THE DERIVE - NEWSLETTER #15

ISSN 1990-7079

THE BULLETIN OF THE

DERIVE®

USER GROUP

Contents:

revised 2007 **September 1994**

[1] **Numerical Analysis via DERIVE**, Steven Schonefeld, 530 pages
MathWare, 604 E. Mumford Dr. , Urbana, IL 61801, 1994

[2] **Precalculus Investigations using DERIVE**, David M. Mathews, 128 pages
Harper Collins College Publishers, 1994

[3] **The DERIVE Calculus Workbook**, L. Townsley Kulich and Barbara Victor
Harper Collins College Publishers, 1994

[4] **Mathematics with Excel**, David Sjöstrand, 189 pages
Chartwell Bratt, 1994

[5] **Matematikk for Økonom og Samfunnsfag**, H.Bjornestad and others, 489 pages
(Mathematics for economics and social studies; the first textbook of mathematics using DERIVE in Norway)Hoyskole Forlaget - Norwegian Academic Press, 1994

[6] **LAB Projects, NSF Calculus Institute, Using Computer Algebra Systems**, 510 pages
Editor: J.Birdsall, Santa Fe Community College, Gainesville, FL 32606

[7] **Derive per la scuola**, S. Rossetto, 138 pages
McGraw - Hill Libri, Milano/Italy, 1992
(Thank you Giuseppe, the other book you mentioned is already on our shelf!)

e-mail .... e-mail .... e-mail .... e-mail .... e-mail .... e-mail .... e-mail

We have received many requests concerning e-mail numbers of Soft Warehouse or of the BBS ......
I am glad to be able to give an answer now. Yes, there are some e-mail numbers:

Soft Warehose:	**74521.3051**	**(Compuserve)**
	74521.3051@compuserve.com	**(Internet)**

DERIVE-NEWS: Derive-News is a new electronic discussion list set up after the International Derive Conference at Plymouth in July. You can join by sending a message to

mailbase@mailbase.ac.uk with the single line
`join derive-news firstname lastname`

(eg join derive-news pam bishop) as the text of the message.

European DERIVE Bulletin Board: cain@can.nl (Internet)

e-mail .... e-mail .... e-mail .... e-mail .... e-mail .... e-mail .... e-mail

For our German members:

In Deutschland sind viele Schulen an das „Offene Deutsche Schulnetz“ ODS angeschlossen, das den Schulen kostenlosen Internet-Zugang ermöglicht. Dort gibt es sogenannte NEWS-Gruppen für gewisse Themen - so zB die Gruppe “schule.math“. (*Hinweis von G.Noll, Erpel/Rhein*)

H. Scheuermann würde gerne mit anderen Kollegen in Kontakt treten und bittet mich seine Adresse zu veröffentlichen. (Aus Gründen des Datenschutzes will ich keine Liste der Mitglieder weiterleiten.) Also, wer Lust auf einen Gedankenaustausch hat wendet sich an: Hellmut Scheuermann, Ubierstr. 13, 65719 Hofheim/Taunus, Tel.: 06192/27566.

Two offers:

I have a copy of DERIVE, Version 3 Release Notes and I have a copy of “The Evolution of DERIVE“. If you are interested in one of these papers (- or both) then please write or call I´ll send you the papers.

Liebe DUG-Mitglieder,

Ein schöner Sommer - für Österreich und einen großen Teil Europas - geht zu Ende und ich freue mich, Ihnen die 15. Ausgabe des DNL präsentieren zu können. Für die DERIVE-Gemeinde war die Konferenz in Plymouth sicher das herausragende Ereignis dieses Sommers. Hier ist zu wenig Platz, jeden einzelnen der vielen hervorragenden Vorträge zu würdigen. Sie können sich aber sicher vorstellen, daß besonders der Hauptvortrag von David Stoutemyer und Albert Rich über das „Innenleben" von DERIVE und über die Möglichkeiten von DERIVE 3.x ganz besonderes Interesse gefunden haben. An dieser Stelle möchte ich im Namen der DUG John Berry und seinem Team für ihren Einsatz recht herzlich danken.
Ich habe im letzten DNL einen Beitrag von Sergey Biryukow versprochen, der die Beschriftung und Skalierung von DERIVE-Plots aus der DERIVE-Umgebung heraus ermöglicht. Da ich Ihnen nicht zumuten will, in mühsamer Arbeit die Daten für die Zeichensätze einzugeben, werde ich diesen Artikel in der nächsten Nummer drucken und kann die zugehörigen Datendateien auf der Diskette 94 gleich mitliefern. Bis dahin bitte ich um Geduld. (Wenn jemand wirklich nicht mehr warten kann, dann soll er mich bitte anrufen.) Sergej überraschte uns in Plymouth nicht nur mit einem spannenden Vortrag über DERIVE-Anwendungen (Optik) sondern auch mit einem DERIVE - Gedicht (Seite 48).
Auch die Freunde von Th. Weths „Kurvenlexikon" bitte ich um Geduld für den nächsten DNL. Aus Platzgründen einerseits und aus Rücksicht auf unsere vielen nicht deutschsprachigen Freunde andererseits wird die Serie mit Folge 5 (mit der Konchoide des Nikomedes) im nächsten DNL fortgesetzt.
Sie finden dieses mal auf unserem Bücherbrett auch ein Buch, das sich nur am Rande mit DERIVE beschäftigt. David Sjöstrand ist ein sehr aktives Mitglied der DERIVE-Gruppe und schrieb ein Buch über Mathematik mit Excel. Für ihn - und für viele andere - sind Tabellenkalkulation und DERIVE kein entweder - oder sondern ein sowohl - als auch.
Aus dem angekündigten Splines - Wettkampf Böhm - Reichel hat sich durch einen Beitrag von LeoKlingen, Bonn, ein Dreikampf entwickelt. Nach meiner ersten Einschätzung dürfte die reiche Erfahrung Dr. Klingens ihm einen beträchtlichen Vorsprung sichern.

Mit den besten Grüßen
bis zum nächsten Mal

[illegible]

Dear DUG Members,

A wonderful summer - for Austria and a large part of Europe - is going to have an end now and I have the pleasure to present the 15th issue of the DNL. For the DERIVE-community the Plymouth Conference in July was the outstanding event of this summer. Here is not room enough to mention honorably each one of the interesting lectures. But you surely can imagine that we all especially appreciated David Stoutemyer´s and Albert Rich´s plenary lecture about "DERIVE´s interior" and about DERIVE 3.x´s new features. Let me say a warm THANK YOU in the DUG´s name to John Berry and his team for organizing the conference and making it a success.

In the last DNL I´ve announced a contribution from Sergey Biryukow which enables us to label and to scale a DERIVE-plot within the DERIVE-environment. Because it is a very hard and boring work to edit the fonts´ data I´ve decided to publish this article in the next DNL, so you can find then the .MTH-file and the data files on the diskette of ´94. I hope that is in your sense. Sergey surprised us in Plymouth not only with an excellent lecture about DERIVE applications (optics) but also with a DERIVE-poem
(page 8).
I also ask the friends of Thomas Weth´s "Lexicon of curves" for patience. There are two reasons to continue his series (with the Konchoide of Nikomedes) in the next DNL: arrangement in the layout and consideration for our many non German speaking friends.
On the Book-Shelf you can find this time one title which is dealing with DERIVE only very little. D.Sjöstrand is a very active DERIVIAN and he wrote a book about teaching Mathematics with Excel. For him - and for many others - DERIVE and spread sheets don´t represent a either - or but an as well - as.
The promised spline - competition Böhm vs Reichel seems to become a triathlon because of a SPLINE.MTH file from Leo Klingen, Bonn. I think that Dr. Klingen´s rich experience will save him a considerable lead.

Sicerely yours

[illegible]

ie Tagungsbände von Krems ´92 **„Teaching Mathematics with DERIVE"** und Krems ´93 **„DERIVE in Education"** sind zum Preis von öS 330.- /öS 450.- bzw. DM 50.- / DM 65.- (inkl. Verpackung und Porto) bei mir zu beziehen. Anruf, Fax oder kurzes Schreiben genügen.

The *DERIVE-NEWSLETTER* is the Bulletin of the *DERIVE User Group*. It is published at least four times a year with a contents of 30 pages minimum. The goals of the *D-N-L* are to enable the exchange of experiences made with *DERIVE* as well as to create a group to discuss the possibilities of new methodical and didactical manners in teaching mathematics.

