,IF(MOD(j_,r_↓3)≠1,col_,0)],
j_,1,r_↓3)))
```

`PARAMETRIC_CURVE_DATA([x(t),y(t),z(t)],[a,b,n],col_)` produces data of n points of the parametric curve $[x(t), y(t), z(t)]$ for t in the interval $[a, b]$. This interval is subdivided into $n-1$ parts, i.e. n equidistant points $p_1 = [x(a), y(a), z(a)]$, ..., $p_n = [x(b), y(b), z(b)]$ are calculated. *col_* is an integer value $0 \le col\_ \le 15$. 0 denotes black that is no colour (back ground colour).

```
PARAMETRIC_CURVE_DATA([COS(t),SIN(t),0.3·t],[0,4π,100],15)
```

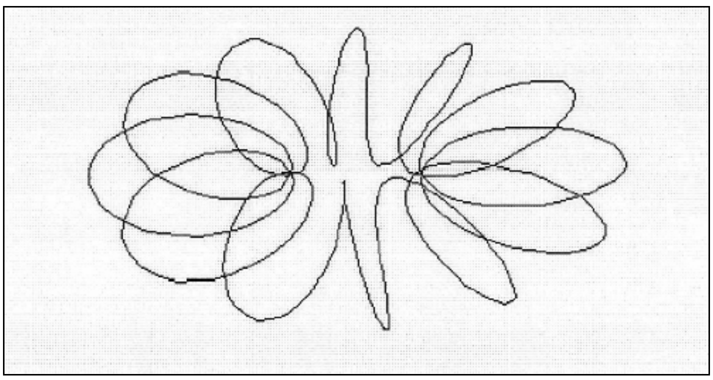
results in a white helix (on a black background).



As you can see, the procedure is still working – under certain conditions. My PC runs under WIN XP. Applying command to switch to the DOD-mode works but 3dv.exe does not work. Then I tried a utility DOS Box – and it works with one deficiency: The program documentation says that pressing any button leads back to the initial 3dV-screen. I have to end the program by applying the task manager.

An **example** (Preload the utility file TO3DV.MTH):

```
#1: "A spiral winding around a torus"

#2: TORUS(v):=[(2+v↓3COS(v↓1))COS(v↓2),(2+v↓3COS(v↓1))SIN(v↓2),v↓3SIN(v↓1)]

#3: LINE(t,m,r):=[t,m·t,r]

#4: PARAMETRIC_CURVE_DATA(TORUS(LINE(t,1/12,6/5)),[0,24π,400],15)
```



```
PARAMETRIC_SURFACE_DATA(expr_,xr_,yr_,col_):=
          APPEND(POINTS_(expr_,xr_,yr_),LINES_(xr_,yr_,col_))
```

`PARAMETRIC_SURFACE_DATA([x(s,t),y(s,t),z(s,t)],[a,b,n],[c,d,m],[cs,ct])` produces data of the parametric surface $[x(s,t), y(s,t), z(s,t)]$ with $s \in [a, b]$ and $t \in [c,d]$, with n points in interval $[a,b]$ and m points in interval $[c,d]$. *cs* and *ct* specify the colours of the *s*- and *t*-parameter curves respectively, so the two parameters can be distinguished graphically. If you don't want different colours for the parameter lines, then you can use [c] as fourth argument.

The order of the intervals is prescribed by the built-in function `VARIABLES`. For example `VARIABLES([s,s*t^2,t]` returns `[s,t]`, so the first interval belongs to the parameter *s*, the second one to *t*.

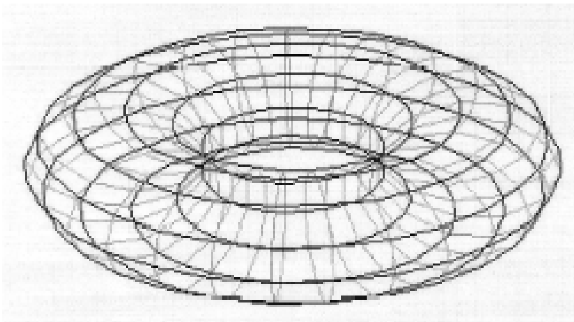
The next example uses two colours:

```
#5: PARAMETRIC_SURFACE_DATA(TORUS([s,t,1]),[0,2π,12],[0,2π,30],[8,15])
```

The figures displaying the spiral and the torus are original figures from DNL#32 (1998).

All other figures are from 2014.

At the end of Stefan's contribution we will generate the objects with *DERIVE* 6 and *TI-Nspire* as well (page 15).

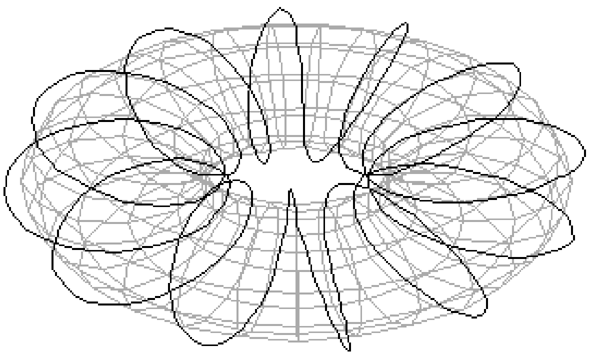


Combining data of two objects

```
MERGE_DATA(v_,w_):=F2_(v_,w_)+F3(w_,v_)
```

`MERGE_DATA(data1,data2)` combines data of two distinct pre-3dV objects to one single pre-3dV data object. Now we merge the torus and the spiral:

```
#6: MERGE_DATA(#5,#4)
```



This picture is made 2014 applying again the DOS Box (page 21) and the *DERIVE* file from 1998. I followed Stefan's instructions and produced a screen shot. Finally I reversed the colours using a graphics program, Josef.

Procedure of conversion to 3d-format

PARAMETRIC_CURVE_DATA, PARAMETRIC_SURFACE_DATA and MERGE_DATA produce vectors of a structure which is appropriate for conversion to 3d-format of Garcia's 3dV (= 3d Viewer). In order to convert the data for 3dV proceed as follows:

You first need the free awk-interpreter mawk.exe which is described by O. Garcia in his DNL#25 contribution. Then you need the script file convert.awk which is a simple ASCII-file:

convert.awk

```
# mawk -f convert.awk datei.in > datei.out

{RS="\]"; z=NF ;
 { for(i=1;i<=z;i++)
  {
    getline < FILENAME      ;
    gsub("\*POW\(10,","E")  ;
    gsub("\)","")           ;
    gsub("\[","",$0)        ;
    gsub(","," ",$0)        ;
    print($0)
  }
 }
}
```

MAWK strips brackets and commas off in the input.pas file. Finally you need a batch file convert.bat which manages the conversion job:

convert.bat

```
REM Conversion batch file  infile.pas -> outfile.3d
mawk -F "\]" -f convert.awk %1 > %2
del %1
```

After *DERIVE* has calculated the data, save the resulting data-vector as a PASCAL-file (File > Write > Pascal file ...), say data.pas. Then call the batch file convert.bat with the input file data.pas and say figure.3d as arguments: convert data.pas figure 3d. Now you are ready to load figure.3d with 3dV.exe and animate the figure **with the mouse**.

mawk.exe, convert.awk and convert.bat must be in the same directory (folder). Notice that the original *.pas-file will be deleted after conversion.

Limitations

Limitations are due to three reasons:

1 Memory of *DERIVE*. The DOS version does better that *DfW*. Too many points produce too long lines in *DfW*. In the DOS version the limit on my machine seems to be a 15× 15 grid for the surfaces and 300 points for curves.

2 Memory usage of the mawk interpreter when converting the *.pas files to *.3d files.

3 The maximum number of 10∗(number of points)+4∗(moves and draws) < 58280 for 3dV.exe.

In my practise the first item is the real restriction while the last one is only a theoretical restriction. I assume that *DERIVE XM* overcomes these difficulties. Because I have no access to this version I calculated the data for the examples in this paper with *Mathematica.*

(*I tried with DERIVE XM and could reproduce Stefan's examples without any problems. From earlier experiences with similar problems (ACD, ACROSPIN) I recommend to reduce the DERIVE-output to 3 decimal digits. DERIVE versions 4 for DOS and WINDOWS as well didn't accept the MERGE command. Trying to overcome this obstacle I found out that in two auxiliary functions several times a SUB[1,1] was used which worked in earlier DERIVE versions. Now you have to write SUB 1 SUB 1 instead. I sent a message to Stefan before inspecting my mailbox and then found his message that he has found out that ..., so we had the same idea nearly at the same time. It might be of interest for you to change your DERIVE code if you want to use well-tried DERIVE files with an upgraded DERIVE version. See also Albert Rich's message in the User Forum. Josef*)

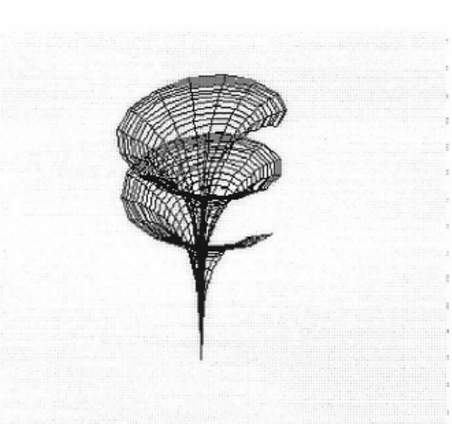
Further examples

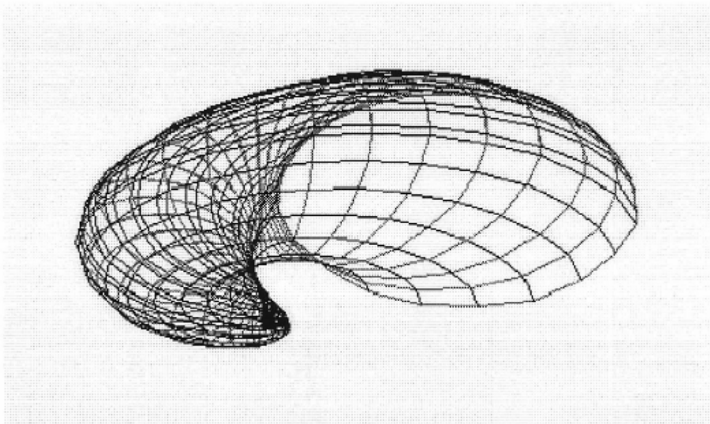
I close with two examples. Dini's surface is a twisted pseudosphere. This is a surface with constant negative curvature. Its explicit parameterization is:

```
DINI(s,t):=[COS(s)SIN(t),SIN(s)SIN(t),COS(t)+LOG(TAN(t/2))+0.2·s]

PARAMETRIC_SURFACE_DATA(DINI(s,t),[0,4π,30], [0.01,2,20],[4,1])
```

The sea shell is defined as follows:

```
SHELL(r,s):=a·r·[COS(r)(b+COS(s)),SIN(r)(b+COS(s)),c+SIN(s)]

