THE DERIVE - NEWSLETTER #33

THE BULLETIN OF THE

DERIVE

USER GROUP

+ TI 92

Contents:

revised version 2014 **March 1999**

Call for submission - Call for participation

ACDCA
Austrian Center for Didactics of Computer Algebra

Summer Academy

Recent Research on DERIVE/TI-92-Supported Mathematics Education

August 25-28, 1999, Gösing, Lower Austria

This conference is the fifth in a series of conferences on DERIVE/TI-92-supported mathematics education initiated by ACDCA: Krems (Austria) 1992, Krems 1993, Honolulu (USA) 1995, Sarö (Sweden) 1997.

Typical topics within the scope of the conference are

- Reports about successful classroom experiments
- New ways of mathematics teaching and learning
- Improvements in the design of mathematics software systems based on mathematical experience
- Examples of successful, new computer-based mathematics texts, lectures, training units, etc.
- Influence on assessment
- Similar studies with other computer algebra systems

How to submit / Time table:
Please send title and abstract (15-20 lines) to Bernhard Kutzler.
E-mail address: b.kutzler@eunet.at
Postal address: Kariglstrasse 5/6, A-4060 Leonding, Austria

- May 15, 1999: Deadline for submissions
- May 31, 1999: Notification of acceptance and publication of a detailed program
- June 25, 1999: Deadline for early bird registration

Local and travel information:
The early bird conference fee is ATS 700 ($ 55) and applies to registrations received by June 25. After that the conference fee is ATS 1000 ($ 80). The expected price for a single room with full board included is ATS 990 ($ 78,-) per day.

Gathering point is St. Pölten (capital of Lower Austria), from where we will take an ancient nostalgic train to the hotel in Gösing. (The train will depart from St. Pölten August 25 at 3:25pm.)

St. Pölten can be reached as follows:
- Plane: Fly to Vienna, then take a train to St. Pölten.
- Train: Take a train to St. Pölten from most major cities in the neigbouring countries.
- Car: Take the A1 motorway towards Vienna, leave it at the St. Pölten exit.

More detailed travel information will be sent out with the detailed program end of June.

The Alpenkurhotel Gösing, situated at 900 meters sea level 60 km far from St. Pölten, is a well hidden residence admidst the charming surroundings of the National Park "Ötscher Tormäuer", eye to eye with the huge, impressive Ötscher Mountain. Nothing but 15,000 acres of forest around you. The hotel offers indoor and outdoor swimming pool, outdoor tennis court, sauna and steam bath, solarium and fitness studio. In cooperation with the hotel management we will offer a program for accompanying guests / family members.

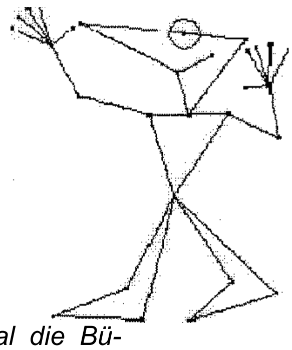
Conference Co-chairmen:

Josef Böhm	nojo.boehm@pgv.at
Helmut Heugl	hheugl@netway.at
Bernhard Kutzler	b.kutzler@eunet at

Liebe DERIVE- und TI-Freunde,

DNL#33 leitet das nun schon 9. DUG-Jahr ein, zu dem ich Sie alle recht herzlich begrüße. An meiner Stelle lacht Ihnen dieses Mal Filou entgegen. Sie werden ihm im nächsten DNL mit einem Gespräch mit seinem Schöpfer JJ Dahan wieder und ausführlich begegnen.

Ich möchte Sie gleich auf die Ankündigung der ACDCA Sommerakademie 1999 auf der gegenüberliegenden Seite hinweisen und Sie herzlichst dazu einladen. Die vorangekündigte Konferenz in Hagenberg, OÖ, musste leider abgesagt werden Aber wir sind sicher, mit der Sommerakademie mehr als nur einen Ersatz anbieten zu können. Wegen dieser Ankündigung muss dieses Mal die Bücherschau und auch die Sammlung der websites entfallen.



Bon jour, je suis Filou.

Neben den ewig jungen Themen DGL und Fraktale möchte ich Sie besonders auf den Artikel von Richard Schorn zur Gruppentheorie hinweisen. Er bietet uns eine nicht alltägliche, aber umso attraktivere Anwendungsmöglichkeit eines CAS. Die DERIVE-Implementation kann leicht auf den TI übertragen werden.

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.

Zum Schluss noch eine sehr wichtige Serviceleistung von uns gemeinsam mit Soft Warehouse Europe: Sie können von nun an alle DERIVE- und TI-files des aktuellen DNLs über eine neue Internetquelle abrufen. Klaus Jürgen Kutzler - die nächste Generation - hat auf der Homepage von SWHE eine DUG-Sektion eingerichtet, auf der Sie alle files finden:

www.derive-europe.com/english/dugengl.htm[*]

Damit müssen im DNL nicht mehr alle Listings abgedruckt werden. Falls Sie keinen Internetzugang haben aber gerne etwas haben wollen, dann rufen Sie bitte einfach an oder schreiben Sie. Ich schicke Ihnen eine Diskette, wenn Sie nicht auf die Jahresdiskette warten wollen.

Herzlichen Dank Klaus Jürgen für die angebotene Hilfe und .... Servus.

Damit wünsche ich Ihnen viel Spaß mit diesem DNL und im 9. DUG Jahr.

Bis zum nächsten Mal - und denken Sie bitte an Gösing!!

Josef Böhm, Editor

Dear *DERIVE*- and TI-friends,

DNL#33 opens a new DUG year. Yes, it is number 9. Welcome to you all. This time Filou is smiling to you on my place. In the next DNL you will meet him again, when he has a interesting TI-92 discussion with his creator Jean-Jacques Dahan.

First of all I' d like to point at the announcement of the ACDCA Summer Academy 1999 on the opposite page and simultaneously invite you to come to Lower Austria. Unfortunately the Hagenberg Conference had to be cancelled but we are sure to offer with the Summer Academy in one of the most attractive regions of our federal state more than only a substitute. Because of this announcement I had to omit both Book Shelf and web sites as well.

Besides the always young topics ODEs and fractals I want to point at Richard .Schorn's group theory article. This is a field where - at least in the DNL - we do not find so often CAS. The DERIVE implementation can easily be transferred onto a TI.

I am very grateful that Johann Wiesenbauer "composed" his TITBITS from a long e-mail discussion concerning an effective algorithm for calculating Stirling numbers. He promised to inform us in one of the next DNLs about the use of these numbers.

Finally I am very happy to announce an important service together with Soft Warehouse Europe. You can download all files (DERIVE and TI) mentioned in the actual DNL from SWHE's web site. K J Kutzler - the next generation - has established a DUG section on SWHE's home page where you can find all files for downloading:

www.derive-europe.com/english/dugengl.htm[*]

It is no longer necessary for me to print each listing in the DNL - and for you to type it in. If you don't have any access to the Internet then feel free to call or to write and 1'11 send a diskette, if you don't want to wait for the Diskette of the Year.

Once more, many thanks Klaus Jürgen for your support and cooperation.

I wish much pleasure with this DNL and for the whole DUG year number nine.

Hope to meet you in Gösing

Josef Böhm
Editor

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

As many of the *DERIVE* Users are also using the *TI-92/89* (and TI-NspireCAS) the *DNL* tries to combine the applications of these modern technologies.

Editor: Mag. Josef Böhm
A-3042 Würmla
D´Lust 1
Austria
Phone/FAX: 43-(0)660 31 36 365
e-mail: nojo.boehm@pgv.at

Contributions:
Please send all contributions to the Editor. Non-English speakers are encouraged to write their contributions in English to reinforce the international touch of the *DNL*. It must be said, though, that non-English articles will be warmly welcomed nonetheless. Your contributions will be edited but not assessed. By submitting articles the author gives his consent for reprinting it in the *DNL*. The more contributions you will send, the more lively and richer in contents the *DERIVE & TI-92 Newsletter* will be.

Next issue: June 1999
Deadline 15 May 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
Quaternion Algebra, Sirota, RUS
Various Training Programs for the TI
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
Share Holder´s Considerations using a CAS, Böhm, AUT
Kaprekar´s "Self numbers", Finite Groups, Schorn, GER
Linear Programming - Graphic Solution on the *TI*, Kirmse, GER
Implicit Multivalue Bivariate Function 3D Plots, Biryukov, RUS
Physical Problems, Magiera, POL
LU-Decomposition, Morales, COL
An Investigation on Calculus with the TO-92, Meagher, AUT

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

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

Jean-Jacques Dahan, Toulouse, FRA

Dear Friend,

I supposed you are interested by publishing into DERIVE letter the article I've promised to you at the end of the last year. I send the translated (in English) version in order that a great number of readers can read it. I am working on a Spanish version.

DNL: Many thanks for your charming, enchanting and inspiring contribution about your talks with your little friend *Filou*. As you can see he took my place in the Letter of the Editor and he said that he wouldn't leave that position until he hasn't seen his conversation with his master published.

Julio Cesar Morales, Universidad Nacional des Colombia, COL

Dear Mr. Boehm,

I send you the LU_ACTR.MTH file along with the corresponding .DOC and .DMO-files to see if they qualify for inclusion in your newsletter. LU_ACTR.MTH includes the functions U_P(A) and L_P(A) which (unlike the function that once appeared on the DERIVE web site) give the LU factorization (LU decomposition) of matrix A even if row interchanges are necessary.

DNL: *Dear Mr. Morales,*

As Editor I am feeling very happy and proud to have - after a long time of absence - an active South American member in the DUG and your article will be printed in our bulletin in the next future. Muchas gracias.

Günter Scheu, Pforzheim, GER

Dein Artikel über die "Logos, ... " hat mir ausgezeichnet gefallen und in Deinem Artikel "Das Pascal'sche..... " hat mich sehr gefreut und sogar überrascht was der TI-92 doch alles kann. Ich habe über 160 Seiten für mein Buch geschrieben und hoffe, dass es in diesem Halbjahr noch beim Dümmler-Verlag erscheint.

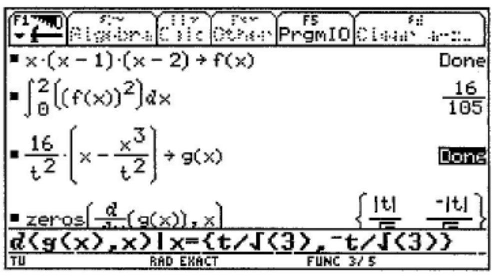
Der Unterricht mit dem TI-92 macht mir und meinen Schülern enorm Freude. Fast in jeder Stunde lerne ich noch was Neues.

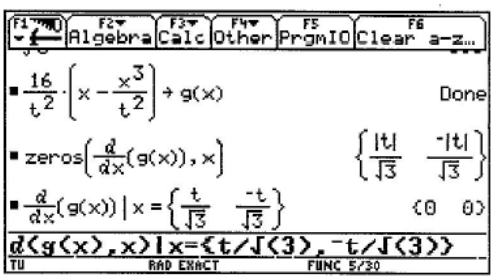
Now in English: I have two questions concerning the TI-92 (Standard Model):

☹ Why is the TI-92 unable to calculate the integral $\int_0^2 (x(x-1)(x-2)^2\,dx$?

☹ There are also problems in calculating the 1st derivative of $f(x)=\frac{16}{t^2}\left(x-\frac{x^3}{t^2}\right)dx$ for $x = t$.

Calculate the zeros of the first derivative and then substitute them into f without the ABS. Why does the TI present the solutions as ABS-expressions?





DNL: Dear Günter, many thanks for you friendly words about the last DNL. I really appreciate the comments of experienced teachers and "advocates of the GAS" - to quote Carl Leinbach. To your questions: As you can see the integral works, but it needs an incredible amount of time to find the integral value. Changing the upper bound to 1.9 or 2.1 accelerates the process?

I sent both requests to Michelle Miller - my very reliable link to the TI-experts - and via her I received a very short answer from the experts:

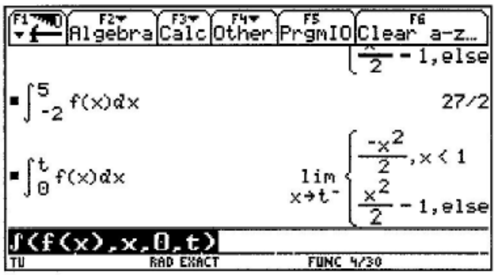
1. and 2. are TI-92 bugs that were fixed on the TI-92+.

There was a third answer from Michelle, because I added one more problem. We – Wolfgang Pröpper and I came across this bug (???) when we worked on our book about integration methods. In one section we dealt with the representation of integral functions. This is a problem on the PLUS, which doesn't occur on the Standard Model. Compare two TI-92 screen shots given below:

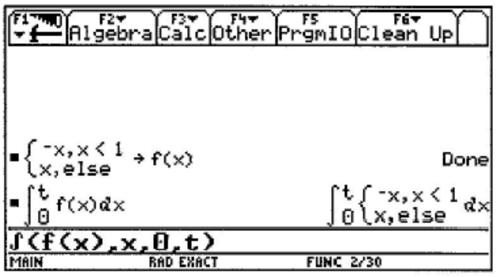
Let's take a piecewise defined function, e.g. $f(x)=\begin{cases}-x & for \ \ x<1\\ x & for \ \ x\geq 1\end{cases}$ ⇔ when(x<1,-x,x) → f(x)

☹ All integrals work – with one exception: a general upper bound:

Standard TI-92



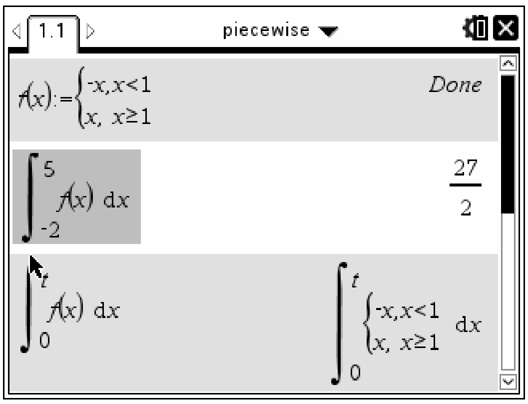
TI-92 PLUS

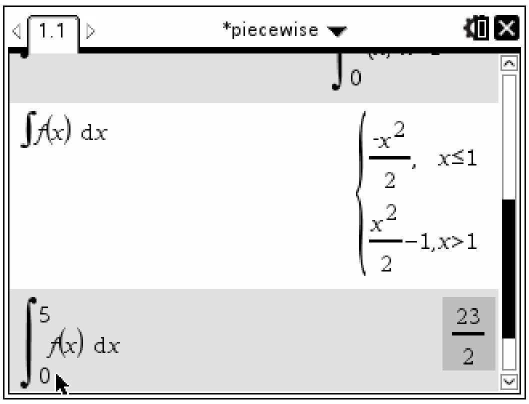


The TI's answer: 3. occurs on the 89/92+ because of a change that was made in symbolic integration due to a problem report.

My conclusion: This is another example for the fact that there is no advantage without any accompanying disadvantage.

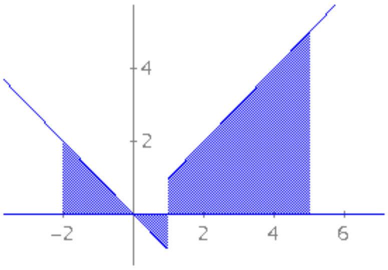
See how TI-NspireCAS performs:





It is interesting to see how DERIVE6.10 – and other CAS – are coping with this integral:

```
        f(x) :=
          If x < 1
#1:          -x
             x

#2:     AreaBetweenCurves(0, f(x), x, -2, 5)
```



$$\#3: \quad \int_{-2}^{5} f(x)\,dx = \frac{29}{2}$$

$$\#4: \quad \int_{0}^{t} f(x)\,dx = \mathrm{IF}\left(t < 1, -\frac{t^2}{2}, \frac{t^2}{2}\right)$$

Using the IF-construct gives a wrong result. *DERIVE* simply substitutes the boarders into the both functions!

$$\#5: \quad g(x) := -x\cdot\chi(-\infty, x, 1) + x\cdot\chi(1, x, \infty)$$

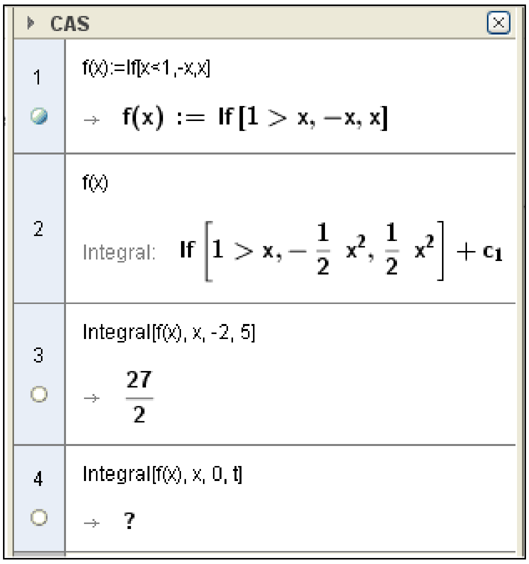
$$\#6: \quad \int_{-2}^{5} g(x)\,dx = \frac{27}{2}$$

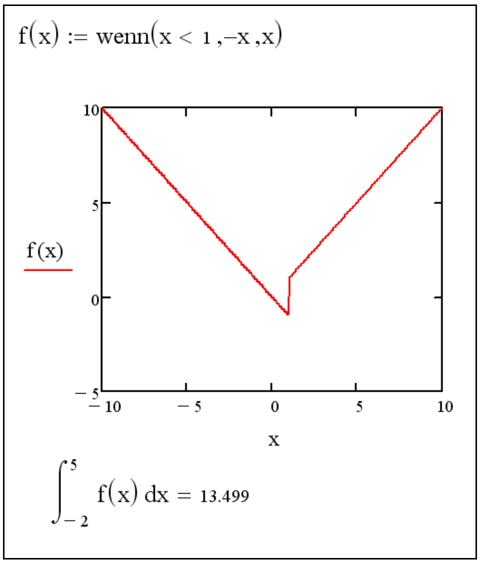
$$\#7: \quad \int_{0}^{t} g(x)\,dx = \left(\frac{t^2}{2} - \frac{1}{2}\right)\cdot \mathrm{SIGN}(t - 1) - \frac{1}{2}$$

Taking the CHI-function works correct. See below the integral with two symbolic boarders. The TIs refuse to give any result.

$$\int_{a}^{b} g(x)\,dx = \left(\frac{1}{2} - \frac{a^2}{2}\right)\cdot \mathrm{SIGN}(a - 1) + \left(\frac{b^2}{2} - \frac{1}{2}\right)\cdot \mathrm{SIGN}(b - 1)$$

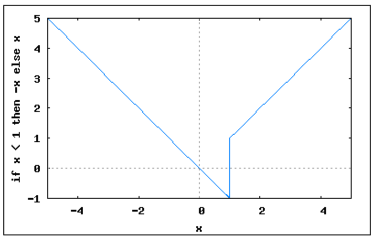
I recommend reading the very interesting article contributed by our Canadian friends Geneviève, Michel and Henri *Integration of Piecewise Continuous Functions* in DNL#91.





wxMaxima & CASIO ClassPadII

```
(%i1) f(x):=if x < 1 then -x else x;
(%o1) f(x):=if x<1 then -x else x

(%i3) wxplot2d([f(x)], [x,-5,5])$
(%t3)
```




```
(%i4) integrate(f(x), x, -2, 5);
```

(%o4) $\int_{-2}^{5} \text{if } x<1 \text{ then } -x \text{ else } x \, dx$

define $f(x)=\begin{cases}-x, & x<1\\ x, & x\geq 1\end{cases}$

done

$\int_{-2}^{5} f(x)dx$

13.5

$\int_{\square}^{\square} f(x)dx$

$\int \begin{cases}-x, & x<1\\ x, & x\geq 1\end{cases} dx$

Mathematica

In[14]:= `f[x_] := Piecewise[{{-x, x < 1}, {x, x ≥ 1}}]`

In[16]:= `f[x]`

Out[16]= $\begin{cases}-x & x<1\\ x & x\geq 1\end{cases}$

In[17]:= `Integrate[f[x], {x, -2, 5}]`

Out[17]= $\frac{27}{2}$

In[18]:= `Integrate[f[x], x]`

Out[18]= $\begin{cases}-\frac{x^2}{2} & x\leq 1\\ \frac{x^2}{2}-1 & \text{True}\end{cases}$

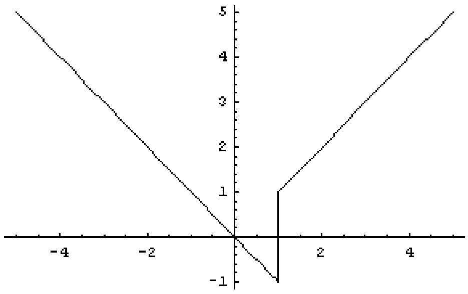
In[19]:= `Integrate[f[x], {x, 0, t}]`

Out[19]= $\int_0^t \begin{cases}-x & x<1\\ x & x\geq 1\end{cases} dx$

In[20]:= `Integrate[f[x], {x, y, b}]`

Out[20]= $\int_y^b \begin{cases}-x & x<1\\ x & x\geq 1\end{cases} dx$

In[21]:= `Plot[f[x], {x, -5, 5}]`



Peer van de Sanden, NED

Hello, I want to compute Stirling numbers of the first and second kind:
The following recursive functions work but are slow ...

DNL: This email (you will find it on page 23) was the beginning of a very interesting discussion on generating Stirling Numbers. Johann Wiesenbauer dedicated his TITBITS column to this topic (page 23).

Rüdeger Baumann, Celle, GER

I send a little "Knobelei" (= Brain Teaser) – puzzle for the DNL-readers which shows *DERIVE*'s capabilities to work with lists:

What is the rule for the sequence: **1, 11, 21, 1112, 3112, 122213, 312213, 212223,** ...?

Rüdeger's original file from 1999 does not work with DERIVE 6.10. I had to replace his oroginal definition of H(v,k) (see below) by a SELECT-construct which is expression #1 in Rüdeger's hint file. Josef

$$H(v, k) := \sum_{i=1}^{\text{DIMENSION}(v)} \text{IF}(v_i = k, k)$$

Rüdeger's Hint:

```
#1:   H(v, k) :≔ DIM(SELECT(a = k, a, v))
#2:   COUNTS(v) :≔ APPEND(VECTOR(IF(H(v, k) > 0, [H(v, k), k], []), k, 1, 9))
#3:   COUNTS([1, 1, 2, 3, 3, 4, 1, 2, 4, 3, 3])
#4:   [3, 1, 2, 2, 4, 3, 2, 4]
#5:   COUNTSEQ(v0, n) :≔ ITERATES(COUNTS(v), v, v0, n)
                              DIMENSION(v) - i
#6:   POT(v) :≔ VECTOR(10                 , i, 1, DIMENSION(v))
#7:   ZF(v0, n) :≔ VECTOR(w·POT(w), w, COUNTSEQ(v0, n))
#8:   Now simplify:
#9:   ZF([1], 10)
```

I believe that it would not be so easy to program this sequence e.g. in PASCAL. On the other hand the Hofstadter Sequence

1, 3, 7, 12, 18, 26, 35, 45, 56, 69, ...

can easily be produced by a PASCAL program. Who of the *DERIVE* family can help with a *DERIVE* program?

DNL: Dear Rüdeger, I can imagine that your request is a challenge for some of the DNL-readers. (Among others I count on Johann Wiesenbauer.) By the way, I know your Teaser-Sequence in a slightly different variation:

1, 11, 21, 1211, 111221, 312211,13112221,1113213211, ...

In the very recommendable *CRC Concise Encyclopedia of Mathematics* (*Eric W. Weisstein*) I found a nice and for me unknown name for this sequence: "Look and Say Sequence".

The Hofstadter Figure-Figure Sequence is defined as follows (also from CRC):

Define $F(1) = 1$ and $S(1) = 2$ and write:

$$F(n) = F(n-1) + S(n-1),$$

where the sequence $\{S(n)\}$ consists of those integers not already contained in $\{F(n)\}$. For example, $F(2) = F(1) + S(1) = 3$, so the next term of $S(n) = S(2) = 4$, giving $F(3) = F(2) + S(2) = 7$, so $S(3) = 5$ and $F(4) = F(3) + S(3) = 12$. Continuing in this manner gives the "figure" sequence as shown in Rüdeger's problem.

(*Hofstadter, Gödel, Escher, Bach, p. 73* (*Seite 79 deutsche Ausgabe*)

Sebastiano Cappuccio, Forli, ITA

Dear Derivers,

`VECTOR(IF(x<0,2x+k),k,-5,5)` works well with *DERIVE 2.08* and I can simplify and plot it, getting a sheaf of parallel half-lines.

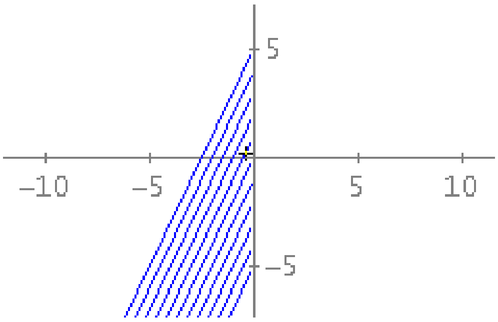
Why does it neither work with *DERIVE 3.01* nor with *DfW 4.11*? How can I get the same sheaf with them?