Editor: Mag. Josef Böhm
A-3042 Würmla
D´Lust 1
Austria
Phone: 43-(0)2275/8207

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 *D-N-L*. It must be said, though, that non-English articles are very welcome nonetheless. Your contributions will be edited but not assessed. By submitting articles the author gives his consent for reprinting it in *D-N-L*. The more contributions you will send, the more lively and richer in contents the *DERIVE Newsletter* will be

Preview: (Contributions for the next issues):

Stability of systems of ODEs, Kozubik, SLO
Prime Iterating Number Generators, Wild, UK
Graphic Integration, Probability Theory, Linear Programming, Böhm, AUS
DERIVE in Austrian Schools, some examples, Lechner, Voigt, Eisler a.o., AUS
Tilgung fremderregter Schwingungen, Klingen, GER
Der Fermat - Punkt im Dreieck, Geyer, GER
Continued Fractions and the Bessel Functions, Cordobá a.o., ESP
Turtle Commands in DERIVE, Lechner, AUS
DREIECK.MTH, Wadsack, AUS
Newton Method and Ill-Conditioned Problems, Lopes, POR
2D Plots Labeling, Biryukov, RUS
Plotting t-periodic functions, Verhoosel, NED
IMP Logo and Misguided Missiles, Sawada, HAWAII
“Reverse“ Discussion of Curves, Reichel, AUS
Reichel - Klingen - Böhm - Splines, A thriathlon, AUS & GER
3D Geometry, Reichel, AUS
Parallel- and Central Projection, Böhm, AUS
Conic Sections, Fuchs, AUS
The IMP Logo, Sawada, USA
Setif, France; Vermeylen, Belgium; Leinbach, USA and others

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

Steven Schonefeld, Angola, IN, USA

Can DERIVE decide if two numbers are equal?

The answer is definitely YES if you are using version 2.58 or later. In some earlier versions, DERIVE's IF(,,,) function cannot always decide whether two numbers are equal, especially if one or both of the numbers are complex. For example, the DERIVE function COMPARE(x, y), defined below, should decide whether the numbers x and y are "Equal" or "Not equal".

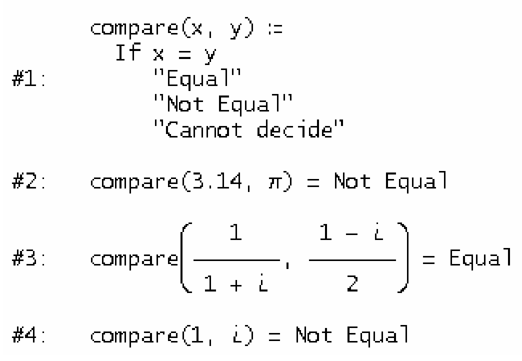
```
COMPARE(x, y) := IF(x = y, "Equal", "Not equal", "?")

COMPARE(3.14, π)    [S]implifies to    "Not equal"

COMPARE(1/(1 + î), (1 - î)/2)    [S]implifies to    "Equal"

COMPARE(1, î)    [S]implifies to "?" when we use version 2.09.  However,

COMPARE(1, î)    [S]implifies to   "Not equal"   when we use version 2.58.
```

There is a simple way to fix this problem so that you get the correct results for all versions of DERIVE containing the IF(,,,) function. This fix relies on the fact that numbers x and y are equal if, and only if, the absolute value of their difference is zero. Thus, we replace "x = y" with "ABS(x - y) = 0" in the function COMPARE(,) to get.

```
COMPARE_2(x, y) := IF(ABS(x - y) = 0, "Equal", "Not equal", "?")
```

The function COMPARE_2(x, y) now gives the correct results for all versions of Derive containing the IF(,,,) function. See DERIVE 6:

No problems with recent versions of DERIVE

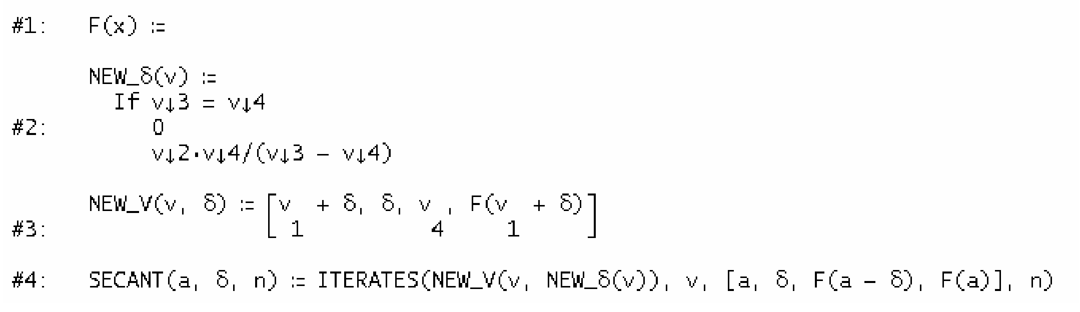


```
      compare(x, y) :=
        If x = y
#1:        "Equal"
           "Not Equal"
           "Cannot decide"

#2:   compare(3.14, π) = Not Equal
```

$$\#3:\quad \text{compare}\left(\frac{1}{1+i}, \frac{1-i}{2}\right) = \text{Equal}$$

```
#4:   compare(1, i) = Not Equal
```

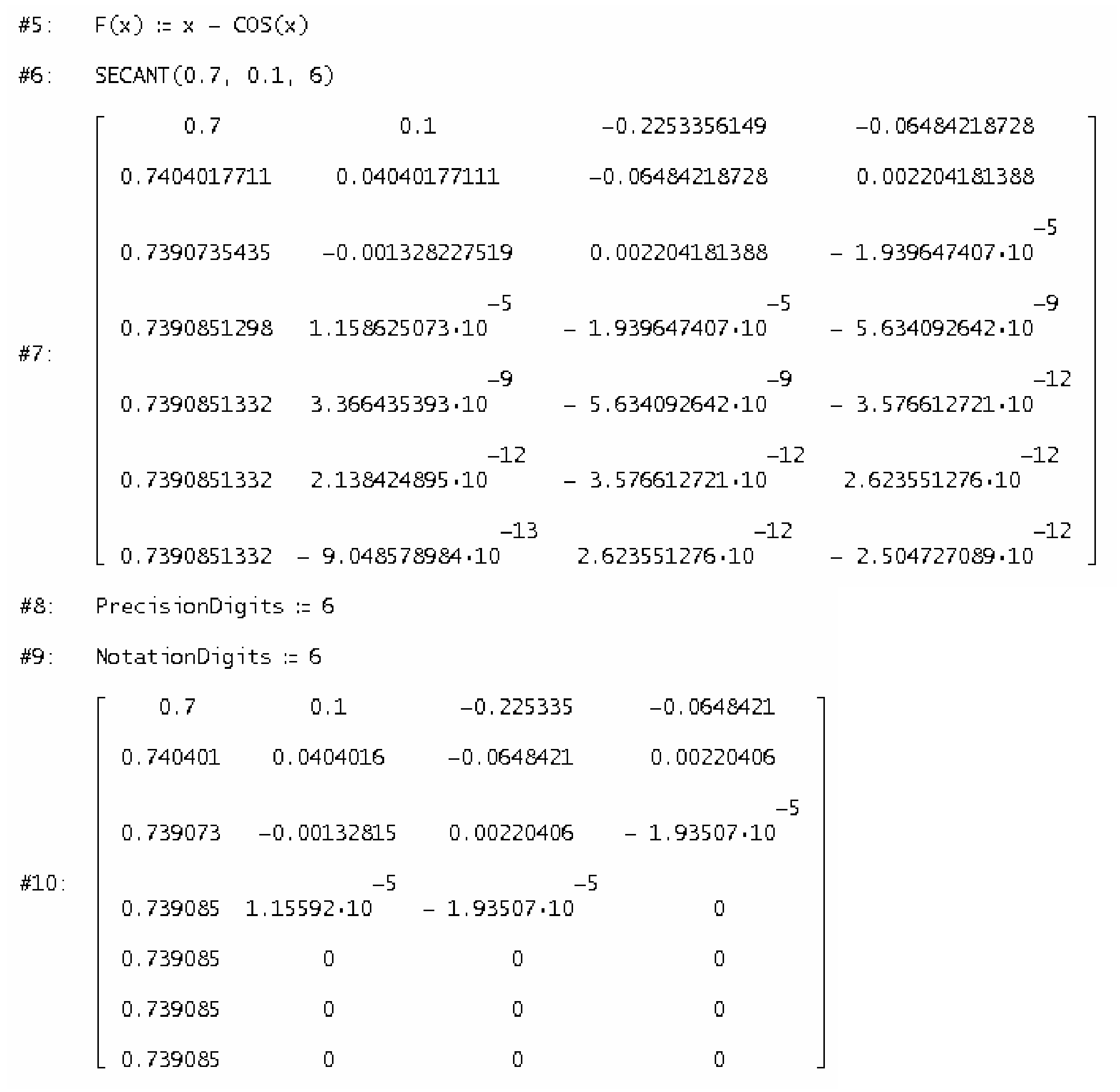
Below is an implementation of the secant method (for approximating solutions to F(x) = 0). We see that DERIVE needs to determine whether the two numbers ELEMENT(v, 3) (= v↓3) and ELEMENT(v, 4) (=v↓4) are different in order to prevent division by zero in the calculation of NEW_δ(v).



```
#1:   F(x) :=

      NEW_δ(v) :=
        If v↓3 = v↓4
#2:        0
           v↓2·v↓4/(v↓3 - v↓4)
```