[a ≔ 1, b ≔ 1, c ≔ 1.3]
PARAMETRIC_SURFACE_DATA(SHELL(r,s),[0,2π,30], [0,3π,30],[4,5])
```



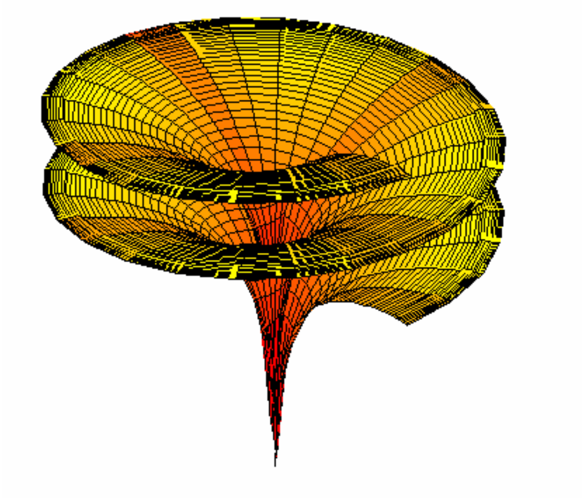


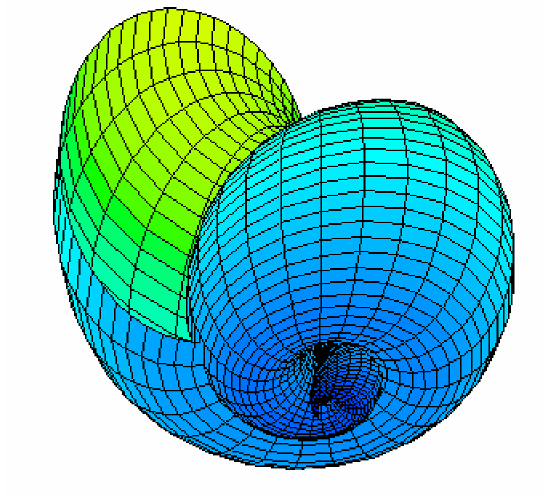
Originals from 1998

References

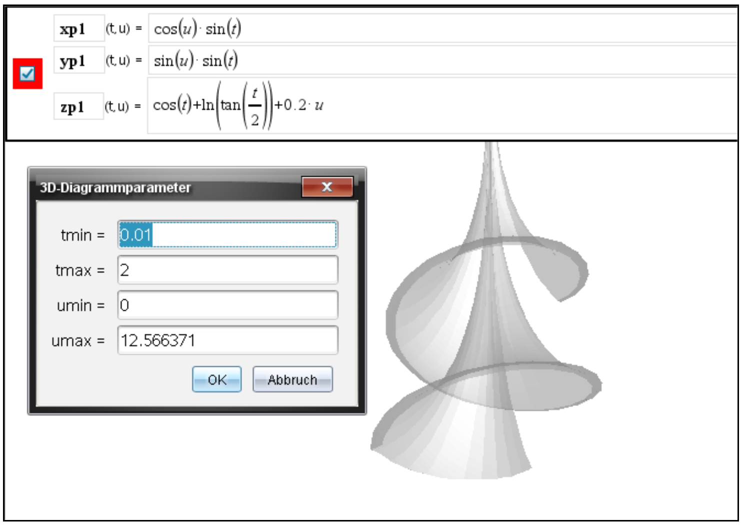
1. O. Garcia, DNL#25, Derive User Forum
2. 3DV25.zip and mawk122.zip can now (2014) be found at
   http://cd.textfiles.com/simtel/simtel20/MSDOS/GRAPHICS/
   http://www.filewatcher.com/m/mawk122x.zip.102254-0.html
3. A. Gray, Moder Differential Geometry of Curves and Surfaces, CRC PRESS, 1993

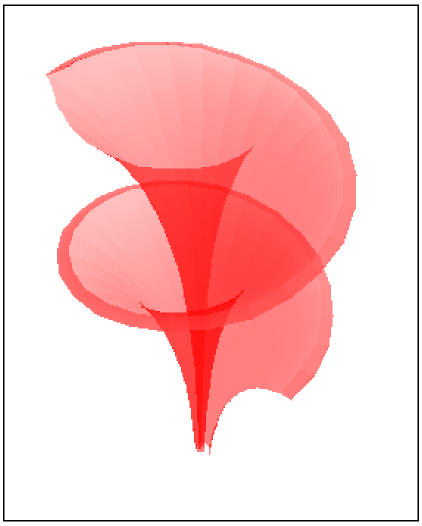
   and among the DNL#32 files

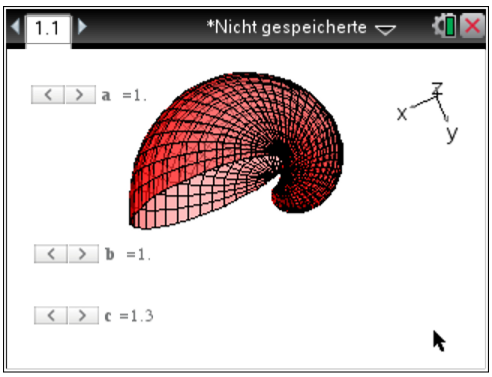


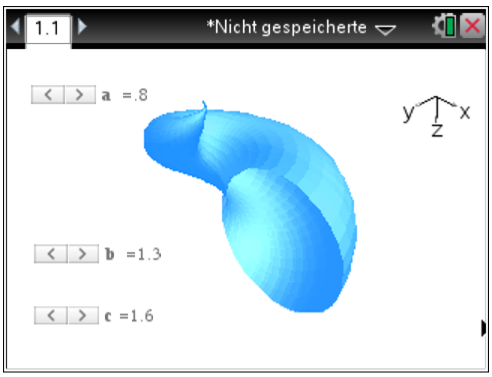


DERIVE 6 (above) and *TI-NspireCAS* (below)









AN EXERCISE ON THE EVALUATION OF THE MODIFIED BESSEL FUNCTIONS IN DERIVE

S.Abraham Ibrahim and P. Fernandez de Córdoba
Departamento de Matemática Aplicada,Universidad Politécnica de Valencia, Spain

Abstract

Using the continued fraction method in *DERIVE* we present an algorithm to calculate Modified Bessel functions of fractional and integer order based on this method. Using Derive we also compare with the traditional Miller s algorithm. *DERIVE* program files and some illustrations of the results are inc1uded.

1. Introduction

When a given function is approximated using the continued fraction representation, we observe that it generally converges faster than other approximations i.e. Taylor expansion. We insist on this in section 2, where we also present as an example the continued fraction representation of the exponential integrals. In sections 3 and 4, we present the algorithm to calculate Modified Bessel functions (MBFs) of integer and fractional orders, based on the continued fraction method.

We compare our algorithm (which as we show is very efficient), with the usual numerical methods (in section 5) which calculate MBFs taking into account normalization relations [1,2].

In this paper we introduce an algorithm and corresponding *DERIVE* codes to evaluate regular and irregular MBFs without any re-calculation through normalization relations. Furthermore, the method maintains the stability of each recurrence relation, i.e., we use forward recurrence relations for the MBFs of the second kind and backward ones[3] for the MBFs of the first kind. The algorithm uses forward recurrence relations to generate irregular MBFs and takes into account the continued fraction method to evaluate high order regular MBFs. From these values we can generate regular MBFs applying backward recurrence relations.

In Section 6 we present the structure of the programs. A comparative table with some results is presented in section 7.

For the importance of *DERIVE* software as support for teaching and researching in science, we evaluate the continued fractions representation of a function and its application to the calculation of MBFs with a code to be used with *DERIVE*.

2. Evaluation of continued fractions

A continued fraction is given by[4,5]

$$f(x) = b_0 + \cfrac{a_1}{b_1 + \cfrac{a_2}{b_2 + \cfrac{a_3}{b_3 + \cfrac{a_4}{b_4 + \dots}}}} \tag{1}$$

Printers prefer using for expression (1) the symbol

$$f(x) = b_0 + \frac{a_1}{b_1 +}\frac{a_2}{b_2 +}\frac{a_3}{b_3 +}\frac{a_4}{b_4 + \dots} \tag{2}$$

In the previous expressions the a's and b's can themselves be functions of x. We consider, as an example[4], the following continued fraction representation oft he exponential integral

$$E_n(z) = e^{-z}\left[\cfrac{1}{z+\cfrac{n}{1+\cfrac{1}{z+\cfrac{n+1}{1+\cfrac{2}{z+\cfrac{n+2}{1+\cfrac{3}{z+\cfrac{n+3}{1+\ldots}}}}}}}}\right. \tag{3}$$

i. e. for this case[*]

$$\begin{aligned} &a_1 = 1; a_{2k} = n + k - 1; a_{2k+1} = k \qquad (k \ge 1) \\ &b_1 = z; b_2 = 1; b_k = b_{k-2} \qquad\qquad\; (k \ge 3) \end{aligned} \tag{4}$$

Continued fractions frequently converge much more rapidly than power series expansions, and in a much larger domain (not necessarily including the domain of convergence of the series). How- ever, sometimes, the continued fraction converges best where the series does worst, although this is not a general rule. You can just evaluate equation (1) from left to right, stopping when the change is small. Written in the form of (l), the only way to evaluate the continued fraction is from right to left, first deciding how far out to start. This is not the right way[4].

The appropriate way to do this, is to use the following result that relates the continued fractions expansions and rational approximations, and gives a means of evaluating (l) from left to right. Let f_n denote the result of evaluating (1) with coefficients though a_n and b_n.

Then[4]

$$f_n = \alpha_n / \beta_n , \tag{5}$$

where α_n and β_n are given by the following recurrence

$$\begin{aligned} &\alpha_{-1} = 1, \beta_{-1} = 0 \\ &\alpha_0 = b_0, \beta_0 = 1 \end{aligned} \tag{6}$$

and

$$\begin{aligned} &\alpha_k = b_k\, \alpha_{k-1} + a_k\, \alpha_{k-2}; \quad k = 1,2,....n \\ &\beta_k = b_k\, \beta_{k-1} + a_k\, \beta_{k-2}; \quad k = 1,2,....n \end{aligned} \tag{7}$$

In the next sections, we apply this method to evaluate Modified Bessel functions of integer and fractional order.

3. Modified Bessel functions of integer order

We are interested in presenting a code in *DERIVE* to generate the Modified Bessel functions (MBFs). (We restrict our attention to real values of the argument z).

[*] When the a_1's and b_1's) are all equal then we may use the ITERATES command of *DERIVE* for calculating the continued fraction.

We use the standard Abramowitz and Stegun [5] notation and we introduce the MBFs $I_n(z)$ and $K_n(z)$, as a particular solution of the differential equation

$$z^2\omega''(z) + z\,\omega'(z) - (z^2 + n^2)\,\omega(z) = 0 \qquad n = 1,2,3,...$$

In the code, we calculate the MBFs of all orders below N_{max}, i.e. we generate the set

$$MB(z) = \{I_n(z), K_n(z);\; n = 0,1,2,...,N_{max}\}.$$

For this, we use an algorithm organized according the following steps:

- Evaluate all the MBFs of the second kind $\{K_n(z),\, n = 0, 1, 2, ..., N_{max}\}$, taking into account the known values[5] of $K_0(z)$ and $K_1(z)$ and using the forward recurrence relation

$$K_{n+1}(z) = \frac{2n}{z} K_n(z) + K_{n+1}(z). \tag{8}$$

- Use the continued fraction method [5] to evaluate the ratio

$$FC(z) = \frac{I_{N_{max}}(z)}{I_{N_{max}-1}(z)} = \cfrac{1}{\cfrac{2N_{max}}{z} + \cfrac{1}{\cfrac{2(N_{max}+1)}{z} + \cfrac{1}{\cfrac{2(N_{max}+2)}{z} + \cfrac{1}{\cfrac{2(N_{max}+3)}{z} + ...}}}} \tag{9}$$

- Calculate the upper order MBFs of the first kind, $I_{N_{max}}(z)$, using the already known values $K_{N_{max}}(z)$ and $K_{N_{max}-1}(z)$, the ratio $FC(z)$, and the value of the Wronskian of the MBFs [5]

$$W\{I_{N_{max}-1}(z), K_{N_{max}-1}(z)\} = I_{N_{max}}(z)\,K_{N_{max}-1}(z) + I_{N_{max}-1}(z)\,K_{N_{max}}(z) = \frac{1}{z}$$

Using the previous expressions, we can write:

$$I_{N_{max}}(z) = \frac{1}{z\left(K_{N_{max}-1}(z) + \dfrac{K_{N_{max}}(z)}{FC(z)}\right)} \tag{10}$$

and then,

$$I_{N_{max}-1}(z) = \frac{I_{N_{max}}(z)}{FC(z)}. \tag{11}$$

Notice that we have calculated not only $I_{N_{max}}(z)$ but also $I_{N_{max}-1}(z)$.

- Evaluate all the MBFs of the first kind, $\{I_n(z),\, n = 0, 1, 2, ..., N_{max}\}$, taking into account the calculated values of $I_{N_{max}}(z)$ and $I_{N_{max}-1}(z)$, and using the backward recurrence relation:

$$I_{n-1}(z) = \frac{2n}{z} I_n(z) + I_{n+1}(z). \tag{12}$$

4. Modified Bessel functions of fractional order

We extend the algorithm in the previous section, to generate the spherical modified Bessel functions (SMBFs) of the third kind and first kinds, restricting our attention to real values of the argument z.

We introduce the SMBFs of the third kind, $k_n(z)$, and the first kinds, $i_n(z)$ as particular solutions of the differential equation

$$z^2\omega''(z)+2z\,\omega'(z)-[z^2+n(n+1]\;\omega(z)=0$$

In the code, wie calculate SMBFs of all orders below N_{max}, i.e., we generate the set

$$SMB(z) = \{i_n(z), k_n(z);\ \ n = 0, 1, 2, ..., N_{max}\}$$

- Evaluate all the SMBFs of the third kind, $\{k_n(z), n = 0, 1, 2, ..., N_{max}\}$, taking into account the known values of $k_0(z)$ and $k_1(z)$ using the forward recurrence relation:

$$k_{n+1}(z)=\frac{2n+1}{z}k_n(z)+k_{n-1}(z). \qquad (13)$$

- Use the continued fraction method to evaluate the ratio

$$FC(z)=\frac{i_n(z)}{i_{n-1}(z)}=\frac{I_{n+\frac{1}{2}}(z)}{I_{n-\frac{1}{2}}(z)}=\cfrac{1}{\cfrac{2\left(n+\frac{1}{2}\right)}{z}+\cfrac{1}{\cfrac{2\left(n+\frac{3}{2}\right)}{z}+\cfrac{1}{\cfrac{2\left(n+\frac{5}{2}\right)}{z}+...}}} \qquad (14)$$

- Calculate the upper order SMBFs of the first kind, i_{Nmax} using the already known values of $k_{N_{max}}(z)$ and $k_{N_{max-1}}(z)$, the ratio $FC(z)$, and the value of the Wronskian of SMBFs [5]

$$W\{i_{N_{max-1}}(z),k_{N_{max-1}}(z)\}=i_{N_{max}}(z)k_{N_{max-1}}(z)+i_{N_{max-1}}(z)k_{N_{max}}(z)=\pi/2z^2 \qquad (15)$$

As in the previous expressions, we can write:

$$i_{N_{max-1}}(z)=\pi/2z^2[FC(z)k_{N_{\max-1}}(z)+k_{N_{\max}}(z)] \qquad (16)$$

and then,

$$i_{N_{max-1}}(z)=FC(z)i_{N_{\max-1}}(z). \qquad (17)$$

Notice that we have calculated not only $i_{N_{max}}(z)$ but also $i_{N_{max-1}}(z)$.

- Evaluate all SMBFs of the first kind, $\{i_n(z), n = 0,,1,2, ..., N_{max}\}$. taking into account the calculated values of $i_{Nmax}(z)$ and $i_{Nmax\text{-}1}(z)$ using the backwards recurrence relation

$$i_{n-1}(z)=\frac{2n+1}{z}i_n(z)+i_{n+1}(z). \qquad (18)$$

We would like to point out that $i_0(z)$ and $i_1(z)$ can be used as checks on the accuracy of the procedure, because both are calculated at the end of the method.

5. Miller's algorithm

Miller's algorithm initiates[3] calculating directly an sufficient high order m SMBFs of the first kind ($m >> N_{max}$) with the values of "0" and "1" respectively for $i''_m(z)$ and $i''_{m-1}(z)$ using the backward recurrence relation

$$i''_{n-1}(z) = \frac{2n+1}{z} i''_n(z) + i''_{n+1}(z) \tag{19}$$

i.e., we generate the set $\left\{ i''_n(z), n = 0,1,2,\ldots,m \right\}$.

Taking into account the calculated values of $i''_0(z)$ and $i''_1(z)$ and using the known values [5] of $i_0(z)$ and $i_1(z)$ we calculate the re-normalization constant

$$p(z) = \frac{i_0(z)}{i''_1(z)}. \tag{20}$$

Finally we generate the set $\quad i_n(z) = p(z) i''_n, \quad (n = 0, 1, 2, \ldots, N_{max})$ (21)

6. Program specification

In this section, we introduce three *DERIVE* programs. A first code to evaluate the SMBFs using Miller's algorithm, a second one to calculate the continued fractions representation of the integral function and the last one to evaluate the SMBFs using the continued fractions. We pass to detail the structure of these programs.

1. ***DERIVE* program to evaluate SMBFs using Miller's algorithm**
   (see appendix A)
   The code is organized as follows:
   - with expressions #1 and #2 we calculate the known[5] values of $i_0(z)$ and $i_1(z)$,
   - with expression #3 we generate the set i''_n, (n = 0, 1, 2, ..., N_{max}),
   - with expression #4 we evaluate the re-normalization constant,
   - and finally with expression #5 we generate the set i''_n, (n = 0, 1, 2, ..., N_{max})

2. ***DERIVE* program to evaluate the continued fraction representation of the integral function** (see appendix B)
   - with expressions #1 and #2 define the a's and b's in expression (4), we evaluate the values of a_n and b_n respectively with the functions $A(k,n)$ and $B(k,z)$,
   - define the α's and β's in expression (5) using the recurrence relation (6) and (7), we evaluate the values of α_l and β_l with the functions $\alpha(k,n,z)$ and $\beta(k,n,z)$,
   - we store the values of $f_k = \alpha_k/\beta_k$, (k = 0, 1, 2, ..., N) in the H(k,n,z) vector

3. ***DERIVE* program to evaluate SMBFs using Miller's algorithm**
   (see appendix C)
   - with expressions #1 to #11 we calculate the initial values of $K_0(z)$ and $K_1(z)$ using the polynomial approximation [5]. For this we use the functions $K0(z)$ and $K1(z)$.
   - with expression #13 we evaluate the set {$K_n(z)$, n = 0, 1, 2, ..., $Nmax$} using equation (8) with the functions $FBK(n,z)$.

- with the expressions #14 to #18 we evaluate the continued fraction storing it in the *H*(*k*,*n*,*z*)-vector. Finally we calculate it using the *FC*(*k*,*n*,*z*)-function.
- Evaluate the set $\{i_n(z), n = 0, 1, 2, ..., Nmax\}$ using function *FBI*(*k*,*n*,*z*) in expression #19.

Unfortunately I did not receive the respective files for 2. and 3. in 1998. I wrote a mail to the authors in 2014. They answered but they could not retrieve these files, Josef.

7. Results

In this section, we have presented some numerical examples of the accuracy of the method.

1. Results of the program to evaluate the integral function $E_n(z)$ using a continued fraction representation.

   We present the results for some values of "*n*" and "z" in the following table using the fraction continued approximation:

Table 1. Calculated values of the integral function $E_n(z)$ using the continued fraction method.

k	*n*	*z*	$E_n(z)$	*Abramowitz and Stegun*[5]
20	20	0.10	0.047359994	0.0473600
20	10	0.69	0.052505381	0.0525055
25	4	1.15	0.071164022	0.0711632

2. Results of the evaluation the MBFs are represented in the next table:.

Table 2. We compare our results with those implemented in DERIVE's files.

n	z	MILLER (*l*=5, m=5)	CONT. FRACT. Method	*k*	BESSEL_I(*n*,*z*)	BESSEL_I_ SERIES(*n*,*z*,*m*) (*m* = 10)	Abramowitz and Stegun
1	5	24.335642097	24.3356421099	10	24.3356421424	24.335642065	24.33564214
5	5	2.157973762	2.157974553	10	2.157974547	2.157974547	2.157974547
10	2	$3.016963885*10^{-7}$	$3.016963839*10^{-7}$	11	dub. Accuracy	$3.016963878*10^{-7}$	$3.016963879*10^{-7}$
15	2	$8.139432548*10^{-13}$	$8.139432410*10^{-13}$	5	dub. Accuracy	$8.139432530*10^{-13}$	$8.139432531*10^{-13}$

8. Conclusions

We have introduced the *DERIVE* version of an algorithm to evaluate the continued fraction representation of a rational approximation of a function and an efficient algorithm to generate MBFs. This code illustrates the se of the continued fraction method and gives an accurate way to evaluate the MBFs although it is slower than the implemented BESSEL.MTH file.

References

1. Yu.L.Ratis, P.Fernández de Córdoba, J.Segura, *A code to evaluate modified Bessel functions based on the continued fraction method.* Comput. Phys. Commun. 105 (1997) 263-272
2. Y.L.Luke, *Mathematical Functions and their approximations*. Academic, New York, (1975)
3. J,C.P.Miller, *Bessel Functions, Part II, Functions of Positive Integer Order, Mathematical Tables, Vol 10.* Cambridge Press, (1952)
4. W.H.Press, B.P.Flannery, S.A.Teukolsky, W.T.Vetterling, *Numerical Recipes.* Cambridge University Press (1989)
5. M.Abramowitz, I.A.Stegun, Eds. *Handbook of Mathematical Functions.* Dover (1972)

Logos, Stars and other Figures with *DERIVE* and the *TI-92*

Josef Böhm, Würmla

I am a teacher at a Business College and our pupils come to our school when they have finished lower secondary education at different types of schools. So one of my tasks is to find a common base to teach mathematics repeating basic mathematical knowledge. As we have to deal with functions and their graphic representation it is obvious to freshen up the system of coordinates. Some years ago I did this successfully supported by *DERIVE* and now class room bounded I am using the *TI-92.* Using the DATA-tool simultaneously with the Data plot on the Graph screen it is very easy and entertaining to explore the plane divided by the axes.

I could see the students find transformations like translations and reflections by themselves. The points can be connected to draw figures.

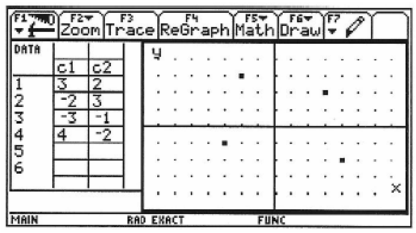
> Draw a square which has one vertex in each quadrant by defining its points. (It is always nice to see them explore how to close the polygon)
>
> Draw a house. Using more pairs of data columns you can show more separated objects.
>
> Draw an equilateral triangle with a given base line. (They have to remember the formula for the altitude and work exactly using √3)
>
> Complete the triangle to a regular hexagon.
>
> Shift the hexagon to another midpoint.
>
> (Some students had the idea to apply the translation(s) immediately on the data columns: c3=cl + 2 and c4=c2 – 4 produces a translation by the vector (2,-4)).

If you are interested in the work sheets then please write, email or call. The paper is in German, but it is very easy to follow even in other languages because there are a lot of pictures. If there are many of you interested I would try to produce a translation.

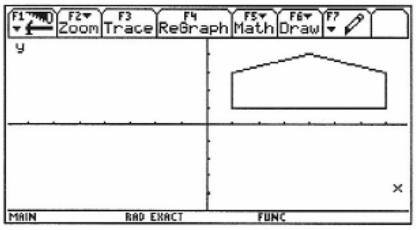
See two short parts of the work sheets:

4. Wechsle mit wieder ins Datenblatt und übertrage nun Punkt für Punkt die Punkte A, B, C und D aus Aufgabe 2 ins Grafikfenster.

   Switch to the DATA-MATRIX-Editor and plot points A, B, C, and D from task 2 into the GRAPH-Window.



11. Zeichne den Umriss eines Hauses im 1. Quadrant.
(*Plot the graph of a house into the 1st quadrant.*)



(← nur ein mögliches Beispiel)

Überlege zuerst die optimale Reihenfolge der Punkte im Polygonzug.

Consider the best order of the points for plotting the polygon.

12. Spiegle dieses Haus zuerst an der x-, dann an der y-Achse und abschließend am Koordinatenursprung. Gib für jede Spiegelung die Konstruktionsvorschrift an.

(*Plot the mirror image of your house wrt the x- and to the y-axis and then wrt the origin. Describe the respective construction rule.*)

Most of this can be done with *DERIVE* and the *TI-92* as well.

At last I gave a home work:

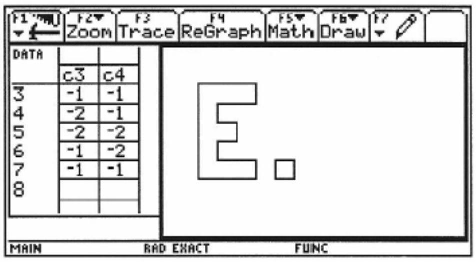
The students should design their own personal logo, their initials, a fantasy star or any other object they liked to represent in the plane using coordinates to describe it point for point.

I promised to provide a program for their devices which would animate their creations.

20. Als Abschluss sollst Du Dir Dein eigenes LOGO – Initialen oder sonst etwas – im Koordinatensystem schaffen. Erzeuge ein neues Datenblatt LOGO. Jedes Polygon (Vieleck) wird durch ein Spaltenpaar beschrieben.

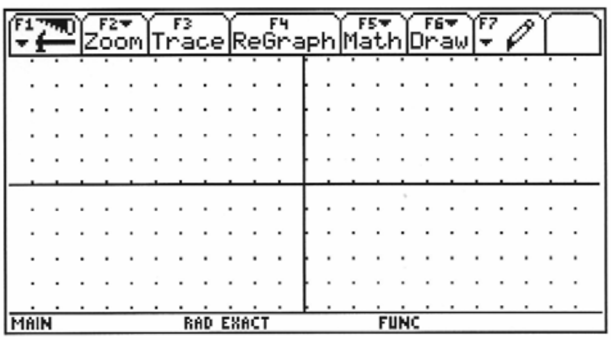
Als Muster siehst Du hier ein "**E.**". Die Achsen, ihre Bezeichnung und das Gitter werden am Ende ausgeblendet. Du kannst damit rechnen, den ganzen Schirm zur Verfügung zu haben.

(-11 ≤ x ≤ 11, -5 ≤ y ≤ 5)

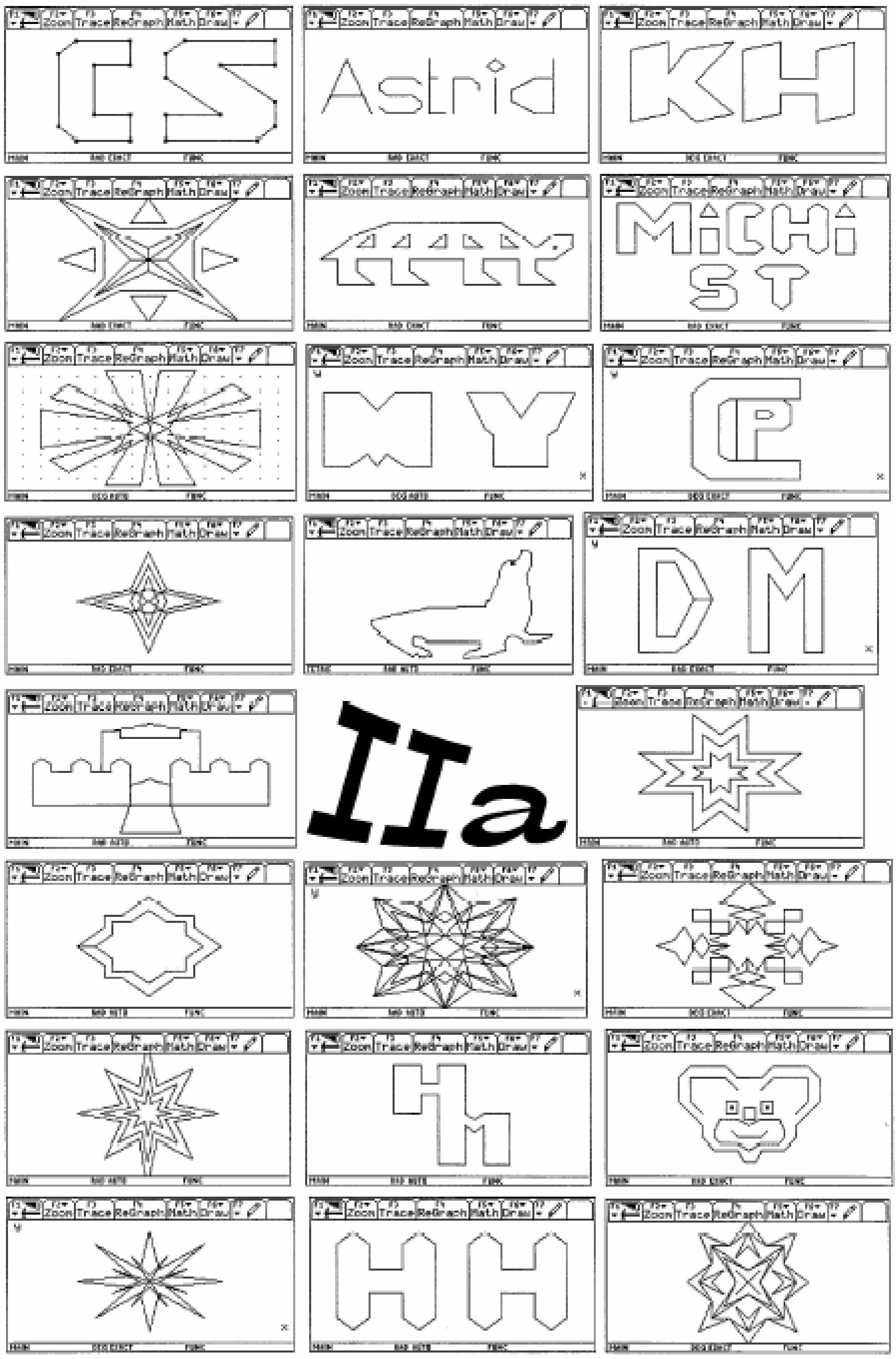


Wenn das LOGO fertig ist, werden wir es mit Hilfe eines Programms animieren, d.h. einen kleinen Film draus machen.

MEIN LOGO



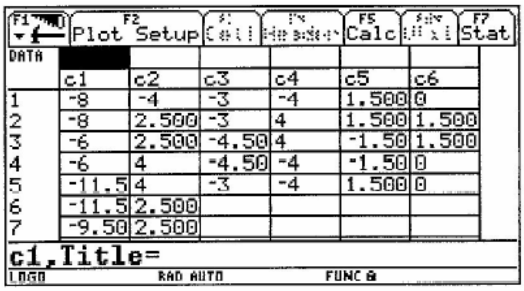
We – my colleague Tania Koller and I – gave this homework in four classes and we didn't expect too many nice results. But sometimes even teachers have their wonderful experiences. After the week end when we asked the pupils to present their designs using the view screen our eyes became bigger and bigger: some of the boys and girls might have spent hours to realize their ideas. Next day I took my notebook with me and using GraphLink collected the Four Classes Picture Gallery. You can see the IIa collection below.

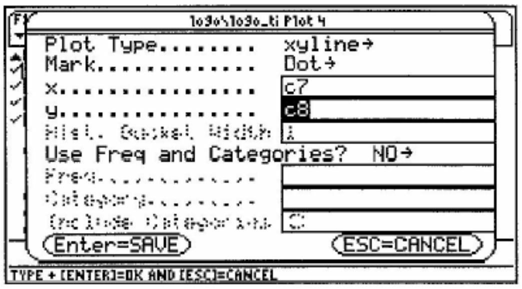


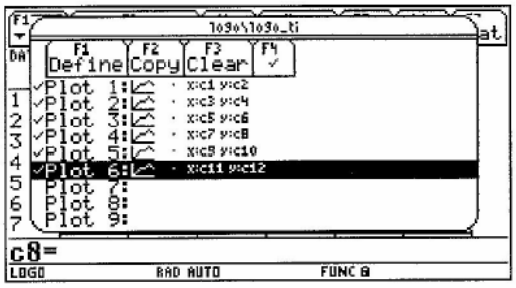
And nobody asked: why are we doing that in such a "boring" way? We could use Paintbrush, Corel Draw or any other painting program. They had just fun and proudly presented their screens. Surprisingly some of the weak mathematicians had excellent ideas and could realize them.

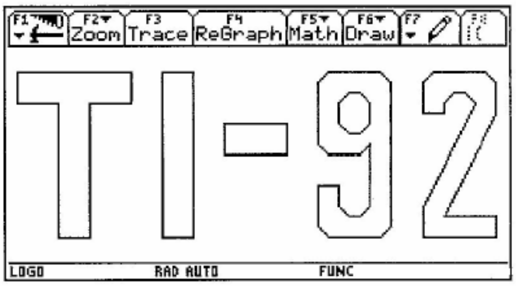
But then they reminded me on my promise to make a "movie" of their pictures.

In pairs of columns I produced letter after letter according to a design on a grid paper and wrote the data into a data table called "`logo`". Then using `F2` Plot Setup I called `F1` Define and there defined one polygon after the other. The screen shot shows the creation of the data plot of columns 7 and 8 representing the contour of "9". 12 columns make the whole picture.

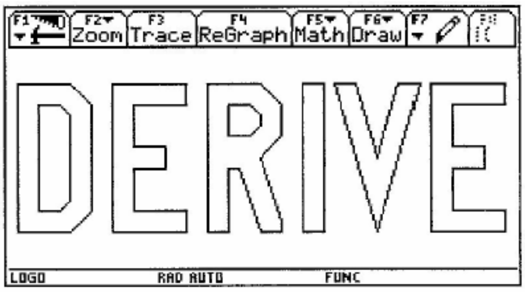








I wanted to produce a movie merging the *TI*'s logo and "*DERIVE*". So I had another data plot with the data fixed in variable "`logod`".



Now I needed a program `BILDER( )` to produce the pictures which should be "cycle-pictured" by the next program to perform the animation:

I came across some problems: I was not able to address the variable "`logo`" - the name of the data table - as a program parameter. I wanted to call `BILDER(logo)` to make the program more flexible - also for `BILDER(logod)` - but I failed. The same reason made it impossible to use `dim(logo)` etc, but fortunately and inconsequently `logo[i]` returns the i-th column of the data. So I used the number of columns as parameter. The program produces three families of pictures "`dr`"s, "`ki`"s and "`st`"s. You see the pictures appearing on the screen, each one is saved in one extra file. For my students I made a version including the choice how to represent the movie:

(1)	`Rotate`	Rotation around the vertical axis
(2)	`Switch`	Rotation around the horizontal axis
(3)	`Flash`	Zooming in and out

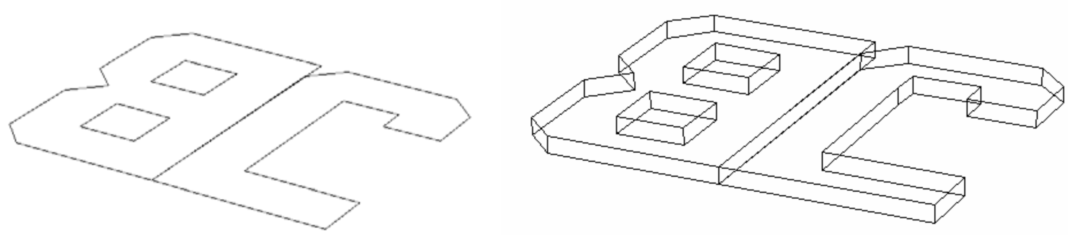
The second program `film()` does the animation.

The smaller versions are on the diskette (BILDER1, FILM!). The super version for merging the *TI-92* and the *DERIVE-Logo* is also on the diskette (BILDER2, FILM2).

You can work very similar using DERIVE. I have three different ways to show the figures (see below). Try and be surprised: logo*.acd-files and logo.mth. You have to use ACROSPIN in combination with ACD (Diskette of the year 97, DNL#28). Inspired by Stefan Welke's 3dV-contribution I produced a DERIVE-function to transfer polyhedrons into 3d-format. You can find the logo*.3d-files on the diskette.

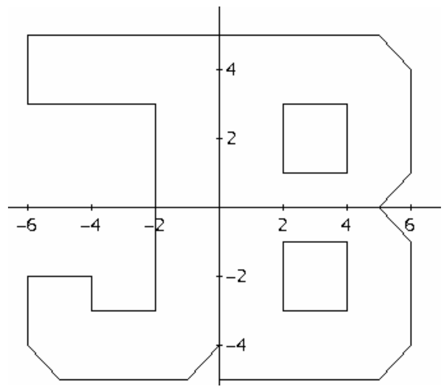
```
Bilder(θn)
Prgm
Local i,j,θt,x1,y1
ClrGraph:FnOff
ZoomDec
setGraph("Axes","Off")
setGraph("Grid","Off")
setGraph("Labels","Off")
For θt,0,π,π/10
ClrDraw
For i,1,θn,2
  logo[i]→x1:logo[i+1]→y1
  For j,1,dim(x1)-1
    Line x1[j]*cos(θt),y1[j],x1[j+1]*cos(θt),y1[j+1]
  EndFor
EndFor
StoPic #("dr"&string(10*θt/(π)+1))
EndFor
For θt,0,π,π/10
ClrDraw
For i,1,θn,2
  logo[i]→x1:logo[i+1]→y1
  For j,1,dim(x1)-1
    Line x1[j],y1[j]*cos(θt),x1[j+1],y1[j+1]*cos(θt)
  EndFor
EndFor
StoPic #("ki"&string(10*θt/(π)+1))
EndFor
For θt,1,11
ClrDraw
For i,1,θn,2
  logo[i]→x1:logo[i+1]→y1
  For j,1,dim(x1)-1
    Line x1[j]*(11-θt)/10,y1[j]*(11-θt)/10,x1[j+1]*(11-θt)/10,y1[j+1]*(11-
θt)/10
  EndFor
EndFor
StoPic #("st"&string(θt))
EndFor
EndPrgm
```