Thank you, Sebastiano

Peer van de Sanden, NED and Carlos Fleitas, Comenar Viejo, ESP

Hello, I don't know why it does not work in DfW 4.11 but I think that it has something to do with nesting formulas. This will work:

```
F(x,k):=IF(x<0,2x+k)
VECTOR(F(x,k),k,-5(5)
```



John Alexiou, USA

The problem seems to be that simplification does not occur for the 2nd or 3rd arguments of the IF( ) function for example , an expression

```
#1: IF(u>d,u-d,u+d)
```

when applied to

```
#2: LIM(#1,d,t)
```

you would expect the result `IF (u>t,u-t,u+t )` but the simplified result is

```
#3: IF(u>t,LIM(u-d,d,t),LIM(u+d,d,t))
```

no matter how many times you simplify it does not simplify the 2nd and 3rd argument of IF( ).
I always use the STEP( ) or CHI( ) functions which work better, unless you call STEP(0) which returns 1/2 +/– 1/2 which cannot be evaluated.

The problem exists for: `VECTOR(IF()), LIM(IF()), and ITERATE(IF())` and possibly more ...

Soft Warehouse, Inc. (Albert Rich)

When DERIVE simplifies an IF statement, it simplifies the "conditional" clause (i.e. the first argument) of the IF statement. If its truth-value can be determined, the "then" clause (i.e. the second argument) OR the "else" clause (i.e. the third argument) of the IF statement is simplified, as appropriate. Note that the other clause is NOT simplified. Therefore, if the truth-value of the "conditional" clause cannot be determined, neither the "then" clause nor the "else" clause is, or should be, simplified.

For example, if x is unknown and IF(x<0, 1+1,2+2) is simplified, DERIVE does not simplify 1+1 or 2+2, since the truth-value of x<0 is unknown.

However, the arguments of user-defined functions, as distinct from the clauses of IF statements, ARE always evaluated before calling the function. Therefore, you can define replacement functions for the IF statement that do evaluate the "then" and "else" clauses as follows:

```
IF_THEN(condition,then):=IF(condition,then)
IF_THEN_ELSE(condition,then,else):=IF(condition,then,else)
```

Then, to generate the vector of IF statements that you want, you can simplify the command

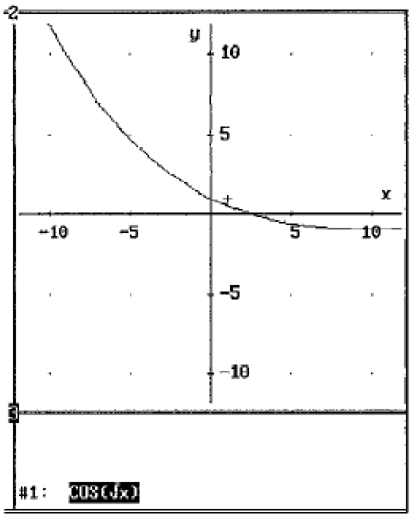
```
VECTOR(IF_THEN(x<0,2x+k) ,k,-5,5)
```

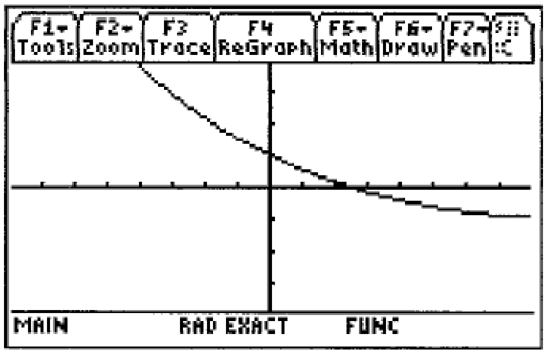
Hope this helps clarify the situation.
Aloha, Albert D. Rich

There was another discussion concerning "Isolated points", raised by an experience made by one of James Eckerman's students.

James Eckerman, Sacramento, California

A student of mine asked why her TI-89 gave her a graph that has points in quadrant two for the function `cos(sqrt(x))` when she had the mode set on "REAL". I got the same picture with DERIVE. I'm sure some- one out there knows the correct explanation. Please help us out! Thanks.

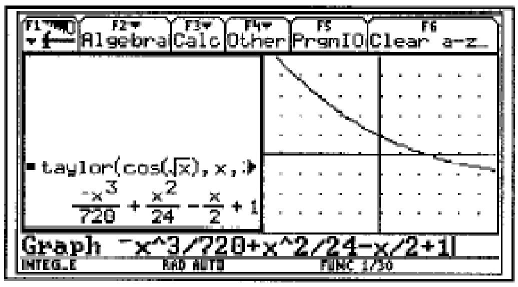




Harald Lang, SWE

Look at the Taylor expansion of cos(*x*):

$$\text{TAYLOR}(\cos(x), x, 0, 6) = -\frac{x^6}{720} + \frac{x^4}{24} - \frac{x^2}{2} + 1$$



If we substitute sqrt(*x*) for *x*, then we get:

$$\text{TAYLOR}(\cos(\sqrt{x}), x, 0, 6) = \frac{x^6}{479001600} - \frac{x^5}{3628800} + \frac{x^4}{40320} - \frac{x^3}{720} + \frac{x^2}{24} - \frac{x}{2} + 1$$

where the right hand side makes perfect sense also for negative values for *x*. I suppose what you see is the right hand side above.

Adri van der Meer, NED

Because $\cos(i\,x) = \cos(-i\,x) = \dfrac{e^x + e^{-x}}{2}$, and this is real if *x* is real.

If you choose the "real" mode in *DERIVE*, you don't prevent it from using complex calculus for "intermediate results". You only force *DERIVE* to choose the real branch (if possible) in the case of multivalued functions.

Peer van de Sanden, NED

Hi Jim, here is a simple function that also gives a plot for negative *x*-values:

`F(x) := RE(x + SQRT(x))`

In this case it is expected. I am asking for the real part of a complex number. The output is real although the part in the brackets `(x + SQRT(x))` is complex.

Your function `COS(SQRT(x))` does the same thing.

The real mode works on the input and on some simplifying algorithms. It doesn't affect the inbetween results. In your case:

$$x\text{(real)} \xrightarrow{\text{SQRT()}} z\text{(complex)} \xrightarrow{\text{COS()}} y\text{(real)}$$

Look at the following *DERIVE* simplification for the complex numbers $a + b\,i$ and $b\,i$ and compare the results:

$$[\mathrm{COS}(a + b\cdot i),\ \mathrm{COS}(b\cdot i)] = \left[\frac{e^{-b + i\cdot a}}{2} + \frac{e^{b - i\cdot a}}{2},\ \frac{e^{b}}{2} + \frac{e^{-b}}{2}\right]$$

The second expression has no imaginary part in its simplification! The output is real.

Therefore (for $x < 0$): $\cos(\sqrt{x}) = \dfrac{e^{\sqrt{-x}}}{2} + \dfrac{e^{-\sqrt{-x}}}{2}$.

The following function gives the expected plot: `cos(if(x≥0,√(x))`.

Johannes Wiesenbauer, Vienna, AUT

James examp1e reminded me of another one which is still a mystery to me:

Why doesn’t DfW 4.11 show any points in the third quadrant when plotting the curve SQRT(*x*)^2? Anyone out there who has a clue? (It goes without saying that "dirty tricks" like simplifying this expression are strictly forbidden!).

Cheers, Johann

J. H. Frisbee, Houston, USA

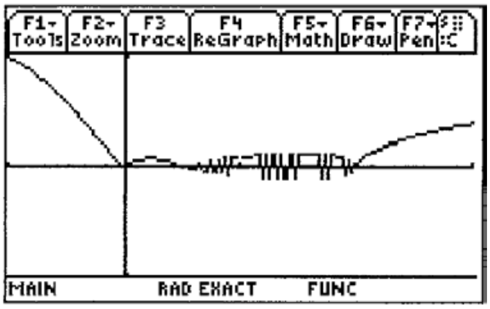
The SQRT(*x*)^2 problem may have to do with the way roots and powers are computed. First, it is interesting to note that if one uses the product of SQRT(*x*) ∗ SQRT(*x*) the curve is complete. Now suppose SQRT(x) is replaced by EXP(LN(*x*)/2). The more original form EXP(LN(*x*)/2)^2 exhibits the incomplete character and, unlike the above product form EXP(LN(*x*)/2) ∗ EXP(LN(*x*)/2) is also incomplete.

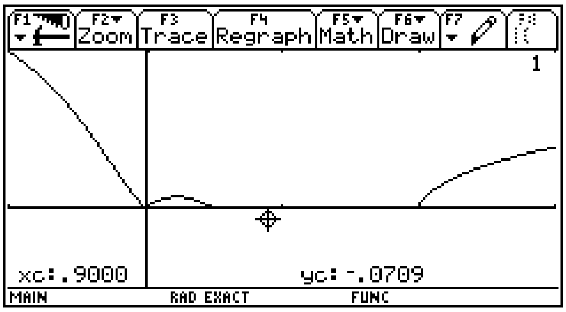
I have no idea how *DERIVE* actually handles the LN and EXP functions but I suspect that they-are probably used for powers (and roots) and that is the more fundamental cause of the incomplete plot.

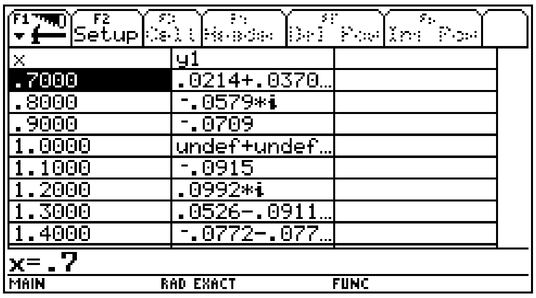
James Eckerman, Sacramento, California

When I read James Eckerman's email, I remembered a strange function, which I obtained last fall. I tested my just bought TI-89 trying to calculate the locus of the maximum curvature of the power function curves.

In the course of that calculation I plotted the function $f(x)=\left(\dfrac{x-2}{x^2(2x-1)}\right)^{\frac{1}{2(x-1)}}$ and was rather astonished to see isolated points within a gap. Further work with *DERIVE* led me to some interesting results as you can see with the enclosed MTH-file: LOCMAXC.MTH (which is among the files in MTH33.zip).

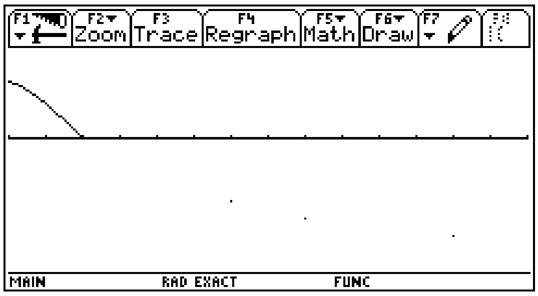






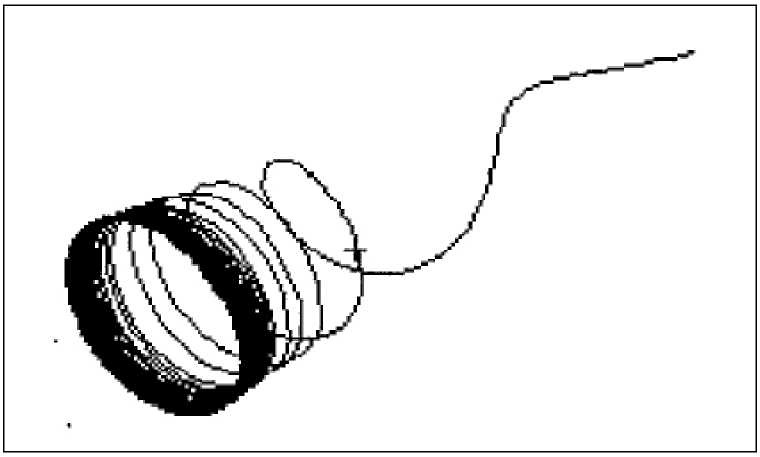
x	y1
.7000	.0214+.0370…
.8000	-.0579*i
.9000	-.0709
1.0000	undef+undef…
1.1000	-.0915
1.2000	.0992*i
1.3000	.0526-.0911…
1.4000	-.0772-.077…

x=.7



Compare the plots: TI-89 left above, Voyage 200 others. You can see the “isolated points”, Lorenz is writing about, Josef.

As I mentioned in the MTH-file-comments the reason for the occurrence of the isolated points is the negative sign of the basis combined with integer values of the exponent of the power term. The simplest example for this phenomenon is the (mostly complex valued) function F(x):=(–l)^x or "squeezed" in F(x):=(–l)^(1/x). Compare the well known F(x):= x^x, They all produce spirals in the complex space (x, RE(y), IM(y)).



This is the spiral in an isometric representation (using ISOMETRIC from GRAPHICS.MTH – DOS-Times, Josef).

Ray Girvan, UK

I haven’t looked through your derivation in detail; but I did try checking the plot of *f*(*x*) against *GrafEq*, which is normally very good at ‘difficult’ functions with singularities. *GrafEq* suggests only one singularity (1.5, -1/9) in the gap between $x = 0.5$ and $x = 2$.

Johann Wiesenbauer, Vienna, AUT

Hi all, the function

```
F(x):=((x-2)/(x^2(2x-1)))^(1/(2(x-1)))
```

introduced by Lorenz Jaeneke is a very remarkable one, indeed, though some things are far less mysterious when viewed from the right angle. (Hence, those who love mysteries are advised to skip the rest of his note!)

When drawing its parametric representation in the complex plane, namely `[RE(F(x),IM(F(x)]`(by the way pros will input this term in the form `[refx,imfx]` and leave it to *DERIVE* to decipher it!) for *x* in [0.5, 0.99] and [0.99, 2] (adjust a suitable range before plotting, e.g. [-0.12,0.12] both for *x* and *y*) you'll see at once what is going on here: If *x* approaches 1 then its image f(*x*) will move on one of two spirals that both have the same circle around the origin as limiting curve. Its radius is given by

```
LIM(ABS(F(x)),x,1)  =  ê^(-5/2)
```

Those "isolated points within a gap" are simply the countable many intersections of the two spirals with the real axis. In particular, they are certainly not singularities, as claimed by Ray Girvan (though I like his homepage very much!), because there is only one genuine singularity of f(*x*) when viewed in the complex plane, namely *x* = l. I hope you all will be enchanted by the picture of the two spirals as I was and there will be no troubles with older versions of Derive.

Needless to say that I'm always referring to the current version of DfW which is 4.11 at the moment. If you have an older version DfW 4.xx instead use the option of a free update on Derive's homepage! Johann

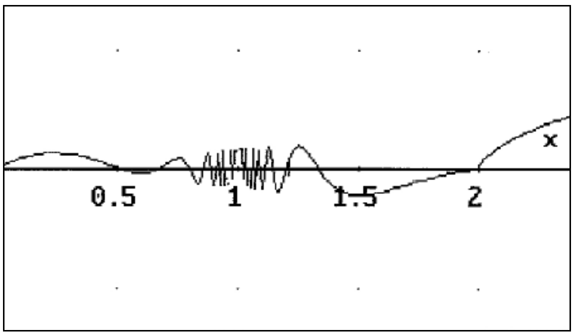
Cheers, Johann

Ray Girvan, UK

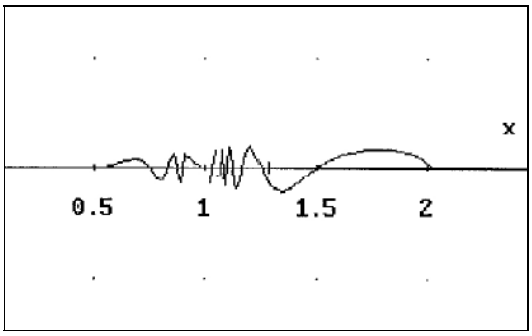
Very neat. Another useful view is to plot `RE(F(x))` and `IM(F(x))` separately, which shows F(x) as out-of-phase real and imaginary sinusoidal components, both increasing in frequency as they approach x = 1. The "isolated points" are where IM = 0 and RE has maxima and minima. (In this case, it looks as if *GrafEq* was on the wrong track).

One thing I notice: when I plot `RE(F(x))` I get a smooth plot. But if I app1y Simplify/Basic to RE(F(x)) and plot the result, the section of the curve between 0.5 and 2 becomes discontinuous. Any idea what's going on.

I am sorry! I know: singularities are places where the function blows up to infinity, not isolated solution points. I wasn't thinking. Thanks for the clarification.

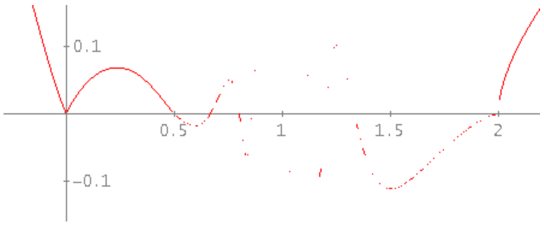


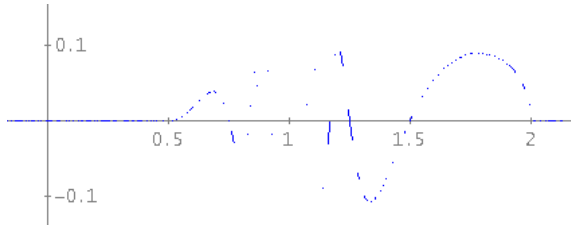
Plot of RE(F(x))



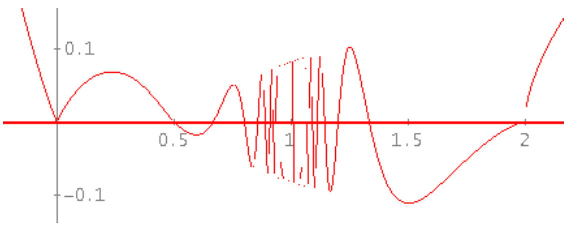
Plot of IM(F(x)), (Josef 1999)

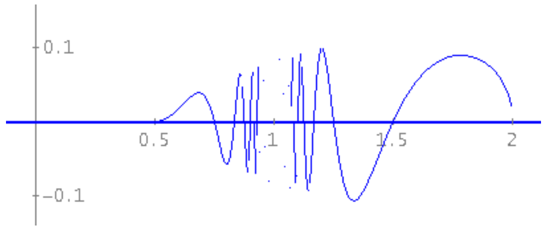
Plotting with DERIVE 6.10 only the real parts first:





Plotting real and imaginary parts now:





Soft Warehouse, Inc (Albert Rich)

Hello Ray Girvan

The result of simplifying RE(F(x)) produces an expression mathematically equivalent to RE(F(x)) but expanded out. When the expanded expression is evaluated, roundoff error produces a small imaginary component in the result that prevents it from being plotted.

The moral of the story: If you wrap RE around a complex expression in order to plot the real part, do not simplify it.

Ray Girvan, UK

It also looks quite good with an offset to show it from a different viewpoint:

`[x+RE(F(x)),IM(F(x)-x)]`, (setting the parameters to [-3,3] plots the function reasonably in one operation).

I like the way an apparently discontinuous function like F(x) makes far more sense in complex space: just a smooth curve that twirls into imaginary over parts of its range.

Johann Wiesenbauer, Vienna, AUT

Hi all,

I'm very sorry, but the second interval for *x* should be of course [1.01,2], otherwise you won't see the second spiral... As I just tried out it will also work, if you draw both spira1s at once by choosing the interval [0.5,2] for *x*, though I prefer the first option because it shows the limiting circle between the two spirals more distinctly.

By the way, isn't it strange that Derive draws this highly complicated curve without batting an eyelid, but has troubles with the simple function sqrt(*x*)^2? Well, actually Derive is on the horns of a di-lemma: In the latter case simplifying the expression before plotting it would fix the problem, but in the ease of our spirals simplifying the parametric expression above is not exactly advisable as Ray has noticed ...

On this occasion, let me point out that Lorenz did a great job, when computing the actual values of the isolated points, namely

$$1+\frac{1}{2m} \qquad (m \text{ a nonzero integer})$$

so we'll look over the enormous troubles he had when computing the two real limit points of f(x). Just one remark that might be helpful to throw light upon matters of this kind: When computing the limit of an expression like

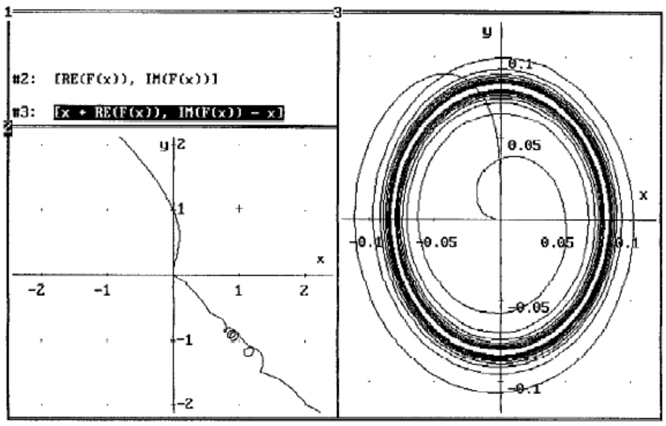
$$\lim_{m\to\infty^-}\left(\frac{8m^2(1-4m)}{\left((2m+1)(4m+1)^2\right)^{2m}}\right)$$

you have to bear in mind that m is viewed as a complex variable by Derive (no matter what adjustments had been made before!), so Derive was perfectly right in saying that this limit doesn't make sense. It took me some time to find out why Lorenz "succeeded" in finding a "limit" all the same: It's because ln x and ln(abs(x)) have the same derivative, so when applying L'Hospital's rule the outcome is the same as if you had taken the limit of the absolute value of the expression above which does exist! A nice fallacy, don't you think so?

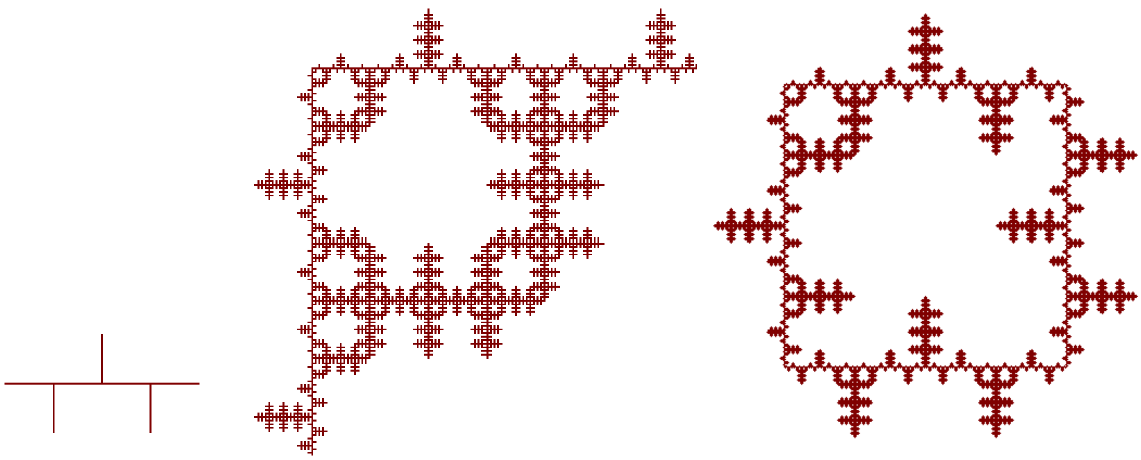
Cheers, Johann

You can see the enchanting picture of the two spirals – Johannes Wiesenbauer – together with Ray Girvan's curve "twisting into imaginary", Josef.

I cannot reproduce the plots with DERIVE 6.10, Josef.



Two more "generator – generated" "Koch like curves (page 50)



A Spigot – Algorithm for π and *e*

Peter Witthinrich, Lübeck, Germany

In 1995 Rabinowitz and Stanley [1] published a new algorithm for computing the decimal digits of the irrational numbers π and *e.* It uses no floating point routines but only integer arithmetic and produces the digits one at a time without using them after computing. So there was no need to store them in an array or string [5]. The authors called it 'spigot algorithm' (called "Tröpfelverfahren" in [3] - in an exercise and example for Delphi programming). (*For those of you who have never heard the word spigot – in German it is a "Zapfhahn", Prost! Josef*)

The authors explicitly refer to a method published by Sale [2] in 1968 - nearly thirty years earlier, for computing the digits of *e,* the basis of the natural logarithm. It is very short, and it also pumps the digits of *e* one at time, again without using them after computing. Therefore the use of arrays for storing the already computed digits, i.e. in [4], can be avoided.