$$\#3:\quad \text{NEW\_V}(v, \delta) := [v_1 + \delta, \delta, v_4, F(v_1 + \delta)]$$

```
#4:   SECANT(a, δ, n) := ITERATES(NEW_V(v, NEW_δ(v)), v, [a, δ, F(a - δ), F(a)], n)
```



```
#5:   F(x) := x - COS(x)

#6:   SECANT(0.7, 0.1, 6)

      ┌      0.7             0.1            -0.2253356149        -0.06484218728     ┐
      │                                                                            │
      │ 0.7404017711    0.04040177111     -0.06484218728        0.002204181388      │
      │                                                                    -5      │
      │ 0.7390735435   -0.001328227519     0.002204181388     - 1.939647407·10      │
      │                              -5                 -5                 -9      │
#7:   │ 0.7390851298   1.158625073·10     - 1.939647407·10    - 5.634092642·10      │
      │                              -9                 -9                -12      │
      │ 0.7390851332   3.366435393·10     - 5.634092642·10    - 3.576612721·10      │
      │                             -12                -12                -12      │
      │ 0.7390851332   2.138424895·10     - 3.576612721·10     2.623551276·10       │
      │                             -13                -12                -12      │
      └ 0.7390851332 - 9.048578984·10      2.623551276·10     - 2.504727089·10      ┘

#8:   PrecisionDigits := 6

#9:   NotationDigits := 6

      ┌    0.7         0.1        -0.225335      -0.0648421   ┐
      │                                                        │
      │ 0.740401    0.0404016    -0.0648421      0.00220406    │
      │                                                  -5    │
      │ 0.739073   -0.00132815    0.00220406   - 1.93507·10    │
#10:  │                    -5               -5                 │
      │ 0.739085   1.15592·10   - 1.93507·10          0        │
      │                                                        │
      │ 0.739085        0              0              0        │
      │                                                        │
      │ 0.739085        0              0              0        │
      │                                                        │
      └ 0.739085        0              0              0        ┘
```

In this example, we re-defined the function F(x) on line #5, and appro[X]imated #6 to get #7. This told Derive to perform n = 6 iterations starting with initial guess $x_2 = a = 0.7$ and $\delta_1 = 0.1$ (nearly any small number δ will do as well).

If we let $\delta_{k-1} = x_k - x_{k-1}$, the secant method approximation x_{k+1} is calculated from:

$$x_{k+1} = x_k + \delta_k \quad \text{where} \quad \delta_k = -\frac{F(x_k)\delta_{k-1}}{F(x_k) - F(x_{k-1})}.$$

A typical row of the display matrix, #7, is of the form:

$$[x_{k+1}, \delta_k, F(x_k), F(x_{k+1})].$$

The secant method may be used to locate complex solutions to F(x) = 0, provided that a reasonable initial guess is given (and #2 is modified for earlier versions of DERIVE.) In fact, the secant method can locate a zero of (nearly) any function, F(x), that the Newton-Raphson method can locate. Moreover, the secant method usually requires less time than the Newton-Raphson method. For more information and examples concerning the secant method, see my book: *Numerical Analysis via Derive* published by MathWare, 604 E Mumford Drive, Urbana, IL.

Terence Etchells, Bolton, UK

Here is a little function for the inverse normal distribution:

$$\#1: \quad \mathrm{INVNORM}(p,\ \mu,\ \sigma) := \mathrm{ITERATE}\left(a - \frac{\dfrac{\mathrm{SIGN}(\sigma)\cdot\mathrm{ERF}\left(\dfrac{\sqrt{2}\cdot a}{2\cdot\sigma} - \dfrac{\sqrt{2}\cdot\mu}{2\cdot\sigma}\right)}{2} + \dfrac{1}{2} - p}{\dfrac{\sqrt{2}\cdot e^{-a^2/(2\cdot\sigma^2) + a\cdot\mu/\sigma^2 - \mu^2/(2\cdot\sigma^2)}}{2\cdot\sqrt{\pi}\cdot|\sigma|}},\ a,\ \mu,\ 5 \right)$$

Example:

$$\#2: \quad \mathrm{NORMAL}(109.8,\ 105,\ 7.5) = \frac{\mathrm{ERF}\left(\dfrac{8\cdot\sqrt{2}}{25}\right) + 1}{2}$$

$$\#3: \quad \mathrm{NORMAL}(109.8,\ 105,\ 7.5) = 0.7389137003$$

$$\#4: \quad \mathrm{INVNORM}(0.7389,\ 105,\ 7.5) = 109.7996839$$

Working now with NSOLVE:

$$\#5: \quad \mathrm{NSOLVE}(\mathrm{NORMAL}(x,\ 105,\ 7.5) = 0.7389,\ x) = (x = 109.7996839)$$

Josef Böhm, Würmla, A

Two tipps for working with DERIVE:

Random Numbers

I worked in the classroom and wanted to demonstrate an experiment using random numbers. All the students loaded the file, simplified and then we compared the results. You will understand that I was not very happy when we found out, that all the results were identical. We tried once more: other results, but identical again .....
So my DERIVE show was not a big success. What to do?
I took the manual and tried to find a way. That is it:

> Before starting the simulation simplify the command
>
> `RANDOM(0)`
>
> then go on in the usual way. Doing this you initialize the random number generator with the time elapsed since starting DERIVE.

Text processing and DERIVE

There are problems to convert DERIVE-files saved in extended ASCII-Code, into WORD for Windows (in ANSI-Code). Now in WORD for Windows 6.x it is possible:

- Save the DERIVE - file with Transfer Print File FILENAME.TXT.
- Open the file from WINWORD and then format with the font **MS LineDraw**. Compress the lines, eg. line distance 0.75. Don´t be surprised that the text now may look a bit strange. A result of this effort can be seen above.

(This advice is no more valid in times of DERIVE for Windows.)

Message 2622: From HARALD LANG to PUBLIC about BUG IN V 2.52

Version 2.52 has a bug. Solving a-b*tan(x)^3 produces a wrong answer! Are there more bugs?

Message 2625: From DAN to HARALD LANG about #2622 / BUG IN V 2.52

Could you give your answer? What branch was your default on? Did you try the real branch?
-- Dan Frezza bz223@Cleveland.Freenet.Edu

Message 2627: From HARALD LANG to DAN about #2625 / BUG IN V 2.52

With Branch=real, solving e.g. 8-8*TAN(x)^3=0 produces the answer x=ATAN(1/4). In general, a-b*TAN(x)^3 gives the answer x=ATAN[(a^(1/3)/b] rather than x=ATAN[(a/b)^(1/3)]. Also other powers, like 2, produces an incorrect answer. Have You got the same problem?

Message 2630: From DAN to HARALD LANG about #2627 / BUG IN V 2.52

Harold, I believe you have found a bug. I tried the same thing, and got the same answer.

How is DERIVE 6 responding?

$$\mathrm{SOLVE}(a - b\cdot\mathrm{TAN}(x)^3 = 0,\ x) = \left[x = -\mathrm{ATAN}\left(\frac{a^{1/3}\cdot\left(\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}\right)}{b^{1/3}}\right) - \pi \lor x = \pi - \mathrm{ATAN}\left(\frac{a^{1/3}\cdot\left(\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}\right)}{b^{1/3}}\right) \lor x = -\mathrm{ATAN}\left(\frac{a^{1/3}\cdot\left(\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}\right)}{b^{1/3}}\right) \lor x = -\mathrm{ATAN}\left(\frac{a^{1/3}\cdot\left(\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2}\right)}{b^{1/3}}\right) - \pi \lor x = \pi - \mathrm{ATAN}\left(\frac{a^{1/3}\cdot\left(\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2}\right)}{b^{1/3}}\right) \lor x = -\mathrm{ATAN}\left(\frac{a^{1/3}\cdot\left(\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2}\right)}{b^{1/3}}\right) \lor x = \mathrm{ATAN}\left(\frac{a^{1/3}}{b^{1/3}}\right) - \pi \lor x = \mathrm{ATAN}\left(\frac{a^{1/3}}{b^{1/3}}\right) + \pi \lor x = \mathrm{ATAN}\left(\frac{a^{1/3}}{b^{1/3}}\right)\right]$$

$$\mathrm{SOLVE}(8 - 8\cdot\mathrm{TAN}(x)^3 = 0,\ x) = \left[x = -\frac{5\cdot\pi}{4} - \frac{i\cdot\mathrm{LN}(2 - \sqrt{3})}{2} \lor x = -\frac{5\cdot\pi}{4} - \frac{i\cdot\mathrm{LN}(\sqrt{3} + 2)}{2} \lor x = \frac{3\cdot\pi}{4} - \frac{i\cdot\mathrm{LN}(2 - \sqrt{3})}{2} \lor x = \frac{3\cdot\pi}{4} - \frac{i\cdot\mathrm{LN}(\sqrt{3} + 2)}{2} \lor x = -\frac{\pi}{4} - \frac{i\cdot\mathrm{LN}(2 - \sqrt{3})}{2} \lor x = -\frac{\pi}{4} - \frac{i\cdot\mathrm{LN}(\sqrt{3} + 2)}{2} \lor x = \frac{5\cdot\pi}{4} \lor x = -\frac{3\cdot\pi}{4} \lor x = \frac{\pi}{4}\right]$$

$$\mathrm{SOLVE}(8 - 8\cdot\mathrm{TAN}(x)^3 = 0,\ x,\ \mathrm{Real})$$

$$x = \frac{5\cdot\pi}{4} \lor x = -\frac{3\cdot\pi}{4} \lor x = \frac{\pi}{4}$$

Message 3098: From HARALD LANG to HADUD about #3091 / SIMPLIFYING CUBE ROOTS

Here is another example: 8*SIN(π/18)*COS(π/9)*COS(2π/9)

This is equal to 1 (new challenge: prove it!). We can get reasonably convinced by approximating it by DERIVE, but DERIVE fails to **simplify** it to 1. A Bug!! Hello, Soft Warehouse, are you there <the-re...echo...echo> :-)

Message 3100: From SOFT WAREHOUSE to HARALD LANG about #3098 / SIMPLIFYING TRIG EXPRESSIONS

Your questions are exactly what I like to see on this BBS. Believe me, improving DERIVE's knowledge of mathematics is much more interesting than user-interface programming.