```
film()
Prgm
Lbl start
CyclePic "st",11,.12,1,-1
CyclePic "dr",11,.12,1,-1
CyclePic "ki",11,.12,1,-1
Goto start
EndPrgm
```



The *DERIVE*-graphs are based on the logo given below. The file is among the accompanying files of the DNL.

$$\text{logo} := \left[\begin{bmatrix} -2 & -3 \\ -2 & -2 \\ -2 & 3 \\ -6 & 3 \\ -6 & 5 \\ 0 & 5 \\ 0 & -4 \\ -1 & -5 \\ -5 & -5 \\ -6 & -4 \\ -6 & -2 \\ -4 & -2 \\ -4 & -3 \\ -2 & -3 \end{bmatrix}, \begin{bmatrix} 0 & 5 \\ 5 & 5 \\ 6 & 4 \\ 6 & 1 \\ 5 & 0 \\ 6 & -1 \\ 6 & -4 \\ 5 & -5 \\ 0 & -5 \\ 0 & 5 \end{bmatrix}, \begin{bmatrix} 2 & 3 \\ 4 & 3 \\ 4 & 1 \\ 2 & 1 \\ 2 & 3 \end{bmatrix}, \begin{bmatrix} 2 & -3 \\ 4 & -3 \\ 4 & -1 \\ 2 & -1 \\ 2 & -3 \end{bmatrix}\right]$$



More interesting for the revised version of DNL#32 is how to perform the movie on *TI-NspireCAS.* The coordinates of the logo points are given in lists in a Lists & Spreadsheet page. Take care that columns are of equal length, i.e. fill up empty cells with the coordinates of the last point (see l1_1 and l2_2).

Collecting all lists in another list (in the Calculator) results in a matrix (see next page). I called this matrix logo. Three small programs produce list l1, l2, …, ln with *n* = number of columns of the matrix (= 2 × number of polygons).

	A l1_1	B l1_2	C l2_1	D l2_2	E l3_1	F l3_2	G l4_1	H l4_2	I
=									
1	-2	-3	0	5	2	3	2	-3	
2	-2	-2	5	5	4	3	4	-3	
3	-2	3	6	4	4	1	4	-1	
4	-6	3	6	1	2	1	2	-1	
5	-6	5	5	0	2	3	2	-3	
6	0	5	6	-1	2	3	2	-3	
7	0	-4	6	-4	2	3	2	-3	
8	-1	-5	5	-5	2	3	2	-3	
9	-5	-5	0	-5	2	3	2	-3	
10	-6	-4	0	5	2	3	2	-3	
11	-6	-2	0	5	2	3	2	-3	
12	-4	-2	0	5	2	3	2	-3	
13	-4	-3	0	5	2	3	2	-3	
14	-2	-3	0	5	2	3	2	-3	

C11:D14

```
logo:={l1_1,l1_2,l2_1,l2_2,l3_1,l3_2,l4_1,l4_2}
 [-2 -2 -2 -6 -6  0  0 -1 -5 -6 -6 -4 -4 -2]
 [-3 -2  3  3  5  5 -4 -5 -5 -4 -2 -2 -3 -3]
 [ 0  5  6  6  5  6  6  5  0  0  0  0  0  0]
 [ 5  5  4  1  0 -1 -4 -5 -5  5  5  5  5  5]
 [ 2  4  4  2  2  2  2  2  2  2  2  2  2  2]
 [ 3  3  1  1  3  3  3  3  3  3  3  3  3  3]
 [ 2  4  4  2  2  2  2  2  2  2  2  2  2  2]
 [-3 -3 -1 -1 -3 -3 -3 -3 -3 -3 -3 -3 -3 -3]
rot_x(logo)                     Fertig
DelVar t                        Fertig
rot_y(logo)                     Fertig
DelVar t                        Fertig
```

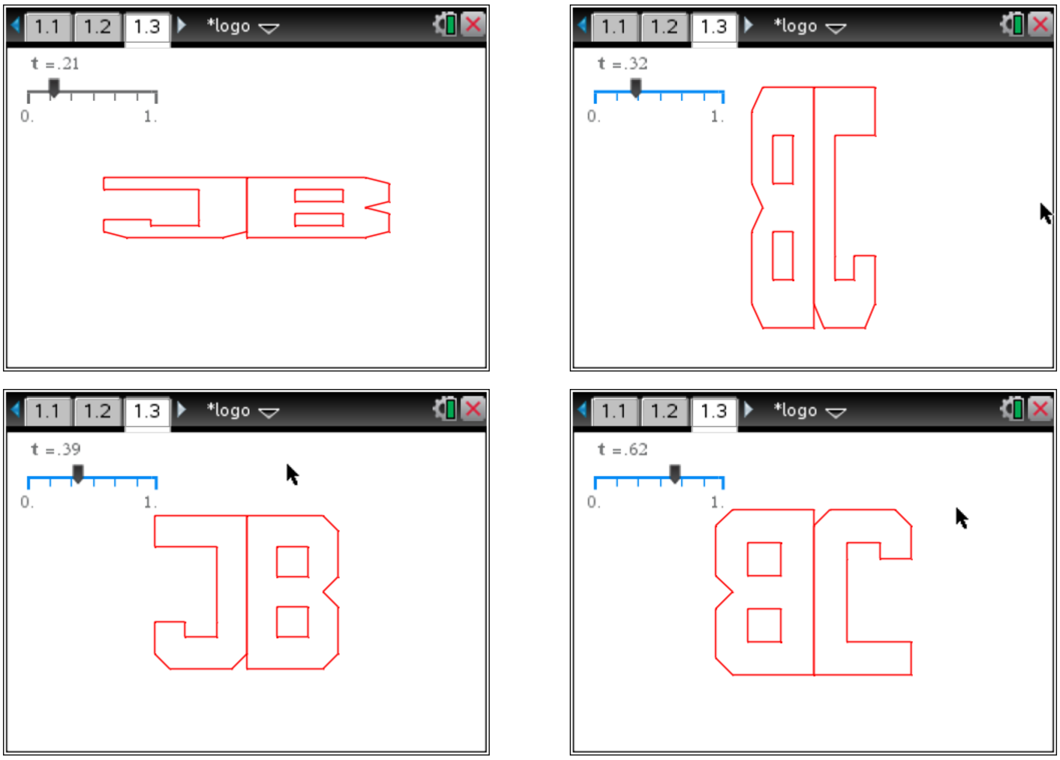
```
rot_x                                             6/6
Define rot_x(pic)=
Prgm
Local d,i
d:=dim(pic)[1]
For i,1,d,2
mat▸list(pic[i])→#("l"&string(i))
mat▸list(pic[i+1])·cos(t·2·π))→#("l"&string(i+1))
EndFor
EndPrgm
```

```
rot_y                                             0/6
Define rot_y(pic)=
Prgm
Local d,i
d:=dim(pic)[1]
For i,1,d,2
mat▸list(pic[i])·cos(t·2·π))→#("l"&string(i))
mat▸list(pic[i+1])→#("l"&string(i+1))
EndFor
EndPrgm
```