The algorithm is based upon the well known series of *e*

$$e = 1+\frac{1}{1!}+\frac{1}{2!}+\frac{1}{3!}+\frac{1}{4!}+\frac{1}{5!}+\frac{1}{6!}+\ldots\ldots = 1+1\left(1+\frac{1}{2}\left(1+\frac{1}{3}\left(1+\frac{1}{4}\left(1+\frac{1}{5}\left(1+\frac{1}{6}\left(1+\ldots.\right)\right)\right)\right)\right)\right)$$

$$= 2 + \frac{1}{2}\left(1+\frac{1}{3}\left(1+\frac{1}{4}\left(1+\frac{1}{5}\left(1+\frac{1}{6}\left(1+\ldots.\right)\right)\right)\right)\right)$$

The sum right of 2 is <1. I shift the decimal point, i.e. I divide by 10 and simultaneously I multiply by 10 each of the single summands. I obtain the following decimal place from the integer part of the remainder within the parentheses. Again we find a fraction as remainder:

$$2+\frac{1}{10}\left(\frac{1}{2}\left(10+\frac{1}{3}\left(10+\frac{1}{4}\left(10+\frac{1}{5}\left(10+\frac{1}{6}\left(10+\ldots\right)\right)\right)\right)\right)\right)=$$

$$=2+\frac{1}{10}\left(7+\frac{1}{2}\left(0+\frac{1}{3}\left(1+\frac{1}{4}\left(0+\frac{1}{5}\left(1+\frac{1}{6}\left(5+\ldots\right)\right)\right)\right)\right)\right)$$

The digits of the intermediate results are calculated by integer division and stored. The last carry delivers the next decimal digit. Algorithmically you need a loop:

```
0→carry
  For j,s,2,-1
    coeff[j]*10+carry→temp
    intDiv(temp,j)→carry
    remain(temp,j)→coeff[j]
  EndFor
Output row,column,iPart(carry)
```

$$2.7+\frac{1}{100}\left(\frac{1}{2}\left(0+\frac{1}{3}\left(10+\frac{1}{4}\left(0+\frac{1}{5}\left(10+\frac{1}{6}\left(50+\ldots\right)\right)\right)\right)\right)\right)=$$

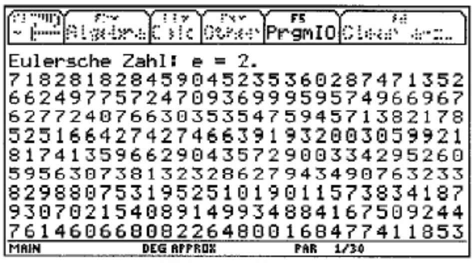
$$=2.7+\frac{1}{100}\left(1+\frac{1}{2}\left(1+\frac{1}{3}\left(1+\frac{1}{4}\left(3+\frac{1}{5}\left(4+\frac{1}{6}\left(2+\ldots\right)\right)\right)\right)\right)\right)$$

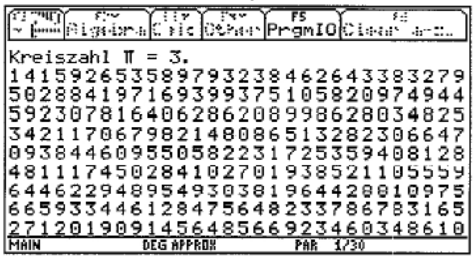
The algorithm in [1] was written down in Algol 60, a good structured language, and it is not very difficult to translate it into TI-92 programming language (filename: spigote.92p):

The output on the TI-92 display has been formatted to 270 decimal places, nine rows with 30 digits each. Computing time for the 270 decimal digits was about 35 minutes. The computed digits for *e* are correct compared to the result in [4]. Unfortunately the corresponding algorithm for π[1] is more complicated and longer than the algorithm for *e.* Both programs are rewritings of the programs given in the references in the TI92 language, they both don't include new own ideas.

A result of 270 decimal places here needs about seven hours computing time, the TI-92 is too slow:

```
spigote()
Prgm
Local digits,s,test,row,column,i,j,coeff,carry,temp
ClrIO
Request "Number of decimals: ",digits
expr(digits)→digits
4→s
(digits+1)*2.30258509→test
While s*(ln(s)-1)+.5*ln(6.2831852*s)≤test
s+1→s
EndWhile
int(s)→s
Output 0,0,"Euler's number: e = 2."
setMode("Exact/Approx","Exact")
10→row
0→column
seq(1,n,1,s,1)→coeff
For i,1,digits
  0→carry
  For j,s,2,-1
    coeff[j]*10+carry→temp
    intDiv(temp,j)→carry
    remain(temp,j)→coeff[j]
  EndFor
  Output row,column,iPart(carry)
  column+8→column
  If column>239 Then
    0→column
    row+10→row
  EndIf
EndFor
setMode("Exact/Approx","Approximate")
EndPrgm
```





spigotpi() is included in MTH33.zip.

References:

[1] *Rabinowitz, Wagon*: A Spigot Algorithm for the Digits of π. American Mathematical Monthly, Band 102, Heft 3, 1995, S. 195-203

[2] *Sale, A.H.J.*: The calculation of *e* to many significant digits. Computer Journal, Band 11, 1968, S. 229-230

[3] *Kaiser, R.*: Object Pascal mit Delphi. Springer, Berlin 1977, S.l73 (Loesung: S.650)

[4] *Engel, A.*: Mathematisches Experimentieren mit dem PC. Klett, Stuttgart 1991, S. 19

[5] *Fietkau, R. u. M.*: Die Berechnung der Kreiszahl π. Peeker, Hüthig-Verlag, Heft 5, Mai 1986, S. 42-51

Subject: Shading with *DERIVE*

Peer van de Sanden, Netherlands

Dear Derivers,

I started this new year (happy new year to you all) with creating an utility file that I needed often in the past but never had the time to make it. It's called `shade.mth`. This file contains four functions for shading an area between the two functions $f(x)$ and $g(x)$ on the interval $[a, b]$:

```
SHADE(a,b,space)
SHADE_UNDER(a,b,space)
SHADE_BACK(a,b,space)
SHADE_BACK_UNDER(a,b,space)
```

You must define the functions $f(x)$ and $g(x)$ before using the functions. The parameters a and b are the boundaries of the interval $[a,b]$ (with $a < b$). The parameter *space* is the space between two spacing lines. This parameter is optional. The standard value for *space* = 0.2. The vertical boundaries $x = a$ and $y = b$ will not be calculated and therefore will not be plotted.

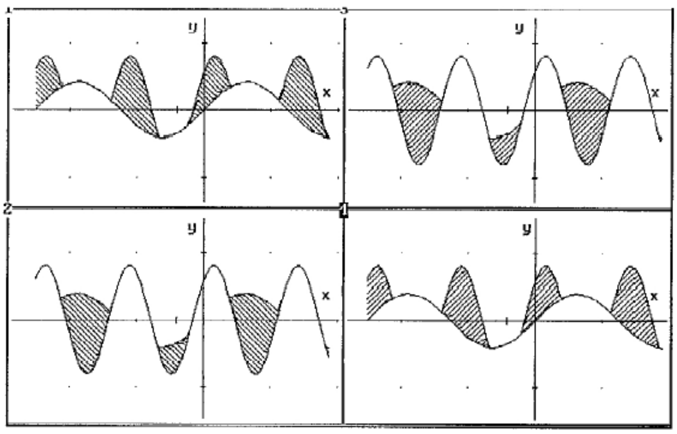
The `SHADE`-function draws lines in the area where $f(x) > g(x)$ on the interval $[a,b]$. The angle of the lines is 45° if the x-scale and the y-scale in the plot are equal.

The `SHADE_UNDER`-function draws lines in the area where $g(x) > f(x)$ on the interval $[a,b]$. The angle of the lines is 45° if the x-scale and the y-scale in the plot are equal.

The `SHADE _BACK`-function draws lines in the area where $f(x) > g(x)$ on the interval $[a,b]$. The angle of the lines is -45° if the x-scale and the y-scale in the plot are equal.

The `SHADE_BACK_UNDER`-function draws lines in the area where $g(x) > f(x)$ on the interval $[a,b]$. The angle of the lines is -45° if the x-scale and the y-scale in the plot are equal.

The shading works with various functions, The functions do not work on intervals on which either $f(x)$ or $g(x)$ has an asymptotic behaviour.



As I produced a shading tool some times ago (DNL#15) and G P Speck presented his "Concentric Curve Shading" in DNL#31, you may think that there is some demand on tools to beautify plots. (W.Pröpper's contribution in this DNL works in the same intention). I had some ideas to improve Peer's useful tool and contacted him how to add the boundaries, to combine both shadings and how to produce a SHADE_BETWEEN-function. After a very short while he wrote back:

I tried several times to develop an easy to use shading-tool. The functions you added make it possible to make life even more easier. Here are my suggestions:

- Include the boundaries in my original user functions. Then they can be deleted from your functions.
- Rename the functions in a more logical way
- Add one more which shades normally and backwards between the two functions *f*(*x*) and *g*(*x*)

Then we will have the following functions:

User functions (not frequently used):

```
SHA_A(a,b,space)
SHA_U(a,b,space)
SHA_AB(a,b,space)
SHA_UB(a,b,space)
SHA_ABO(a,b,space)
SHA_UBO(a,b,space)
```

User functions (frequently used):

```
SHADE(a,b,space)
SHADE_BACK(a,b,space)
SHADE_BOTH(a,b,space)
```

Most people will only need your functions. Thus these functions should have the most comfortable names. The `SH_A`- and `SH_U`-function look a little like the `SHADE`-option on the TI-92

Note: in the Options Menu on the plot screen you must select Points-connected and Points-small and perhaps no color cycling.

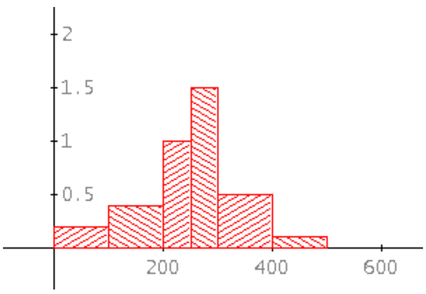
Note: I remember that I had problems with functions with asymptotic behaviour like *f*(*x*) = 1/*x*^2 with *g*(*x*) = 0. Derive will not be able to simplify function `SHADE(-1,1)`. With space = 0.2 the maximum will be at *x* = 0 and you will get a vector with an infinite amount of elements. In this case an other value for space will do e.g. `SHADE(-1,1,0.3)`.

I wanted to have shading functions for plotting histograms (Dutch word, I don't know if the English name will be the same). I have included an example in `histogram.mth`. The original shading-functions are adjusted for this application.

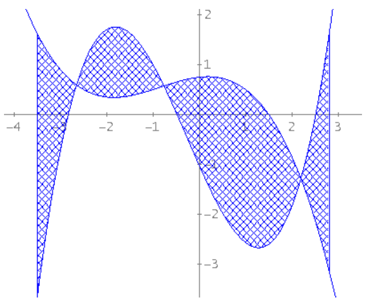
You are welcome to publish it in the International DERIVE & TI-92 Newsletter. You additions deserve that the original MTH-file will be adjusted, preferable in the way that I suggested.

But, would it not be nice if Derive had a shading option or an area color option in the plot screen like the TI-92 is providing? I would like to right click with the mouse on an area and choose my own background color or shading for that area.

Groeten van Peer van de Sanden




```
k1 := [0, 100, 200, 250, 300, 400, 500]
freq := [20, 40, 50, 75, 50, 10]
PLOT_HISTOGRAM(k1, freq)
```

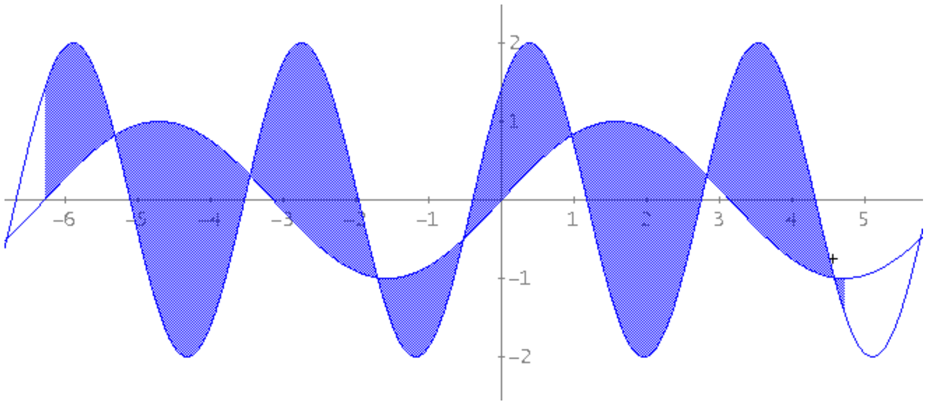


$F(x) := 0.3\cdot(x + 2.5)\cdot(x + 1)\cdot(x - 2.7) + 1$

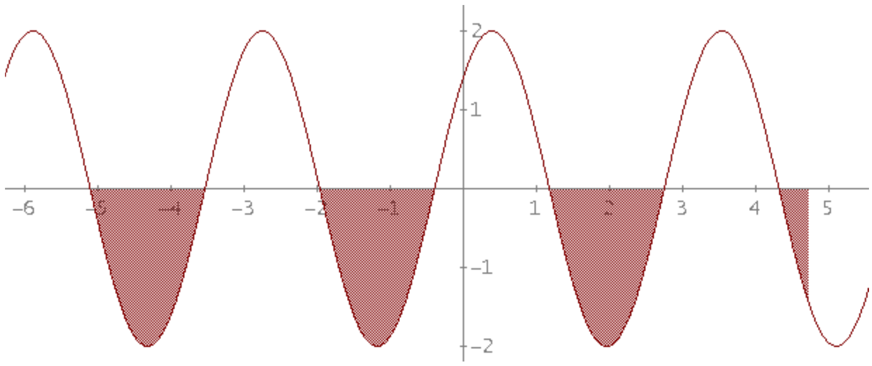
$G(x) := 0.05\cdot(3 - 2\cdot x)\cdot(x^2 + 4\cdot x + 5)$

```
SHADE_BOTH(-3.5, 2.8, 0.15)
```

See some two examples:

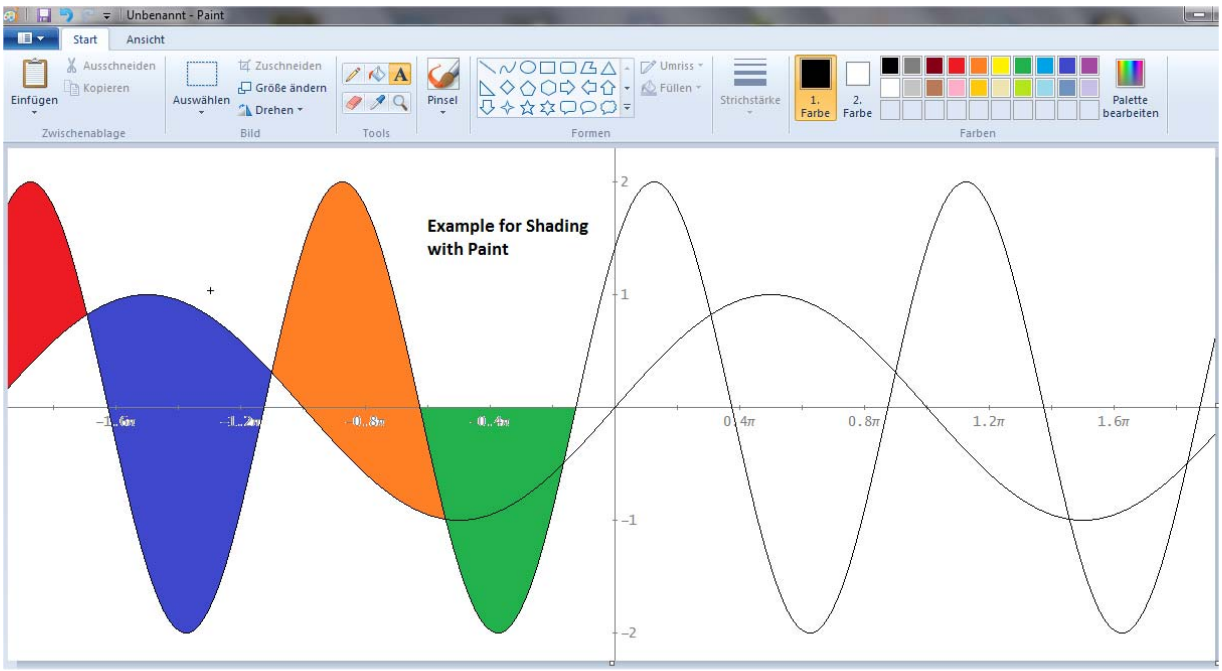


$$\text{AreaBetweenCurves}\left(\text{SIN}(x),\ 2\cdot\text{COS}\left(2\cdot x - \frac{\pi}{4}\right),\ x,\ -2\cdot\pi,\ \frac{3}{2}\cdot\pi,\ y\right)$$



$$\text{AreaOverCurve}\left(2\cdot\text{COS}\left(2\cdot x - \frac{\pi}{4}\right),\ x,\ -2\cdot\pi,\ 1.5\cdot\pi,\ y\right)$$

The next plot was created using the DERIVE 6 option to transfer the plot for a short while to the very comfortable Graphics Utility Paint. Click on the plot after copying it in the Algebra window and open the Edit Menu for proceeding.



Graphic - Tools for the TI-92

von W. Pröpper, Nürnberg

1. Einleitung

Nach dem Entpacken der Datei GRA_TOOD.EXE haben Sie außer dieser Dokumentation RA_TOOD.DOC eine TI-92 Gruppierung GRA_TOOD.92G erhalten. Mit Hilfe von TI-Graph-Link kann dieses group file in 5 TI-92 PRGMs (*.92P files) entpackt werden. Diese können dann (auch unter Verwendung von Graph-Link) in den TI-92 übertragen werden. (Es empfiehlt sich, vorher mit 2nd [VAR-LINK] ein eigenes Verzeichnis auf dem TI-92 anzulegen - Z.B. UTY - die Programme dorthin zu laden und in diesem Verzeichnis ablaufen zu lassen.)

Die Programme wurden von W. Pröpper erstellt und werden zur nichtkommerziellen Nutzung für den allgemeinen Gebrauch freigegeben. Obwohl die Programme mit Sorgfalt angefertigt wurden, kann eine Gewährleistung, gleich welcher Art, nicht übernommen werden.

Verbesserungsvorschläge, Anregungen und Kritik werden dankbar (und möglicherweise zähneflet-schend) entgegengenommen.

Snail-Mail: W. Pröpper, Josef-Simon-Str. 59, 90473 Nürnberg

E-Mail: w. proepper@wpro.franken.de;

2. Programmbeschreibung

Der TI-92 erlaubt schnell und einfach Graphen von Funktionen darzustellen. Will man diesen Graphen jedoch ein vorzeigbares Äußeres geben, stößt man schnell an Grenzen. Die Plot Styles "Dot", "Square" und "Thick" stellen, wenn überhaupt wirksam, nicht das Optimum an Sauberkeit dar und die über die Funktionstaste F7 im Graphik Fenster erreichbaren Linien- und Textwerkzeuge dienen in den wenigsten Fällen dazu, einen Graph mit aussagekräftigen Zusätzen zu versehen. Die fünf Programme der GRA _ TOOLs Gruppe sollen hier eine gewisse Abhilfe schaffen. Mit einem der Programme können Graphen mit verschieden geformten Markierungen für Punkte versehen werden und mit den vier restlichen Programmen lassen sich strichlierte Linien zeichnen.

(Short explication in English follows at the end of the contribution, Josef)

pntat(x,y,p,l) zeichnet Punkte mit den Koordinaten (*x*,*y*). Dabei können *x* und *y* sehr unterschiedliche Formen im Aufruf annehmen (Beispiele s. unten). Die Form der Markierung wird mit dem Parameter **p** bestimmt. Ist p = 1, so wird ein "+"-Kreuz und bei p = 2 ein "×"-Kreuz gezeichnet. Einen kleinen Kreis erhält man mit p = 3. Bei p = 4 entsteht ein gefülltes Quadrat (■) und bei p = 5 eine gefüllte Raute (♦). Der Parameter **l** bestimmt die Größe der Objekte. Bei l = 2 beträgt ihre maximale Ausdehnung 5 Pixel, bei l = 3 ist sie 7 Pixel. Nach regulärer Ausführung des Programms schaltet der TI-92 in das Graphik Fenster. Liegen jedoch alle Punkte außerhalb des Plotbereichs oder wenn für p bzw. l unzulässige Werte angegeben wurden, bleibt der Home Screen aktiv und der Programmaufruf meldet done (obwohl eigentlich nichts passiert ist).

Beispiele: pnt(1,2,1,2) erzeugt ein kleines "+"-Kreuz im Punkt (1,2*),* während pnt(2,1,2,3) bei (2,1) ein großes "×"-Kreuz zeichnet. pntat({-1,2,3},{1,-1,2},1,3) setzt große "+"-Kreuze an die Stellen (-1,1), (2,-1) und (3,2).

Wenn die Funktion *f*(*x*) deklariert ist und bei $x = 2$ eine stetig behebbare Definitionslücke besitzt, dann setzt pntat(2,limit(f(x),x,2),3,2) an diese Stelle einen Kreis mit Radius 2 Pixel.

Um alle Punkte mit horizontalen Tangenten (in den meisten Fällen die Extrema) mit einem großen "+"-Kreuz zu markieren, ruft man pntat(zeros(*d*(f(x),x),x),f(zeros(*d*(f(x),x),x)),1,3) - oder besser: zeros(*d*(f(x),x),x), was eine Liste der Nullstellen der 1. Ableitung berechnet, und dann pntat(ans(1),f(ans(1)),1,3) - auf.

Das Markieren aller Nullstellen einer Funktion mit einer großen Raute würde mit pntat(zeros(f(x),x),0,5,3) allerdings nicht funktionieren, da der *x*-Parameter (bei mehr als einer Nullstelle) eine Liste mit mehr als einem Element liefert, der Wert 0 für den *y*-Parameter jedoch nur als einelementige Liste interpretiert wird.

Mit f(zeros(f(x),x)) anstelle von 0 erhält man jedoch das gewünschte Resultat.

slv(x,l,p) zeichnet eine vertikale strichlierte Linie an der Stelle **x**. Die Länge der Striche ist **l**, die Zwischenräume haben die Länge **p**. Da ein Funktionsgraph durchaus mehrere vertikale Asymptoten besitzen kann, darf der Parameter **x** auch eine Liste sein.

slh(y,l,p) zeichnet entsprechend eine horizontale Linie bei **y**. Die Parameter **l** und **p** haben die gleiche Wirkung wie bei der vertikalen strichlierten Linie. (Für y ist nur eine Zahl bzw. ein numerischer Ausdruck zulässig.)

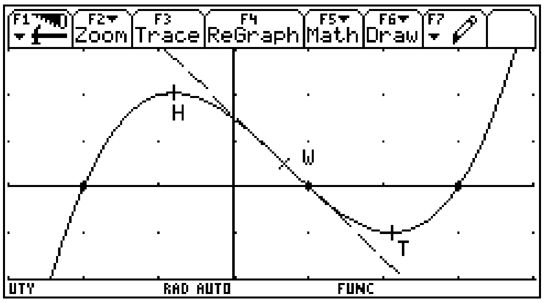
sls(x1,y1,x2,y2,l,p) erzeugt eine strichlierte Linie durch die Punkte (*x1*,*y1*) und (*x2*,*y2*) mit Strichlänge **l** und Zwischenraum **p**. Diese Prozedur wird normalerweise für schiefe Asymptoten oder auch Wendetangenten verwendet.

sls ruft slv bzw. slh, wenn die *x*- bzw. die *y*-Koordinaten gleich sind aus dem Folder uty. Falls die Programme in einem anderen Folder des TI-92 installiert sind, führt dies zu einem Programmfehler.

slf(t,l,p) liefert schließlich eine strichlierte Näherungskurve der Funktion $y = t(x)$. Dazu muss **t** ein Ausdruck mit der unabhängigen Variablen *x* sein. Die Näherungskurve besteht aus Geradenstücken, deren Anfangs- und Endpunkt auf der Kurve $y = t$ liegen. Die angegebenen Parameter für Länge und Zwischenräume beziehen sich auf horizontale Stücke und sind bei einem Koordinatensystem mit gleichen Längen der *x*- und der *y*-Einheiten (z.B. ZoomDec) stimmig. Bei starker Verzerrung (z.B. ZoomStd) sind die für den Betrachter sichtbaren Strecken gleich lang, obwohl, bezogen auf das Koordinatensystem, die steileren Stücke länger sind als die flacheren.

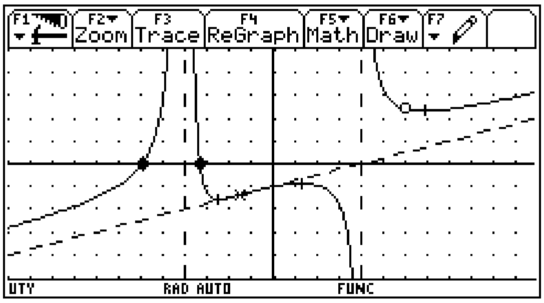
Der Term t darf auch längere Lücken im Definitionsbereich haben (z.B. $\sqrt{(x^2 - 1)}$, der in] -1,1 [nicht definiert ist). Allerdings läuft **slf** dann einigermaßen langsam. Um „passende“ Werte für **l** und **p** zu erhalten muss man ggf. etwas experimentieren.

Die fünf Programme laufen in den 4 zweidimensionalen Graph Modes. Es empfiehlt sich, zuerst den Graph der Funktion zeichnen zu lassen und anschließend die oben gezeigten „Verschönerungsarbeiten“ vorzunehmen. Das Kommando ReGraph (F4) im Graph Screen löscht alle von den oben genannten Funktionen erzeugten Linien und Punkte.