Your trig example is a much easier problem than the fractional power example in your previous message. To simplify the expression to 1, the product of trig factors needs to be collected into a single term. This is done by using the Manage Trig command to select "Collect" trig mode before simplifying your expression.

Aloha, Al Rich, Soft Warehouse, Inc.

Message 3107: From HARALD LANG to SOFT WAREHOUSE about #3100 / SIMPLIFYING TRIG EXPRESSIONS

Hmm, you caught me there. But now I am impressed. If I understand correctly, when "collect" is on, she (DERIVE) will rewrite COS and SIN to complex exponentials, perform the multiplication and go back to SIN and COS. I won't bore you with the trivial details, but she will end up simplifying the sum (or an equivalent sum)

SIN(π/18) – COS(π/9) + COS(2π/9) (1) which is =0.

But **how** does she know that??? In order to outsmart her, I must give her a somewhat different problem: simplify

SIN(π/7) – SIN(10π/21) + COS(13π/42) (2) which is also =0 (challenge ...)

This one beats her, I have tried all 9 combinations of options. Note that I am not complaining on DERIVE, on the contrary. I will try later to sort out what I am getting at. – Harald

Message 3109: From HARALD LANG to PUBLIC about INTERACTING WITH DERIVE

Expression (2) in my message #3107 was SIN(π/7) – SIN(10π/21) + COS(13π/42) (2)

This is =0, which DERIVE can't simplify it to. **I** know it is equal to 0 for a peculiar reason: I looked at the 3-degree equation

x^3 – (3/4)*x + (1/4)*SIN(3π/7) = 0 (3) and it is easy to see, using the identities SIN(3x) = 3*SIN(x) – 4*SIN(x)^3 and COS(3x) = 4*COS(x)^3 – 3*COS(x), that this equation has the following roots:

x1 = SIN(π/7), x2 = COS(13π/42), x3 = -SIN(10π/21)

(this is also essentially the solution DERIVE gives with Manage Trig 'Direction'= Auto, 'Toward'=Sines.) But since the sum of the roots of a 3-degree equation (with x^3-coefficient =1) equals the negative of the x^2-coefficient, i.e., =0, we get my identity (2) = 0.

Here is another way to look at it:

consider the expression SIN(x) – SIN(x + π/3) + COS(x + π/6) (4)

This is quite simple to simplify to =0, isn't it? We just expand the second and third terms using the usual trig formulas. Also, DERIVE HAS NO PROBLEMS TO SIMPLIFY IT TO =0 ('direction'=Expand, 'towards'=Auto)! But if we substitute π/7 for x, and **then** simplify with DERIVE, she fails! Indeed, we get (2)! So we have the apparent paradox that DERIVE can simplify the more general expression (4), but not the special case (2). But surely, this is true also for us humans (at least we find (2) more difficult than (4), don't we?) Of course, there is a similar story behind my earlier ‘fractional power’ example.

When I analyse a model or a problem, I usually have some information that can not be formalised: I know what interpretation I have in mind, where the problem come from, **why** I made a certain specification, etc. There is no way (in general) that I could give all information to the programme (in a source code, for instance) and then ‘turn the crank’, and hope to get a ‘solution’ **relevant for the application**. Applied mathematics just isn't that way. My examples above are of course artificial, but you see the point: there is no way that DERIVE could know that I got the three terms in (2) as the roots of the equation (3). Nor could she know that the relevant way to interpret the seemingly random numbers 10π/21 and 13π/42 is π/7 + π/3 and π/7 + π/6. But **I** know, since I put these numbers there. The punchline is that I need an **interactive** programme: I am the master, DERIVE is my clever assistant ("DERIVE -- a Mathematical Assistant"; very well put!), and I must have the option in each step to help her and to push her the way I want.

[As an aside: I think SWH is making a mistake in marketing DERIVE: you focus so much on it as a tool for education that it gives the impression that if you want something for research in applied mathematics, you should look for something else. But this is totally wrong: DERIVE must be (I believe, I have no experience) a wonderful tool for education, since one can **explore** mathematics with it, not just ‘turn a crank’, but this is also precisely why it is so useful in research! (my experience is from economics, but isn't it generally true?) **Don't** pretend that with DERIVE, you just formulate the problem and lean back while she does all the job for you, because that is not true! Some people get disappointed and give up (I know examples of this.) It **can't** be true either, this is an inherent property of mathematics, NOT a limitation to this very software; thus nothing to be ashamed of. And the right way to adopt to this reality is to make the programme interactive, as DERIVE.]

But for me to work in an efficient way interactively with DERIVE, I must know how she "thinks", i.e., what she does (tries) when she simplifies, calculates an `antiderivative', solves an equation, etc. I would like to hear on this BBS what experiences other users have on this, and how they manage to push DERIVE to perform what they want in certain situations, etc. Furthermore, SWH: I would definitely be willing to pay some for an additional documentation on how DERIVE "thinks", I think I would gain at least as much efficiency from that as from an upgrade to XM. What do you say, SWH, any hope for that?

Finally, here is an example I think DERIVE **should** manage to simplify to =0:

(sqrt(5)+2)^(1/3) + (8*sqrt(5)+16)^(1/3) - (27* sqrt(5)+54)^(1/3)

I have tried different option settings, but DERIVE fails. Again, one can help her by substituting x for sqrt(5); **then** she gets =0. Again it is easier for her to simplify a more general expression than a special case! SWH: I hope you realise how useful it would be if one could anticipate such things. – Harald

Message 3110: From SOFT WAREHOUSE to HARALD LANG about #3107 / SIMPLIFYING TRIG EXPRESSIONS

In your message #3107 you wondered how DERIVE is able to simplify

SIN(π/18) - COS(π/9) + COS(2π/9)   to 0.

The more fundamental question is how DERIVE simplifies SIN(π/18) + COS(2π/9) to COS(π/9). Since COS(x) = SIN(x + π/2) for all x, **internally** DERIVE stores the above expression as SIN(π/18) + SIN(13π/18)

Then DERIVE applies the trig identity SIN(x) + SIN(y) = 2 * SIN((x + y)/2) * COS((x – y)/2) to the above expression. This results in the product 2 * SIN(7π/18) * COS(–π/3)

The above SIN factor simplifies to COS(π/9) and the COS factor simplifies to 1/2, which then conveniently cancels the 2 factor. Your done! No magic required.

The real trick to doing computer algebra is to know which identities to apply and when to apply them. For example, the above trig identity should not be applied if the result is more complicated than the original expression.

A bug prevented DERIVE from simplifying your second example using the same algorithm. The problem will be fixed in the next release. Examples like these are invaluable for finding deficiencies and improving DERIVE. Keep um' coming!

Note that DERIVE does **not** transform SIN and COS to complex exponentials when in trig "Collect" mode. Rather it applies the identities listed on the bottom of page 98, Section 6.3, of the DERIVE User Manual.

Aloha, Al Rich, Soft Warehouse, Inc.

Message 3136: From JERRY GLYNN to PUBLIC about ALGEBRAIC EXPANSION

We think of expanding (x + 1)^3 as algebra and when we do Taylor expansions we think we're doing calculus and we generate polynomials for sin(x) and exp(x) and others. Try simplifying Taylor((x+1)^3,x,0,3). Looks like we can do algebra with calculus tools. It might also make better sense for students to start with a familiar expansion when first doing taylor. (x+1)^3=a0+a1*x+a2*x^2+a3*x^3 and differentiate the whole thing and simplify and substitute 0 for x and solve to work out the coefficients.