```
"flash" erfolg. gespeichert
Define flash(pic)=
Prgm
Local d,i
d:=dim(pic)[1]
For i,1,d,2
mat▸list(pic[i])·sin(t·2·π))→#("l"&string(i))
mat▸list(pic[i+1])·sin(t·2·π))→#("l"&string(i+1))
EndFor
EndPrgm
```

After running one of the three programs insert a Graphs & Geometry page, introduce a slider for parameter t with $0 \le t \le 1$, step size 0.01, and enter the lists l1 through ln in the Scatter Plot entry lines. Moving or animating the slider shows the "movie".

If you want to have another movie, first delete t in the calculator, run the next program, switch to the Geometry page and initiate again the slider. Have fun!



SUPER DUPER OSCULANTS (3)

David Halprin, North Balwyn, Australia

This is the procedure for any *Whewell* Curves Type-1 to be tested in OSCWHEl.MTH.

Declare Function, answer v or V, then when asked for the value, answer with the function of Φ, (e.g. tan(Φ), for Catenary).

Then Derive writes the line V(Φ) := tan(Φ), then Simplify #15 as next step, which means that Derive interprets that SIMPLIFY command to mean Substitute TAN(Φ) wherever V(Φ) appears, and differentiate where appropriate.

Then, when it returns the result, use the commands MANAGE SUBSTITUTE for that expression, and answer 0 for the first variable, which is Φ and when it asks for t just press ENTER.

When the expression is returned, use the command SIMPLIFY, which has a different meaning than earlier. Here it means evaluate, and when it returns the next expression it is a parametric pair of equations in t only, of the form [f(t),g(t)], which is ready for plotting.

Should there be a division by zero after substituting 0 for Φ, then REMOVE those expressions and use $\frac{\pi}{2}$ for Φ and it should do. Once the correct expression for plotting is there, REMOVE the previous two expressions.

$$3.21 \quad x = t.\dot{V}.\cos\phi + \frac{t^2}{2}\left[\cos\phi.\ddot{V} - \sin\phi.\dot{V}\right] + \frac{t^3}{3!}.\left[\cos\phi.(\dddot{V} - \dot{V}) - 2\sin\phi.\ddot{V}\right] +$$

$$+ \frac{t^4}{4!}.\left[\cos\phi.\left(\frac{d^4V}{d\phi^4} - 3\ddot{V}\right) - \sin\phi.\left(3\dddot{V} - \dot{V}\right)\right] +$$

$$+ \frac{t^5}{5!}.\left[\cos\phi.\left(\frac{d^5V}{d\phi^5} - 6\dddot{V} + \dot{V}\right) - 4\sin\phi.\left(\frac{d^4V}{d\phi^4} - \ddot{V}\right)\right] +$$

$$+ \frac{t^6}{6!}.\left[\cos\phi.\left(\frac{d^6V}{d\phi^6} - 10\frac{d^4V}{d\phi^4} + 5\ddot{V}\right) - \sin\phi.\left(5\frac{d^5V}{d\phi^5} - 10\dddot{V} + \dot{V}\right)\right] +$$

$$+ \frac{t^7}{7!}.\left[\cos\phi.\left(\frac{d^7V}{d\phi^7} - 15\frac{d^5V}{d\phi^5} + 15\dddot{V} - \dot{V}\right) - 2\sin\phi.\left(3\frac{d^6V}{d\phi^6} - 10\frac{d^4V}{d\phi^4} + 3\ddot{V}\right)\right] +$$

$$+ \frac{t^8}{8!}.\left[\cos\phi.\left(\frac{d^8V}{d\phi^8} - 21\frac{d^6V}{d\phi^6} + 35\frac{d^4V}{d\phi^4} - 7\ddot{V}\right)\right] -$$

$$- \frac{t^8}{8!}.\left[\sin\phi.\left(7\frac{d^7V}{d\phi^7} - 35\frac{d^5V}{d\phi^5} + 21\dddot{V} - \dot{V}\right)\right] +$$

$$+ \frac{t^9}{9!}.\left[\cos\phi.\left(\frac{d^9V}{d\phi^9} - 28\frac{d^7V}{d\phi^7} + 70\frac{d^5V}{d\phi^5} - 28\dddot{V} + \dot{V}\right)\right] -$$

$$- \frac{t^9}{9!}.\left[8\sin\phi.\left(\frac{d^8V}{d\phi^8} - 7\frac{d^6V}{d\phi^6} + 7\frac{d^4V}{d\phi^4} - \ddot{V}\right)\right] +$$

$$+ \frac{t^{10}}{10!}.\left[\cos\phi.\left(\frac{d^{10}V}{d\phi^{10}} - 36\frac{d^8V}{d\phi^8} + 126\frac{d^6V}{d\phi^6} - 84\frac{d^4V}{d\phi^4} + 9\ddot{V}\right)\right] -$$

$$- \frac{t^{10}}{10!}.\left[\sin\phi.\left(9\frac{d^9V}{d\phi^9} - 84\frac{d^7V}{d\phi^7} + 126\frac{d^5V}{d\phi^5} - 36\dddot{V} + \dot{V}\right)\right]$$

3.22 $$y = t.\sin\phi.\dot{V} + \frac{t^2}{2}.[\cos\phi.\dot{V} + \sin\phi.\ddot{V}] + \frac{t^3}{3!}.[2.\cos\phi.\ddot{V} + \sin\phi.(\dddot{V} - \dot{V})] +$$

$$+ \frac{t^4}{4!}.\left[\cos\phi.(3\dddot{V} - \dot{V}) + \sin\phi.\left(\frac{d^4V}{d\phi^4} - 3\ddot{V}\right)\right] +$$

$$+ \frac{t^5}{5!}.\left[4\cos\phi.\left(\frac{d^4V}{d\phi^4} - \ddot{V}\right) + \sin\phi.\left(\frac{d^5V}{d\phi^5} - 6\dddot{V} + \dot{V}\right)\right] +$$

$$+ \frac{t^6}{6!}.\left[\cos\phi.\left(5\frac{d^5V}{d\phi^5} - 10\dddot{V} + \dot{V}\right) + \sin\phi.\left(\frac{d^6V}{d\phi^6} - 10\frac{d^4V}{d\phi^4} + 5\ddot{V}\right)\right] +$$

$$+ \frac{t^7}{7!}.\left[\cos\phi.\left(6\frac{d^6V}{d\phi^6} - 20\frac{d^4V}{d\phi^4} + 6\ddot{V}\right) + \sin\phi.\left(\frac{d^7V}{d\phi^7} - 15\frac{d^5V}{d\phi^5} + 15\dddot{V} - \dot{V}\right)\right] +$$

$$+ \frac{t^8}{8!}.\left[\cos\phi.\left(7\frac{d^7V}{d\phi^7} - 35\frac{d^5V}{d\phi^5} + 21\dddot{V} - \dot{V}\right)\right] +$$

$$+ \frac{t^8}{8!}.\left[\sin\phi.\left(\frac{d^8V}{d\phi^8} - 21\frac{d^6V}{d\phi^6} + 35\frac{d^4V}{d\phi^4} - 7\ddot{V}\right)\right] +$$

$$+ \frac{t^9}{9!}.\cos\phi.\left(8\frac{d^8V}{d\phi^8} - 56\frac{d^6V}{d\phi^6} + 56\frac{d^4V}{d\phi^4} - 8\ddot{V}\right) +$$

$$+ \frac{t^9}{9!}.\sin\phi.\left(\frac{d^9V}{d\phi^9} - 28\frac{d^7V}{d\phi^7} + 70\frac{d^5V}{d\phi^5} - 28\dddot{V} + \dot{V}\right) +$$

$$+ \frac{t^{10}}{10!}.\cos\phi.\left(9\frac{d^9V}{d\phi^9} - 84\frac{d^7V}{d\phi^7} + 126\frac{d^5V}{d\phi^5} - 36\dddot{V} + \dot{V}\right) +$$

$$+ \frac{t^{10}}{10!}.\sin\phi.\left(\frac{d^{10}V}{d\phi^{10}} - 36\frac{d^8V}{d\phi^8} + 126\frac{d^6V}{d\phi^6} - 84\frac{d^4V}{d\phi^4} + 9\ddot{V}\right)$$

Whewell Type 2, $\Phi = W(s) = W_s$, requires

$$\frac{dx}{ds}, \frac{d^2x}{ds^2}, \frac{d^3x}{ds^3} \dots \dots \frac{d^nx}{ds^n} \text{ and } \frac{dy}{ds}, \frac{d^2y}{ds^2}, \frac{d^3y}{ds^3} \dots \dots \frac{d^ny}{ds^n}.$$

To use DERIVE declare W = W(s), differentiate with respect to *s* using DERIVE.

4.00 $\phi = W(s) = W_s = W$

4.01 $\frac{dx}{ds} = \cos\phi = \cos W$

4.02 $\frac{d^2x}{ds^2} = -W'.\sin W$

4.03 $\frac{d^3x}{ds^3} = -W''.\sin W - (W')^2.\cos W$

4.04 $\frac{d^4x}{ds^4} = -\cos W.(W' + 2W'W'') + \sin W.\left[(W')^2 - W'''\right]$

4.11 $\frac{dy}{ds} = \sin\phi = \sin W$

4.12 $\frac{d^2y}{ds^2} = W'.\cos W$

4.13 $\frac{d^3y}{ds^3} = W''.\cos W - (W')^2.\sin W$

4.14 $\frac{d^4y}{ds^4} = -\cos W.\left[(W')^2 - W'''\right] - \sin W.(W'' + 2W'W'')$

4.21 $x = t - \frac{t^3}{6}.(W')^2 - \frac{t^4}{24}.[W' + 2W'W'']$

4.22 $y = \frac{t^2}{2}.W' + \frac{t^3}{6}.W'' - \frac{t^4}{24}.\left[(W')^2 - W'''\right]$

where $\phi = 0$

Euler Type 1 is ρ = E(Φ) = E_Φ, requiring

$$\frac{dx}{d\phi}, \frac{d^2x}{d\phi^2}, \frac{d^3x}{d\phi^3}.... ...\frac{d^nx}{d\phi^n} \text{ and } \frac{dy}{d\phi}, \frac{d^2y}{d\phi^2}, \frac{d^3y}{d\phi^3}.... ...\frac{d^ny}{d\phi^n}.$$

To use DERIVE declare E = E(Φ), differentiate with respect to Φ using DERIVE.
This leads to:

$$5.21 \quad x = E.t + \frac{\dot{E}.t^2}{2} + \frac{(\ddot{E} - E).t^3}{6} + \frac{(\dddot{E} - 3\dot{E}).t^4}{24}$$

$$5.22 \quad y = \frac{E.t^2}{2} + \frac{\dot{E}.t^3}{3} + \frac{(3\ddot{E} - E).t^4}{24}$$

where Φ = 0.

Euler Type 2 is Φ = F(ρ) = Fρ, requiring

$$\frac{dx}{d\rho}, \frac{d^2x}{d\rho^2}, \frac{d^3x}{d\rho^3}.... ...\frac{d^nx}{d\rho^n} \text{ and } \frac{dy}{d\rho}, \frac{d^2y}{d\rho^2}, \frac{d^3y}{d\rho^3}.... ...\frac{d^ny}{d\rho^n}.$$

To use DERIVE declare F = F(ρ), differentiate with respect to ρ using DERIVE to obtain:

$$6.21 \quad x = t.\left[\rho.\frac{dF}{d\rho}\right] + \left(\frac{t^2}{2}\right).\left[\frac{dF}{d\rho} + \rho.\frac{d^2F}{d\rho^2}\right] + \left(\frac{t^3}{6}\right).\left[2.\frac{d^2F}{d\rho^2} + \rho.\frac{d^3F}{d\rho^3} - \rho.\frac{dF}{d\rho}\right]$$

$$6.22 \quad y = \left(\frac{t^2}{2}\right).\left[\rho.\frac{dF}{d\rho}\right] + \left(\frac{t^3}{3}\right).\left[\frac{dF}{d\rho} + \rho.\frac{d^2F}{d\rho^2}\right]$$

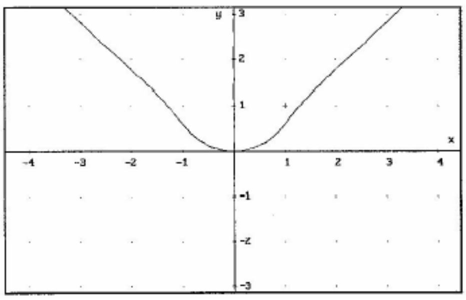
where Φ = 0.

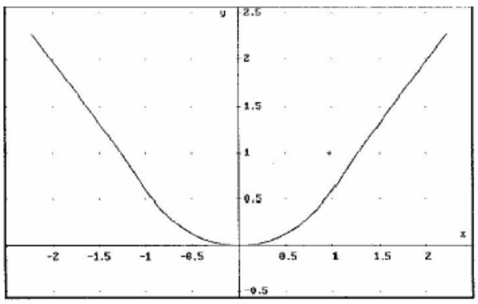
TABLE OF OSCULANTS

(1) Quality, (2) Order. (3) Number, (4) Equation Identity

(1)	(2)	(3)	NAMES(S)	EQUATION	(4)
Q1	1	1	Gradient	$\frac{dy}{dx} = \tan\phi$	1.11
Q1a	1	2	Elasticity	$\frac{y}{x}.\frac{dx}{dy}$	6.10
Q2	2	1	Radius of Curvature	$\rho = \frac{ds}{d\phi}$	2.20
Q2a	2	2	Gradient Change per Abscissa Increase	$\frac{d^2y}{dx^2} = \frac{\sec^3\phi}{\rho}$	
Q3	3	1	Spiralation	$\frac{d\rho}{ds} = \rho' = \tan\alpha = \frac{\dot{\rho}}{\rho}$	4.11
Q3a	3	2	Aberrancy or Deviation	$\frac{1}{3}.\frac{d\rho}{ds} = \frac{\rho'}{3} = \frac{\dot{\rho}}{3\rho} = \tan\delta$	4.10
Q3b	3	3	Involation	$\frac{d\rho}{d\phi} = \dot{\rho} = \rho\rho'$	19.0

(1)	(2)	(3)	NAMES(S)	EQUATION	(4)
Q3c	3	4	Whewell Gudermannian	$\frac{\rho'}{\sqrt{\rho^2 - 1}} = \frac{\dot{\rho}}{\rho\sqrt{\rho^2 - 1}}$	16.6
Q3d	3	5	Spiralation-2	$\frac{\rho'}{\rho} = \frac{\dot{\rho}}{\rho^2}$	20.0
Q3e	3	6	Clothoidation	$\frac{\rho'}{\rho^2} = \frac{\dot{\rho}}{\rho^3}$	21.0
Q4	4	1	Radius of Tau-ature Radius of Q4	$\tau = \frac{ds}{d\alpha} = \frac{1 + (\rho')^2{}_{''}}{\rho} = \frac{\rho(\rho^2 + \dot{\rho}^2)}{\rho\ddot{\rho} - \dot{\rho}^2}$	22.0
Q4a	4	2	Catenation	$\frac{d^2\rho}{ds^2} = \rho'' = \frac{\rho\ddot{\rho} - \dot{\rho}^2}{\rho^3}$	9.10
Q4b	4	3		$\frac{d^2\rho}{d\phi^2} = \ddot{\rho} = \rho[(\rho')^2 + \rho\rho'']$	
Q4c	4	4	Cycloidal Displacement	$\frac{\ddot{\rho}}{\rho} = \rho\rho' + (\rho')^2$	23.0
Q4d	4	5	Tractration	$\frac{\ddot{\rho}}{\rho\dot{\rho}} = \frac{(\rho')^2 + \rho\rho''}{\rho\rho'}$	24.0
Q5	5	1	Betation	$\frac{d\tau}{ds} = \tan\beta = 2\rho' - \frac{\rho'''[1 + (\rho')^2]}{(\rho'')^2}$	27.0
Q5a	5	2		$\rho''' = \frac{\rho^4\dddot{\rho} - 4\rho^3\dot{\rho}\ddot{\rho} + 3\rho^2\dot{\rho}^3}{\rho^7}$	
Q6	6	1	Radius of Q6	$\frac{ds}{d\beta}$	
Q7	7	1	Gammation	$\frac{d\left(\frac{ds}{d\beta}\right)}{ds} = \tan\gamma$	
Q8	8	1	Radius of Q8	$\frac{ds}{d\gamma}$	





There are different scalings of the 4th order Curve of constant Tau-ature, which is the forth in the homologous series of curves of constant geometric quality, whose first three are the Straight Line, the Circle and the Logarithmic Spiral.

Leibnizens silberne Taschenuhr

Leibniz' Pocket Watch of Silver

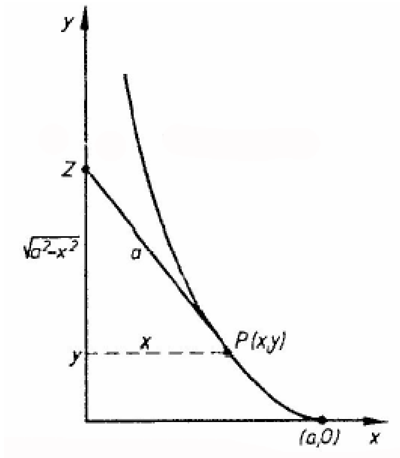
Josef Lechner, Viehdorf, Austria

In H. Heuser's worth reading book "Gewöhnliche Differentialgleichungen" [1] you can find the following problem as an introduction (Problem of the Tractrix – Drag Curve, Curve of Pursuit):

Leibnizens silberne Taschenuhr oder die Traktrix

Gottfried Wilhelm Leibniz (1646-1716) behandelte 1693 in den Leipziger *Acta eruditorum* das folgende Problem (s. Abb.): in der xy-Ebene ziehe man einen Punkt P an einer straff gespannten Schnur PZ der Länge a. Der Zugpunkt Z soll auf der y-Achse fortrücken, und zu Beginn des Vorgangs befinde sich P in $(a,0)$ (= „Anfangsbedingung").

Welche Kurve beschreibt P?



Des „Verständnisses wegen" imaginierte Leibniz eine *horologio portatili suae thecae argenteae*, die an ihrem Kettchen *per tabulam* gezogen wird. Erfinder des Problems sei *Claudius Perralus, Medicus Parisinus, tum et Mechanicis atque Architectonis studiis egregius ...* Gemeint ist der Pariser Architekt Claude Perrault (1613-1688), der die berühmte Säulenfassade an der Ostfront des Louvre entworfen hat; die Heilkünste des Mannes dürfen wir auf sich beruhen lassen.

Da die Zugschnur PZ zur gesuchten „Zugkurve" oder „Traktrix" $y = y(x)$ offenbar immer *tangential* ist, können wir aus der Figur sofort die Gleichung

$$y' = -\frac{\sqrt{a^2 - x^2}}{x}$$

ablesen. Da in ihr die Ableitung y' der gesuchten Funktion auftritt, ist sie eine Differentialgleichung, wenn auch nur von de extrem einfachen Form $y' = f(x)$. Bereits die elementare Integralrechnung lehrt, dass alle Funktionen

$$y(x) = -\int \frac{\sqrt{a^2 - x^2}}{x} \mathrm{d}x + C = a \ln\left(\frac{a + \sqrt{a^2 - x^2}}{x}\right) - \sqrt{a^2 - x^2} + C$$

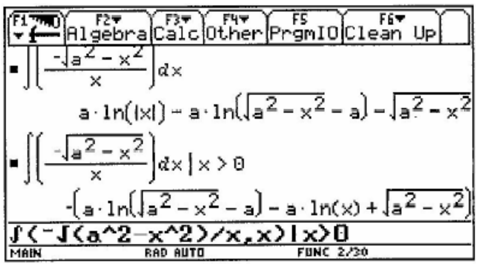
und keine anderen dieser Differentialgleichung genügen ($C \in$ R beliebig; siehe Formel Nr. 9 im Anhang 1). Wegen der Anfangsbedingung $y(a) = 0$ muss $C = 0$ sein. Die gesuchte Gleichung der Traktrix ist also

$$y(x) = a \ln\left(\frac{a + \sqrt{a^2 - x^2}}{x}\right) - \sqrt{a^2 - x^2}.$$

Was aber weder *Claudius Perraltus* noch *Gothofredus Guilielmus Leibnitius* im entferntesten ahnen konnten: Wenn man ihre verspielte Traktrix um die y-Achse rotieren lässt, entsteht eine Fläche mit konstanter negativer Gaußscher Krümmung – und diese „Pseudosphäre" ist nichts weniger als ein Modell für die *Nichteuklidische Geometrie Lobatschewskis* (1792-1856).

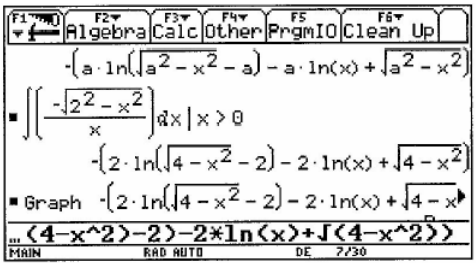
1. Attempt: with the *TI*-92

Trying to find the tractrix by integration using the *TI*-92 you will find another result as given in Heuser's book:

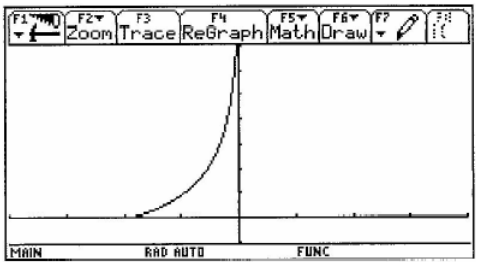


We integrate as usual.

Integration with condition $x > 0$.



The tractrix shall start in point (2,0)

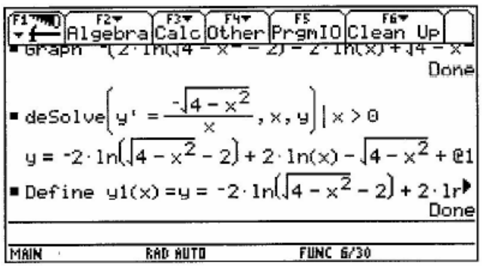


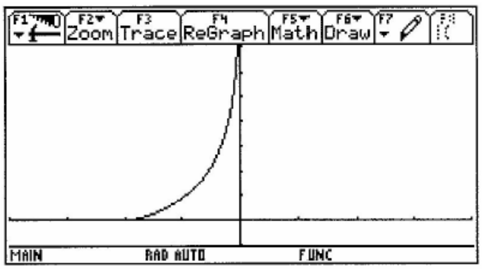
The graph shows the – unexpected – result that we find the curve reflected with respect to the y-axis. Why that?

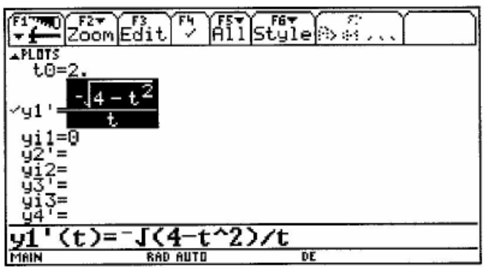
Something must have gone wrong with the sign ...

2. Attempt: again with the *TI*-92 (PLUS Module or *TI*-89 and the Voyage 200 as well)

The PLUS (and the *TI*-89) offers the opportunity to solve differential equations using the `deSolve` command. Applying this command to integrate yields the same result as before:



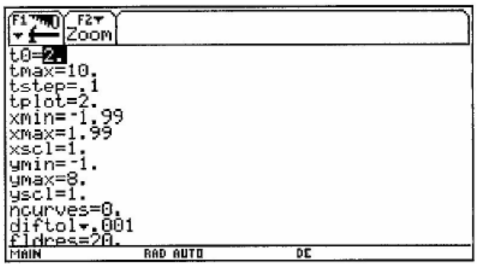


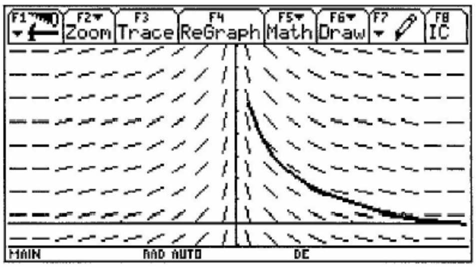


You'd better choose a numerical approach:

(Don't forget first to set `MODE 6:DIFF EQUATIONS`)

If you take care to have a defined radicand you will obtain the correct “drag curve”.

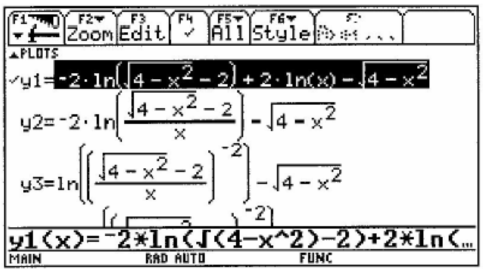


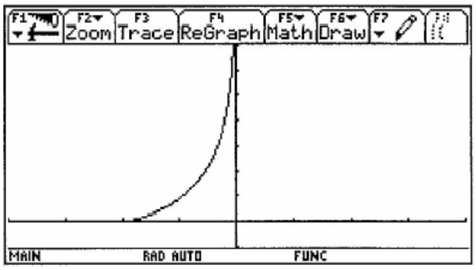


Let’s go on exploring ...

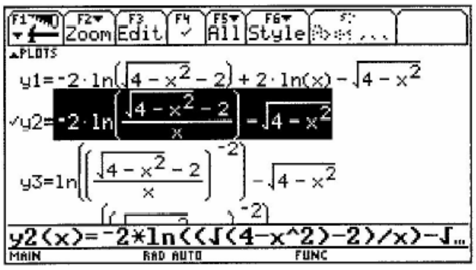
3. Attempt: once more with the *TI*s

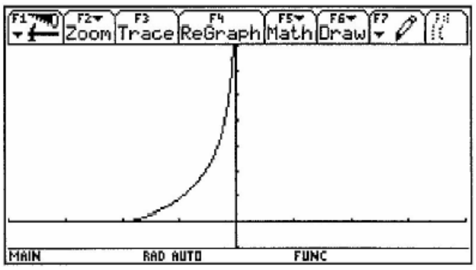
If we want to find the reason for this “bug” then we have to investigate the expression to be integrated very accurately. Just with the first summand we are lucky: the root is defined only for |*x*| > 0 and returns a value between 0 and 2. Subtracting 2 leads to the fact that the whole summand is undefined for any real number. It is more than astonishing that a curve is plotted yet!



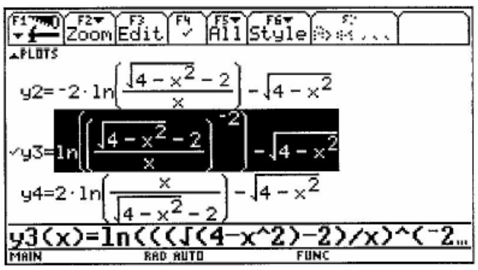


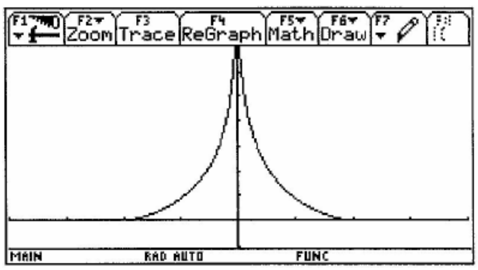
Let us now manipulate this expression (unfortunately we can do this only “manually” - or as we Austrians are saying “händisch = zu Fuß (by foot)” – because from my point of view the *TI* is too poor in manipulating logarithmic expressions) and then plot the respective expression. By collecting the first two summands we change the domain and `y2(x)` is now in fact defined for [-2,0].



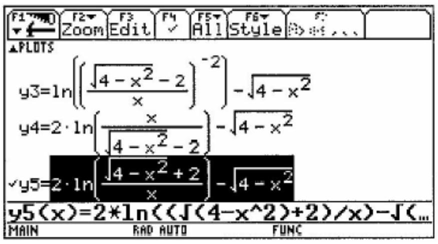


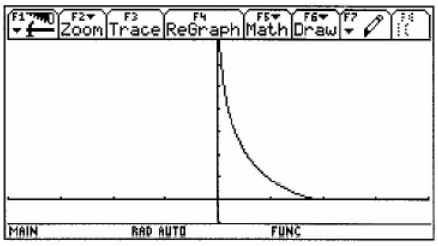
Dragging the factor -2 into the log-function the domain changes again: `y3(x)` and `y4(x)` are defined for [-2,0] ∪]0,2].





Plotting function `y4(x)` we obtain the same curve, what is quite clear. We have to remove the root from the denominator to receive the requested and expected result because the log-expression is now defined for only]0,2].





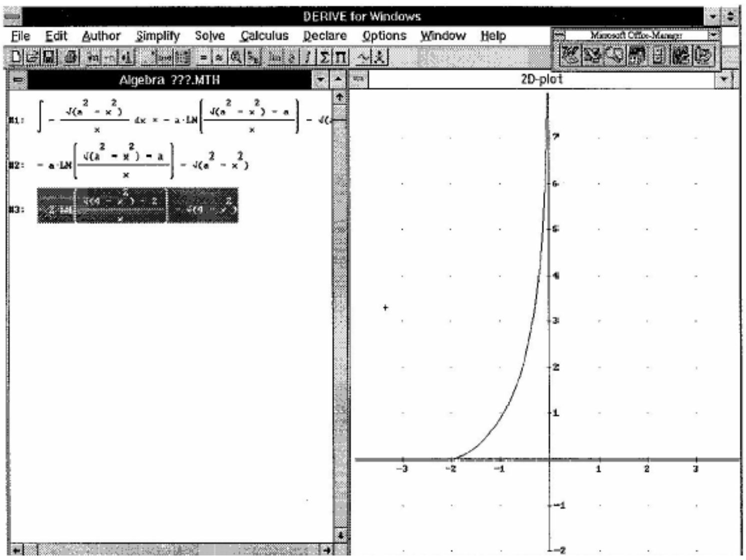
We can see that the manipulations connected with logarithm, root and square are influencing the domain of the graph in a very high degree. The more accurate inspection has shown that the "bug" in fact is no bug. But should we not wish to have immediately the solution as presented in `y5(x)`. As a user only I cannot estimate how overbold this wish really is ...

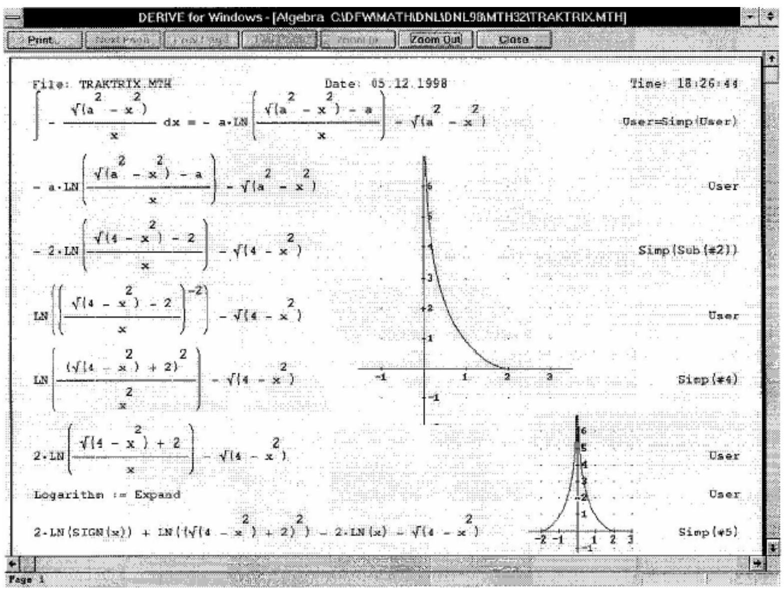
Now let us see how *DERIVE* is treating this integral:

4. Attempt: with *DERIVE* 4.10

Here manipulation of the expression goes one step further and so it is defined for]-2,0]. Function and graph are here – which is a difference to the first *TI*-solution - corresponding.

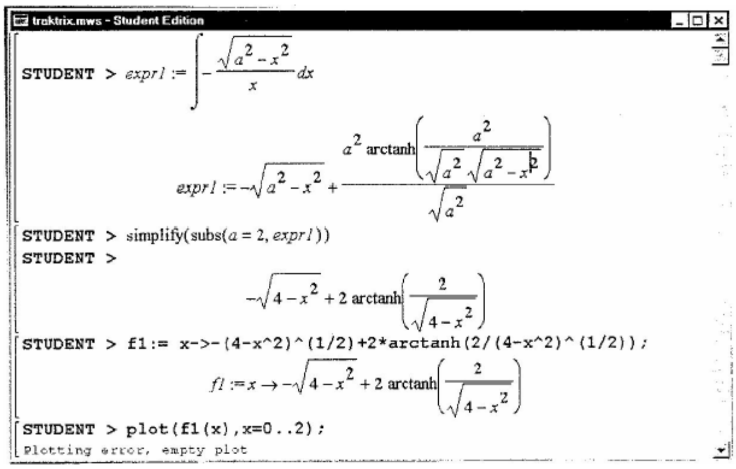
Again we need some manipulations (like on the *TI* but with the difference that the decisive step – removing the root from the denominator – is performed by *DERIVE*) to find the requested result (see #6:))





5. Attempt: with Maple V (R4 Student Version)

In this connection it might be interesting to compare how other CAS are dealing with this problem. Maple returns a new representation of the integral. But you will fail plotting the function graph - message of the system: "empty plot". Obviously here is nothing to plot: the Area function *arctanh*(x) is defined for $|x|<1$. This leads to the consequence that for no $x \in [0,2]$ the argument lies in the domain of *arctanh*(x).

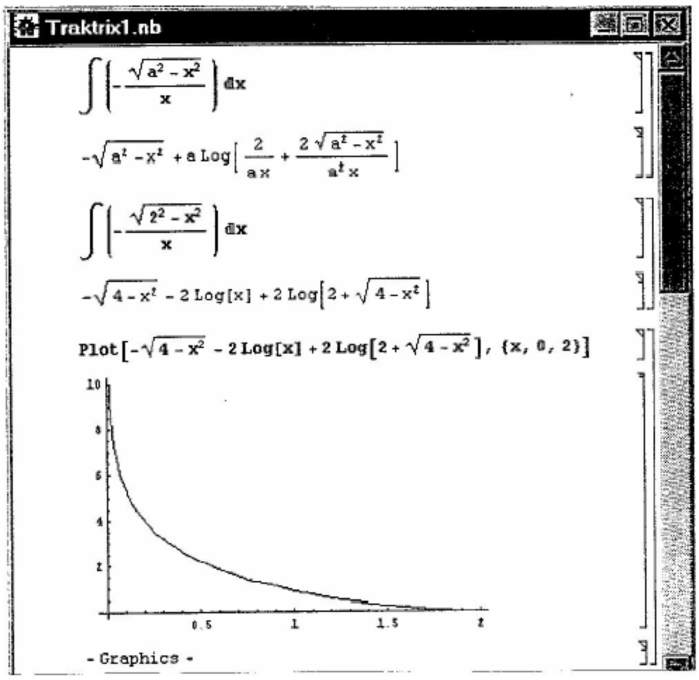


Using the relation $\operatorname{arctanh}(x) = \frac{1}{2}\ln\frac{1+x}{1-x}$ (with $|x|<1$) we find one of the known representations of our problem:

$$\operatorname{LN}\left(\frac{1+\frac{2}{\sqrt{4-x^2}}}{1-\frac{2}{\sqrt{4-x^2}}}\right) - \sqrt{4-x^2} = \operatorname{LN}\left(-\frac{\left(\sqrt{4-x^2}+2\right)^2}{x^2}\right) - \sqrt{4-x^2}$$

The log expression can again be transformed such that it represents the expected curve.

6. Attempt: with Mathematica 3.0 (Teacher's Edition)



MATHEMATICA behaves very brilliant. Version 2.21 returns exactly the function given by Heusler, version 3.0 a slightly modified one.

It leads directly to the graph.

[1] H. Heuser, Gewöhnliche Differentialgleichungen, Teubner Verlag, Stuttgart 1995
[2] Thomas Weth, A Lexicon of Curves (5), DERIVE Newsletter #16

DIOPHANTINE EQUATIONS (2)

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

Third Problem (presented in ACDC 6/DNL#29): The tiles problem could be solved as follows: let x^2 be the number of tiles used for the first nave, and y^2 for the third one. Then

$$x^2+\left(y+\frac{y}{2}\right)^2+y^2+1=2x^2 \Rightarrow x^2-\frac{13}{4}=1.$$

Putting $y/2$ = z, we have $x^2 - 13z^2 = 1$, which is **Pell's equation**.
For solving it we will use the second file explained in ACDC 7, adding some new functions:

```
#10:  [M(n):=,L(n):=]
#11:  M_AUX(n):=IF(n=1,0,w SUB 1)
#12:  L_AUX(n):=IF(n=1,1,num^2-w SUB 1^2)
#13:  L(n):=IF(n>2,(num^2-M(n)^2)/L(n-1),L_AUX(n))
#14:  M(n):=IF(n>2,L(n-1)*w SUB (n-1)-M(n-1),M_AUX(n))
#15:  TABL(k):=VECTOR([n,w SUB n,P(n),Q(n),M(n),L(n)],n,1,k)`
```

Pell's equation verifies that $P(n)^2 - num^2\, Q(n)^2 = (-1)^n\, L(n+1)$. In our case $num = \sqrt{13}$, and you will obtain the data for finding the solution of our equation, adding for example:

```
#16:  S_L(k):=VECTOR(IF(P(n)^2-num^2*Q(n)^2=1,[P(n),Q(n)],0),n,1,k)
#17:  SOL(k):=SELECT(i/=0,i,S_L(k))
```

and then approximating `SOL(10)`. The final sum of tiles then is: $2x^2 = 2 * 649^2 = 842\,402$.

For the last two problems we will make use of a trick involving the function `ROW_REDUCE`(*).

Fourth Problem:

$$\begin{cases} a+10b+100c=26(a+b+c) \\ c+10b+100a=48(a+b+c) \end{cases} \Rightarrow \begin{cases} 25a+16b=74c \\ 52a-38b=47c \end{cases}$$

Then simplify

$$\text{ROW\_REDUCE}\begin{bmatrix} 25 & 16 & 74\,c \\ 52 & -38 & 47\,c \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2\,c \\ 0 & 1 & \frac{3\,c}{2} \end{bmatrix}$$

so that we can conclude that the two possibilities are 234 and 468.

```
SOLUTIONS([25 a + 16 b = 74 c, 52 a - 38 b = 47 c], [a, b])
```

$$\left[\left[2\,c,\ \frac{3\,c}{2}\right]\right]$$

does the same job, Josef.

(*) *DERIVE*'s´ `ROW_REDUCE` function is equivalent to the *TI*'s rref() function!

But the **fifth problem** is not so immediate to be solved. You have to write nine equations: eight of them for rows, columns and diagonals are equal to 15; the ninth with the sum of digits from 1 through 9, which is 45. Then we can use `ROW_REDUCE` like in the former question.

As the matrix is singular, some rows result in zeros. Hence we can write $x_1 = 10-t$, $x_2 = 10-z$, $x_3 = -5+t+z$, $x_4 = -10+2t+z$, $x_5 = 5$, $x_6 = 20-2t$, $x_7 = 15-t-z$, $x_8 = z$, and $x_9 = t$. Now we can calculate all possibilities which can created in a statement like

```
VECTOR(VECTOR([10-t,10-z,-5+t+z,-10+2t+z,20-2t-z,15-t-z,t,z],z,1,9),t,1,9)
```

and then select from it the solutions which have no numbers repeated (how?).
(MAGSQ.MTH)

Finally a challenge for which I have been thinking about a lot of time: Has anyone any idea how to obtain the 880 different order 4 magic squares with *DERIVE*?

I gave a workshop at TIME 2014 – "Brain Twisters – and how to tackle them". There were also problems like Alfonso's fourth problem, Josef.

`vars(v,k)` creates all permutations (with repetitions) of order *k* of the elements given in vector *v*. The two conditions are described as equations. `SELECT` finds out all three element vectors (out of 1000) which satisfy both conditions.