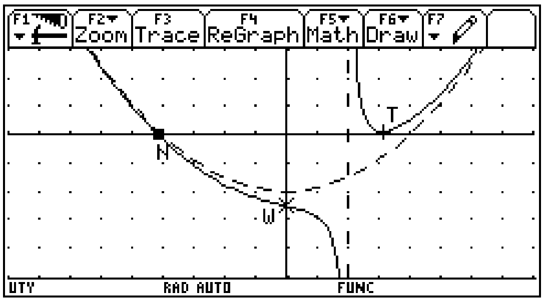
Ein Polynom 3. Grades: Extrempunkte mit "+"-Kreuzen, Wendepunkt mit "×"-Kreuz, Nullstellen mit Rauten. Wendetangente strichliert.

Cubic: Turning points "+", inflection point "×", zeros "♦", tangent in inflection point dashed.



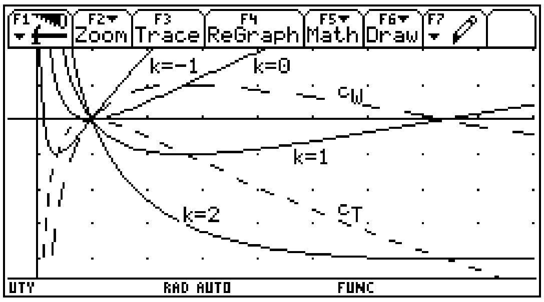
Eine gebrochen rationale Funktion mit vertikalen und schiefen Asymptoten. Punkte sind gekennzeichnet wie vorher.
Die stetig behebbare Definitionslücke ist mit einem Kreis markiert.

Rational function with two vertical and one slant (oblique) asymptote, points are marked like on the left graph, the removable discontinuity is marked by a circle.



Eine gebrochen rationale Funktion mit einer vertikalen Asymptote und einer Parabel als asymptotischer Kurve. Besondere Punkte sind wie vorher gekennzeichnet (Nullstelle mit "■")

Rational function with a vertical asymptote and a parabola as asymptotic curve. Zero is marked by a "■".



Eine Kurvenschar ($f_k(x) = (-2\ k + \ln x)\cdot\ln x$), in der die Kurve der Tiefpunkte kurz gestrichelt, die der Wendpunkte mit längeren Strichen eingezeichnet ist.

Family of curves ($f_k = (-2k + \ln x)\cdot\ln x$), with locus of turning points dashed (short segments) and locus of inflection points (long segm.).

Summary of the tools

pntat(x,y,p,l) point at (x,y), p for the mark (1,2,3,4,5) for + , ×,●, ■, ♦;
l for size: 2: small, 3: large

slv(x, l, p) vertical dashed line at position x, with segments of length l and spaces of length p

slh(x, l, p) horizontal dashed line at position x, with segments of length land spaces of length p

sls(x1, y1, x2, y2, l, p) segment between points (x1,y1) and (x2,y2), length and space

slf(t, l, p) dashed function graph for function t(x), length and space

Titbits from Algebra and Number Theory(14)

by Johann Wiesenbauer, Vienna

Back again! As some of the readers may know there has been a vivid discussion about Stirling numbers of the first and second kind via the DERIVE-mailbase quite recently. In the aftermath Josef has asked me to sum up the results of this discussion and to tell you more about Stirling numbers in general, in particular how they are defined and what they are good for. Well, actually this stuff belongs to combinatorics (in fact, their DERIVE-implementations can be found in the new utility-file COMBINAT.MTH !), it is true, but I have no guilty conscience at all when treating this topic here, as Stirling numbers also have a number of important applications in algebra and number theory.

The discussion mentioned above was triggered off by the following problem posed by Peer van de Sanden on the DERIVE-mailbase(derive-news@mailbase.ac.uk):

Hello,

I want to compute Stirling numbers of the first and the second kind. The following recursive functions work but are slow:

Stirling numbers of the first kind:

```
stirl1(n, k) :=
  If n = k
     1
     If k = 1
        (n - 1)!
        (n - 1)·stirl1(n - 1, k) + stirl1(n - 1, k - 1)
```

Stirling numbers of the second kind:

```
stirl2(n, k) :=
  If n = k
     1
     If k = 1
        k
        k·stirl2(n - 1, k) + stirl2(n - 1, k - 1)
```

For this last function I have found a faster one:

$$\text{stirl\_2}(n, k) := \sum_{p=0}^{k} \frac{\text{COMB}(k, p)\cdot(k - p)^n\cdot(-1)^p}{k!}$$

Does anyone know a function for the Stirling numbers of the first kind which is faster then the one from above:

Here are the recursive rules:

Stirling1(n,1) = (n-1)!

Stirling1(n,n) = 1

Stirling1(n,k):= (n-1) · Stirling1(n-1,k) + Stirling1(n-1,k-1)

Peer van de Sanden

I named the function `stirl` because later we will use STIRLING (name of the respective DERIVE functions). Johann will give a short description what the St. n. are good for. In this revised version I'll take the space to add some more accurate notes on them, Josef

Some words about Stirling numbers:

James Stirling was a Scottish mathematician who lived from 1692 – 1770)

Stirling numbers of the first kind S1(*n*,*k*) is the number of permutations of a set of *n* elements having exact *k* cycles.

Set {1,2,3,4} can be distributed in 2-cycle permutations as follows:

(1,2,3)(4), (1,3,2)(4), (1,2,4)(3), (1,4,2)(3), (1,3,4)(2), (1,4,3)(2),(2,3,4)(1), (2,4,3)(1), (1,2)(3,2), (1,3)(2,4), (1,4)(2,3) which makes exact 11 possible circles.

Function says: `stirl1(4, 2) = 11`

Stirling numbers of the second kind S2(n,k) is the number nonempty subsets of a set with *n* elements having *k* elements.

Set {1,2,3,4} can be partitioned into two nonempty subsets as follows:

{{1,2},{3,4}},{[1,3},{2,4}},{{1,4},{2,3}}, {{1,2,3},{4}},{{1,2,4},{3}}, {{1,3,4},{2}},{{2,3,4},{1}} which makes exact 7 partitions. [1]

Function says: `stirl_2(4, 2) = 7`

Now let's Johannes proceed:

Before going on I should explain where the recursive rules above come from. Basically, Stirling numbers are used to convert back and forth between the standard representation of a polynomial over a commutative ring with identity and its representation using so-called factorial powers

$$x^{\underline{k}} := x(x-1)(x-2)...(x-k+1) \quad (k \geq 0).$$

(For $k = 0$ take the empty product 1 as usual). More precisely the following defining equations hold:

$$x^{\underline{n}} := \sum_{k=0}^{n} \begin{bmatrix} n \\ k \end{bmatrix} (-1)^{n-k} x^k \quad \text{and} \quad x^n := \sum \begin{Bmatrix} n \\ k \end{Bmatrix} x^{\underline{k}} \quad (n \geq 0)$$

where we used the notations $\begin{bmatrix} n \\ k \end{bmatrix}$ and $\begin{Bmatrix} n \\ k \end{Bmatrix}$ for Stirling1(n,k) and Stirling2(n,k), respectively.

From the definition of the Stirling numbers of the first kind it follows immediately that the polynomial

$x(x+1)(x+2)(x+3)...(x+n-1)$

is their generating function for any fixed $n \geq 0$. (The reader should be warned here that some authors choose the signs of Stirling numbers of the first kind in such a way that the n^{th} factorial power

$x(x-1)(x-2)(x-3)...(x-n+1)$

becomes the generating function for fixed $n \geq 0$. They are usually called the "signed" Stirling numbers of the first kind in order to differ them from our "unsigned" ones.) In particular, by expanding both sides of the equation

$$x(x+1)(x+2)(x+3)...(x+n-1) = \left(\sum_{k=0}^{n} \begin{bmatrix} n \\ k \end{bmatrix} x^k \right)(x+n-1)$$

[1] Graham, Knuth, Patashnik, *Concrete Mathematics,* Addison-Wesley, 1989

and comparing the coefficient of x^k for $0 \le k \le n$, the recursion formula for Stirling numbers of the first kind mentioned by Peer is easily obtained.

My answer contained the following passage:

.... To the best of my knowledge in the current version of the new utility file combinat.mth the following formulas are used for Stirling numbers:

```
                      1    k                   k - j_   n
stirling2(n, k) := ───· ∑   COMB(k, j_)·(-1)        ·j_
                     k!   j_=1

stirling1(n, k) :=
  If n < k
     0
     If n = k
        1
        (-1)^(n + k)·∑((-1)^j_·COMB(n - 1 + j_, n - k + j_)·
        COMB(2·n - k, n - k - j_)·stirling2(n - k + j_, j_), j_, 1, n - k)
```

The implementation of ST1RL1NG1(n,k) above is certainly faster than that of Peer van de Sanden (by a factor 2 or so). Since it essentially follows a suggestion of mine, I would like to pass on the question: Is there still a faster one? ...

Yes, it is true, Al Rich had asked me exactly the same question so me months before and I had suggested in my answer the implementation of STIRLINGl(n,k) above (however, without considering the cases $n < k$ and $n = k$; by the way, Peer too overlooked the case $n < k$ and also the case $k = 0$ in his implementation). Since my implementation was so much faster on average than Peer's (speaking of a factor 2 only was actually a colossal understatement!), I thought that my final asking whether there was still a faster one was only a rhetorical question. Little did I know then, how much I was mistaken!

This is where the third protagonist, Ralph Freese of Hawaii, comes into play with the following message:

Here's a faster Stirling1 version:

```
stirling1_aux(v, n) := VECTOR(IF(k = 1, n!, IF(k = n + 1, 1, n·v  + v      )),
                                                                k    k - 1
                       k, 1, n + 1)

                       n - k
stirling1(n, k) := (-1)     ·(ITERATE(stirling1_aux(v, DIM(v)), v, [1], n))
                                                                           k
```

Notes:

1. *This is only valid if 1 <= k <= n; it really should have an 1F statement like Derive's version in front of it.*
2. *I've included the sign as Derive did; Peer van de Sanden's numbers are called unsigned Stirling numbers of the first kind.*

Concerning calculation time:

Peer van de Sanden algorithm is slow because to calculate stirling1(2n,n) two calls are made to stirlingl(2n-2,n-1), four calls are made to stirling1(2n-4,n-2), ... , and 2^n calls are made to stirling1(1,l). So this implementation requires exponential time.

The program above is really the same recursion as Peer van de Sanden's except that it remembers what it has previously calculated.

The Derive program is much better: it has running time proportional to n"3. The program above has running time proportional to n/: 2.

Shortly after he added the following Post Scriptum.

The final n should be n – 1 and I noticed that Derive was returning the unsigned stirling1 so the (-1)$^{n+k}$ should be removed. stirling1(n,k) should be defined:

```
stirling1(n,k):=ITERATE(stirling1_aux(v,DIM(v)), v, [1], n - 1))
                                                                 k
```

Ralph

Well, should I really have to swallow the bitter pill and accept that I had clearly failed in finding the best way of tackling this problem? At first, I wasn't prepared to do this as you may conclude from my following reaction:

Hm, ... although Ralph's implementation of Stirling numbers of the first kind is certainly streets ahead of anything I have seen so far on that score, it remains to be seen whether it is really faster than mine in practice (though it should be asymptotically faster, if the estimates above are correct).

Unfortunately, when trying to check this I came across unexpected difficulties. For small n, say n<100, my implementation is notably faster, it is true, but for larger n (n=103 and k = 1 seems to be the smallest example) the function SUM( ) in stirling1(n,k)

```
                      1    k                         k - j_    n
stirling2(n, k) ≔ ————· ∑    COMB(k, j_)·(-1)             ·j_
                     k!   j_=1


stirling1(n, k) ≔
  If n < k
     0
     If n = k
        1
        (-1)^(n + k)·∑((-1)^j_·COMB(n - 1 + j_, n - k + j_)·
        COMB(2·n - k, n - k - j_)·stirling2(n - k + j_, j_), j_, 1, n - k)
```

all of a sudden goes to pieces returning senseless values such as fractions.

To see that my routine should work and SUM () is actually the culprit all you have to do is to replace SUM( ) by SUM(VECTOR(...)) in stirling1(n,k), but it goes without saying that this remedy is sheer madness when it comes to performance.

Thus in my opinion in terms of speed a fair comparison between the two implementations seems to be impossible before a fix of this bug and has to be postponed ... On the other hand, it may well be that very few Derivers, if any, are troubled by this ...

(By the way, according to Albert Rich this SUM error was simply caused by a dummy variable name conflict: stirling1 and stirling2 both used the same dummy variable j_ …)

This is what Ralph answered:

Absolutely right. (Referring to the first paragraph of my email!)

I didn't get the SUM error. On my computer computing S1(2n,n) was faster with your version than mine even with n=100. But the time as a function of n did seem to be growing at a slower rate than yours. Viewing S1(n,k) as a lower triangular matrix, may method calculated the first n rows and returned the $(n,k)^{th}$ entry. But it really only needs the entries in the parallelogram bounded by (1,1), (k,k), (n,k), (n–k+1,1).

Here is a program to do that:

```
stirling1_next2(k, row, shift, offset, u) ≔ [row, offset + shift,

  VECTOR(IF(j = 1, (row - 1)·u , IF(j = row, 1, (row - 1)·u
                              1                           j - offset + 1

  + u           )), j, offset + shift, IF(row < k, row, k))]
     j - offset

stirling1_next1(v, k, n) ≔ stirling1_next2(k, 1 + v , IF(v  > n - k, 1,
                                                   1     1

  0), v , v )
       2   3

stirling1(n, k) ≔ (ITERATE(stirling1_next1(v, k, n), v, [1, 1, [1]], n -


  1))
     3,1

stirling1(10, 5) = 269325
```

(Sorry, this isn't very readable.) The area of the parallelogram is k·(n-k+1). The worst case is k = n/2. it is only about twice as fast for k=n/2 but for k near 1 or n, it is much faster. Yours is worst when k is small since it sums from 1 to n–k so this one is faster than yours when k is small.
Ralph

When checking Ralph's arguments above it dawned on me at last that he was right and I had to change my basic strategy. This is from my answer:

(...) *After looking into the whole business once more, I must admit that you have a point there. on top of leading to that SUM-bug the formula I used is also very inappropriate when it comes to small values of k. Therefore I have made up my mind to change sides and try to beat you at your own game. What about the following implementation?*

```
S1(n,k):=IF(k>n,0,IF(k=1,(n-1)!,IF(k>n/2,(ITERATE([DELETE(n_·APPEND(v_,[0])+
         APPEND([0],v_)),n_+1],[v_,n_],ITERATE([n_·APPEND(v_,[0])+APPEND([0],v_),
         n_+1],[v_,n_],[[1],0],n-k),k))↓1↓1,(ITERATE([DELETE(n_·APPEND([0],v_)+
         APPEND(v_,[0])),n_+1],[v_,n_],ITERATE([n_·APPEND([0],v_)+APPEND(v_,[0]),
         n_ + 1], [v_, n_], [[1], 0], k), n - k))↓1↓1)))
```

```
S1(10, 5) = 269325

S1(100, 2) =

  48318504964586796816005505663040585914405716072348750119187884695326280735810632874 28~
  459115887885635393993680616672360632116716255156633600000000000000000000
S1(1000, 999) = 499500
```

And here are some execution times on a Pentium 116 PC (1999)

S1(*1000,2*): *0.5s vs. 2.3s*
S1(*1000,500*): *20.2s vs. 161.9s*
S1(*1000,999*): *0.5s vs. 2.2s*

Now (in 2014) are the execution times: 0.063s (2566 digits number), 2.27s (1627 digits number) and 0.063s for the 6 digits number. Josef

I hope this top performance will reconcile you even if you don't agree with my programming style due to the missing auxiliary functions. (Sorry!)

Cheers, Johann

I'd like to explain the decisive idea behind my implementation above, because it could turn out useful in your own programs. Similar to Pascal's triangle one can compute the "*Triangle of Stirling Numbers of the first kind*", e.g.

```
VECTOR([n, VECTOR(S1(n, k), k, 0, n)], n, 0, 8)
```

0	[1]
1	[0, 1]
2	[0, 1, 1]
3	[0, 2, 3, 1]
4	[0, 6, 11, 6, 1]
5	[0, 24, 50, 35, 10, 1]
6	[0, 120, 274, 225, 85, 15, 1]
7	[0, 720, 1764, 1624, 735, 175, 21, 1]
8	[0, 5040, 13068, 13132, 6769, 1960, 322, 28, 1]

Now it is obvious that it should be much faster to compute each row of this triangle as a whole vector by using the built-in vector operations than computing it element wise. Take the row number 5 for example: Instead of computing each of its components

$$0 = 4\cdot 0 + 0,\ 24 = 4\cdot 6 + 0,\ 50 = 4\cdot 11 + 6, \text{ etc.}$$

we can compute it at once from two modified copies of row number 4 (zeros have to be inserted in its beginning and end, respectively) in the following way:

$$4\cdot[0, 6, 11, 6, 1, 0] + [0, 0, 6, 11, 6, 1].$$

This accounts for the frequent occurrence of expressions such as

```
n_·APPEND(v_, [0]) + APPEND([0], v_)
```

in my implementation above.

Only later did I exploit Ralph's idea that only the Stirling numbers of the first kind in a certain parallelogram are actually needed to the full extent. It is contained in my following email to all Derivers connected to the mailbase (skipping many other interesting contributions in-between in order to cut a long story short and come to an end):

Just in case you also belong to the people who dot the i's and cross the t's like me, you might enjoy the following a little bit tidier'version' of my STIRLING1(n,k):

```
stirling1(n,k):=
IF(k ≥ n,
   MAX(1 - k + n, 0),
   IF(k,
      0,
      IF(k = 1,
         (n - 1)!,
         IF(k = n - 1,
            COMB(n, 2),
            IF(k > n/2,
               (ITERATE([n_·DELETE(v_) + DELETE(v_, 1 - n_ + n), n_ + 1],
               [v_, n_], ITERATE([n_·APPEND(DELETE(v_), [0]) + v_, n_ + 1],
               [v_, n], ITERATE([n_·APPEND(v_, [0]) + INSERT(0, v_), n_ + 1],
               [v_, n_], [[1], 0], n - k), 2·k - n), n - k))↓1↓1,
               (ITERATE([n_·DELETE(v_, 1 - n_ + n) + DELETE(v_), n_ + 1],
               [v_, n_], ITERATE([n_·v_ + APPEND(DELETE(v_), [0]), n_ + 1],
               [v_, n_], ITERATE([n_·INSERT(0, v_) + APPEND(v_, [0]), n_ + 1],
               [v_, n_], [[1], 0], k), n - 2·k), k))↓1↓1)))))
```

Numerical experiments seem to indicate that the asymptotic growth of computation times for Stirling numbers of the first kind is much better than that for Stirling numbers of the second kind, though the latter can be given explicitly by the simple formula

```
                      1   k                    k - j_   n
stirling2(n, k) := ———· ∑   COMB(k, j_)·(-1)       ·j_
                     k!  j_=1
```

This leads to the following interesting question: Is this inevitable or does exist an implementation for Stirling numbers of the second kind as well that is asymptotic faster than the one above?

Shortly after I gave the answer myself:

Yes, there is an implementation with a far better behaviour and as a matter of fact it is so simple that it is almost embarrassing. (Why couldn't this guy do his job properly from the very start?!)

```
stirling2(n, k) := (ITERATE([APPEND(v_, [0]) + INSERT(0, v_), n_ + 1], [v_, n_],
[[1], 0], k))↓1·VECTOR((-1)^(k - j_)·j_^n, j_, 0, k)/k!
```

For small n and k, you may gain some fractions of a second by using the old implementation, but in my opinion it doesn't pay off to split the computation depending on the size of the arguments ...

Again it was Ralph who found a flaw (though harmless this time!) in the routine above thereby putting the finishing touches to it:

Nice!

I did note that you didn't need to have the n_ since it is not used. So it can be simplified a little to:

```
stirling2(n, k) := ITERATE(APPEND(v_, [0]) + INSERT(0, v_), v_, [1], k)·
VECTOR((-1)^(k - j_)·j_^n, j_, 0, k)/k!
```

Ralph

Well, there may still be minor improvements, e.g.

```
stirling2(n, k) ≔ IF(n < k, 0, ITERATE(APPEND(v_, [0]) + INSERT(0, v_), v_, [1],
k)·VECTOR((-1)^(k - j_)·j_^n, j_, 0, k)/k!)
```

but its performance is already staggering in the present form. For example, try to compute `stirling2(2000,1000)` with DERIVE and other CAS (nomina sunt odiosa!) and you'll see the difference, believe me!. The same goes for `stirling1` above, where I recommend comparing the computation times for `stirling1(1000,500)`.

> Note: (`stirling2(2000,1000)` is a 3355 digit number, which needs 1.41 sec calculation time, `stirling2(1000,55)` gives a 1627 digit number needing 0.93 seconds.)

Concluding another nice programming task: `stirling1(n,k)` can also be defined as the number of permutations on *n* letters having *k* cycles, whereas `stirling2(n,k)` is the number of ways to partition a set of *n* elements into *k* disjoint subsets. How could DERIVE-routines for `stirling1(n,k)` and `stirling2(n,k)` look like that make use of this?

Additional comments on Stirling numbers:

I found many interesting explanations, proofs and identities in one of the "computer science bibles" – *Concrete Mathematics by Graham, Knuth & Patashnik.*

Here is a small selection together with the *DERIVE* verification:

[1] $$\left[\begin{matrix} n \\ k \end{matrix}\right] = \left\{\begin{matrix} -k \\ -n \end{matrix}\right\} \quad \text{integer } k, n$$

[2] $$\left\{\begin{matrix} n \\ m \end{matrix}\right\} = \frac{1}{m!}\sum_{k=1}^{m}\binom{m}{k} k^n (-1)^{m-k}$$

[3] $$\left[\begin{matrix} m+n+1 \\ m \end{matrix}\right] = \sum_{k=m}^{n}(n+k)\left[\begin{matrix} n+k \\ k \end{matrix}\right]$$

[4] $$\binom{n}{m} = \sum_{k=0}^{n}\left\{\begin{matrix} n+1 \\ k+1 \end{matrix}\right\}\left[\begin{matrix} k \\ m \end{matrix}\right](-1)^{m-k}$$

[1] does not work with the functions given in Titbits14, but it works with the *DERIVE* - functions.

```
[STIRLING1(10, 7), STIRLING2(-7, -10)] = [9450, 9450]

stirling2(13, 7) = 5715424

 1    7                13      7 - k
 ——·  Σ  COMB(7, k)·k   ·(-1)       = 5715424
 7!  k=1

                                   19
stirling1(19 + 8 + 1, 19) =        Σ  (8 + k)·stirling1(8 + k, k)
                                   k=1

60383004803151030 = 60383004803151030

 20                                            11 - k
 Σ   stirling2(21, k + 1)·stirling1(k, 11)·(-1)       = 167960
k=11

COMB(20, 11) = 167960
```

A finite Group with linear rational Functions

Richard Schorn, Kaufbeuren, GER
richard.schorn@t-online.de

Expressions **A** and **B** (see below) can be used to construct a finite group. The binary operation **P⊗Q**= **PQ** is defined by replacing variable *x* in expression **Q** by the expression **P** which is "**P in Q**".

$$\mathbf{A} = \frac{x}{x-1}, \qquad \mathbf{B} = 3 - x$$

Find the order of the group by building the complete group table. Explain your work using *DERIVE*.

Solution:

```
CaseMode := Sensitive
```

$$\left[A := \frac{x}{x - 1}, B := 3 - x\right]$$

B — Let's find **AB**

3 − x

3 − A — Replace *x* in **B** by **A** (**A in B**)

$$2 - \frac{1}{x - 1}$$ and factorize!

$$\frac{2\cdot x - 3}{x - 1}$$ this is now **AB**!

$$OP(E1, E2) := \lim_{x \to E1} E2$$ Introduce function OP(E1,E2) for "**E1 in E2**"

OPI(e1, e2) := ITERATE(e2, x, e1, 1) — same for purists who don't like the limit at this occasion.

$$OP(A, B) = 2 - \frac{1}{x - 1}$$

$$OPI(A, B) = 2 - \frac{1}{x - 1}$$

$$FACTOR(OP(A, B)) = \frac{2\cdot x - 3}{x - 1}$$ once more **AB**.