Message 3170: From SOFT WAREHOUSE to PUBLIC about ROOTS OF UNITY

The following definition returns a vector of the nth roots of unity ordered counter-clockwise in the complex plain beginning with the real root:

$$\text{NTH\_ROOTS\_OF\_UNITY}(n) :\!= \text{VECTOR}\left(e^{i\cdot\pi\cdot 2\cdot k/n}, k, 0, n - 1\right)$$

For example, to compute the 5th roots of unity enter and simplify the expression NTH_ROOTS_OF_UNITY(5) giving

NTH_ROOTS_OF_UNITY(5)

$$\left[1, \frac{\sqrt{5}}{4} - \frac{1}{4} + \frac{i\cdot\sqrt{(2\cdot\sqrt{5} + 10)}}{4}, -\frac{\sqrt{5}}{4} - \frac{1}{4} + \frac{i\cdot\sqrt{(10 - 2\cdot\sqrt{5})}}{4}, -\frac{\sqrt{5}}{4} - \frac{1}{4} - \frac{i\cdot\sqrt{(10 - 2\cdot\sqrt{5})}}{4},\right.$$
$$\left.\frac{\sqrt{5}}{4} - \frac{1}{4} - \frac{i\cdot\sqrt{(2\cdot\sqrt{5} + 10)}}{4}\right]$$

Message 3172: From SOFT WAREHOUSE to JERRY GLYNN about MULTIPLE EQUATION SOLVING

The following definition is in response to Jerry Glynn's request for a function that individually solves each equation in a list of equations and returns the result as a list of lists of roots:

```
SOLVE_LIST(v,x):=VECTOR(SOLVE(ELEMENT(v,n_),x),n_,1,DIMENSION(v))
```

Then for example, simplifying SOLVE_LIST([x-1,x^2-1,x^3-1,x^4-1],x) results in the list of lists:

```
[[x=1], [x=1,x=-1],
 [x=1,x=-1/2-SQRT(3)*#i/2,x=-1/2+SQRT(3)*#i/2],
 [x=1,x=-1,x=-#i,x=#i]]
```

SOLVE from 1994 has changed its behaviour from 1994 up to now. As you know there is a difference between SOLVE and SOLUTIONS. We can also write the VECTOR-expression in a more compact form:

#1: $\text{equs} := \left[x - 1,\ x^2 - 1,\ x^3 - 1,\ x^4 - 1\right]$

#2: $\text{SOLVE\_LIST}(v,\ x) := \text{VECTOR}(\text{SOLVE}(v_{n\_},\ x),\ n\_,\ 1,\ \text{DIM}(v))$

#3: SOLVE_LIST(equs)

#4: $\left[x = 1,\ x = -1 \lor x = 1,\ x = -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} \lor x = -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} \lor x = 1,\ x = -i \lor x = i \lor x = -1 \lor x = 1\right]$

#5: $\text{SOLUTIONS\_LIST}(v,\ x) := \text{VECTOR}(\text{SOLUTIONS}(v_{n\_},\ x),\ n\_,\ 1,\ \text{DIMENSION}(v))$

#6: SOLUTIONS_LIST(equs)

#7: $\left[[1],\ [1,\ -1],\ \left[1,\ -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2},\ -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}\right],\ [1,\ -1,\ i,\ -i]\right]$

#8: $\text{SOLVE\_L\_new}(v,\ x) := \text{VECTOR}(\text{SOLVE}(v\_,\ x),\ v\_,\ v)$

#9: SOLVE_L_new(equs)

#10: $\left[x = 1,\ x = -1 \lor x = 1,\ x = -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} \lor x = -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} \lor x = 1,\ x = -i \lor x = i \lor x = -1 \lor x = 1\right]$

#11: $\text{SOLS\_L\_new}(v,\ x) := \text{VECTOR}(\text{SOLUTIONS}(v\_,\ x),\ v\_,\ v)$

#12: SOLS_L_new(equs, x)

#13: $\left[[1],\ [1,\ -1],\ \left[1,\ -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2},\ -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}\right],\ [1,\ -1,\ i,\ -i]\right]$

During the last years I received a lot of letters and I had a lot of talks and very often they ended:

It would be fine if we could .... in DERIVE.

I repeat some of your ideas:

Functions like real(x), integer(x), ...; GCD and LCM for polynomials; the commands of the menu should be "official" DERIVE-commands, eg. FACTOR(expression),....; the parameters of the Options should be implemented in commands; let´s have a command to declare a variable´s domain; (Dr. Künzer, Bruneck, Italy in his last letter).

We want to create our own menus in German (or any other language) and without some commands - our students shouldn´t be able to use soLve in the beginning; .....

I wished I had an easy way to create own DEMO-files;

Couldn´t it be possible to have a RENUMBER command to avoid a line number chaos?

When I want to load a .MTH - file I often have to use MANAGE>EXECUTE>DIR to remember the file´s name ....

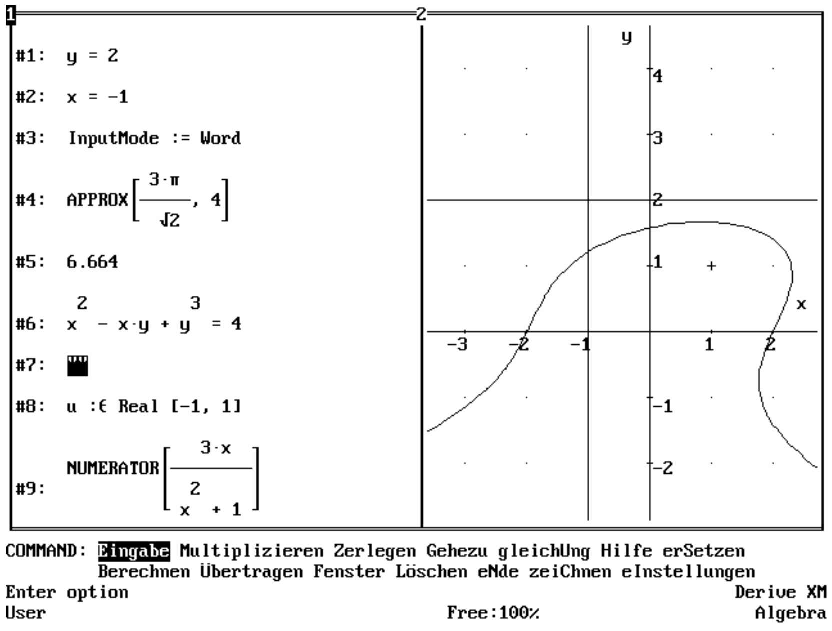
I need another software to insert the scale

I cannot explain my students why x = 3 in DERIVE is a horizontal line!!

and, and, and ......

Now please look at the Hardcopy: that is DERIVE 3.0:

I think you will see many of your ideas realized. I am sure that the DERIVE USER GROUP as a whole took an important part in improving DERIVE. And I´m also sure that a product like DERIVE never will be perfect - and I hope so - because we and the „Fathers of DERIVE" need the challenge to go on.


#1: $y = 2$

#2: $x = -1$

#3: InputMode := Word

#4: $\text{APPROX}\left[\frac{3\cdot\pi}{\sqrt{2}}, 4\right]$

#5: 6.664

#6: $x^2 - x\cdot y + y^3 = 4$

#7:

#8: $u :\in \text{Real } [-1, 1]$

#9: $\text{NUMERATOR}\left[\frac{3\cdot x}{x^2 + 1}\right]$

```
COMMAND: Eingabe Multiplizieren Zerlegen Gehezu gleichUng Hilfe erSetzen
         Berechnen Übertragen Fenster Löschen eNde zeiChnen eInstellungen
Enter option                                                    Derive XM
User                                     Free:100%                Algebra
```

I´ve changed the ALGEBRA-Menu for DERIVE novices (in German). 'erSetzen' is a shortcut for Manage Substitute, some command the students don´t need have been omitted (= hidden) (Calculus, Manage,....). Expression #6 is an implicit function with, you can see the respective plot in the Plot Window.

See also on page 17 a short DERIVE 3.0 Listing. Look there at the effect of the new **Annotate** Option

If you would like to have more information about DERIVE´s new features, then call or write. I´ll send a copy of the **Version 3 Release Notes.**

Dear colleague
I am a lecturer at the Faculty of Education. I have a Ph.D. in Mathematics and a Ph.D. in Computer Science. My works are mainly about applications of computers to Mathematics and about Teaching Mathematics with computers......

Eugenio Roanes

I met Eugenio in Plymouth and we had lots of fun together. Eugenio is an excellent tennis player and a great milk drinker and he is an expert in railways. In the Dartmoor Nature Park he gave a lecture about the railway there when we walked along an old railway line.

Some of Eugenio´s functions are still implemented in later DERIVE versions. Nevertheless they are a fine example of sophisticated programming with DERIVE especially using its recursive capabilities. Muchas gracias Eugenio!

DEVELOPMENT OF SOME ALGEBRAIC OPERATIONS ON POLYNOMIALS IN DERIVE

Eugenio Roanes, Madrid, Spain

Introduction.

Derive is nowadays a common assistant for teaching mathematics both at high schools and in the first years of university courses. We understand that it is intended for use as an algebraic calculator (it supports exact arithmetic) with graphical capabilities.

We think that, because of this orientation, it is somewhat lacking, especially in polynomial handling. We aim to show in this paper how we have tried to overcome this, using the possibility of defining new functions of the 2.xx versions.

We know that, as we cannot access internal representations of polynomials, there will be no quick way to obtain, for instance, a coefficient. However, there are times when these functions are really needed. Implementations of the functions of §2.1 and §2.2 are included in the file MISC.MTH, but they use differentiation.