```
vars(v, k, b, k_, m_ := 0, n_, s_ := [], t_) :=
  Loop
    b := DIM(v)
    If m_ = b^k
       RETURN REVERSE(s_)
    k_ := k
    n_ := m_
    t_ := []
    Loop
      t_ := ADJOIN(v↓(MOD(n_, b) + 1), t_)
      n_ := FLOOR(n_, b)
      k_ :- 1
      If k_ = 0 exit
    s_ := ADJOIN(t_, s_)
    m_ :+ 1

all := vars([0, 1, 2, 3, 4, 5, 6, 7, 8, 9], 3)
```

$[a := v_1, b := v_2, c := v_3]$

cond1 := $100 \cdot a + 10 \cdot b + c = 26 \cdot (a + b + c)$

cond2 := $100 \cdot c + 10 \cdot b + a = 48 \cdot (a + b + c)$

$$\text{SELECT}(\text{cond1} \wedge \text{cond2}, v, \text{all}) = \begin{bmatrix} 0 & 0 & 0 \\ 2 & 3 & 4 \\ 4 & 6 & 8 \end{bmatrix}$$

needs 0.3 sec calculation time.

What you always wanted to know about the Undefined Variable Error but never dared to ask

Wolfgang Pröpper Nürnberg, Germany

Each TI-92 user is familiar with the error message shown in the title. From the experience the undersigned has gained in many teacher training courses on the TI-92 (as well as from his own inadequacy) it might even be one of the most frequently occurring errors. In this contribution the reasons for it are identified and methods are shown, how one can avoid and overcome it.

1. As the error message says, the TI-92 has come across an undefined variable. Depending on the situation it can be easy or more difficult to identify this (or those) variable(s).
A first remark: This error message only occurs with graphing and/or tabulating functions (which are closely related). Generally algebraic manipulations (like limits, derivatives, etc.) in the Home Screen do not ask whether a variable is declared or not.
When searching for the reason, one should look at the error message exactly: There are two slightly different forms of the message.

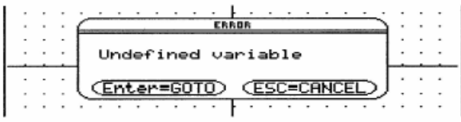


Fig. 1

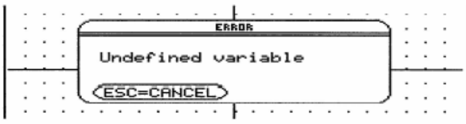


Fig. 2

With the message shown in Fig. 1, i.e. the one with **two alternatives**, you have already nearly found the reason:
With an [ENTER] you return to the Home Screen and the entry line shows for example (s. Fig. 3)

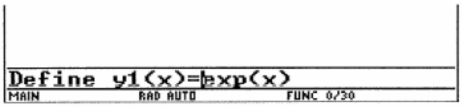


Fig. 3

Fig. 4

Looking into the Y= Editor (s. Fig. 4) belonging to it, one sees that the function which is said to be defined in the entry line is activated there. Possibly in this example the exponential function was to be plotted. And indeed it is called exp(x) in nearly every programming language. With the TI-92 however it is reached by [2nd] [LN].
The same case also occurs when one refers to a function in the Y= Editor, which was declared once, but has since been deleted (e.g. with DelVar f or by pressing [F6]). However the corresponding line in the Y= Editor is still activated. Changing to the Graph Screen causes the error message. The case described here in Graph Mode "Function" is analogous to the four other plot modes.

A similar situation to the case described above occurs when declaring a term in x with an additional parameter in the Y= Editor (s. Fig. 5). When assigning values to that parameter using the "with" operator ([2nd] [K]), everything is okay and the TI-92 plots (like in this case) the graph of a family of curves (s. Fig. 6).

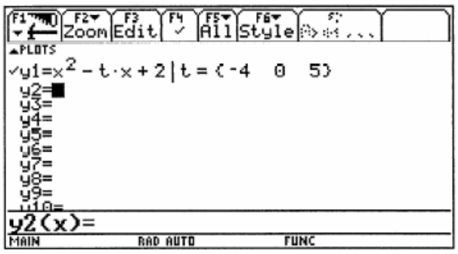


Fig. 5

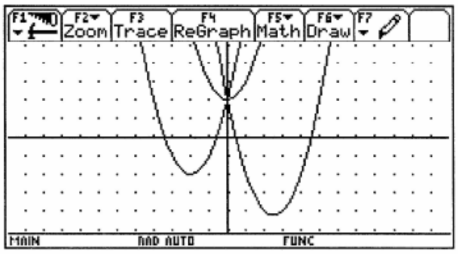


Fig. 6

However one must be careful: When declaring a function with parameters in an analogous way in the Home Screen (Fig. 7) and then trying to plot the family (Fig. 8), one gets "shipwrecked". The TI-92 does not pass the "with" statement to the symbolic term f(x). Instead it generates an error message like in Fig. 1 and displays it on [ENTER] like in Fig. 3.

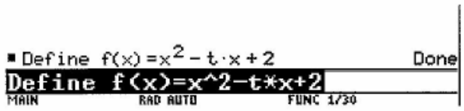


Fig. 7

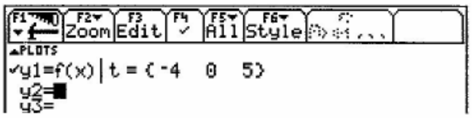


Fig. 8

(This flaw has been eliminated in the TI-92 Plus and the situation shown in Fig. 7/8 works without any error message.)

There are two ways to overcome the errors described above:

- One either deletes or changes the functions that caused the error message - either in the Home Screen or in the Y= Editor - or
- one deactivates the functions with [F4] (in the Y= Editor) or executes the FnOff statement in the Home Screen ([F4] [8]).

2. A totally different way to generate the undefined variable error involves using the Graph statement of the TI-92. It is copied from the Other menu in the Home Screen to the entry line by [F4] [2].

Everybody will regard it a truism that the statement Graph f(x) will cause an error message if the function f was not declared. The message comes in the Graph Screen and can only be closed by [ESC] (Fig. 2). Subsequently one returns to the Home Screen with [♦] [Q] and finds, as a result of the Graph statement, the familiar error message (Fig. 9, 1st line).

The case becomes interesting if one now declares a function with another name and tries to plot it with Graph g(x). The same error message occurs. One feels totally lost if a direct Graph statement as shown in Fig. 9 is added. Here the same undefined variable error arises!

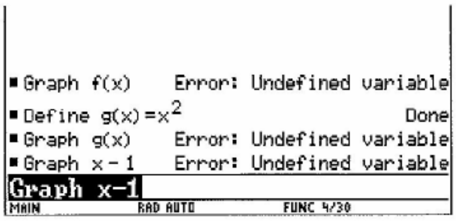


Fig. 9

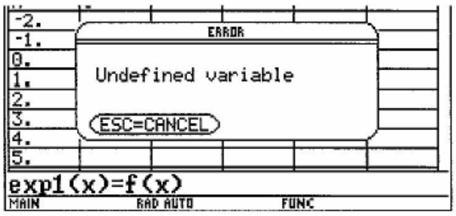


Fig. 10

An explanation is found, in the Table Screen ([♦] [Y]). Here an undefined variable errror message is issued first. After closing it with [ESC] one sees the reason (Fig. 10). The table for an expression exp1(x)=f(x) with an undeclared function f is to be generated. That means the Graph statement has done more than its name says. It did not only try to plot in the Graph Screen but has also written its argument into the Table Screen.

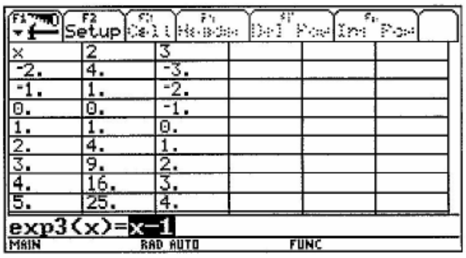


Fig. 11

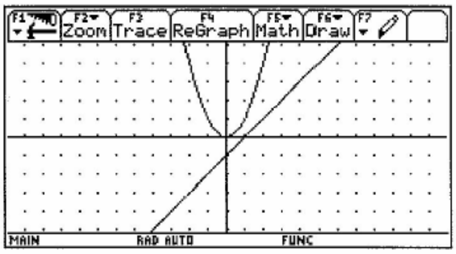


Fig. 12

After deleting the highlighted f(x) with the [CLEAR] key (and not forgetting [ENTER] !) one finds two tables which are numbered 2 and 3 (Fig. 11). One can be easily convinced that these are the tables of the two functions which were to be plotted with the 2nd and 3rd Graph statement. In the Graph Screen ([♦] [R]) one sees that the plot statements have been executed (Fig. 12).

The easiest way to remove the error message in this case is the ClrGraph statement (it can be found in [F4] [5]). It deletes all entries in the Table Screen including the cause for the error. Extremely cautious people use a ClrGraph prior to each Graph statement.

3. Finally, a very insidious situation:

The TI-92 giudebook deals on page 188 ff. with a "Preview of Statistics and Data Plots". Carry out steps 1 to 4 and 12 to 14. Now switch with [♦] [R] to the Graph Screen. Normally you will see nothing because your coordinate system will not be dimensioned in the right way. A simple ZoomData ([F2] [9]) however should produce a picture like Fig. 13. One can be easily convinced, (for instance by tracing: [F3]) that the set of points is nothing but the data (pairs of numbers) stored in build.

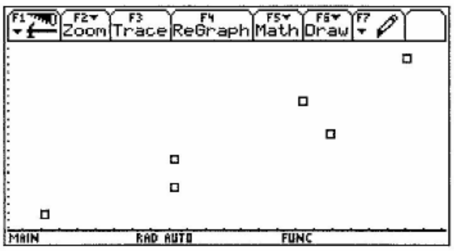


Fig. 13

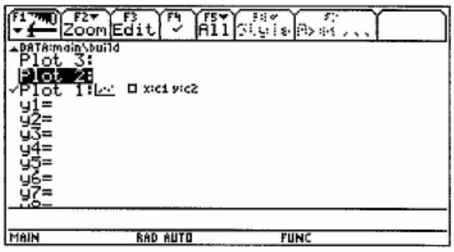


Fig. 14

If you now change to the Home Screen, delete the data variable with DelVar build, and then switch to the Graph Screen again with [♦] [R], our "popular" error message is back again. (Instead of deleting the variable build and possibly having to repeat the entry procedure mentioned above, one can rename it in the Var-Link Screen ([2nd] [-]) using [F1] [3] there.)

Of course one knows the reason for the error. However if this data variable was deleted some time previously and you have not accessed the Graph Screen since, you may not remember the deletion and have difficulty localizing the error. You will not find any illegal Graph statement in the Home Screen and you will find practically no hints to that error in the Y= Editor. A close examination shows the word PLOTS with an up arrow (τ) between the Y1= line and the toolbar in very small characters. A few ⌃ yield the picture shown in Fig. 14. The little tick in front of PLOT1 indicates that this line is active. With the [F4] key this line can be deactivated or the PLOT can be deletetd with [CLEAR].

Conclusion: If the undefined variable error occurs after the ClrGraph statement has been executed and all functions in the Y= Editor have been deactivated, then there onla remains to examine the PLOTS area (which is the same in all graph modes).

In case, however, you find other occurences of this error please let me know. My eMail-adress is: w.proepper@wpro.franken.de

The next contribution has its source in a lecture given by Dr. Maria Koth at the Vienna University at the occasion of a so called "Teachers' Day". Every year one day in spring time – the first Friday after Easter – the university opens its doors for math teachers. Professors and teachers give lectures on various fields of interests for secondary school teachers. Dr. Koth is one of the university teachers who try to inform the students about CAS and how to use it in school. Her lectures are always well prepared and I have the pleasure to present the essentials of her talk "Graphics with DERIVE". I believe that it might be useful for beginners and for more advanced users of DERIVE as well. Josef

Computergrafik mit *DERIVE* – Computer Graphics with *DERIVE*

Dr. Maria Koth, University of Vienna, Austria

Contents:

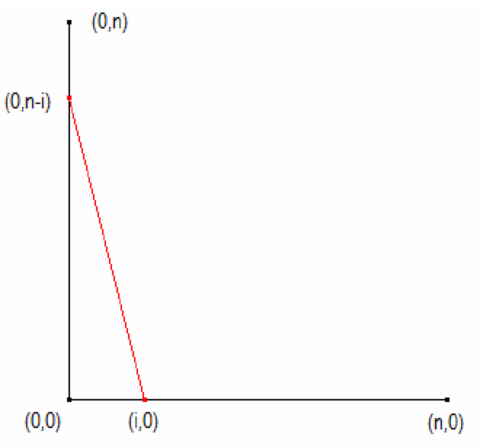
0. Introduction

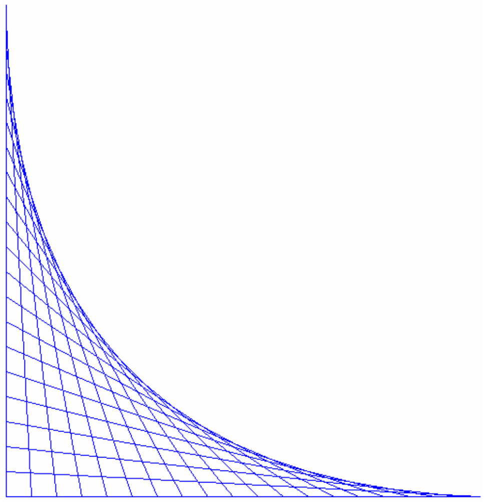
First take care for the following settings: Work in Approximate Mode with not more than 3 digits to display. In the Plot Window switch off the Axes and Color Cycling. Point Size should be Small and Points should be Connected.

1. Elementary Graphics with Lines

1.1 Graphics over a right angle

The two segments on both axes with equal lengths are divided in *n* equal intervals.
Every i[th] dividing point on one segment is connected with the (n-i)[th] dividing point on the other segment.
Then the following graphic is created:





GRAFIK1(20)

We choose (0,0), (0,*n*) and (*n*,0) as endpoints of the two segments because then it is easy to shift, stretch or shrink the graphic using Center and Scale.

To produce a similar graphic using a programming language needs a lot of work. The "*DERIVE* program" consists of only one single line:

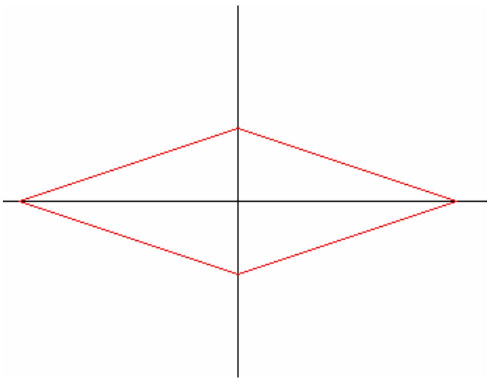
```
GRAFIK1(n):= VECTOR([i,0;0,n-i],i,0,n)
```

Enter now `GRAFIK1(20)` and plot, done.

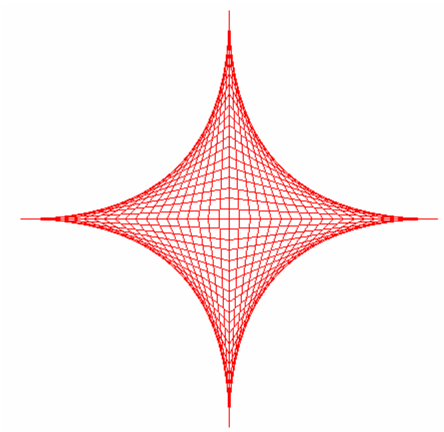
(Later we will try to reproduce the graphics with *TI-Nspire*, Josef)

You can describe the graphics below by simple *DERIVE* commands very similar.

```
GRAFIK2(n):= VECTOR([i,0;0,n-i;-i,0;0,i-n;i,0],i,0,n)
```

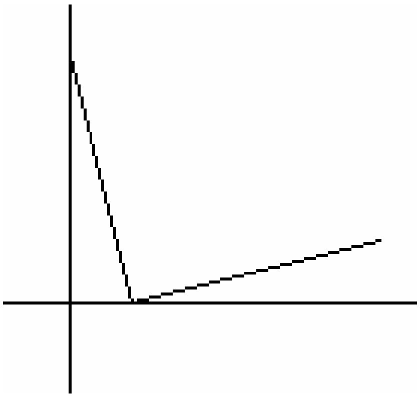


GRAFIK2(20)SUB 16

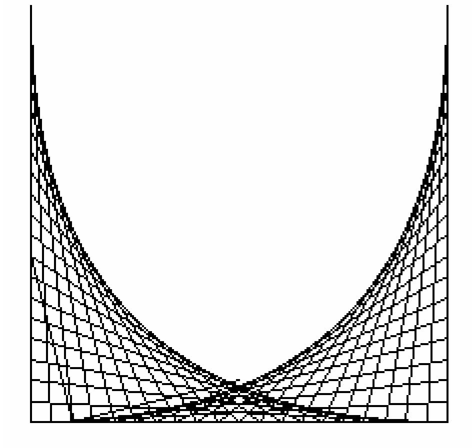


GRAFIK2(20)

GRAFIK3(n):= VECTOR([0,n-i;i,0;n,i],i,0,n)

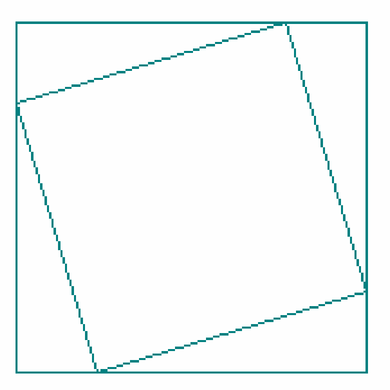


GRAFIK3(10)SUB 3

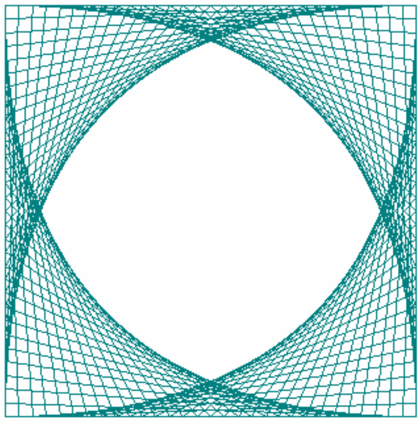


GRAFIK3(20)

GRAFIK4(n):= VECTOR([0,n-i;i,0;n,i;n-i,n;0,n-i],i,0,n)



GRAFIK4(30)SUB 8

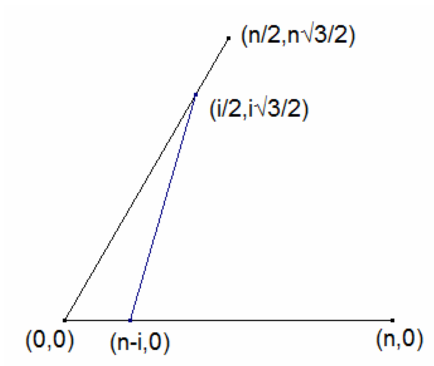


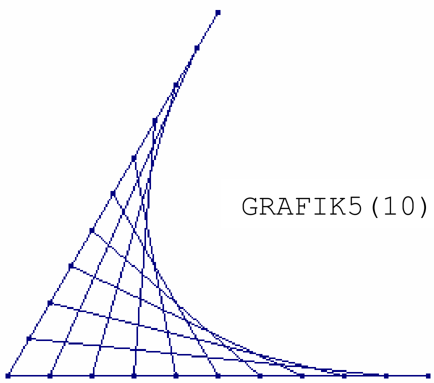
GRAFIK4(30)

1.2 Graphics over a 60° angle

Starting with two segments enclosing an angle of 60° the segments form two sides of an equilateral triangle. From lower secondary school the pupils (should) know the altitude of such a triangle of side length n with $\frac{n\sqrt{3}}{2}$.

GRAFIK5(n):= VECTOR([n-i,0;0.5 i,0.5 i √3],i,0,n)

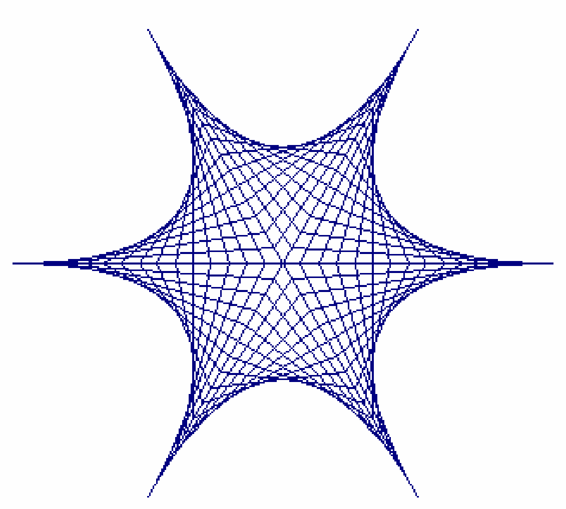




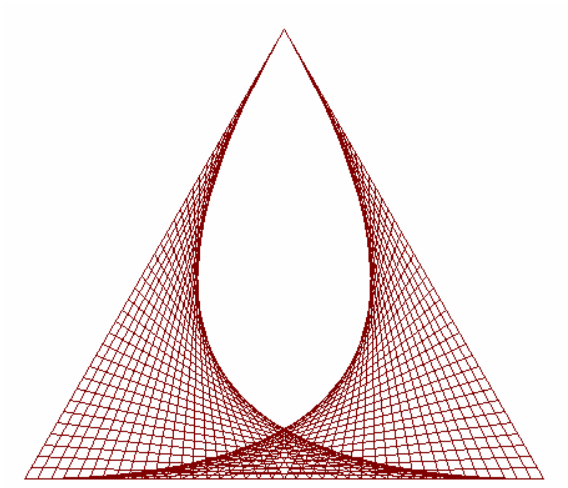
GRAFIK5(10)

The following patterns are produced by `GRAFIK6(n)`, `GRAFIK7(n)` and `GRAFIK8(n)`.

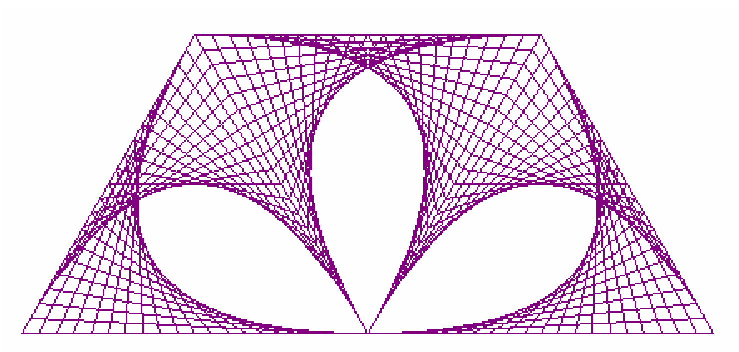
```
GRAFIK6(n):= VECTOR([i,0;0.5 (n-i),0.5 (n-i) √3;-0.5 i,0.5 i √3;
             i-n,0;-0.5 i,-0.5 i √3;0.5 (n-i),0.5 (i-n) √3;i,0],i,0,n)