OP(A, A) = x — The powers of **A**: $\mathbf{A}^2$

OP(B, B) = x — and $\mathbf{B}^2$ (does not need DERIVE).

$$OP\left(\frac{2\cdot x-3}{x-1}, \frac{2\cdot x-3}{x-1}\right) = \frac{x-3}{x-2}$$

$$OP\left(\frac{x-3}{x-2}, \frac{2\cdot x-3}{x-1}\right) = x$$

$$OP\left(\frac{2\cdot x-3}{x-1}, \frac{x-3}{x-2}\right) = x$$

The group contains cyclic subgroups of orders 2 and 3, thus the group order must be divisible by 6!

$$OP(B, A) = \frac{x-3}{x-2}$$

$$[OPI(A, B), OPI(B, A)] = \left[2 - \frac{1}{x-1}, \frac{x-3}{x-2}\right]$$

$$OP(OP(B, A), OP(B, A)) = \frac{2\cdot x-3}{x-1}$$

Vector of the elements known so far:

$$\left[e1 := x,\ e2 := \frac{x}{x-1},\ e3 := 3 - x,\ e4 := \frac{2\cdot x-3}{x-1},\ e5 := \frac{x-3}{x-2}\right]$$

e1 = A^2 = B^2 = E, e2 = A, e3 = B, e4 = AB, e5 = BA

OP(A, [e1, e2, e3, e4, e5])

$$\left[\frac{x}{x-1}, x, 2 - \frac{1}{x-1}, 3 - x, \frac{2\cdot x-3}{x-2}\right]$$

$$\left[\frac{x}{x-1}, x, \frac{2\cdot x-3}{x-1}, 3 - x, \frac{2\cdot x-3}{x-2}\right]$$

A new element has been born (the last one in the row is **ABA**).

Now we have found all elements of the group:

$$\left[e1 := x,\ e2 := \frac{x}{x-1},\ e3 := 3 - x,\ e4 := \frac{2\cdot x-3}{x-1},\ e5 := \frac{x-3}{x-2},\ e6 := \frac{2\cdot x-3}{x-2}\right]$$

grp := [e1, e2, e3, e4, e5, e6]

We check if there are really no “Strangers in the Group”?

$$\text{FACTOR(OP(e1, grp))} = \left[x, \frac{x}{x-1}, 3-x, \frac{2\cdot x-3}{x-1}, \frac{x-3}{x-2}, \frac{2\cdot x-3}{x-2}\right]$$

$$\text{FACTOR(OP(e2, grp))} = \left[\frac{x}{x-1}, x, \frac{2\cdot x-3}{x-1}, 3-x, \frac{2\cdot x-3}{x-2}, \frac{x-3}{x-2}\right]$$

$$\text{FACTOR(OP(e3, grp))} = \left[3-x, \frac{x-3}{x-2}, x, \frac{2\cdot x-3}{x-2}, \frac{x}{x-1}, \frac{2\cdot x-3}{x-1}\right]$$

$$\text{FACTOR(OP(e4, grp))} = \left[\frac{2\cdot x-3}{x-1}, \frac{2\cdot x-3}{x-2}, \frac{x}{x-1}, \frac{x-3}{x-2}, x, 3-x\right]$$

$$\text{FACTOR(OP(e5, grp))} = \left[\frac{x-3}{x-2}, 3-x, \frac{2\cdot x-3}{x-2}, x, \frac{2\cdot x-3}{x-1}, \frac{x}{x-1}\right]$$

$$\text{FACTOR(OP(e6, grp))} = \left[\frac{2\cdot x-3}{x-2}, \frac{2\cdot x-3}{x-1}, \frac{x-3}{x-2}, \frac{x}{x-1}, 3-x, x\right]$$

This is the result:

$$\mathbf{E} = x,\ \mathbf{A} = \frac{x}{x-1},\ \mathbf{B} = 3-x,\ \mathbf{AB} = \frac{2x-3}{x-1},\ \mathbf{BA} = \frac{x-3}{x-2},\ \mathbf{ABA} = \frac{x-3}{x-2}$$

with among others **BAB** = **ABA**, **ABAB** = **BA**, **BABA** = **AB**, …

See the operation table:

	E	A	B	AB	BA	ABA
E	E	A	B	AB	BA	ABA
A	A	E	AB	B	ABA	BA
B	B	BA	E	ABA	A	AB
AB	AB	ABA	A	BA	E	B
BA	BA	B	ABA	E	AB	A
ABA	ABA	AB	BA	A	B	E

On my request Richard Schorn sent three additional examples:

- **A** $= -\dfrac{\sqrt{5}+3}{2(x+1)}$ which gives a cyclic group of order 5 (see next page).
- **A** $= \dfrac{1}{x}$, **B** $= 1-x$ gives a group which is isomorph to the group constructed by $\left[\dfrac{2x-3}{x-2}, 3-x\right]$ (= group of cover mappings of an equilateral triangle).
- **A** $= 2-x$, B $= \dfrac{2}{2-x}$ leads to a group of order 8.

I tried to create the group table belonging to Richard Schorn's first additional example. ***Cyclic groups*** are groups with all elements being powers of one single element. They can consist of finite or of infinite many elements.

Using the TI-NspireCAS I found out the five elements of this group:

$$\mathbf{U} = x,\ \mathbf{A} = -\frac{\sqrt{5}+3}{2(x+1)},\ \mathbf{A}^2 = -\frac{(\sqrt{5}+3)(x+1)}{2x-\sqrt{5}-1},\ \mathbf{A}^3 = \frac{(\sqrt{5}+3)(2x-\sqrt{5}-1)}{2((\sqrt{5}+1)\cdot x+2\cdot(\sqrt{5}+2))},$$

$$\mathbf{A}^4 = -\frac{(\sqrt{5}-1)\cdot((\sqrt{5}+1)\cdot x+2\cdot(\sqrt{5}+2))}{4x} = \frac{-2x-\sqrt{5}-3}{2x}$$

$\mathrm{op}(e1,e2):=\mathrm{factor}\left(\lim_{x\to e1}(e2)\right)$ ▸ *Done*

$a:=\frac{-(\sqrt{5}+3)}{2\cdot x+2}$ ▸ $\frac{-(\sqrt{5}+3)}{2\cdot(x+1)}$

$aa:=\mathrm{op}(a,a)$ ▸ $\frac{-(\sqrt{5}+3)\cdot(x+1)}{2\cdot x-\sqrt{5}-1}$

$aaa:=\mathrm{op}(aa,a)$ ▸ $\frac{(\sqrt{5}+3)\cdot(2\cdot x-\sqrt{5}-1)}{2\cdot((\sqrt{5}+1)\cdot x+2\cdot(\sqrt{5}+2))}$

$aaaa:=\mathrm{op}(aaa,a)$ ▸ $\frac{-(\sqrt{5}-1)\cdot((\sqrt{5}+1)\cdot x+2\cdot(\sqrt{5}+2))}{4\cdot x}$

$\mathrm{op}(a,aaa)$ ▸ $\frac{-(\sqrt{5}-2)\cdot(2\cdot(\sqrt{5}+2)\cdot x+5\cdot\sqrt{5}+11)}{2\cdot x}$

$\mathrm{op}(a,aaa)-\mathrm{op}(aaa,a)$ ▸ 0⚠

$\mathrm{op}(aaaa,a)$ ▸ x

$\mathrm{op}(a,aaaa)$ ▸ x

And this is the operation table of this commutative cyclic group:

	U	**A**	$\mathbf{A}^2$	$\mathbf{A}^3$	$\mathbf{A}^4$
U	**U**	**A**	$\mathbf{A}^2$	$\mathbf{A}^3$	$\mathbf{A}^4$
A	**A**	$\mathbf{A}^2$	$\mathbf{A}^3$	$\mathbf{A}^4$	**U**
$\mathbf{A}^2$	$\mathbf{A}^2$	$\mathbf{A}^3$	$\mathbf{A}^4$	**U**	**A**
$\mathbf{A}^3$	$\mathbf{A}^3$	$\mathbf{A}^4$	**U**	**A**	$\mathbf{A}^2$
$\mathbf{A}^4$	$\mathbf{A}^4$	**U**	**A**	$\mathbf{A}^2$	$\mathbf{A}^3$

ON THE SOLUTION OF A LINEAR DIFFERENTIAL EQUATION OF THE ORDER *n* WITH CONSTANT COEFFICIENTS

Emilio Defez Candel , Vicente Soler Basauri
Departamento de Matemática Aplicada
Universidad Politécnica de Valencia, P. O. BOX 22.012. Valencia. Spain

1. Introduction:

In this paper we study the solution of the homogeneous linear differential equation of *n*th order with constant coefficients, not covered in **Derive** utility files, by implementing the file:

CONST.MTH

which enables the general solution of the homogeneous linear differential equation of the order *n* with constant coefficients:

$$a_n\, y^{n)}(x) + a_{n-1}\, y^{n-1)}(x) \;+\ldots+\; a_1 y'(x) + a_0 y(x) = 0 \;.$$

The functions in this file are also used to solve the linear differential equation of the *n*th order

$$a_n\, y^{n)}(x) + a_{n-1}\, y^{n-1)}(x) \;+\ldots+\; a_1 y'(x) + a_0 y(x) = f(x) \;,$$

by means of the undetermined coefficients and the constant variation methods.

The organization of this paper is as follows. In section 2 we review the theoretical results on linear differential equation of n^{th} order (see Flett). In section 3 we study the implementation of the file side *CONST.MTH* used for the solution of homogeneous linear differential equation, illustrating its use with several examples. In section 4 we apply the CONST.MTH file to the solution of the complete differential equation, depending on whether we use the undetermined coefficient or the parameter variation method. The file was implemented with **Derive XM**.

2. Linear differential equation of n^{th} order

Definition 2.1 *A linear differential equation of nth order is a differential equation that can be written like this:*

$$a_n(x)\, y^{n)}(x) + a_{n-1}(x)\, y^{n-1)}(x) \;+\ldots+\; a_1(x) y'(x) + a_0(x) y(x) = f(x) . \tag{1}$$

We can write the equation (1) in a more compact way:

$$Ay = f(x)\;,\; A = \sum_{i=0}^{n} a_i(x) D^i \;, \tag{2}$$

where D is the derivative operator, defined as $D = \frac{d}{dx}$, $D^n y = \frac{d^n y}{dx^n}$, $D^0 y = y$.

The existence and uniqueness of local solutions of equation (1) is guaranteed by the following result:

Theorem 2.1 *Let all the functions* $a_i(x)$, $i = 0,1,\ldots,n$ *and* $f(x)$ *be continuous functions in an interval* $]a,b[$*, with* $a_n(x) \neq 0$ *in that interval, and let* $x_0 \in\,]a,b[$, $\{y_0, y_1, \ldots, y_n\}$ *be arbitrary real values. Then there exists a unique solution* $y = y(x)$ *of the differential equation (1) in all the interval* $]a,b[$ *verifying:*

$$y(x_0) = y_0,\; y'(x_0) = y_1, \ldots, y^{n-1)}(x_0) = y_n$$

Definition 2.2 *An homogeneous linear differential equation of the nth order is a differential equation (1) where f(x) = 0. I.e.:*

$$a_n(x)y^{n)}(x)+a_{n-1}(x)\,y^{n-1)}(x)+\ldots+a_1(x)y'(x)+a_0(x)y(x)=0\ , \qquad (3)$$

which can also be written like this:

$$Ay=0\,,\ A=\sum_{i=0}^{n}a_i(x)D^i\ . \qquad (4)$$

As a consequence of the linearity of the derivative operator, the following result is obtained:

Theorem 2.2 If $\{y_1(x),y_2(x),\ldots,y_n(x)\}$ *are solutions of (3), then any linear combination of these functions of the form:*

$$y=C_1\,y_1(x)+C_2\,y_2(x)+\ \ldots\ +C_n\,y_n(x)\ \ ,\ \ C_1\,,C_2,\ \ldots\ ,C_n\ \in\mathrm{R}\,,$$

is a solution of (3).

Theorem 2.2 suggests the following definition:

Definition 2.3 *A set of functions* $\{y_1(x),y_2(x),\ldots,y_n(x)\}$*, all of them continuous in the interval* $]a,b[$*, is said to be* **linearly dependent** *in*]a,b[*if there exist constants* $\alpha_1,\alpha_2,\ldots,\alpha_n$*, not all naught, so that*

$$\alpha_1\,y_1(x)+\alpha_2\,y_2(x)+\ \ldots\ +\alpha_n\,y_n(x)=0\ ,$$

in all the interval $]a,b[$*. Otherwise the functions* $\{y_1(x),y_2(x),\ldots,y_n(x)\}$ *are said to be* **linearly independent.**

Definition 2.3 and theorem 2.2 enable the obtaining of the following result:

Theorem 2.3 *If* $\{y_1(x),y_2(x),\ldots,y_n(x)\}$ *are n functions linearly independent in the interval* $]a,b[$*, solutions of (3), where all the functions* $a_i(x)$ *,* $i=1,2,\ldots,n$ *are continuous in the interval* $]a,b[$ *with* $a_n(x)\neq 0$ *in the mentioned interval, then the general solution of equation(3) is given by:*

$$y=C_1\,y_1(x)+C_2\,y_2(x)+\ \ldots\ +C_n\,y_n(x)\ \ ,$$

where $C_1\,,C_2,\ldots,C_n$ *are arbitrary constants.*

Let's observe the importance presented by the linearly independent sets of solutions of equation (3), $\{y_1(x),y_2(x),\ldots,y_n(x)\}$. This brings about that these sets are given a special name.

Definition 2.4 *The n solutions* $\{y_1(x),y_2(x),\ldots,y_n(x)\}$ *to (3) are said to form a* **fundamental system of solutions of equation** *(3) if they are linearly independent over the definition interval of differential equation (3).*

The following concept will enable us to characterise the fundamental sets of definitions in a simple way, without having to resource to definition 2.3.

Definition 2.5 *Let* $\{y_1(x), y_2(x), \ldots, y_n(x)\}$ *be n functions having derivatives up to order n-1 in the interval* $]a,b[$. *The* ***Wronskian of*** $\{y_1(x), y_2(x), \ldots, y_n(x)\}$ *is defined, evaluated in x, and the value of the following determiner is denoted as* $\mathrm{W}(y_1, y_2, \ldots, y_n; x)$*:*

$$\mathrm{W}(y_1, y_2, \ldots, y_n; x) = \begin{vmatrix} y_1(x) & y_2(x) & \ldots & y_n(x) \\ y'_1(x) & y'_2(x) & \ldots & y'_n(x) \\ \vdots & \vdots & \ddots & \vdots \\ y^{n-1)}{}_1(x) & y^{n-1)}{}_2(x) & \ldots & y^{n-1)}{}_n(x) \end{vmatrix}, \tag{5}$$

The way of characterising fundamental sets of solutions is given by the following result:

Theorem 2.4 *The n solutions* $\{y_1(x), y_2(x), \ldots, y_n(x)\}$ *to (3), defined in the interval* $]a,b[$ *are a linearly dependent set, if and only if*

$$\mathrm{W}(y_1, y_2, \ldots, y_n; x) = 0 .$$

Furthermore, if the n solutions $\{y_1(x), y_2(x), \ldots, y_n(x)\}$ *to (3) are continuous in the interval* $]a,b[$ *and the function* $a_n(x) \neq 0$ *, then only one of the following possibilities is verified:*

1. $\mathrm{W}(y_1, y_2, \ldots, y_n; x) \equiv 0 \;,\; \forall x \in]a,b[$,

2. $\mathrm{W}(y_1, y_2, \ldots, y_n; x) \neq 0 \;,\; \forall x \in]a,b[$.

We will call homogeneous linear differential equation of the n^{th} order a differential equation (1) where:

$$a_i(x) = a_i \;,\; a_i \in R \;,\; i = 1,2,\ldots,n \;,\; f(x) = 0 \;,$$

and which is written in this way:

$$a_n\, y^{n)}(x) + a_{n-1}\, y^{n-1)}(x) + \ldots + a_1 y'(x) + a_0 y(x) = 0 . \tag{6}$$

By using the derivative operator *D*, equation (6) results like this:

$$P(D)y = 0, \quad \mathrm{P(D)} = a_n D^{n)} + a_{n-1} D^{n-1)} + \ldots + a_1 D^{1)} + a_0 D , \tag{7}$$

For equation (6), using theorem 2.1 the existence of solutions is guaranteed. Furthermore, the following result will enable us to find solutions of (6) which have an exponential form.

Proposition 2.1 *The differential equation* $P(D)y = 0$ *has function* $y(x) = e^{r_0 x}$ *as solution, for each one of the roots* r_0 *of the polynomial equation:*

$$P(r) = 0. \tag{8}$$

(8) is said to be a **characteristic equation** *of (6).*

Let us distinguish different cases according to whether the roots of (8) are simple or multiple.

Theorem 2.5 *If (8) has n roots different form each other, real or complex,* $r_1, r_2, \ldots, r_n$*, then the functions*

$$\left\{e^{r_1 x}, e^{r_2 x}, \ldots, e^{r_n x}\right\},$$

form a fundamental set of solutions of (6), and therefore the general solution of equation (6) is given by:

$$y = C_1\, e^{r_1 x} + C_2\, e^{r_2 x} + \ldots + C_n\, e^{r_n x} \quad ,$$

being $C_1, C_2, \ldots, C_n$ *arbitrary constants.*

Note 2.1 *Let us observe that if (8) has a complex root* $\alpha + i\beta$*, it also has its conjugate* $\alpha - i\beta$ *as a root. Therefore, in the general solution of (6) obtained in theorem 2.5 a term of the following form must appear:*

$$C_1\, e^{(\alpha+i\beta)x} + C_2\, e^{(\alpha-i\beta)x} \quad , \tag{9}$$

with C_1, C_2 *arbitrary constants. If we are interested in the real solutions, we can used the identity*

$$e^{(a+ib)} = e^{a}\left(\cos(b) + i\sin(b)\right) \ , \tag{10}$$

to write (9) as follows:

$$C_1\, e^{(\alpha+i\beta)x} + C_2\, e^{(\alpha-i\beta)x} = e^{\alpha x}\left(A\cos(\beta\ x) + B\sin(\beta\, x)\right) \tag{11}$$

where $A = C_1 + C_2$ *and* $B = i(C_1 - C_2)$.

Theorem 2.6 *If* $r = r_0$ *is a root of (8) with m order of multiplicity, then m solutions linearly independent associated to* r_0 *exist,*

$$\left\{e^{r_0 x},\ xe^{r_0 x},\ x^2 e^{r_0 x}, \ldots, x^{m-1} e^{r_0 x}\right\}.$$

Note 2.2 *If (8) has a complex number* $\alpha + i\beta$ *with m multiplicity as a root, then* $\alpha - i\beta$ *is also a root with m multiplicity. Applying theorem 2.6 to the 2m solutions corresponding to these two roots and working as stated in note 2.1, it can be proved that*

$$\left\{e^{\alpha x}\cos(\beta\ x),\ xe^{\alpha x}\cos(\beta\ x),\ x^2 e^{\alpha x}\cos(\beta\ x), \ldots, x^{m-1} e^{\alpha x}\cos(\beta\ x),\right.$$
$$\left. e^{\alpha x}\sin(\beta\ x),\ xe^{\alpha x}\sin(\beta\ x),\ x^2 e^{\alpha x}\sin(\beta\ x), \ldots, x^{m-1} e^{\alpha x}\sin(\beta\ x)\right\}, \tag{12}$$

is a set of 2m independent solutions of differential equation (6).

3. Solution of the homogeneous linear differential equation of n^{th} order with constant coefficients with DERIVE. Implementation. Examples. CONST.MTH

```
KRONECKER_VECTOR(s) := VECTOR(KRONECKER(s , 0), i, 1, DIM(s))
                                         i

VECTOR_AUXILIAR_1(s, i) :=
  If i = 1
     1
     If s↓i ≠ 0
        If s↓(i - 1) ≠ 0
           1
           0
        0
```

$$AUX\_0(v) := VECTOR(x^{DIM(v) - k + 1}, k, 2, DIM(v) + 1)$$

$$AUX\_1(v) := v \cdot AUX\_0(v)$$

$$AUX\_3(v) := SOLUTIONS(AUX\_1(v), x)$$

$$SEPARA\_RAIZ\_1(v) := VECTOR(IF(IM((AUX\_3(v))_i) \neq 0, (AUX\_3(v))_i, 0), i, DIM(AUX\_3(v)))$$

$$SEPARA\_RAIZ\_2(v) := \Sigma(KRONECKER\_VECTOR(SEPARA\_RAIZ\_1(v)))$$

$$AUX\_4(v, i) := VECTOR\left(\lim_{x \to (AUX\_3(v))\downarrow i} \left(\frac{d}{dx}\right)^j AUX\_1(v), j, 0, DIM(v) - 1\right)$$

$$AUX\_5\_1(v, i) := KRONECKER\_VECTOR(AUX\_4(v, i))$$

$$AUX\_5(v, i) := VECTOR\_AUXILIAR\_2(AUX\_5\_1(v, i))$$

$$AUX\_REAL\_1(v, i) := VECTOR\left(c_{i,k} \cdot x^{k - 1} \cdot e^{(AUX\_3(v))_i \cdot x}, k, 1, DIM(v)\right)$$

$$AUX\_REAL\_2(v, i) := AUX\_REAL\_1(v, i) \cdot AUX\_5(v, i)$$

$$AUX\_6(v) := \sum_{i=1}^{SEPARA\_RAIZ\_2(v)} AUX\_REAL\_2(v, i)$$

$$AUX\_COMPLEJA\_1(v, i) := VECTOR\left(d_{i,k} \cdot x^{k - 1} \cdot RE\left(e^{(AUX\_3(v))_i \cdot x}\right), k, 1, DIM(v)\right)$$

$$AUX\_COMPLEJA\_2(v, i) := VECTOR\left(r_{i,k} \cdot x^{k - 1} \cdot SIGN(IM((AUX\_3(v))_i)) \cdot IM\left(e^{(AUX\_3(v))_i \cdot x}\right), k, 1, DIM(v)\right)$$

$$AUX\_COMPLEJA\_3(v, i) := AUX\_COMPLEJA\_1(v, i) + AUX\_COMPLEJA\_2(v, i)$$

$$AUX\_COMPLEJA\_4(v, i) := AUX\_COMPLEJA\_3(v, i) \cdot AUX\_5(v, i)$$

$$AUX\_7(v) := \Sigma(VECTOR(AUX\_COMPLEJA\_4(v, i), i, SEPARA\_RAIZ\_2(v) + 1, DIM(AUX\_3(v)), 2))$$

$$DIFERENCIAL\_HOMOGENEA\_COEFICIENTES\_CONSTANTES(v, x) := EXPAND(AUX\_6(v) + AUX\_7(v))$$

$$V\_C\_AUX\_2(v, c, i, j) := \left(\frac{d}{dx}\right)^j c_i$$

$$V\_C\_AUX\_3(v, c) := VECTOR\left(\left(\frac{d}{dx}\right)^j c, j, 0, DIM(v) - 2\right)$$

$$V\_C\_AUX\_4(v, c) := V\_C\_AUX\_3(v, c)^{-1}$$

$$V\_C\_AUX\_5(v, c, f) := VECTOR\left(\frac{f}{v_1} \cdot KRONECKER(j, DIM(v) - 2), j, 0, DIM(v) - 2\right)$$

$$V\_C\_AUX\_6(v, c, f) := V\_C\_AUX\_4(v, c) \cdot V\_C\_AUX\_5(v, c, f)$$

$$V\_C\_AUX\_7(v, c, f) := \int V\_C\_AUX\_6(v, c, f)\, dx$$

```
V_C_AUX_8(v, c, f) := VECTOR(1, j, 0, DIM(v) - 2)

V_C_AUX_9(v, c, f) := V_C_AUX_7(v, c, f)·c

SOLUCION_PARTICULAR_COMPLETA(v, c, f) := V_C_AUX_9(v, c, f)
```

Given the homogeneous differential equation of the *n*th order with constant coefficients

$$a_n\, y^{n)}(x) + a_{n-1}\, y^{n-1)}(x) \;+\ldots+\; a_1 y'(x) + a_0 y(x) = 0\,,$$

we will introduce in the vector $v = [a_n, a_{n-1}, \ldots, a_0]$ the coefficients of this equation, written in serial order, tapering downward. The CONST.MTH file functions operate in the following way:

1. Function $AUX\_0(v)$ creates vector $\left[x^n, x^{n-1}, \ldots, x, 1\right]$ with the same dimension as *v*.

2. Function $AUX\_1(v)$ starting from v and $AUX\_0(v)$ provides us with the characteristic polynomial of equation (6).

3. Function $AUX\_2(v)$ calculates the roots of the characteristic polynomial (let us remember that **Derive** does not directly provide us with the multiplicity of a roots of a polynomial). Thus $AUX\_2(v)$ gives a vector of the form $[x = r_1, x = r_2, \ldots, x = r_s]$, where $\{r_1, r_2, \ldots, r_s\}$ are the roots of the characteristic polynomial, real or complex, each one with a multiplicity $\{l_1, l_2, \ldots, l_s\}$ and being verified $\sum_{i=1}^{s} l_i = n$.

4. Function $AUX\_3(v)$ provides us with the list of roots calculated by the function $AUX\_2(v)$, giving a vector of the form $[r_1, r_2, \ldots, r_s]$.

5. Functions *SEPARA_RAIZ_1(v)* and *SEPARA_RAIZ_2(v)* separate the real and complex roots of the characteristic polynomial given by *AUX_3(v)*.