1 Some new integer functions.

1.1 Division of integers

DERIVE possesses a division function in exact arithmetic, but doesn´t include a division of integers. A possible way of calculating the remainder of dividing two positive integers would be to use the following:

```
Algorithm:
If a < b: RESTO(a,b) = a
In other cases: RESTO(a,b) = RESTO(a-b,b)
```

Implementation:

```
       resto(a, b) :=
         If a < b
#1:         a
            resto(a - b, b)

#2:    [resto(384, 35), resto(35, 384)] = [34, 35]
```

Version 2.52 includes the function FLOOR(number) - version 2.03 doesn´t. Using the former, it should be better to define:

```
#3:   resto2(a, b) := a - b·FLOOR(a/b)

#4:   [resto2(384, 35), resto2(35, 384)] = [34, 35]
```

1.2 Greatest Common Divisor.

The above makes it possible to define a function GCD in N using Euclid´s algorithm. We have done it this way:

Algorithm:

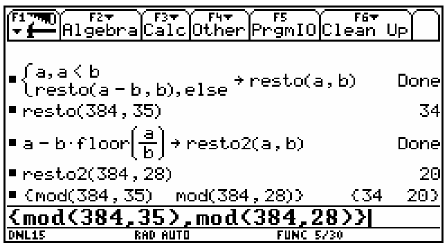
`MCD(a,b) decides which of the entries is greater and sends to AUX(greater{a,b},smaller{a,b}).`
`AUX(a,b) is Euclid´s Algorithm: the division of integers is repeated, and at every step the divisor becomes the dividend and the remainder becomes the divisor. When the remainder is 0, the previous remainder is given as output.`

Implementation:

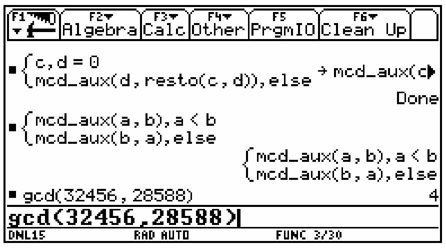
```
      MCD_AUX(c, d) :=
        If d = 0
#5:        c
           MCD_AUX(d, resto(c, d))

      MCD(a, b) :=
        If a > b
#6:        MCD_AUX(a, b)
           MCD_AUX(b, a)

#7:   MCD(32456, 28588) = 4
```

We do have now function GCD(a,b) - which is working for numbers only!!

```
#8:   GCD(32456, 28588) = 4
```



```
■ {a,a < b
  {resto(a - b,b),else → resto(a,b)        Done
■ resto(384,35)                              34
■ a - b·floor(a/b) → resto2(a,b)           Done
■ resto2(384,28)                             20
■ {mod(384,35)  mod(384,28)}          {34   20}
{mod(384,35),mod(384,28)}
DNL15          RAD AUTO          FUNC 5/30
```



```
■ {c,d = 0
  {mcd_aux(d,resto(c,d)),else → mcd_aux(c▸
                                           Done
■ {mcd_aux(a,b),a < b
  {mcd_aux(b,a),else
                       {mcd_aux(a,b),a < b
                       {mcd_aux(b,a),else
■ gcd(32456,28588)                            4
gcd(32456,28588)
DNL15          RAD AUTO          FUNC 3/30
```

And there is a function on the handheld, too

2 Some new functions for univariate polynomials.

Unfortunately, DERIVE only allows the substitution of a variable by a certain value in a function of a polynomial if the user is in the interactive mode. So, we can only find the possibility of defining a substituting function: SUB(polynomial, variable, value) as the limit of the function "polynomial" when "variable" tends towards "value":

```
      sub(p, v, a) := lim p
#9:                   v→a
```

In the meanwhile function `SUBST` is available and we don´t need `sub(p,v,a)` any longer.

2.1 Degree of a polynomial.

The degree 0 term of a polynomial p(x) is p(0). Then p(x) - p(0) is divisible by x. The times we can repeat this process until we obtain 0 is the degree of the original polynomial.

Implementation:

#10: $\mathrm{POL\_AUX}(p, v) := \dfrac{p - \mathrm{SUBST}(p, v, 0)}{v}$

```
       GRADO(p, v) :=
         If p = 0
#11:        -1
            1 + GRADO(POL_AUX(p, v), v)
            1 + GRADO(POL_AUX(p, v), v)
```

#12: $\mathrm{GRADO}\left((3\cdot x^3 + 2\cdot x - 1)\cdot\left(3\cdot x - 3\cdot x^2 + 2\cdot x^4 - \dfrac{x^5}{2}\right), x\right) = 8$

2.2 Coefficient of the term of degree n.

Algorithm:

$p_0(x) = p(x)$

$p_n(x) = (p_{n-1}(x) - p_{n-1}(0))/x$

$\mathrm{COEFF}(p(x),n) = p_n(0)$

Implementation:

#13: $(3\cdot x^3 + 2\cdot x - 1)\cdot\left(3\cdot x - 3\cdot x^2 + 2\cdot x^4 - \dfrac{x^5}{2}\right)$

#14: $-\dfrac{3\cdot x^8}{2} + 6\cdot x^7 - x^6 - \dfrac{9\cdot x^5}{2} + 7\cdot x^4 - 6\cdot x^3 + 9\cdot x^2 - 3\cdot x$

```
       POL_AUX2(p, v, n) :=
         If n = 0
#15:        p
            (POL_AUX2(p, v, n - 1) - SUBST(POL_AUX2(p, v, n - 1), v, 0))/v
```

#16: COEFF(p, v, n) := SUBST(POL_AUX2(p, v, n), v, 0)

#17: $\mathrm{COEFF}\left((3\cdot x^3 + 2\cdot x - 1)\cdot\left(3\cdot x - 3\cdot x^2 + 2\cdot x^4 - \dfrac{x^5}{2}\right), x, 5\right) = -\dfrac{9}{2}$

#18: LCOEFF(p, v) := COEFF(p, v, GRADO(p, v))

#19: $\mathrm{LCOEFF}\left((3\cdot x^3 + 2\cdot x - 1)\cdot\left(3\cdot x - 3\cdot x^2 + 2\cdot x^4 - \dfrac{x^5}{2}\right), x\right) = -\dfrac{3}{2}$

2.3 Quotient and remainder of dividing a polynomial by (x – a).

This case is treated separately in order to apply Ruffini´s Rule[1] and the Remainder Theorem:
The remainder of the division of p(x) by (x –-a) is p(a). So, we can write the function `RESTO_RUF(polynomial, value, variable)` as simply:

```
#20:  RESTO_RUF(p, a, v) := SUBST(p, v, a)
```

$$\#21:\quad \text{RESTO\_RUF}\left(3\cdot x - 3\cdot x^2 + 2\cdot x^4 - \frac{x^5}{2},\ 4,\ x\right) = -36$$

[1] You can find Comments on Ruffini on page 47.

The quotient of the process of Ruffini´s rule can be calculated following the

Algorithm:

```
(RUF_AUX obtains the coefficient of the term of degree n when
dividing by (x - a)).
RUF_AUX(p(x),a,δ(p(x))-1)=leader coeff. of p(x).
If n < δ(p(x))-1 then:
RUF_AUX(p(x),a,v,n)=a*RUF_AUX(p(x),a,v,n+1)+coeff.of the term
of degree (n+1) of p(x).
COC_RUF(p(x),a,v) = sum of the coefficients multiplied by the
variable to the power of the corresponding degree.
```

Implementation:

```
       RUF_AUX(p, a, v, n) :=
         If n < GRADO(p, v) - 1
#22:        a·RUF_AUX(p, a, v, n + 1) + coeff(p, v, n + 1)
            coeff(p, v, n + 1)
```

$$\#23:\quad \text{COC\_RUF}(p,\ a,\ v) := \sum_{n=0}^{\text{GRADO}(p,\ v) - 1} \text{RUF\_AUX}(p,\ a,\ v,\ n)\cdot v^n$$

$$\#24:\quad \text{COC\_RUF}\left(3\cdot x - 3\cdot x^2 + 2\cdot x^4 - \frac{x^5}{2},\ 4,\ x\right) = -\frac{x^4}{2} - 3\cdot x - 9$$

$$\#25:\quad \text{EXPAND}\left(\frac{3\cdot x - 3\cdot x^2 + 2\cdot x^4 - \frac{x^5}{2}}{x - 4},\ \text{Trivial}\right) = -\frac{36}{x - 4} - \frac{x^4}{2} - 3\cdot x - 9$$

2.4 Quotient and remainder of dividing two polynomials.

Algorithm:

If δ(dividend)< δ(divisor), return the dividend, else

$$\text{new dividend} = \text{dividend} - \frac{\text{leader coeff dividend}}{\text{leader coeff divisor}} * \text{variable}^{\text{difference of degrees}} * \text{divisor}$$

Implementation:

```
      RESTO_POL(ddo, dor, v) :=
        If GRADO(ddo, v) < GRADO(dor, v)
#26:       ddo
           RESTO_POL(ddo - dor·LCOEFF(ddo, v)/LCOEFF(dor, v)·v^(GRADO(ddo, v) - GRADO(dor, v)), dor, v)
```

If we are interested in the quotient, the division of polynomials function available from DERIVE can be used. The difference between the dividend and the remainder is divisible by the divisor. So, using the previous `RESTO_POL`, we define:

$$\#27:\quad \mathrm{COC\_POL}(p,\ q,\ v) := \frac{p - \mathrm{RESTO\_POL}(p,\ q,\ v)}{q}$$

To calculate the pseudoremainder we shall multiply the dividend by the leader coefficient of the divisor to the power of the difference of degrees plus one. This way we can ensure that if the polynomials belong to Z(x), no rational coefficient will appear in the calculations.