GRAFIK7(n):= VECTOR([0.5 (n-i),0.5 (n-i) √3;i,0;n-0.5 i,
             0.5 i √3],i,0,n)

GRAFIK8(n):= VECTOR([i,0;n-0.5 i,0.5 i √3;0.5 (n-i),0.5 (n-i) √3;
             -0.5 n+i,0.5 n √3;-0.5 i,0.5 i √3;-0.5 (n+i),
             0.5 (n-i) √3;i-n,0],i,0,n)
```



GRAFIK6(15)



GRAFIK7(40)



GRAFIK8(20)

Try this:

```
GRAFIK9(n):= VECTOR([i,0;n-0.5 i,0.5 i √3;0.5 (n-i),0.5 (n-i) √3;
             -0.5 n+i,0.5 n √3;-0.5 i,0.5 i √3;-0.5 (n+i),
             0.5 (n-i) √3;i-n,0;-0.5 (n+i),0.5 (i-n) √3;-0.5 i,
             -0.5 i √3;-0.5 n+i,-0.5 n √3;0.5 (n-i),0.5 (i-n) √3;
             n-0.5 i,-0.5 i √3;i,0],i,0,n)

GRAFIK9(50)
```

2. Regular Polygons

2.1 Plotting regular polygons

The coordinates of a regular polygon with n vertices, its center in (0,0) and r as radius of its circumcircle are given by:

$$\left[r\cdot\cos\left(\frac{2\pi i}{n}\right), r\cdot\sin\left(\frac{2\pi i}{n}\right)\right] \text{ with } 0 \le i \le n-1.$$

Using the *DERIVE* syntax this n-gon is described by the following list of points:

`POLYGON(n,r):=VECTOR([r·COS(2πi/n),r·SIN(2πi/n)],i,0,n).`

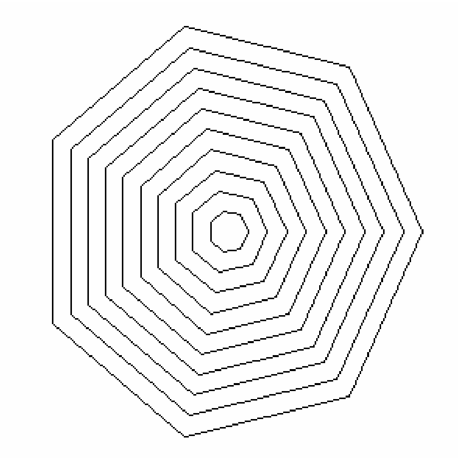
There is another possibility to fix the vertices by:

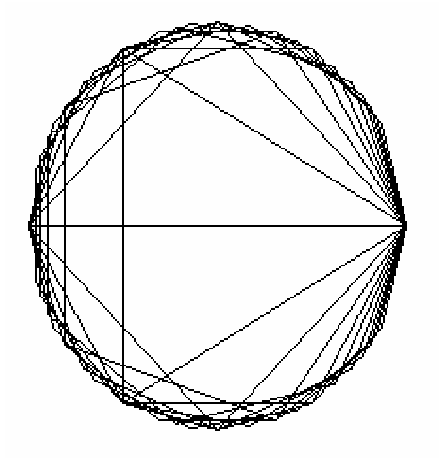
$$\left[r\cdot\cos(i), r\cdot\sin(i)\right] \text{ with } 0 \le i \le 2\pi \quad \text{step } \frac{2\pi}{n}.$$

`POLYGON2(n,r):=VECTOR([r·COS(i),r·SIN(i)],i,0,2π,2π/n).`

Try `POLYGON(9,3)` or `POLYGON2(12,4)`. Explore the `VECTOR` command with

`VECTOR(POLYGON2(7,t),t,0.5,5,0.5)` and `VECTOR(POLYGON2(t,4),t,1,12)`.





The line graphs with TI-NspireCAS & LUA (by Josef Böhm)

Now – several years later – times of *DERIVE*, TI-92 and even Voyage 200 seem to have passed. Instead of this we have new tools like *TI-NspireCAS*, GeoGebra, CASIO, ...

I will show how to create Maria Koth's line graphs with *TI-NspireCAS*, which is not so easy because we miss my favourite *DERIVE* command – `VECTOR`. The students – and we too – are using `VECTOR` as a tool, nothing else than a black box. So let's provide another black box for *Nspire*. The important fact is finding out the "rule" for generating the petty families of segments given by the matrices as first argument in the `VECTOR` construct.

Another problem is that we cannot program the graph window directly – as it was possible with TI-92 and Voyage 200. So, how to plot the family of segments?

One solution is working with a LUA-script, and I will demonstrate how to do this:

First of all I need a program for calculating the coordinates of the vertices of the segments in the right order.

Then I open a Notes page and define the rules as matrices from gr1 through gr9 – and can find my own rules, too.

For using the variables in the LUA-script it is necessary to work in approximate mode (i.e. enter at least one number in the "rule" as a decimal number).

```
lgraph                                                    6/10
Define lgraph(rule,n_)=
Prgm
Local pt,i_,k
l:={[]}
d:=dim(rule)[1]
For i_,0,n_
   For k,1,d
    pt:=rule[k]|i=i_ and n=n_
    l:=augment(augment(l,{pt[1,1]}),{pt[1,2]})
   EndFor
EndFor
m:=dim(l)
EndPrgm
```

$$\mathbf{gr1}:=\begin{bmatrix} i & 0 \\ 0 & n-i \end{bmatrix} \blacktriangleright \begin{bmatrix} i & 0 \\ 0 & n-i \end{bmatrix}$$

$$\mathbf{gr2}:=\begin{bmatrix} i & 0 \\ 0 & n-i \\ -i & 0 \\ 0 & i-n \\ i & 0 \end{bmatrix} \blacktriangleright \begin{bmatrix} i & 0 \\ 0 & n-i \\ -i & 0 \\ 0 & i-n \\ i & 0 \end{bmatrix}$$

$$\mathbf{gr3}:=\begin{bmatrix} 0 & n-i \\ i & 0 \\ n & i \end{bmatrix} \blacktriangleright \begin{bmatrix} 0 & n-i \\ i & 0 \\ n & i \end{bmatrix}$$

$$\mathbf{gr4}:=\begin{bmatrix} 0 & n-i \\ i & 0 \\ n & i \\ n-i & n \\ 0 & n-i \end{bmatrix} \blacktriangleright \begin{bmatrix} 0 & n-i \\ i & 0 \\ n & i \\ n-i & n \\ 0 & n-i \end{bmatrix}$$

$$\mathbf{gr5}:=\begin{bmatrix} n-i & 0 \\ 0.5\cdot i & 0.5\cdot i\cdot\sqrt{3} \end{bmatrix} \blacktriangleright \begin{bmatrix} n-i & 0 \\ 0.5\cdot i & 0.866025\cdot i \end{bmatrix}$$

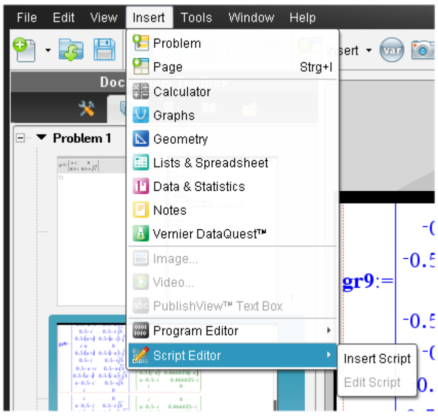
$$\begin{bmatrix} i & 0 \\ n-0.5\cdot i & 0.5\cdot i\cdot\sqrt{3} \\ 0.5\cdot(n-i) & 0.5\cdot(n-i)\cdot\sqrt{3} \\ \dots & \dots \end{bmatrix} \quad \begin{bmatrix} i & 0 \\ n-0.5\cdot i & 0.866025\cdot i \\ -0.5\cdot(i-n) & -0.866025\cdot(i-n) \end{bmatrix}$$

Then I insert a LUA-script.

(For informing about LUA you may go to

http://compasstech.com.au/TNS_Authoring/Scripting/index.html

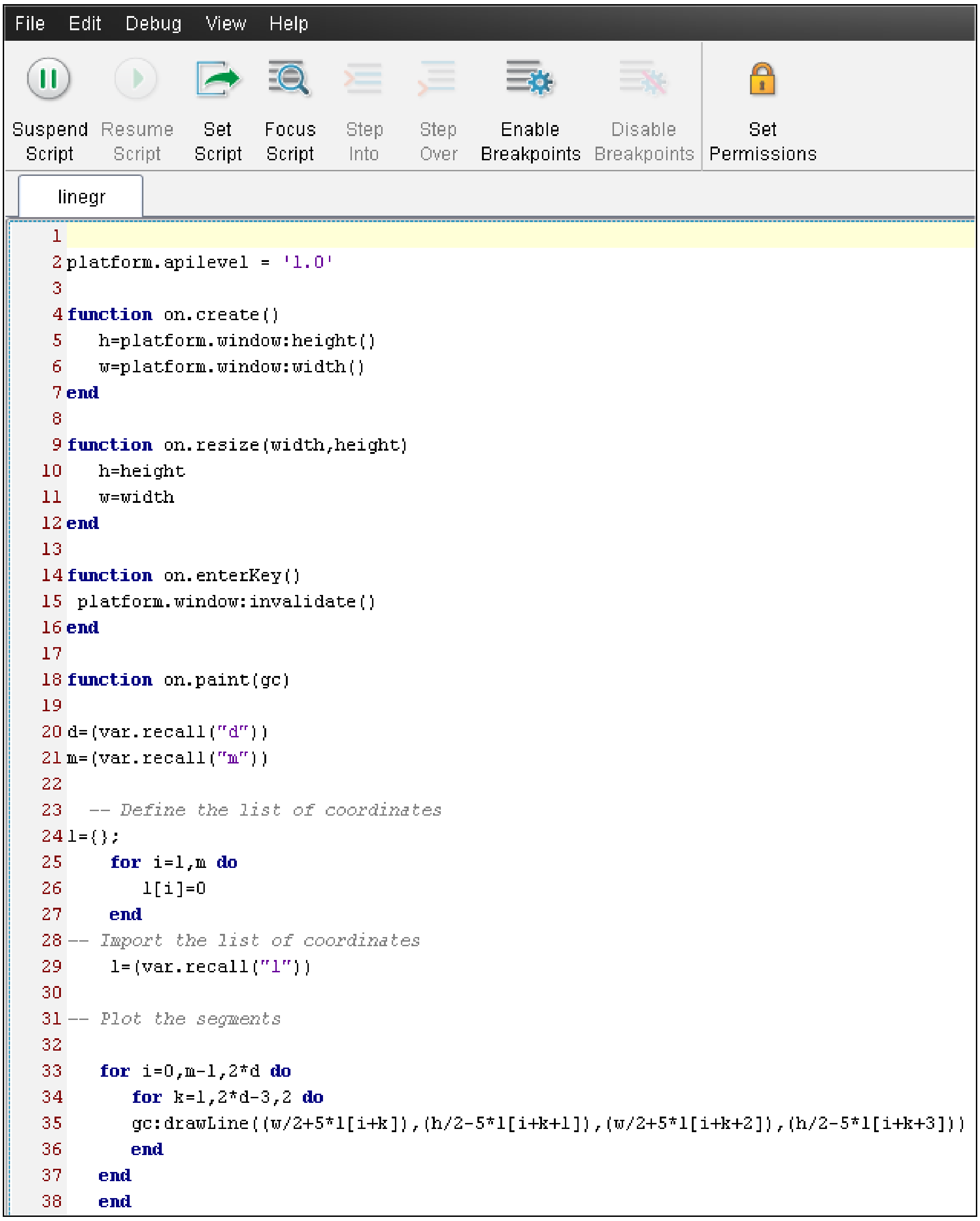
and explore Steve Arnold's great LUA website.)



More LUA resources are given at the end of this article.

Another LUA script is presented in the next contribution.

(Steve recommends working with later version apilevel '2.2'.)

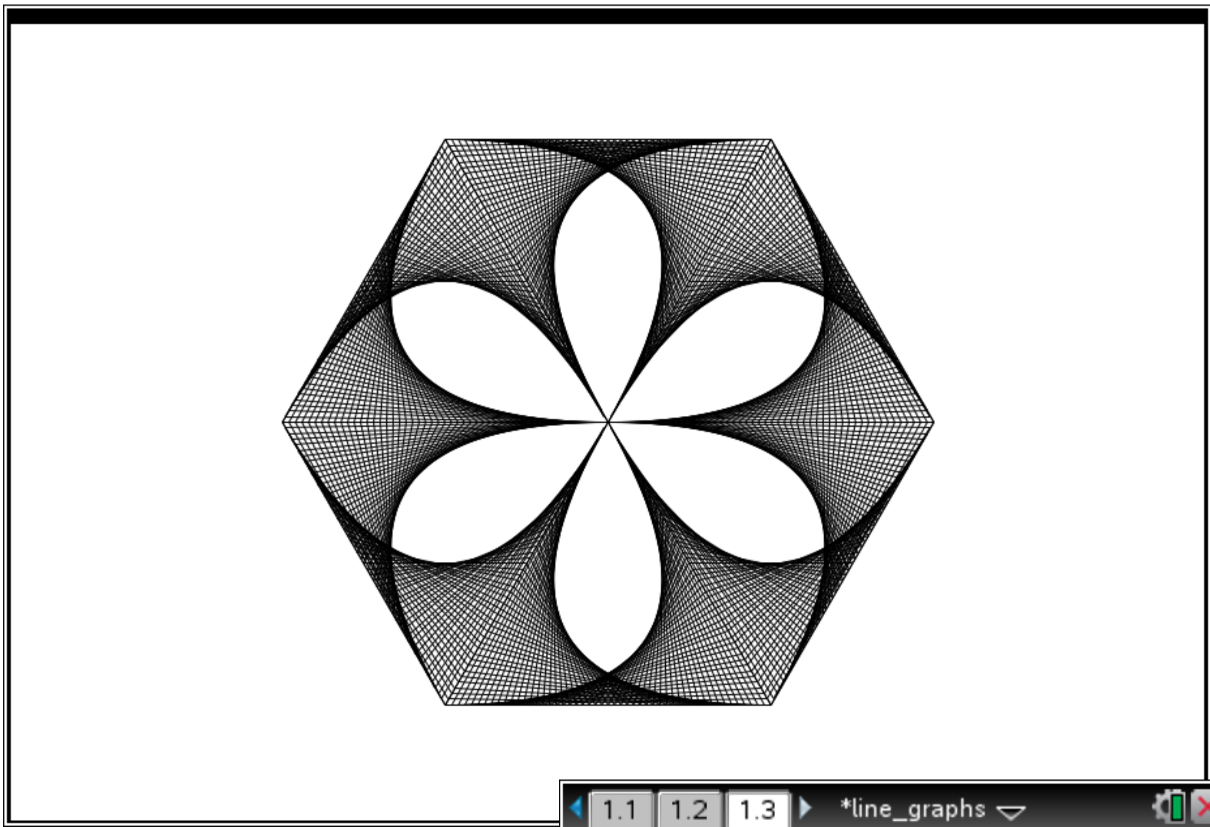


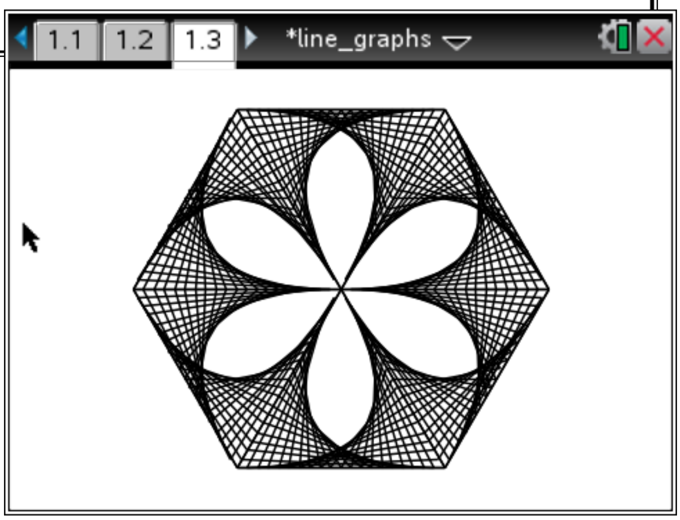

```

platform.apilevel = '1.0'

function on.create()
   h=platform.window:height()
   w=platform.window:width()
end

function on.resize(width,height)
   h=height
   w=width
end

function on.enterKey()
 platform.window:invalidate()
end

function on.paint(gc)

d=(var.recall("d"))
m=(var.recall("m"))

  -- Define the list of coordinates
l={};
    for i=1,m do
       l[i]=0
    end
-- Import the list of coordinates
    l=(var.recall("l"))