6. Function *AUX_4(v,i)* , for every root of the characteristic polynomial r_i, give us a vector with the values of the characteristic polynomial and its n – 1 first derivatives of that root. Thus it produces a vector of the form $[b_0, b_1, \ldots, b_{n-1}]$, where

$$b_j = \left.\frac{d^j P(x)}{dx^j}\right|_{x=r_i}\,, \quad j=0,1,\ldots,n-1\,.$$

It is evident that when a root has multiplicity *s*, it will give rise to a vector of the form

$$\left[\overbrace{0,\ldots,0}^{s+1}, b_{s+2}, \ldots, b_{n-1}\right].$$

7. Starting from a vector $\left[\overbrace{0,\ldots,0}^{s+1}, b_{s+2}, \ldots, b_{n-1}\right]$, obtained by means of the function $AUX\_4(v,i)$, the function $AUX\_5\_1(v,i)$ provides us with a vector of the form:

$$\left[\overbrace{1,\ldots,1}^{s+1}, 0, \ldots, 0\right],$$

for each root r_i of the characteristic polynomial. For that purpose it uses functions *KRONECKER_VECTOR* (s), *VECTOR_AUXILIAR*_1(*s,i*) and *VECTOR_AUXILIAR*_2(*s,i*).

8. Function $AUX\_5(v,i)$ filters function $AUX\_5\_1(v,i)$ so as to avoid anomalies such as the one in the following example:

Example 3.1 *Let us consider the polynomial* $P(x)=x^4+x^2$. *This polynomial verifies*

$$\left.\begin{array}{lcl} P(x)=x^4+x^2 & \Rightarrow & P(0)=0 \\ P'(x)=4x^3+2x & \Rightarrow & P'(0)=0 \\ P''(x)=12x^2+2 & \Rightarrow & P''(0)=2 \\ P'''(x)=24x & \Rightarrow & P'''(0)=0 \\ P^{4)}(x)=24 & \Rightarrow & P^{4)}(0)=24 \end{array}\right\} \Rightarrow x=0 \text{ is a double root of } P(x),$$

Function AUX_4(v,i) for $v=[1,0,1,0,0]$ *provides as a result vector* $[0,0,2,0,24]$ *and function AUX_5_1(v,i) provides vector* $[1,1,0,1,0]$. *Thus strange solutions would be introduced to differential equation (6). Function AUX_5 (v,i) permits us, for this concrete example, to obtain vector* $[1,1,0,0,0]$ *as an answer, which interprets the root* $x=0$ *as a double root of this polynomial.*

9. Functions $AUX\_REAL\_1(v,i)$ and $AUX\_REAL\_2(v,i)$ work with the real roots, operating with each one of them so that they appear with its corresponding multiplicity.

10. Function $AUX\_6(v,i)$ gives us the partial solution of the differential equation obtained of the real roots.

11. Functions $AUX\_COMPLEJA\_1(v,i)$, $AUX\_COMPLEJA\_2(v,i)$, $AUX\_COMPLEJA\_3(v,i)$ and $AUX\_COMPLEJA\_4(v,i)$ operate on the complex roots r_i of the characteristic polynomial. Basically they operate their real and imaginary parts separately, and prepare them so that when they are added in pairs, as indicated in notes 2.1 and 2.2, the imaginary numbers disappear and the root appears with its corresponding multiplicity.

12. Function $AUX\_7(v)$ adds all the partial results obtained when working with the real and complex roots separately, providing us with the solution of the differential equation.

Examples:

$$y'''(x)+3\,y''(x)\ \text{-}4y'(x)-12y(x)=0 \qquad . \qquad (13)$$

In this case vector $v=[1,3,-4,-12]$, and we have:

```
DIFERENCIAL_HOMOGENEA_COEFICIENTES_CONSTANTES([1, 3, -4, -12], x)
```

$$e^{2\cdot x}\cdot c_{1,1}+e^{-2\cdot x}\cdot c_{2,1}+e^{-3\cdot x}\cdot c_{3,1}$$

Let us check that the solution obtained is actually a solution of (13).

$$y(x) := e^{2\cdot x}\cdot c_{1,1}+e^{-2\cdot x}\cdot c_{2,1}+e^{-3\cdot x}\cdot c_{3,1}$$

$$\left(\frac{d}{dx}\right)^3 y(x) + 3\cdot\left(\frac{d}{dx}\right)^2 y(x) - 4\cdot\frac{d}{dx} y(x) - 12\cdot y(x) = 0$$

The next two examples will be solved by applying CONST.MTH and by *MATHEMATICA*:

$$y^{(4)} - 7y'' - 18y = 0,$$
$$y^{(4)} + y'' = 0$$

```
DIFERENCIAL_HOMOGENEA_COEFICIENTES_CONSTANTES([1, 0, -7, 0, -18], x)
```

$$e^{3\cdot x}\cdot c_{1,1} + e^{-3\cdot x}\cdot c_{2,1} + \cos(\sqrt{2}\cdot x)\cdot d_{3,1} + \sin(\sqrt{2}\cdot x)\cdot r_{3,1}$$

```
DIFERENCIAL_HOMOGENEA_COEFICIENTES_CONSTANTES([1, 0, 1, 0, 0], x)
```

$$\cos(x)\cdot d_{2,1} + \sin(x)\cdot r_{2,1} + x\cdot c_{1,2} + c_{1,1}$$

In[1]:= `DSolve[y''''[x] - 7 y''[x] - 18 *y[x] == 0, y, x]`

Out[1]= $\{\{y \to \text{Function}[\{x\}, e^{-3x} c_3 + e^{3x} c_4 + c_1 \cos(\sqrt{2}\, x) + c_2 \sin(\sqrt{2}\, x)]\}\}$

In[2]:= `DSolve[y''''[x] +y''[x] == 0, y, x]`

Out[2]= $\{\{y \to \text{Function}[\{x\}, c_3 + x\, c_4 - c_1 \cos(x) - c_2 \sin(x)]\}\}$

Finally two more examples of order 5:

$$y^{(5)} + \frac{6}{5}y^{(4)} + \frac{237}{25}y''' + \frac{1358}{15}y'' + \frac{108}{25}y' + \frac{72}{125}y = 0$$

$$y^{(5)} - 3y''' + 4y' = 0$$

DERIVE calculation + check of the solution:

$$\text{DIFERENCIAL\_HOMOGENEA\_COEFICIENTES\_CONSTANTES}\left(\left[1, \frac{6}{5}, \frac{237}{25}, \frac{1358}{125}, \frac{108}{25}, \frac{72}{125}\right], x\right)$$

$$x^2\cdot e^{-2\cdot x/5}\cdot c_{1,3} + x\cdot e^{-2\cdot x/5}\cdot c_{1,2} + e^{-2\cdot x/5}\cdot c_{1,1} + \cos(3\cdot x)\cdot d_{2,1} + \sin(3\cdot x)\cdot r_{2,1}$$

$$y(x) := x^2\cdot e^{-2\cdot x/5}\cdot c_{1,3} + x\cdot e^{-2\cdot x/5}\cdot c_{1,2} + e^{-2\cdot x/5}\cdot c_{1,1} + \cos(3\cdot x)\cdot d_{2,1} + \sin(3\cdot x)\cdot r_{2,1}$$

$$y'''''(x) + \frac{6\cdot y''''(x)}{5} + \frac{237\cdot y'''(x)}{25} + \frac{1358\cdot y''(x)}{125} + \frac{108\cdot y'(x)}{25} + \frac{72\cdot y(x)}{125} = 0$$

```
DIFERENCIAL_HOMOGENEA_COEFICIENTES_CONSTANTES([1, 0, -3, 0, 4, 0], x)
```

$$e^{\sqrt{7}\cdot x/2}\cdot d_{2,1}\cdot \cos\left(\frac{x}{2}\right) + e^{\sqrt{7}\cdot x/2}\cdot r_{2,1}\cdot \sin\left(\frac{x}{2}\right) + e^{-\sqrt{7}\cdot x/2}\cdot d_{4,1}\cdot \cos\left(\frac{x}{2}\right) + e^{-\sqrt{7}\cdot x/2}\cdot r_{4,1}\cdot \sin\left(\frac{x}{2}\right) + c_{1,1}$$

$$y(x) := e^{\sqrt{7}\cdot x/2}\cdot d_{2,1}\cdot \cos\left(\frac{x}{2}\right) + e^{\sqrt{7}\cdot x/2}\cdot r_{2,1}\cdot \sin\left(\frac{x}{2}\right) + e^{-\sqrt{7}\cdot x/2}\cdot d_{4,1}\cdot \cos\left(\frac{x}{2}\right) + e^{-\sqrt{7}\cdot x/2}\cdot r_{4,1}\cdot \sin\left(\frac{x}{2}\right) + c_{1,1}$$

$$y'''''(x) - 3\cdot y'''(x) + 4\cdot y'(x) = 0$$

[1] T.M.Flett, *Differential Analysis*. New York. Cambridge University Press. 1980

THE RIEMANN INTEGRAL WITH THE TI-92

Francisco Jose Santonja Gómez
C.U. ESTEMA-ANTONIO DE NEBRIJA UNIVERSITY. Valencia. Spain

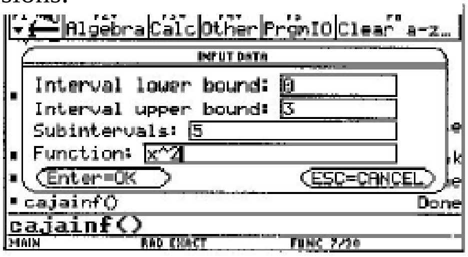
In this paper some TI-programs and functions concerning integration are given.

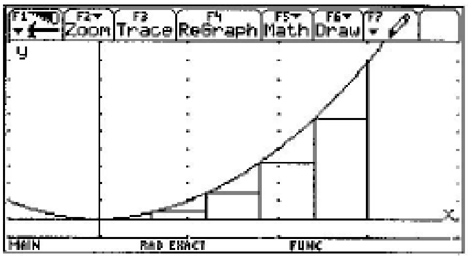
Description of the TI-programs

Firstly the *Lower Riemann Sum is* shown by cajainf().

Let $f(x) = x^2$ on [0,3] with 5 subdivisions.

```
:cajainf()
:Prgm
:setMode("Exact/Approx","EXACT")
:Dialog
:Title "INPUT DATA"
:Request "Interval.lower bound",a
:Request "Interval.upper bound",b
:Request "Subintervals",n
:Request "function",f
:EndDlog
:expr(a)→a:expr(b)→b:expr(n)→n:expr(f)→f
:ClrDraw
:For i,0,n,1
:PtOn a+(b-a)*i/n,0
:EndFor
:limit(f,x,m)→valor(f,m)
:For i,0,n,1
:PtOn a+(b-a)*i/n,valor(f,a+(b-a)*i/n)
:EndFor
:For i,0,n,1
:Line a+(b-a)*i/n,0,a+(b-a)*i/n,valor(f,a+(b-a)*i/n)
:EndFor
:For i,1,n,1
:Line a+(b-a)*(i-1)/n,valor(f,a+(b-a)*(i-1)/n),a+(b-a)*i/n,
 valor(f,a+(b-a)*(i-1)/n)
:EndFor
:EndPrgm
```

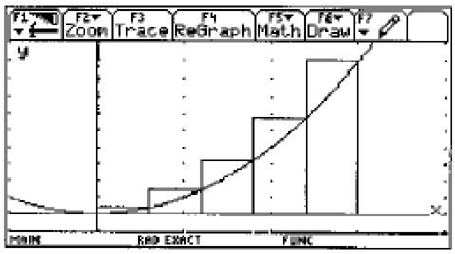


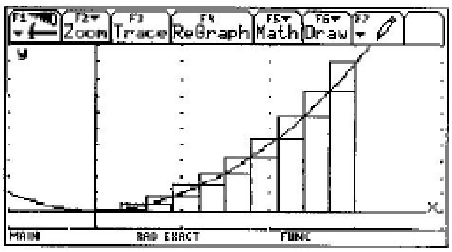


Next, the program cajasup() has been defined to present the *Upper Riemann Sum.*

You only have to replace the marked program block of cajainf() by the block given below.

```
:cajasup()
:Prgm
.......
:For i,0,n-1,1
:PtOn a+(b-a)*i/n,valor(f,a+(b-a)*(i-1)/n)
:EndFor
:limit(f,x,m)→valor(f,m)
:For i,0,n-1,1
:Line a+(b-a)*i/n,0,a+(b-a)*i/n,
 valor(f,a+(b-a)*(i+1)/n)
:EndFor
:Line b,0,b,valor(f,b)
:For i,0,n-1
:Line a+(b-a)*i/n,valor(f,a+(b-a)*(i+1)/n),
 a+(b-a)*(i+1)/n, valor(f,a+(b-a)*(i+1)/n)
:EndFor
```

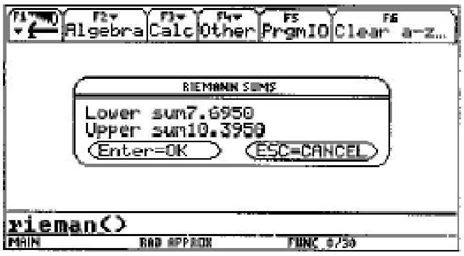




Combining cajainf() with cajasup() gives supinf(), which presents both sums (showing above 10 subdivisions).

Next program rieman() has been defined to calculate both *Riemann Sums*.

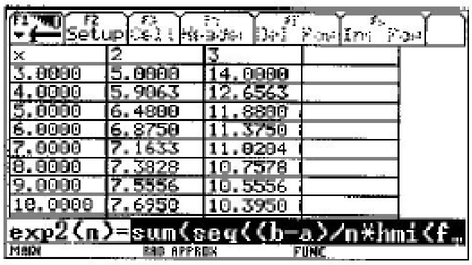
```
:rieman()
:Prgm
:Dialog
:Title "INPUT DATA"
:Request "Interval.lower bound",a
:Request "Interval.upper bound",b
:Request "Subintervals",n
:Request "function",f
:EndDlog
:expr(a)→a:expr(b)→b:expr(n)→n:expr(f)→f
:limit(f,x,m)→valor(f,m)
:max(valor(f,a),valor(f,b))→hma(f,a,b)
:min(valor(f,a),valor(f,b))→hmi(f,a,b)
:sum(seq((b-a)/n*(hmi(f,a+(b-a)*(i-1)/n,a+(b-a)*i/n)),i,1,n))→infsuma
:setMode("Exact/Approx","APPROXIMATE")
:sum(seq((b-a)/n*(hma(f,a+(b-a)*(i-1)/n,a+(b-a)*i/n)),i,1,n))→supsuma
:Dialog
:Title "RIEMANN SUMS"
:Text "Lower sum"&string(infsuma)
:Text "Upper sum"&string(supsuma)
:EndDlog
:EndPrgm
```



$n = 10$

Finally program tableta() shows the integration as the limit of the sums (lower and upper sum) converging to the same value.

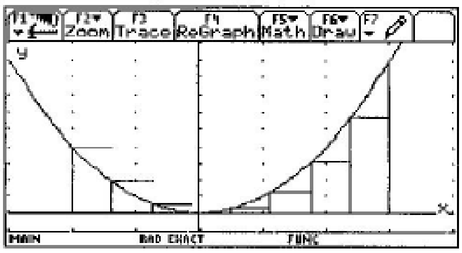
```
:tableta()
:Prgm
:ClrTable
:ClrIO
:setMode("Exact/Approx","APPROXIMATE")
:Dialog
:Title "INPUT DATA"
:Request "interval.lower bound",a
:Request "interval.upper bound",b
:Request "function",f
:EndDlog
:expr(a)→a:expr(b)→b:expr(f)→f
:Table sum(seq((b-a)/n*(hmi(f,a+(b-a)*(i-1)/n,a+(b-a)*i/n)),i,1,n)),n
:Table sum(seq((b-a)/n*(hma(f,a+(b-a)*(i-1)/n,a+(b-a)*i/n)),i,1,n)),n
:EndPrgm
```



References:

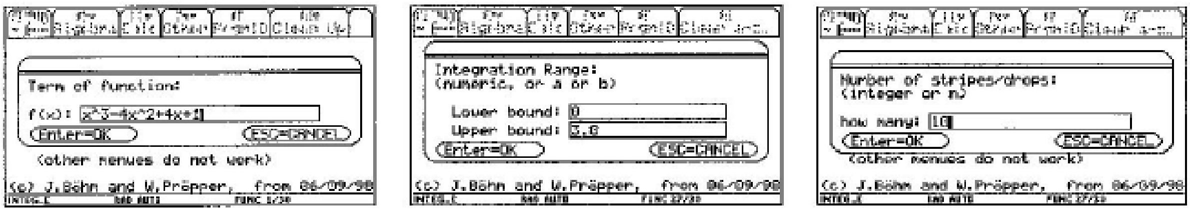
Llorens Fuster, *Una Lección de Matematicas con Ordenador,* EPSILON. n.31-32, p. 81-88. 1995.
Llorens Fuster, *Aplicaciones de DERIVE al Análisis Matemático* 1, Polytechnic University ofValencia Press.

When I received F Santonjas files I immediately tried them because some times ago - at the occasion of the legendary Krems Conference 1 in 1992 - I produced a DERIVE package on the same topic. (Happy memories on Terence Etchells´ pre email - correspondence and diskette exchanging by snail mail). I found the TI-programs useful but had the idea that they could be more userfriendly. Many settings must be set from outside of the program: graph the function, set the [WINDOW] *values, set the* [TblSet] *for the last program.* tableta() *does not work without running* rieman() *before. And look what is happening when using a non increasing function:*

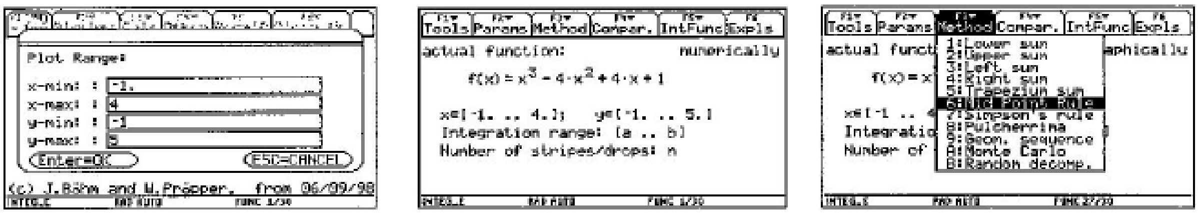


These packages could serve excellent as a starting point for students who want to improve their ideas to represent and to calculate Riemann sums. Step by step they could try to include additional features to the core of the programs and at last try to combine them.

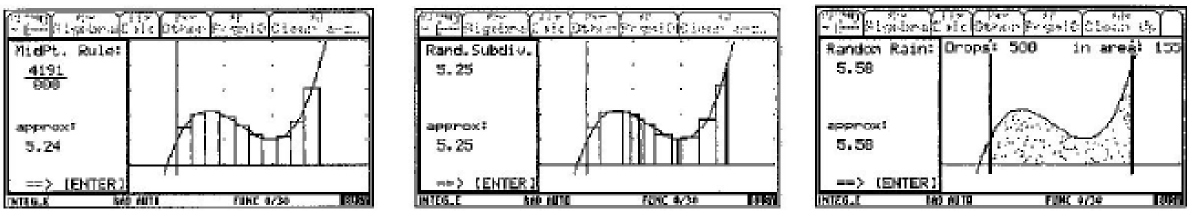
For me this contribution was challenge enough to remember my DERIVE deeds from 1992 and I produced a TI-92 package: :integ(). My friend Wolfgang Pröpper saw my efforts and very similar to me he again found many ways to make this package more user friendly and he added some important features. Encouraged by Bernhard Kutzler we produced a book "The Riemann Integral - From Counting Raindrops to the Fundamental Theorem". I'd like to add some screen shots to show where F. Santonja's inspiring ideas led Wolfgang and Josef:



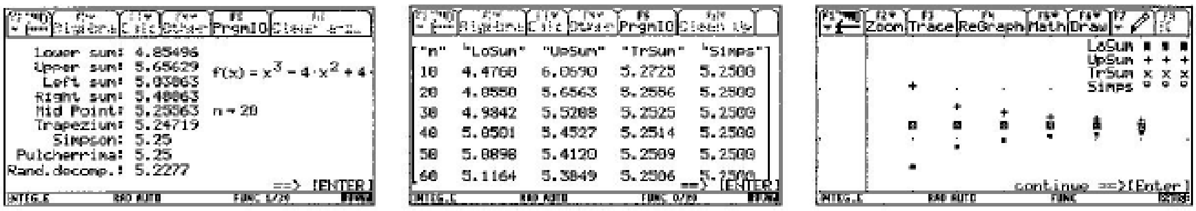
Enter all data using dialogs:



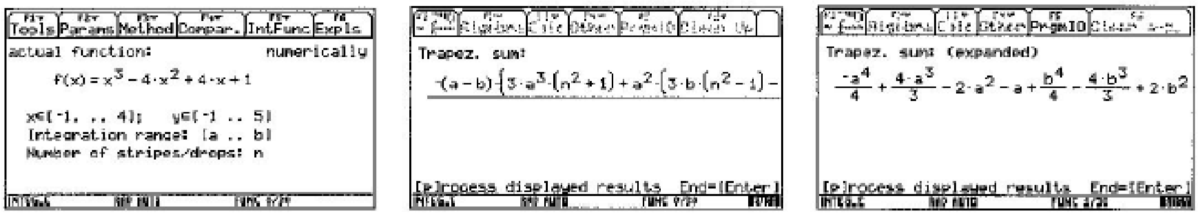
Treat the given function in several ways: numerically;



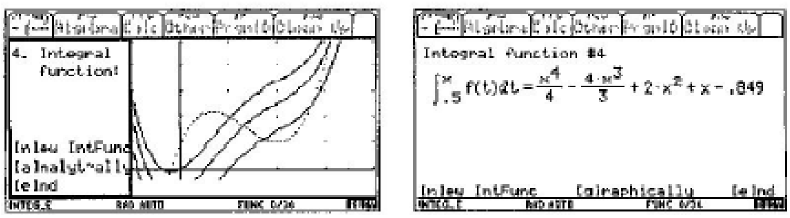
Show the graphic representation of the methods:



Compare the convergence by inspection:



Work symbolically:

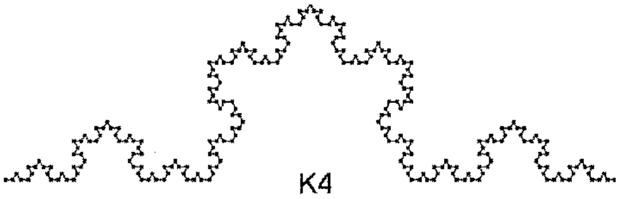


Discuss the integral functions: (… and much more.)

Helge von Koch' s Snowflake

Matija Lokar, Faculty of mathematics and physics, University of Ljubljana, Slovenia
Matija.Lokar@fmf.uni-lj.si

As at the time being it is snowing, let's try to draw some snowflakes with *DERIVE,* too. As basis we take the so called Koch curve. It belongs to the class of recursive defined objects. We get them with applying a certain procedure to a base object. Base object, which can be line, box, circle, triangle or something else, is transformed into severa1 simi1ar objects which serve as the starting point of thea same transformation. We know Peano curve, Dragon curve, Hilbert curve, Sierpinski triangle and several others.


K4

We start with line (Koch curve of order 0). The basis procedure is as follows. We divide the line into three parts. The middle part is substituted with two lines that form en equilateral triangle with deleted line. We get the Koch curve of the 1st order. If we apply the procedure on all four segments then we will get the Koch curve of 2nd order.
An interesting paper about Koch curve can be found on Internet on page

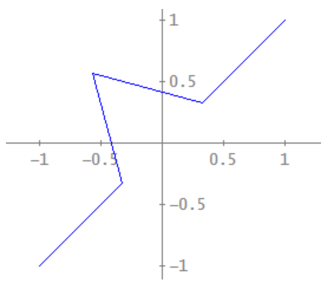
`http://www.eurologo. org/papers/logomov. html`[1].

There LOGO is used. But we would like to draw with *DERIVE.* Let us first recall some basis facts about drawing with *DERIVE.*
A point in *DERIVE* is represented as pair of coordinates in square brackets, divided by a comma. So the origin is represented as [0,0] and a point with co ordinate $x = 1$ and coordinate $y = 2$ as [1,2]. If we plot such an expression ([1,2] for example), we get point (1,2). If we would like to draw several points, we make a vector consisting of two-dimensiona1 vectors, namely an n x 2 matrix, where n is number of points. When we draw several points, *DERIVE* can draw a line which connects points. When we are in 2D-Plot Windows, we choose `Options State` and set `Mode` to `Connected` or `Discrete`. We can also set the size of points. When set to `Small`, point can not be distinguished from the joining line in `Connected mode`.

Now to our Koch curve. First we calculate three new points that our procedure makes. Each one will be calculated with a function. The parameters for all three functions are the starting and ending point of the line. We exploit the fact that we can calculate with points like with vectors.

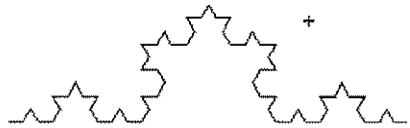
```
TT(a,b):=a+(b-a)/3
TDT(a,b):=TT(b,a)
TO(a,b):=(a+b)/2+SQRT(3)/6*[a SUB 2-b SUB 2,b SUB 1-a SUB 1]
```

Let us check first our attempt. Set `u` as a starting point ([-1,-1]) and `v` as an ending point ([1,1]). We have to draw a polygon 1ine `[u,TT(u,v),TO(u,v),TDT(u,V),v]`.



It seems OK. Now we use the recursion. If the order of curve is 0, then we have a 1ine. Otherwise we have four curves of one 1ess degree on: `u-TT(u,v)`; `TT(u,v)-TO(u,v)`; `TO(u,v)-TDT(u,v)` and finally `TDT(u,v)-v`.