Implementation:

$$\#28:\quad \mathrm{SRESTO\_POL}(p,\ q,\ v) := \mathrm{RESTO\_POL}(\mathrm{LCOEFF}(q,\ v)^{1 + \mathrm{GRADO}(p,\ v) - \mathrm{GRADO}(q,\ v)}\cdot p,\ q,\ v)$$

$$\#29:\quad \mathrm{COC\_POL}(3\cdot x^7 - x^5 + 3\cdot x^4 - x^2 - 2,\ 4\cdot x^3 - 3\cdot x^2 + x + 1,\ x)$$

$$\#30:\quad \frac{768\cdot x^4 + 576\cdot x^3 - 16\cdot x^2 + 420\cdot x + 175}{1024}$$

$$\#31:\quad \mathrm{RESTO\_POL}(3\cdot x^7 - x^5 + 3\cdot x^4 - x^2 - 2,\ 4\cdot x^3 - 3\cdot x^2 + x + 1,\ x)$$

$$\#32:\quad -\frac{903\cdot x^2 + 595\cdot x + 2223}{1024}$$

$$\#33:\quad \mathrm{SRESTO\_POL}(3\cdot x^7 - x^5 + 3\cdot x^4 - x^2 - 2,\ 4\cdot x^3 - 3\cdot x^2 + x + 1,\ x)$$

$$\#34:\quad - 903\cdot x^2 - 595\cdot x - 2223$$

(The computing times were about 10 seconds in 1994. Compare with computing times of today!!)

2.5 GCD of two polynomials

Euclid´s Algorithm also works for calculating the GCD of two polynomials. The only changes which have to be made are:

- Order the polynomials in the first step according to their degree,
- Substitute the function: remainder of the division of integers by the function: remainder of the division of polynomials.

One disadvantage of this algorithm is that the coefficients may increase greatly in size.

Implementation:

```
       MCD_POL_AUX(c, d, v) :=
         If d = 0
#35:        c
            MCD_POL_AUX(d, RESTO_POL(c, d, v), v)
            MCD_POL_AUX(d, RESTO_POL(c, d, v), v)

       MCD_POL(a, b, v) :=
         If GRADO(a, v) > GRADO(b, v)
#36:        MCD_POL_AUX(a, b, v)
            MCD_POL_AUX(a, b, v)
```

$$\#37:\quad \mathrm{MCD\_POL}(x^3 - 8\cdot x^2 + 21\cdot x - 18,\ x^3 + 9\cdot x^2 + 27\cdot x + 27,\ x) = -\frac{144500}{3249}$$

$$\#38:\quad \mathrm{MCD\_POL}(x^5 - 5\cdot x^4 + 4\cdot x^3 + x^2 - 5\cdot x + 4,\ x^4 - 5\cdot x^3 + 5\cdot x^2 - 5\cdot x + 4,\ x) = 2\cdot x^2 - 10\cdot x + 8$$

$$\#39:\quad \mathrm{MCD\_POL}(x^4 - 7\cdot x^3 + 18\cdot x^2 - 22\cdot x + 12,\ 3\cdot x^3 - 13\cdot x^2 + 8\cdot x + 12,\ x) = \frac{34\cdot (x^2 - 5\cdot x + 6)}{9}$$

The pseudoremainder could be used for polynomials of Z(x) (instead of the remainder). In this way all the polynomials in every step of the algorithm should have integer coefficients:

Implementation:

```
      MCD_POL_AUX_Z(c, d, v) :=
        If d = 0
#40:       c
           MCD_POL_AUX_Z(d, SRESTO_POL(c, d, v), v)
           MCD_POL_AUX_Z(d, SRESTO_POL(c, d, v), v)

      MCD_POL_Z(a, b, v) :=
        If GRADO(a, v) > GRADO(b, v)
#41:       MCD_POL_AUX_Z(a, b, v)
           MCD_POL_AUX_Z(a, b, v)

                  4       3       2                    3        2                             2
#42:  MCD_POL_Z(x  - 7·x  + 18·x  - 22·x + 12, 3·x  - 13·x  + 8·x + 12, x) = 34·(x  - 5·x + 6)
```

Bibliography

[1] J.Johnson, B. Evans: *Discovering Calculus with DERIVE.* John Wiley and Sons, 1992

[2] J.L. Llorens: *Introducción al uso de DERIVE.* Universidad Politécnica de Valencia, 1993

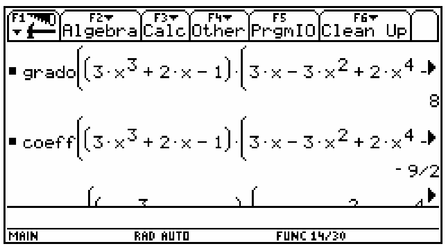
[3] A. Rich, J. Rich, D. Stoutemyer: *DERIVE User´s Manual.* Soft Warehouse, 1990

[4] *The Bulletin of the DERIVE USER GROUP*

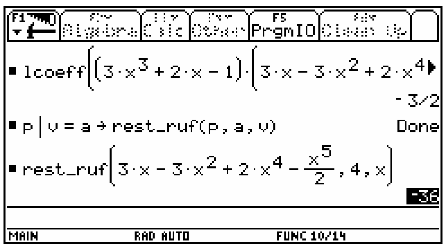
Now with DERIVE 3.0 and higher ...

```
                                                          5
                   3                        2      4     x
#43:  POLY_COEFF((3·x  + 2·x - 1)·(3·x - 3·x  + 2·x  - ——), x, 5) = - 9/2
                                                          2

                4       3       2                    3        2                             2
#44:  MCD_POL_Z(x  - 7·x  + 18·x  - 22·x + 12, 3·x  - 13·x  + 8·x + 12, x) = 34·(x  - 5·x + 6)


               4       3       2                    3        2                  2
#45:  POLY_GCD(x  - 7·x  + 18·x  - 22·x + 12, 3·x  - 13·x  + 8·x + 12) = x  - 5·x + 6

                                                                    4       3      2
                    7     5      4    2         3      2         3·x     9·x      x     105·x     175
#46:  QUOTIENT(3·x  - x  + 3·x  - x  - 2, 4·x  - 3·x  + x + 1) = ———— + ———— - ——— + ————— + ————
                                                                  4       16      64     256      1024

                                                                          2
                     7     5      4    2         3      2          903·x      595·x     2223
#47:  REMAINDER(3·x  - x  + 3·x  - x  - 2, 4·x  - 3·x  + x + 1) = - ——————— - ————— - —————
                                                                    1024       1024      1024
```

Now let me try to implement Eugenio´s procedures on the TI-handheld devices. As you can see in the following, they works without any problems. It is easy to repeat the recursive functions or to re-write them using while-loops. First I worked with the TI-92 / Voyage 200. After installing the new TI-Nspire Computer Link Software I defined the functions and repeated the whole procedure also on the Nspire. It is a pity that it is not able to transmit the TI-92/V 200 functions to the Nspire although the CAS-machine is more or less quite the same. See first the results (the first screens from the V 200):

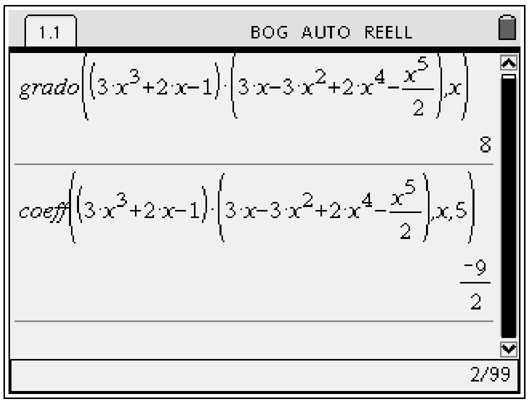


```
■ grado((3·x^3 + 2·x - 1)·(3·x - 3·x^2 + 2·x^4 ▸
                                                 8
■ coeff((3·x^3 + 2·x - 1)·(3·x - 3·x^2 + 2·x^4 ▸
                                              - 9/2
MAIN          RAD AUTO          FUNC 14/30
```



```
■ lcoeff((3·x^3 + 2·x - 1)·(3·x - 3·x^2 + 2·x^4 ▸
                                              - 3/2
■ p|v = a → rest_ruf(p, a, v)                  Done
■ rest_ruf(3·x - 3·x^2 + 2·x^4 - x^5/2, 4, x)
                                                -36
MAIN          RAD AUTO          FUNC 10/14
```

Here are the screen shots from the Nspire. You can download the TI-92/V200 and the Nspire files as well. The PC-Nspire works identically.