-- Plot the segments

   for i=0,m-1,2*d do
      for k=1,2*d-3,2 do
      gc:drawLine((w/2+5*l[i+k]),(h/2-5*l[i+k+1]),(w/2+5*l[i+k+2]),(h/2-5*l[i+k+3]))
      end
   end
   end
```

Run the program from the Notes page.

$$gr9:=\begin{bmatrix} \cdots & \cdots \\ i-n & 0 \\ -0.5\cdot(n+i) & 0.5\cdot(i-n)\cdot\sqrt{3} \\ -0.5\cdot i & -0.5\cdot i\cdot\sqrt{3} \\ -0.5\cdot n+i & -0.5\cdot n\cdot\sqrt{3} \\ 0.5\cdot(n-i) & 0.5\cdot(i-n)\cdot\sqrt{3} \\ n-0.5\cdot i & -0.5\cdot i\cdot\sqrt{3} \\ i & 0 \end{bmatrix} \blacktriangleright \begin{bmatrix} \cdots & \cdots \\ i-n & 0 \\ -0.5\cdot(i+n) & 0.866025\cdot(i-n) \\ -0.5\cdot i & -0.866025\cdot i \\ i-0.5\cdot n & -0.866025\cdot n \\ -0.5\cdot(i-n) & 0.866025\cdot(i-n) \\ n-0.5\cdot i & -0.866025\cdot i \\ i & 0 \end{bmatrix}$$

$lgraph(gr9,50) \blacktriangleright$ *Done*





Just overwrite the parameters in the program call. Don't enter it a second time.

On the handheld we have to take *n*=20. Other wise the graph would become too large.

http://www.lua.org/
http://lua.gts-stolberg.de/
http://www.t3vlaanderen.be/fileadmin/t3-be/cahiers/cahier_35.pdf

The Triangle of PASCAL

Josef Böhm, Würmla, Austria

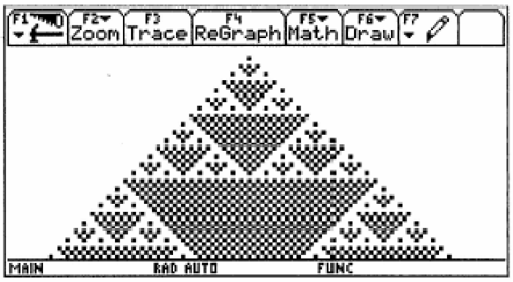
On the last page I'd like to present a nice program which brought a lot of fun in some of my math lessons. My students explored the binomial theorem, built up the Triangle of Pascal and learned how to use its numbers finding the expansion of powers of binomials considering additionally the pattern in its exponents.

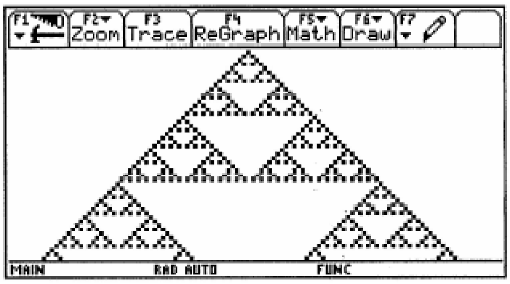
As a home work I gave them a sheet of paper with two 21 lines Pascal Triangles and asked them to mark in one figure all odd numbers and in the other one all numbers which are divisible by 3.

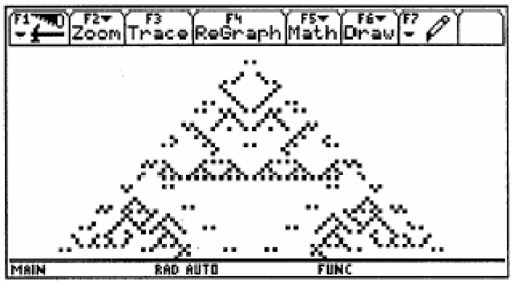
They were very much surprised about the nice pattern which appeared under their hands. Then I transmitted a *TI*-92 program `pas()` to their *TI*s and asked them first to remove the axes from the graph screen. As they had no plotting experience on the *TI* I need not caring for the `[Y=]`-editor.

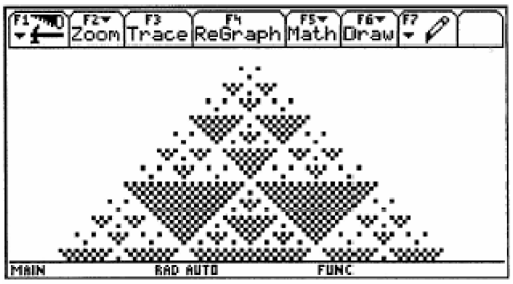
They ran `pas(50,2,0,1)` and then they started investigating:

```
pas(n,a,r,θt)
Prgm
Local i,j,z,pt
Define pt(x,y)=Prgm
PxlOn x,y:PxlOn x,y:PxlOn x+1,y
PxlOn x,y-1:PxlOn x+1,y-1
EndPrgm
ClrDraw
FnOff
For i,0,n
  For j,0,i
    nCr(i,j)→z
    If mod(z,a)=r and θt=1 or
     mod(z,a)≠r and θt=0 Then
    pt(1+2*i,117-2*i+4*j)
    EndIf
EndFor
EndFor
FnOn
EndPrgm
```







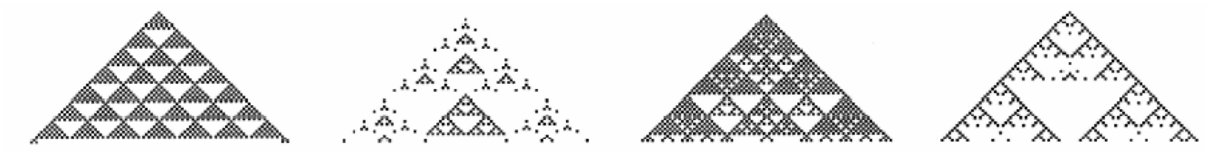


`pas(n,a,r,θt)` gives the first *n* rows of the Triangle. The numbers which leave the remainder *r* by division by *a* are marked as a black spot (if *θt* = 1). If *θt* = 0 then all positions are marked where the remainders is not *r* (i.e. the negative picture).

The students had the task to find nice PASCAL pictures in order to produce their PASCAL GALLERY. Another challenge was to find out the parameters of pas() for given pictures. (The third one was a bit tricky for them.)

Günter Scheu wrote an article on "*Explorations in the Triangle of Pascal*" in DNL#9. Then – this was in 1993 – I was inspired by his contribution and by a chapter in Peitgen's very recommendable book "*Fractals for the Classroom*" and provided the students with some *DERIVE* functions to produce similar figures (see `tripas.dfw`)

The latest versions of *DERIVE* offered the possibility for programming (see `pascal_new.dfw`).



So we can do it with DERIVE and on the TI-92 but how to do it with TI-NspireCAS?

We can again work with a LUA script, but we can also plot the respective points as a scatter diagram on a Geometry page or as a diagram in the statistics page. See the following screen shots. Base of all is a program similar to `pas()` given above.

```
pas
Define pas(nn,aa,rr,s)=
Prgm
Local i,j,z,txt
lx:={ }:ly:={ }
n:=nn:a:=aa:r:=rr
txt:={"is not remainder","is remainder"}
For i,0,nn
    For j,0,i
      z:=mod(nCr(i,j),aa)
      If z=rr and s=0 or z≠rr and s=1 Then
      lx:=augment(lx,{-4·i+8·j}): ly:=augment(ly,{6·i})
       EndIf
     EndFor
EndFor
nl:=dim(lx)
tx:=txt[s+1]
EndPrgm
```

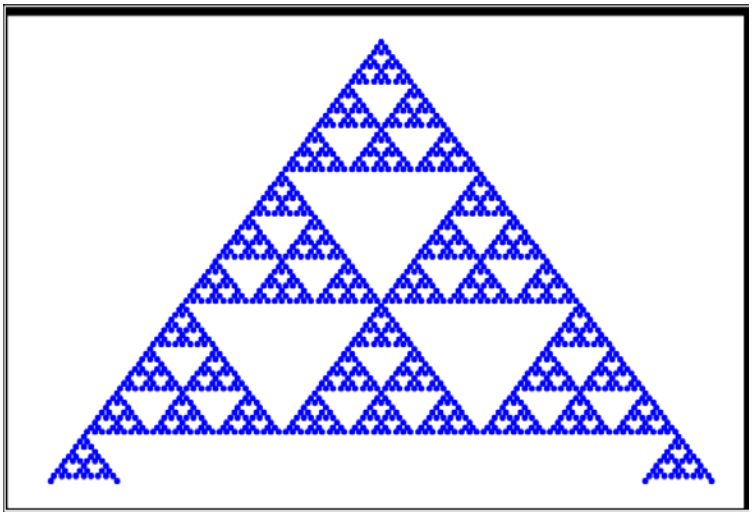
We can run `pas()` from the Notes:

```
pas(90,3,0,1) ▸ Done

Divisor: 3
0   is remainder
Number of Points: 1372
```

You can insert the lists of coordinates in a Lists & Spreadsheet page. Please notice that I reversed the signs of the *y*-values because I wrote the program for being used in connection with the LUA-program.

Running pas() with other arguments will change the plots immediately.

	A x_l	B y_l
=	=lx	=⁻ly
1	0	0
2	⁻4	⁻6
3	4	⁻6
4	⁻8	⁻12
5	0	⁻12
6	8	⁻12
7	⁻12	⁻18
8	12	⁻18
9	⁻16	⁻24
10	⁻8	⁻24
11	8	⁻24
12	16	⁻24

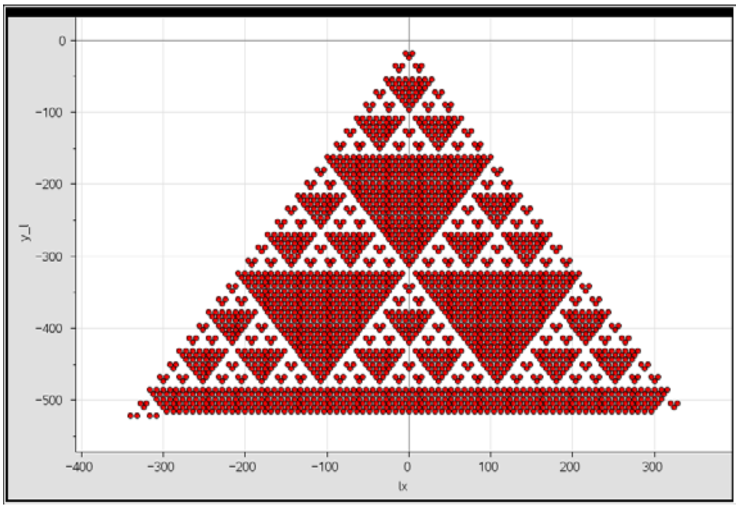


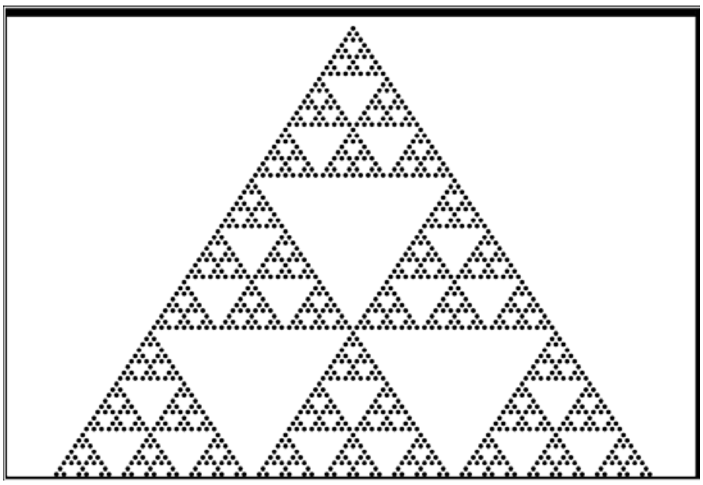
But you can also use the two lists to be presented in a Data & Statistics page: pas(90,3,0,0)

See finally the LUA-generated plot in a Geometry page.

Scatter plot and Data plot as well are restricted to 2500 points.

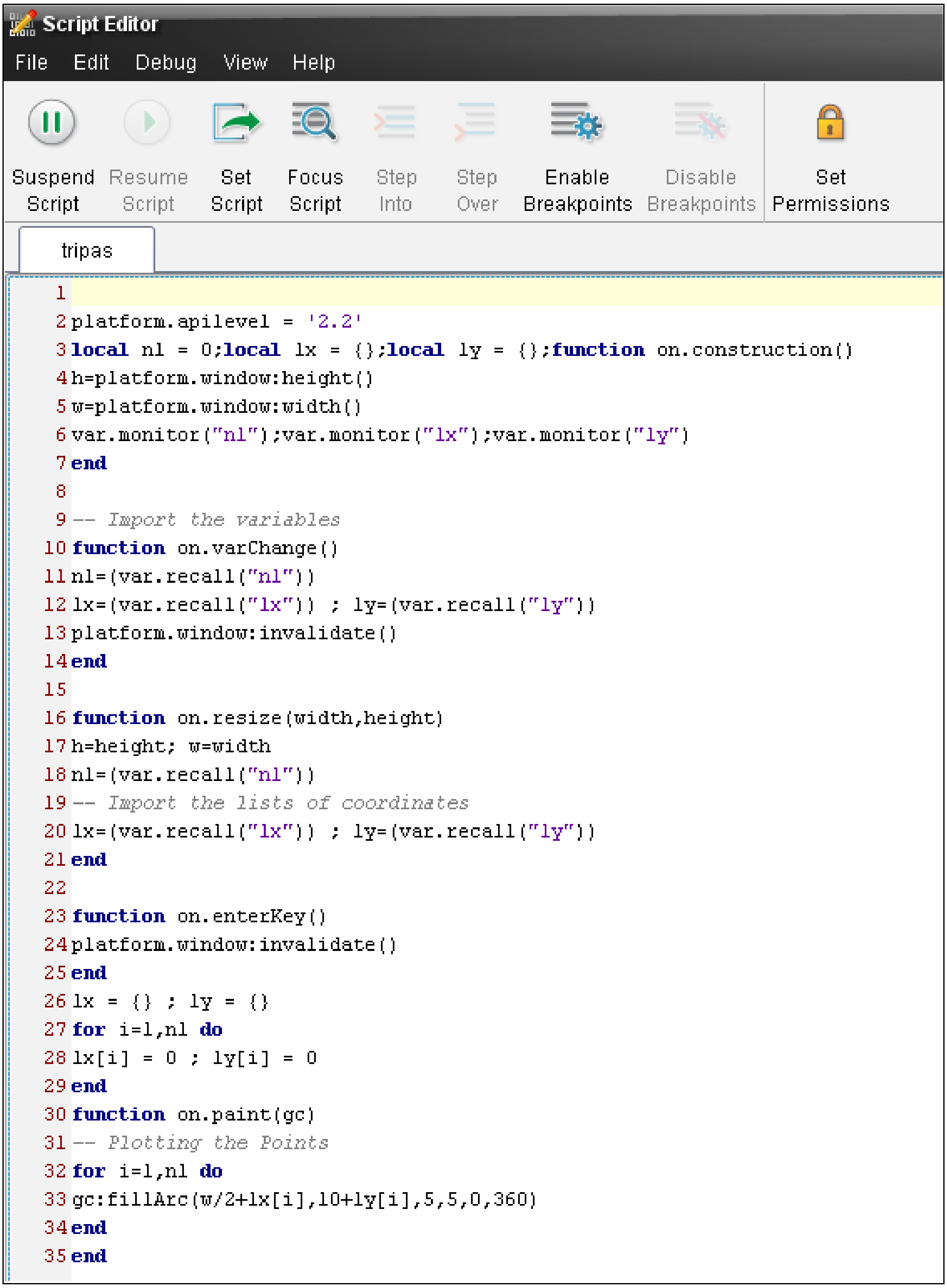
The LUA program allows more points to be plotted.





Many Thanks and Credits:

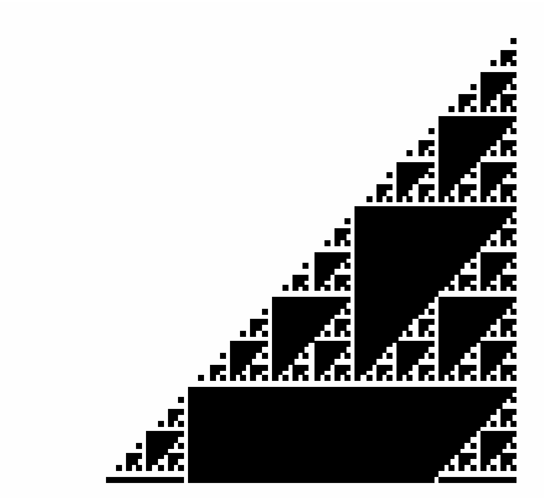
I am very indebted to Steve Arnold who informed about a couple of improvements in my LUA-program due to the recent apilevel 2.2. Once more I am recommending visiting Steve's website. You will find numerous Nspire applications and a very extended tutorial for LUA programs.



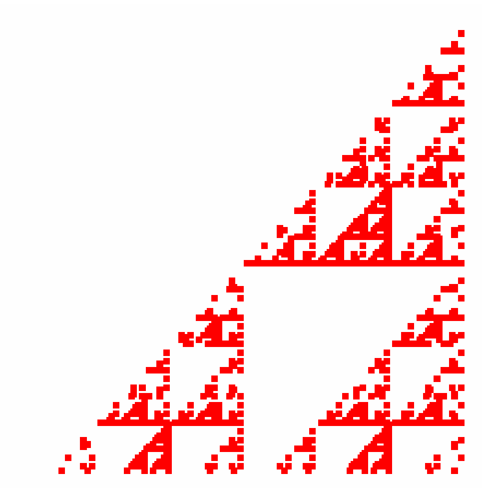
http://compasstech.com.au/TNS_Authoring/Scripting/index.html

Plots created with pascal_new.dfw:

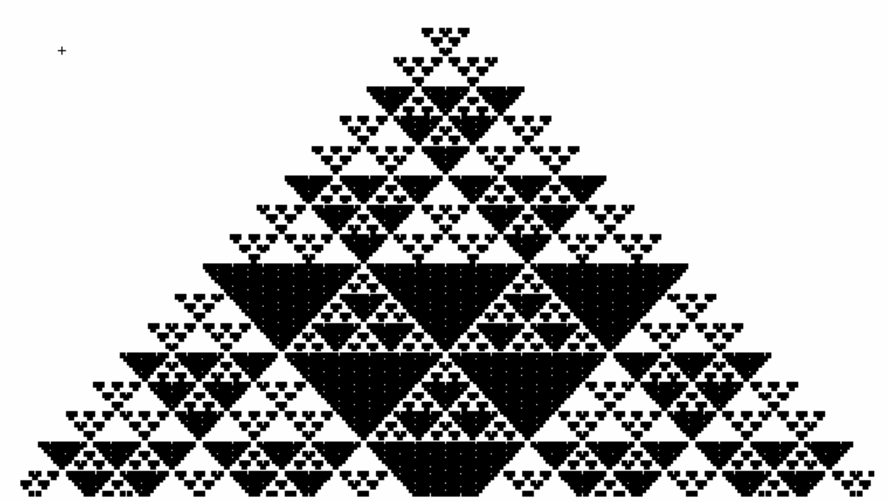
pasptv1(100, 2, 0)



pasptv1(150, 6, 2)



pasptv2(150, 9, 0)



pasinvptv2(150, 9, 0)