We edit and plot `KOCH([-2,0],[2,0],3]`. Simplifying we get a huge expression. Before plotting we turn off automatic color cycling and hide the axes (all in `Options Display`).



But if we observe the received expression, we are not quite satisfied. First there is no need to use exact arithmetic, so we set `Precision := Approximate PrecisionDigits := 2`. This also significantly speeds up the computation. Also on each step we end up with four polygon lines. But we should get only one. So we change function `KOCH` a little:

```
KOCH(a,b,n):= If n = 0
                 [a,b]
                 APPEND(KOCH(a,TT(a,b),n-1),KOCH(TT(a,b),TO(a,b),n-1),
                 KOCH(TO(a,b),TDT(a,b),n-1),KOCH(TDT(a,b),b,n-1))
```

Another improvement is due to observation that we get all points twice – first as an ending point and secondly as a starting point. If we just drop the second point in case of order 0 then we circumvent this obstacle:

```
KOCH(a,b,n):= If n = 0
                 [a]
                 APPEND(KOCH(a,TT(a,b),n-1),KOCH(TT(a,b),TO(a,b),n-1),
                 KOCH(TO(a,b),TDT(a,b),n-1),KOCH(TDT(a,b),b,n-1))
```

Now our function does not behave "properly" for $n = 0$ and we lost the first point of the sequence. So we define:

```
KOCH_CURVE(a,b,n):= IF(n=0,[a,b],APPEND([a],KOCH(a,b,n)))
```

If we draw three Koch curves based on the sides of an equilateral triangle the we get the so called *Koch Snowflake*. First let us define `POLY(n)` which produces the vertices of a regular polygon of order n.

```
POLY(n):= VECTOR([3·COS(i),3·SIN(i)],i,2π,0,-2π/n)
```

Then we define `KS(a,m)` which creates a collection of KOCH curves of order `m` between consecutive points of a vector a.

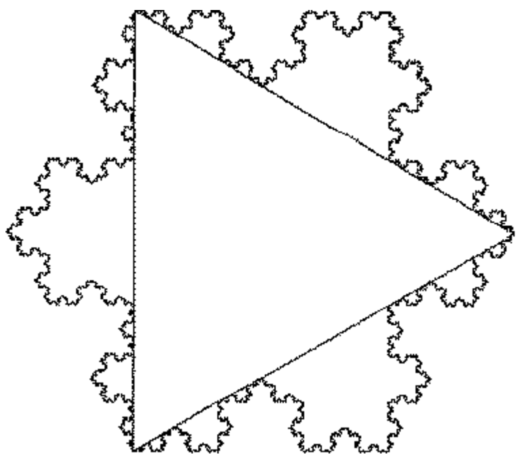
```
KS(a, m):= VECTOR(KOCH(a↓i,a↓(i+1),m),i,1,DIMENSION(a)-1)
```

Simplifying `KS(POLY(3),5)` gives the Snowflake. We see it together with an accompanying triangle.

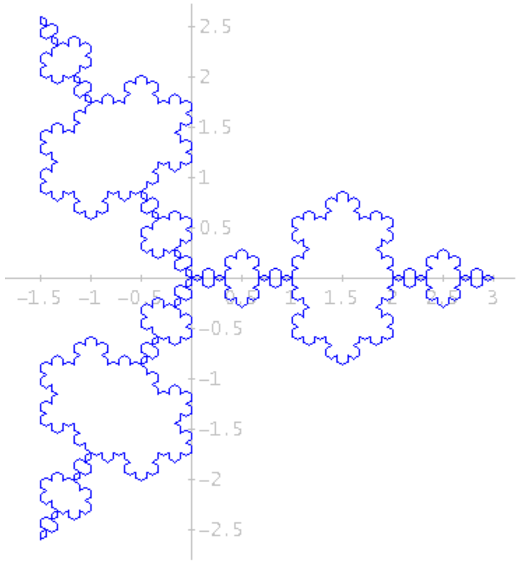
It should be also noted that the order n in which we compute the polygon is important. If we use points computed with

```
POLYB(n):=
VECTOR([3COS(i),3SIN(i)],i,0,2·π,2π/n)
```

then we get Koch curves which are inside the triangle.

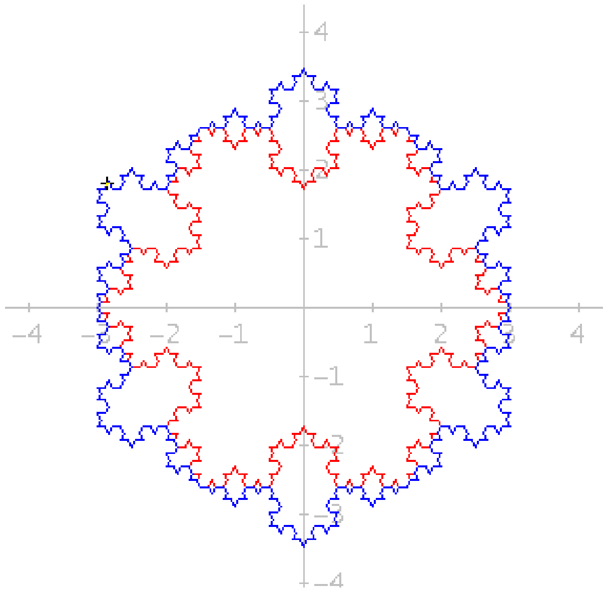


KS(POLYB(3), 4)



KS(POLY(6), 3)

KS(POLYB(6), 3)



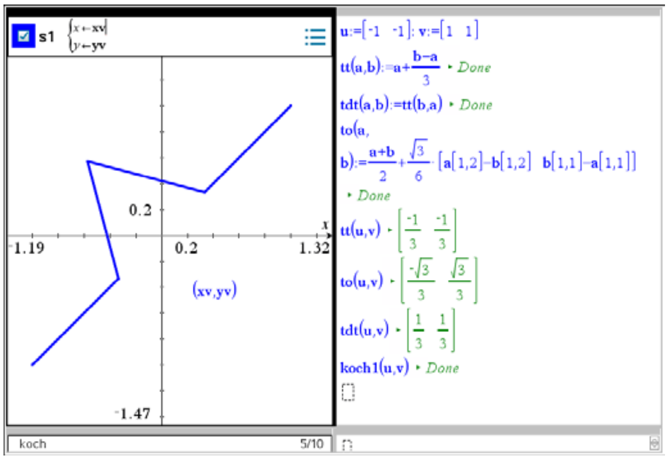
The snowflake is always a challenge to be faced. I think that it may be nice to compare the various ways producing this famous fractal structure. You can do this in this DNL studying the next contribution. Compare also with the Roanes-Lechner-Wiesenbauer implementation of *LOGO in DERIVE (in DNL#25, March 1997) and with Robert Setif' s Treasure Box (DNL#3 and DNL#6 from 1991/92). Josef*

Another comment from 2014 concerning "challenge": I tried to reproduce the recursive procedure given above with TI-NspireCAS. The first steps – were not so difficult – even missing the programming tools for graphics included in the V200 programming language.

But I could not find a way to transfer Matija's recursive creation of the Snowflake. It would be great if any Nspire expert among you would accept the challenge and follow Matija's ideas from 1999 with the nowadays TI-NspireCAS, Josef

This is what I did so far:

```
Define koch(a,b,n)=
Prgm
Local k
If n=0 Then
    k1:=colAugment(a,b)
Else
    k1:=koch(colAugment(k1,tt(a,b)),n−1)
    k1:=koch(colAugment(k1,to(a,b)),n−1)
    k1:=koch(colAugment(k1,tdt(a,b)),n−1)
    k1:=koch(colAugment(k1,b),n−1)
EndIf
k1
EndPrgm
```



[1] Now this website can be accessed at https://eurologo.web.elte.hu/lectures/logomov.htm

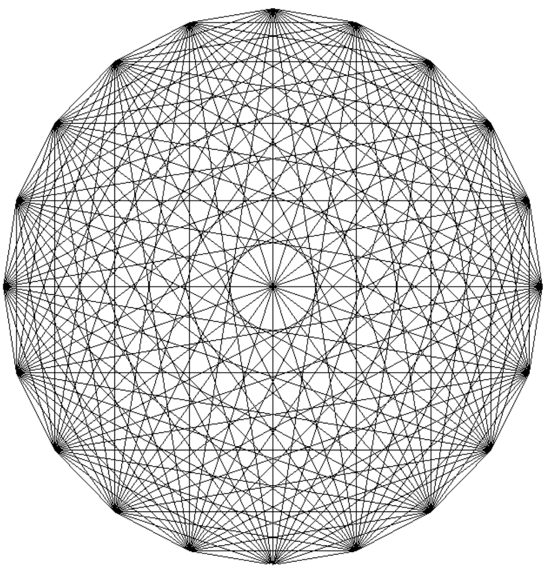
Computergrafik mit *DERIVE* - Computer Graphics *with DERIVE*

Dr. Maria Koth, University of Vienna, Austria

2.2 All diagonals of a regular *n*-gon

A nice graph is created connecting each vertex of an *n*-gon with each of the remaining vertices:
The radius of the circum circle is 1. The line connecting the i^{th} vertex `[COS(i),SIN(i)]` with the j^{th} one `[COS(j),SIN(j)]` is described in *DERIVE* by a list

```
[[COS(i),SIN(i)], [COS(j),SIN(j)]]
```



To avoid plotting lines twice it is sufficient to connect the i^{th} vertex with vertices # (i+1), (i+2), ...*n*. Function `LINIEN(i,n)` creates all the lines starting from point *i*.

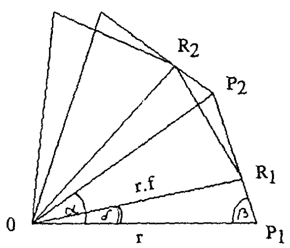
`DIAGON(n)` produces all diagonals by running *i* through all *n* vertices of the *n*-gon.

You can see the result of `DIAGON(20)`.

```
LINIEN(i,n):= VECTOR([[COS(2*π*i/n),SIN(2*π*i/n)],[COS(j),SIN(j)]],j,
              2*π*(i+1)/n,2*π,2*π/n)
DIAGON(n):=VECTOR(LINIEN(i,n),i,n-1)
```

2.3 A family of rotated *n*-gons

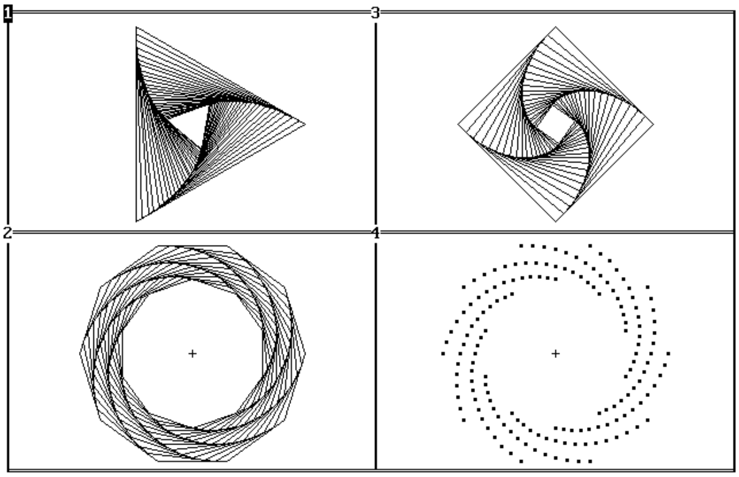
These graphs show families of polygons which are produced by rotating a polygon by a constant angle δ, eg. $\delta = 5°$ and simultaneously shrinking it in such a way that the new polygon is exactly inscribed the preceding one. At first one has to find the shrink factor f:



Given is sector OP_1P_2 of a regular *n*-gon with radius *r*. Its angle α in the center is $\alpha = \angle P_1OP_2 = 2\pi/n$. Angle ß is given by $\beta = (180° - \alpha)/2 =$ $= \pi/2 - \pi/n$. The rotated triangle OR_1R_2 is part of the next polygon with a circumscribed circle of radius $f \cdot r = OR_1$. We apply the sine rule in order to find OR_1:

$$\frac{r\,f}{\sin\beta} = \frac{r}{\sin(180° - \beta - \delta)} \;\rightarrow\; f = \frac{\sin\beta}{\sin(180° - \beta - \delta)} = \frac{\sin(\pi/2 - \pi/n)}{\sin(\pi/2 + \pi/n - \delta)}$$

Using $\sin(90° - \alpha) = \cos\alpha$ gives f:



$$F(n,\delta) := \frac{\cos\left(\frac{\pi}{n}\right)}{\cos\left(\delta - \frac{\pi}{n}\right)}$$

Setting $r = 1$ gives the radius of the k^{th} polygon's circumscribed circle with f^k. This polygon is turned by the angle $k\cdot\delta$.

`ROTECK` returns its vertices.
`ROTSCHAR(n,δ,k)` produces a family of *z* rotated *n*-gons.

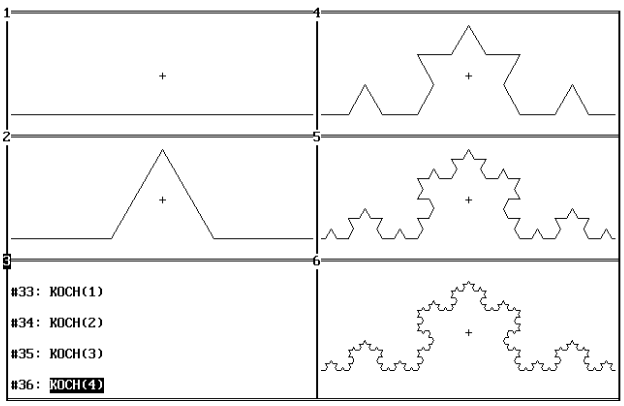
```
ROTECK(n,δ,k):= VECTOR([F(n, δ)^k·COS(i+k·δ),F(n,δ)^k·SIN(i+k·δ)],
                i,0,2·π ,2·π /n)

ROTSCHAR(n,δ,z)≔VECTOR(ROTECK(n,δ,k),k,0,z)

ROTSCHAR(4, 5°, 20)

ROTSCHAR(10 ,30°,20)

ROTSCHAR(3, 2·°,25)
```

3. The Koch Curve

The Koch curve is a well known example of a fractal. It was defined in 1904 by the Danish mathematician Helge von Koch. Koch had the intention to show a "Monster curve", which is continuous on each place but nowhere differentiable.

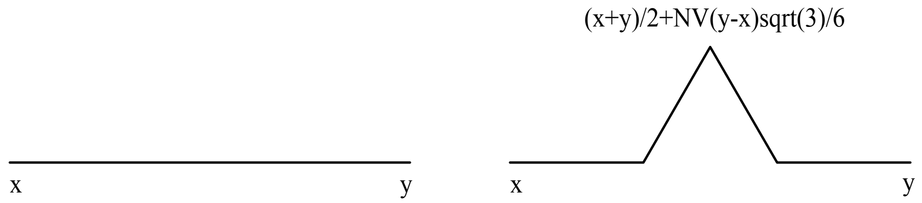
3.1 The construction

We assume that you all know how to obtain this famous curve:



Using *DERIVE* commands you can produce a Koch curve or order *n* in an easy way. You start with the unit segment as Koch curve of order 0 and build step by step curves of next order. The important idea is to relate a segment *(x,y)* to a sequence of segments `generator(x,y)`.

The only difficulty is to express the vertex of the equilateral triangle by the points *x* and *y*. We define the auxiliary function `nv(v)`, which is the normal vector of *v* with equal length and turned to the left. Using `nv(v)` we can define the `generator` - function.



$$\text{nv}(v) := v\cdot\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

$$\text{generator}(x,\ y) := \left[x,\ \frac{2\cdot x + y}{3},\ \frac{x + y}{2} + \frac{\sqrt{3}}{6}\cdot \text{nv}(y - x),\ \frac{x + 2\cdot y}{3}\right]$$

$$\text{next}(v) := \text{APPEND}(\text{VECTOR}(\text{generator}(v_i,\ v_{i+1}),\ i,\ \text{DIM}(v) - 1),\ [[v_{\text{DIM}(v)}]])$$

```
koch(l) :=
  If l = 0
     [0, 0; 1, 0]
     APPEND(next(koch(l - 1)))

koch(1)

koch(2)

koch(3)

koch(4)
```

> Even if the application of matrices to describe mappings in the plane has not been discussed in classes you could provide this function as a "Black Box".

The Koch curve of order n consists of a list of points $v = [v_1, v_2, \ldots v_{DIM(v)}]$. You receive the curve of next order n+1 in applying `generator` to each of the segments (v_i. v_{i+1}). `next(v)` creates these vectors `generator(`v_i`,`v_{i+1}`)` and appends them one after the other. Finally we have to append the last point $v_{DIM(v)}$. Starting with segment `[[0,0],[1,0]]` as Koch curve of order 0 we use a recursive construction process to create Koch curves of order k with `koch(k)`.

I will demonstrate how generator works with the next curve (Koch2-Curve), Josef.

3.2. Koch similar curves

You can generalize the Koch curve in various ways. If you replace `generator(x,y)` by any other appropriate function. `koch(k)` will generate "Koch similar" curves. Four examples of these curves will be presented in the following. `nv` and `next` will remain the same.

3.2.1 The Koch-2-Curve

If you start with the unit segment and choose points [0.4,0.2] and [0.6,-0.2] as vertices of the generator three segments of equal length $\frac{1}{\sqrt{5}}$.

This generator is described by *DERIVE* in the following function:

```
generator(x,y):=[x,0.6*x+0.4*y+0.2*nv(y-x),0.4*x+0.6*y-0.2*nv(y-x)]
```

Demonstration of the "`generator`":

Let's assume that the initial segment is [(1,1), (5,3)] – which is then `koch(0)`. According to the algorithm given above `koch(1)=append(next(koch(0)))`.

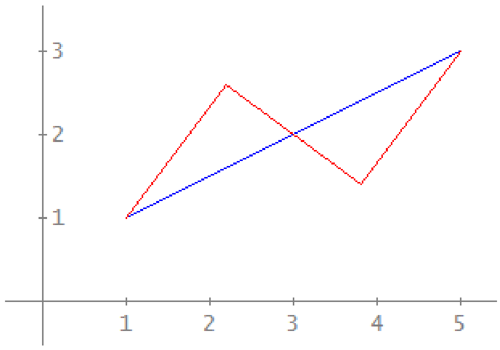
```
next(koch(0))=append(generator(koch(0)_1,koch(0)_2),koch(0)_2)
generator((1,1),(5,3))=((1,1),0.6*(1,1)+0.4*(5,3)+0.2*nv(4,2),
                                          0.4*(1,1)+0.6*(5,3)-0.2*nv(4,2)
```

$$0.2*(4,2)*\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}=(-0.4,0.8)$$

```
generator((1,1),(5,3))=((1,1),(0.6+2-0.4,0.6+1.2+0.8),
                                           (0.4+3+0.4,0.4+1.8-0.8))=
                                         =((1,1),(2.2,2.6),(3.8,1.4))
next(koch(0))=((1,1),(2.2,2.6),(3.8,1,4),(5,3))
```

We check the p&p-calculation:

$$\text{APPEND}\left(\text{next}\begin{bmatrix} 1 & 1 \\ 5 & 3 \end{bmatrix}\right)=\begin{bmatrix} 1 & 1 \\ 2.2 & 2.6 \\ 3.8 & 1.4 \\ 5 & 3 \end{bmatrix}$$



The graph of the first steps of this curve is given on the next page.

3.2.2 The Cross Stitch Curve

Another possibility is to replace the equilateral triangle by a square. Then the generator is built by 5 segments of length 1/3.

```
generator(x,y):=[x,(2*x+y)/3,(2*x+y+nv(y-x))/3,(x+2*y+nv(y-x))/3,(x+2*y)/3]
```

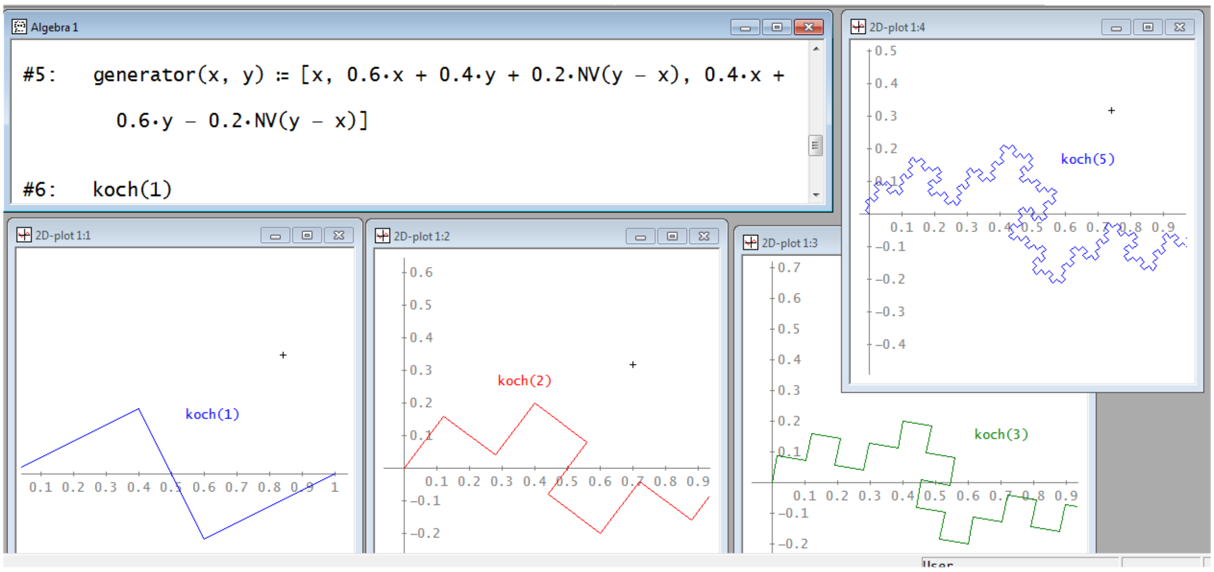
3.2.3 The Ice Curve

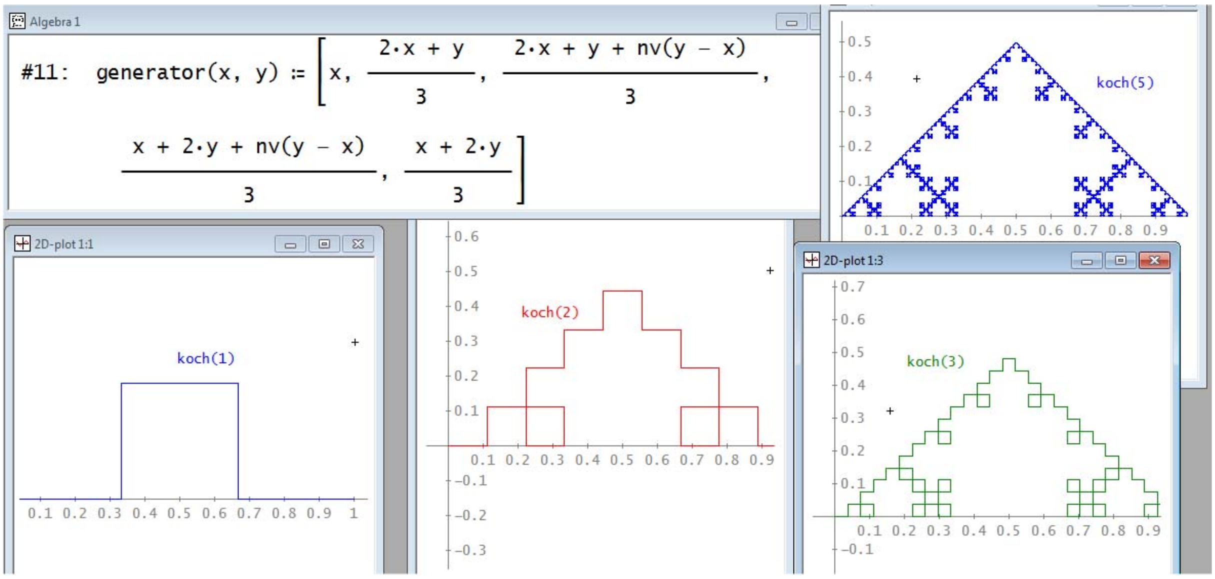
A Koch curve which makes us think on ice needles and ice flowers can be generated very easily.
Let us erect a segment of length 1/3 in the mid point of the unit segment.

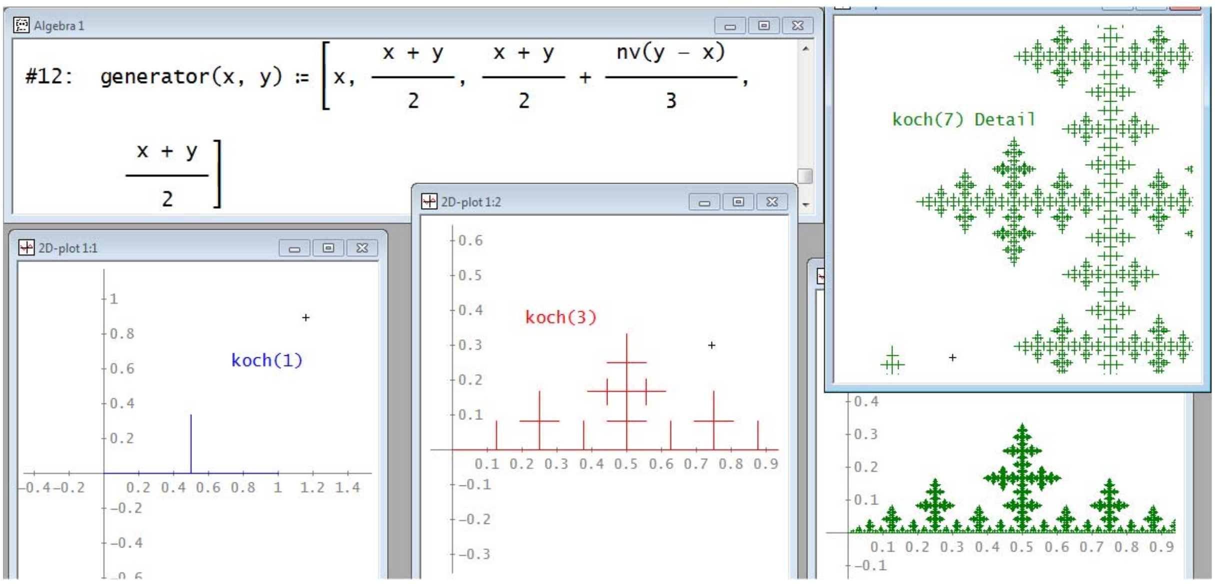
```
generator(x,y):=[x,(x+y)/2,(x+y)/2+nv(y-x)/3,(x+y)/2]
```

The pictures on the next page show these three varieties of Koch curves: (*DERIVE 6*):

Koch-2-Curve
Cross Stitch Curve
Ice Curve


Algebra 1
#5: generator(x, y) := [x, 0.6·x + 0.4·y + 0.2·NV(y – x), 0.4·x + 0.6·y – 0.2·NV(y – x)]
#6: koch(1)
2D-plot 1:1
koch(1)
2D-plot 1:2
koch(2)
2D-plot 1:3
koch(3)
2D-plot 1:4
koch(5)


Algebra 1
#11: generator(x, y) := [x, (2·x + y)/3, (2·x + y + nv(y – x))/3, (x + 2·y + nv(y – x))/3, (x + 2·y)/3]
2D-plot 1:1
koch(1)
koch(2)
2D-plot 1:3
koch(3)
koch(5)


Algebra 1
#12: generator(x, y) := [x, (x + y)/2, (x + y)/2 + nv(y – x)/3, (x + y)/2]
2D-plot 1:1
koch(1)
2D-plot 1:2
koch(3)
koch(7) Detail

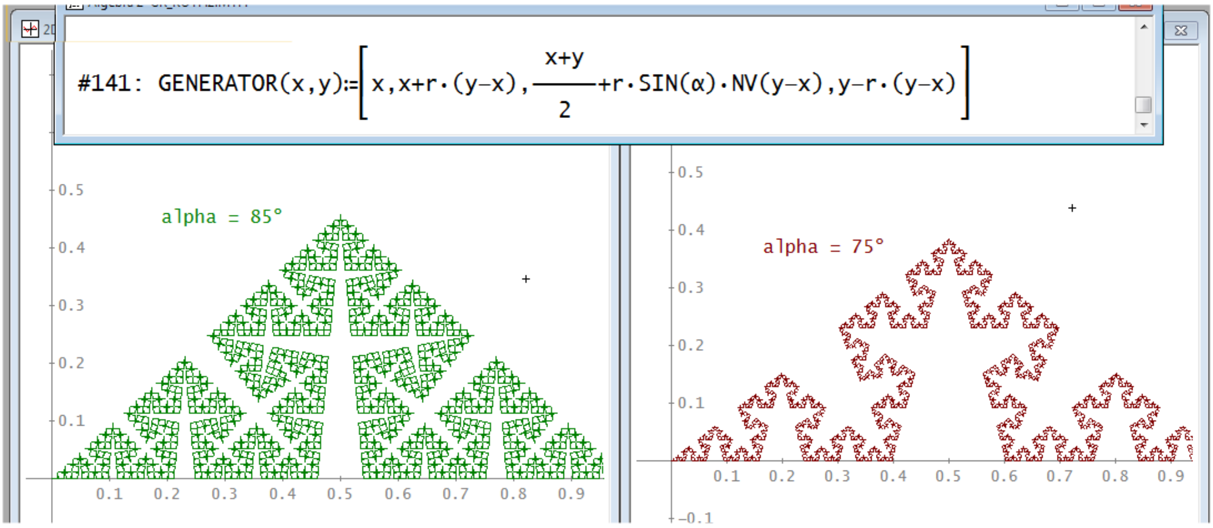
3.2.4 The Cesáro Curve

The equilateral triangle in the original Koch curve is replaced by an isosceles triangle. The length of its sides is calculated in order to obtain a generator consisting of four segments of equal length. Varying the base angle α results in varieties of this kind of Koch curve.

$$\frac{1}{2} - r = r\cos\alpha \;\rightarrow r = \frac{1}{2(1+\cos\alpha)}$$

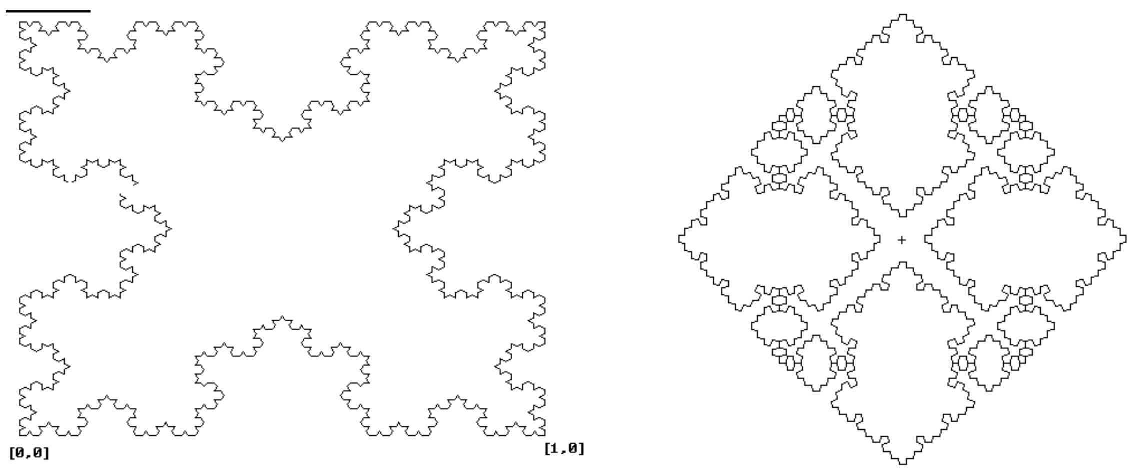
The *DERIVE* - Code:

```
[α:=85°,r:=1/(2*(1+COS(α)))]
generator(x,y):=[x,x+r*(y-x),(x+y)/2+r*SIN(α)*nv(y-x),y-r*(y-x)]
KOCH(6)
```



3.3. Koch curves in a square

Four Snowflake curves can be assembled to form a "Koch Square":



To generate a Koch- or Koch similar curve on any chosen segment *v* needs only a change in `koch(k)` from above:

```
koch1(k,v):=IF(k=0,v,APPEND(next(koch1(k-1,v))))
```

Using the vector v_0 to describe a square, `square(k,v0)` creates the square of order *k*.

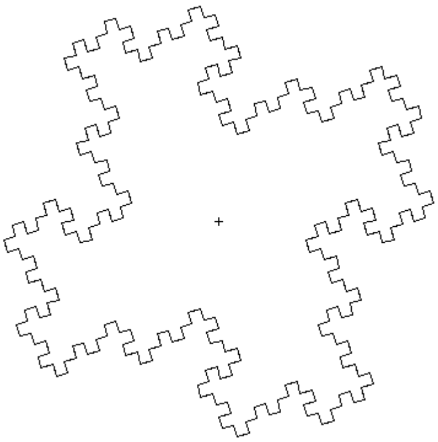
```
v0:=[[0,0],[1,0],[1,1],[0,1],[0,0]]
square(k,v_):=VECTOR(koch1(k,[v_SUB i,v_SUB (i+1)]),i,DIM(v_)-1)
```

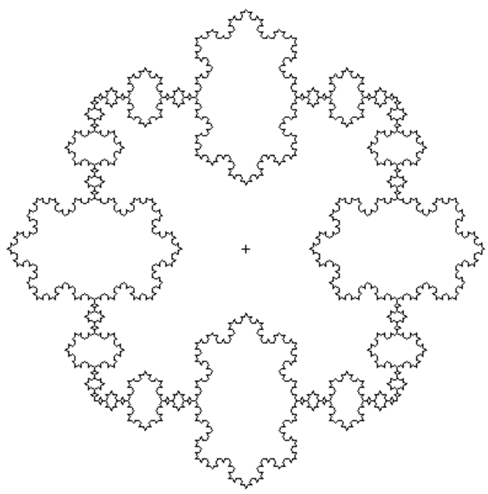
Depending on the chosen generator you can receive the following pictures. Using v_1 returns curves which are directed out of the square. Interesting pictures result in combining both versions.

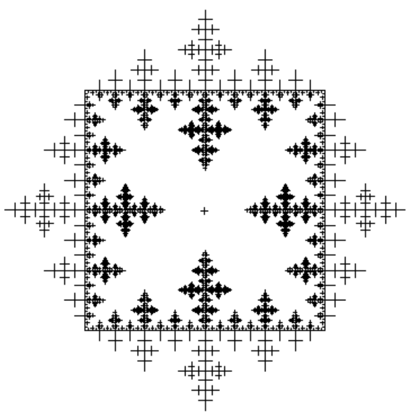
```
v1:=[[0,0],[0,1],[1,1],[1,0],[0,0]]
generator(x,y):=[x,(2*x+y)/3,(x+y)/2+SQRT(3)/6*nv(y-x),(x+2*y)/3]
```

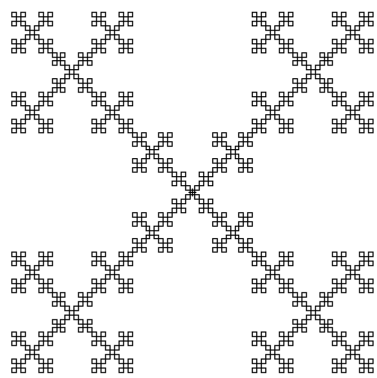
```
square(4,v0)
```

```
v1:=[[0,0],[0,1],[1,1],[1,0],[0,0]]
square(4,v1)
```

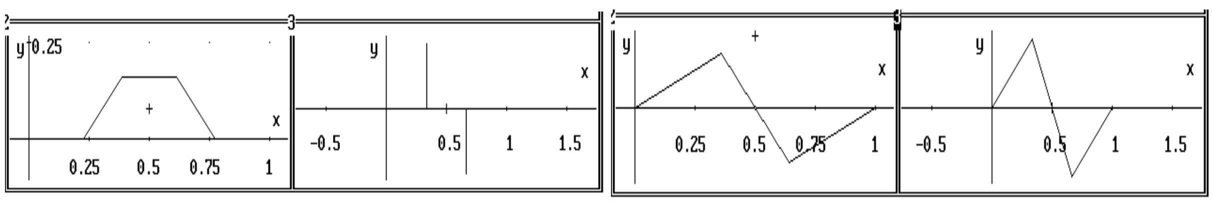


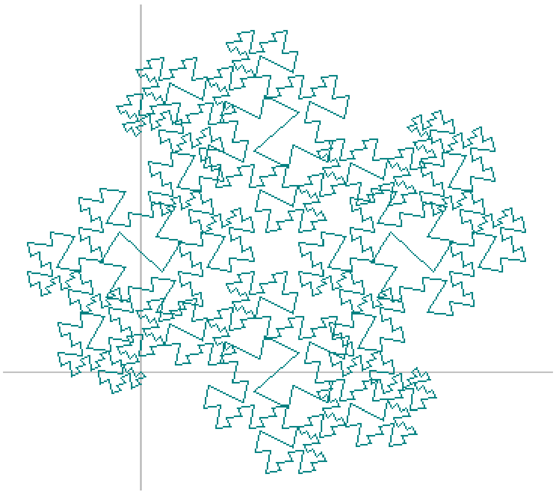


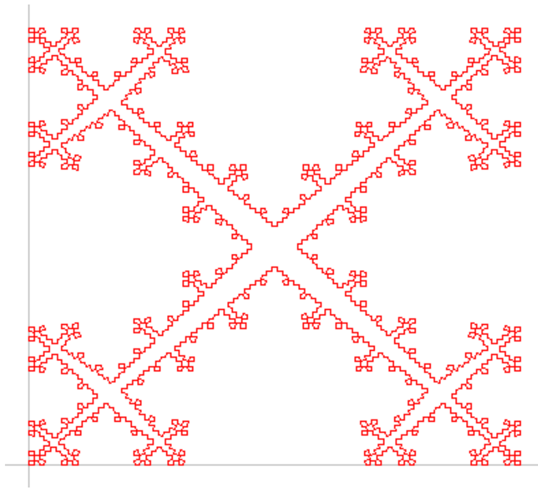


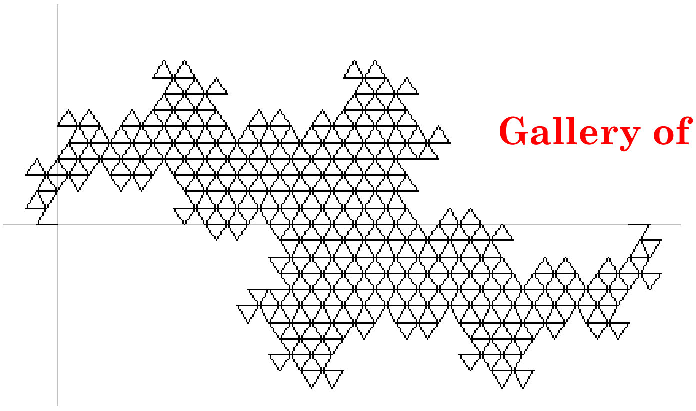


(Students could try to find their own generators. See here some suggestions. Josef)

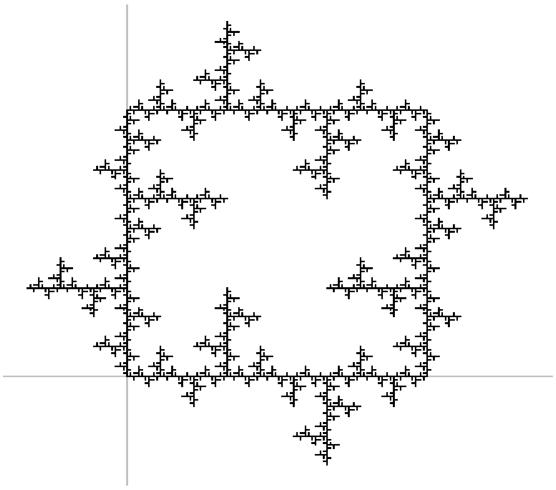


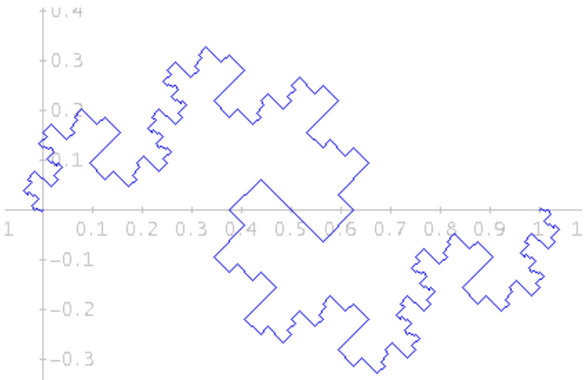


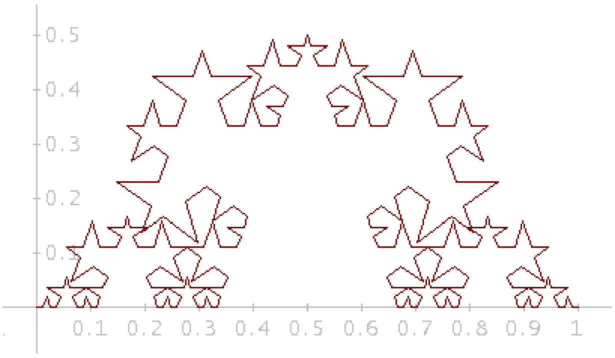




Gallery of 'Koch - Relatives'







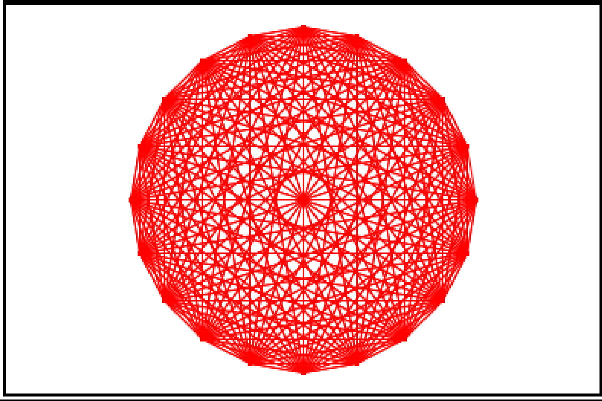
What about the Koch curves and TI-NspireCAS?

lines 4/5

Define **lines**(i,n)=
Prgm
$ls:=\{\}$
For $j,\frac{2\cdot\pi\cdot(i+1)}{n},2\cdot\pi,\frac{2\cdot\pi}{n}$
$ls:=\text{augment}\left(ls,\left\{\cos\left(\frac{2\cdot\pi\cdot i}{n}\right),\sin\left(\frac{2\cdot\pi\cdot i}{n}\right),\cos(j),\sin(j)\right\}\right)$
EndFor
EndPrgm

diagonals 7/8

Define **diagonals**(n)=
Prgm
Local d
$ds:=\{\}$
For $i,1,n-1,1$
$lines(i,n)$
$ds:=\text{augment}(ds,ls)$
EndFor
$xp:=\text{seq}(ds[i],i,1,\dim(ds),2)$
$yp:=\text{seq}(ds[i],i,2,\dim(ds),2)$
EndPrgm



Diagonals of an n-gon

Change the number of vertices:

diagonals(20) ▸ *Done*

There were no problems presenting the *n*-gons, but things changed treating the Koch curves.

nv$(v):=v\cdot\begin{bmatrix}0 & 1\\ -1 & 0\end{bmatrix}$ ▸ *Done*

ca(a,b):=colAugment(a,b) ▸ *Done*

generator(x,y):=$\text{ca}\left(\text{ca}\left(\text{ca}\left(x,\frac{2\cdot x+y}{3}\right),\frac{x+y}{2}+\frac{\sqrt{3}}{6}\cdot\text{nv}(y-x)\right),\frac{x+2\cdot y}{3}\right)$ ▸ *Done*

generator$([0\ \ 0],[1\ \ 0])$ ▸ $\begin{bmatrix}0 & 0\\ \frac{1}{3} & 0\\ \frac{1}{2} & \frac{\sqrt{3}}{6}\\ \frac{2}{3} & 0\end{bmatrix}$

next

Define **next**(v)=
Prgm
$m:=generator(v[1],v[2])$
For $i,2,\dim(v)[1]-1$
$m:=ca(m,generator(v[i],v[i+1]))$
EndFor
$m:=ca(m,v[\dim(v)[1]])$
EndPrgm

I tried to reproduce Maria's recursive procedures with TI-Nspire. It is a shame, I failed. It would be great if somebody from the DUG community could transfer Matija'a and/or Maria's recursive procedures on the Nspire.

Then I changed my strategy and used the *generators* iteratively – and this lucked.

I must use colAugment for collecting the points as a whole and to add *y* as last element of the generator.

$\mathbf{nv}(v):=v\cdot\begin{bmatrix}0 & 1\\ -1 & 0\end{bmatrix}$ ▸ *Done* $\mathbf{ca}(a,b):=\text{colAugment}(a,b)$ ▸ *Done*

Adapt the generator according to your ideas:

This gives the **Snowflake**:

$\mathbf{generator}(x,y):=\mathbf{ca}\left(\mathbf{ca}\left(\mathbf{ca}\left(\mathbf{ca}\left(x,\frac{x+y}{3}\right),\frac{x+y+\mathbf{nv}(y-x)}{2}\right),\frac{x+y}{3}\right),y\right)$ ▸ *Done*

$\mathbf{koch}([0\ \ 0],[1\ \ 0],4)$ ▸ *Done* $\dim(\mathbf{xp})$ ▸ 626

n > 4 is not possible because of restricted resources!!

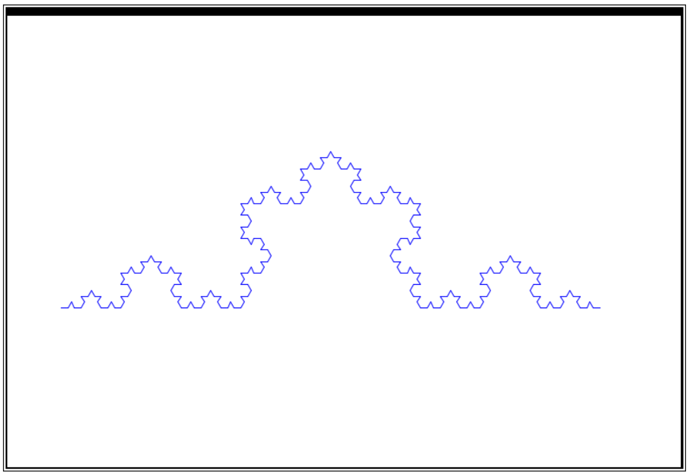
Lists ***xp*** and ***yp*** form the scatter diagram.

Just for fun: what will this look like? Exchange generator and koch given above!

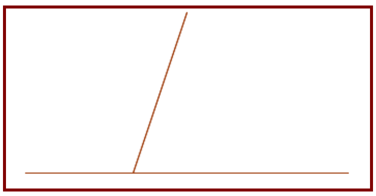
generator(x,y):=ca(ca(ca(ca(x,(x+y)/3),(x+y+nv(y−x))/2),(x+y)/3),y)

koch([0,0],[1,0],4)

```
koch
Define koch(u,v,n)=
Prgm
Local nn,g
fr:=ca(u,v)
fro:=fr
If n=0 Then
  Goto end
EndIf
For i,1,n
   nn:=dim(fro)[1]
   fr:=fro[1]
   For j,1,nn−1
      g:=generator(fro[j],fro[j+1])
      fr:=ca(fr,g)
   EndFor
fro:=fr
EndFor
Lbl end
xp:=mat▸list(froᵀ[1]):yp:=mat▸list(froᵀ[2])
EndPrgm
```



This is the generator of the fractal on the right hand side.



Matija Lokar, University of Ljubljana, Slovenia

ANIMATION IN DERIVE

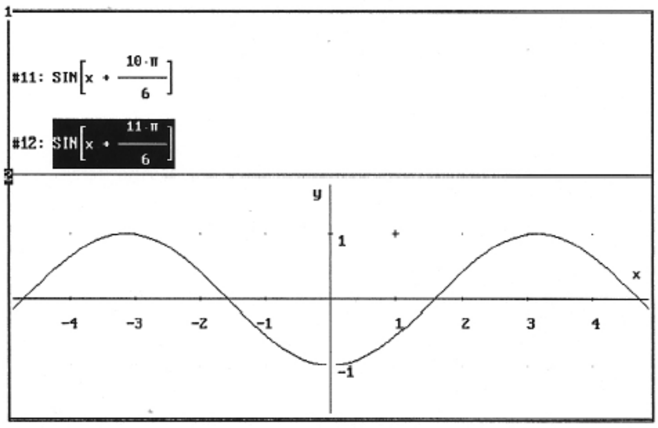
Motivation for this contribution is twofold: During systematic study of all menus in DERIVE (DOS version), I wondered why we need Window > Open and Window > Flip command. Also a colleague asked me how to demonstrate the behaviour of the sine function sin($x + t$) in dependence of time parameter t.

Well, the head line is promising too much. I will not describe any ingenious technique in DERIVE and a proper animation to be the substitute for *AcroSpin* or similar program. But for the task described it was quite successful.

The idea is to create the sequence of graphs and by switching simulating an animation. We exploit the possibility that DERIVE can open many windows on the same position (Window > Open) and switch from one window to the next one (Window > Flip).

Suppose we like to show how sine waves sin($x + t$) "travel" in time. First we prepare a suitable sequence of functions – each one as an own expression:

```
SIN(x)
SIN(x + π/6)
SIN(x + π/3)
SIN(x + π/2)
...
...
SIN(x + 11π/6)
```



It really works!!

Let us plot the last graph:

Plot > Under 10 > Plot
Open a new window

Window > Open > 2D Plot

Go to the Algebra Window, mark the before last expression and plot it

Algebra

Plot

Plot

We open a new window on the same place and repeat the procedure until we plot the function sin(x). Now we have prepared all the pictures and by rapid switching we animate them. Using Window > Flip would be too slow, so we use F2. If our computer is sufficiently fast then we will get the impression of a travelling wave.