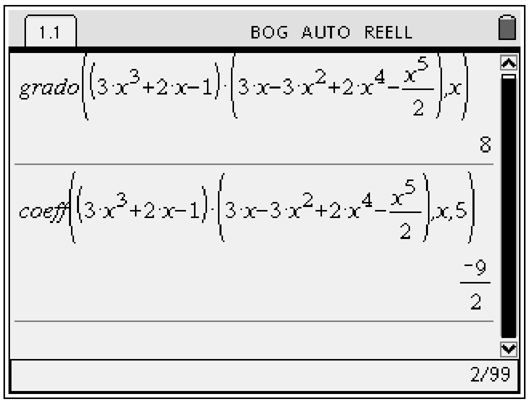


1.1 BOG AUTO REELL

$grado\left((3\cdot x^3+2\cdot x-1)\cdot\left(3\cdot x-3\cdot x^2+2\cdot x^4-\frac{x^5}{2}\right),x\right)$ 8

$coeff\left((3\cdot x^3+2\cdot x-1)\cdot\left(3\cdot x-3\cdot x^2+2\cdot x^4-\frac{x^5}{2}\right),x,5\right)$ $\frac{-9}{2}$

2/99

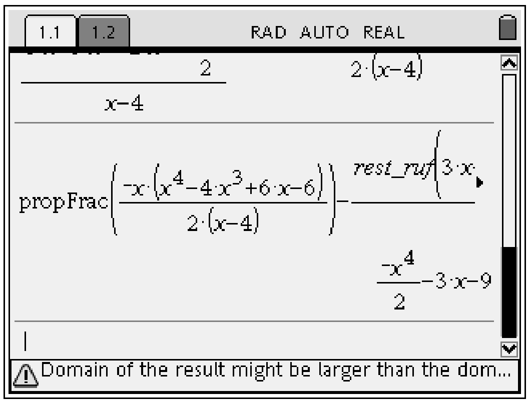


1.1 BOG AUTO REELL

$grado\left((3\cdot x^3+2\cdot x-1)\cdot\left(3\cdot x-3\cdot x^2+2\cdot x^4-\frac{x^5}{2}\right),x\right)$ 8

$coeff\left((3\cdot x^3+2\cdot x-1)\cdot\left(3\cdot x-3\cdot x^2+2\cdot x^4-\frac{x^5}{2}\right),x,5\right)$ $\frac{-9}{2}$

2/99



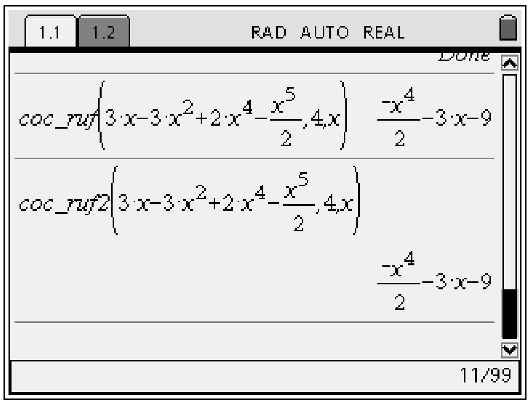
1.1 1.2 RAD AUTO REAL

$\frac{2}{x-4}$ $\quad 2\cdot(x-4)$

$propFrac\left(\frac{-x\cdot(x^4-4\cdot x^3+6\cdot x-6)}{2\cdot(x-4)}\right)-\frac{rest\_ruf(3\cdot x}{}$ ▸

$\frac{-x^4}{2}-3\cdot x-9$

Domain of the result might be larger than the dom...



1.1 1.2 RAD AUTO REAL

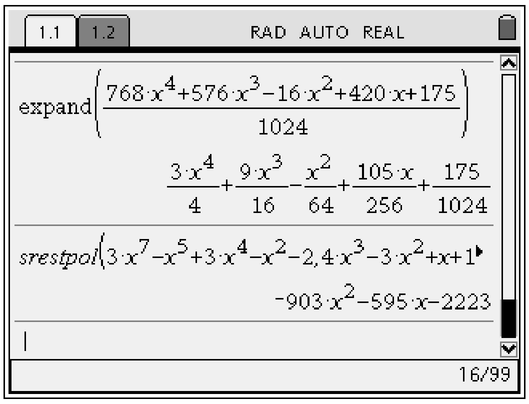
Done

$coc\_ruf\left(3\cdot x-3\cdot x^2+2\cdot x^4-\frac{x^5}{2},4,x\right)$ $\frac{-x^4}{2}-3\cdot x-9$

$coc\_ruf2\left(3\cdot x-3\cdot x^2+2\cdot x^4-\frac{x^5}{2},4,x\right)$

$\frac{-x^4}{2}-3\cdot x-9$

11/99



1.1 1.2 RAD AUTO REAL

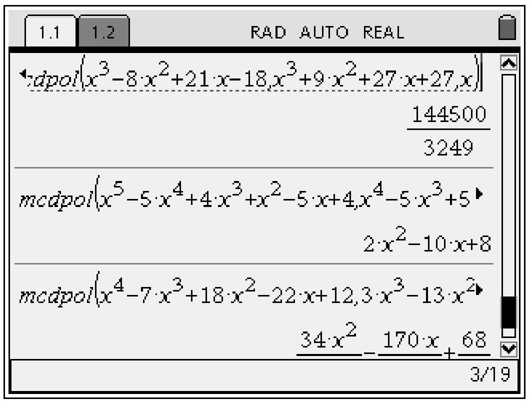
$expand\left(\frac{768\cdot x^4+576\cdot x^3-16\cdot x^2+420\cdot x+175}{1024}\right)$

$\frac{3\cdot x^4}{4}+\frac{9\cdot x^3}{16}-\frac{x^2}{64}+\frac{105\cdot x}{256}+\frac{175}{1024}$

$srestpol(3\cdot x^7-x^5+3\cdot x^4-x^2-2,4\cdot x^3-3\cdot x^2+x+1$ ▸

$-903\cdot x^2-595\cdot x-2223$

16/99



1.1 1.2 RAD AUTO REAL

◂$:dpol(x^3-8\cdot x^2+21\cdot x-18,x^3+9\cdot x^2+27\cdot x+27,x)$

$\frac{144500}{3249}$

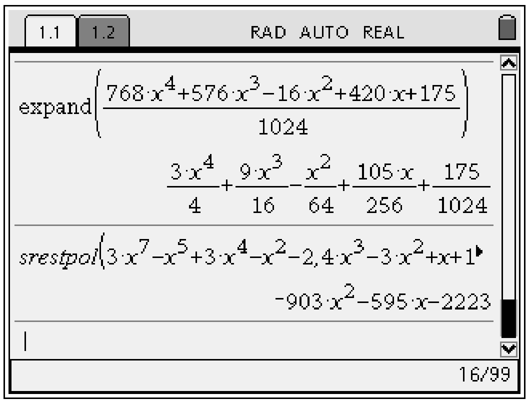
$mcdpol(x^5-5\cdot x^4+4\cdot x^3+x^2-5\cdot x+4,x^4-5\cdot x^3+5$ ▸

$2\cdot x^2-10\cdot x+8$

$mcdpol(x^4-7\cdot x^3+18\cdot x^2-22\cdot x+12,3\cdot x^3-13\cdot x^2$ ▸

$\frac{34\cdot x^2}{}-\frac{170\cdot x}{}+\frac{68}{}$

3/19



1.1 1.2 RAD AUTO REAL

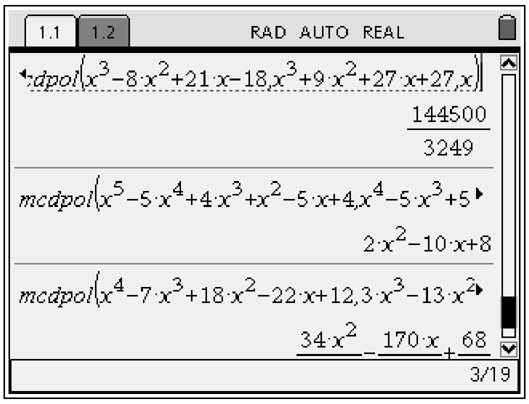
$expand\left(\frac{768\cdot x^4+576\cdot x^3-16\cdot x^2+420\cdot x+175}{1024}\right)$

$\frac{3\cdot x^4}{4}+\frac{9\cdot x^3}{16}-\frac{x^2}{64}+\frac{105\cdot x}{256}+\frac{175}{1024}$

$srestpol(3\cdot x^7-x^5+3\cdot x^4-x^2-2,4\cdot x^3-3\cdot x^2+x+1$ ▸

$-903\cdot x^2-595\cdot x-2223$

16/99



1.1 1.2 RAD AUTO REAL

◂$:dpol(x^3-8\cdot x^2+21\cdot x-18,x^3+9\cdot x^2+27\cdot x+27,x)$

$\frac{144500}{3249}$

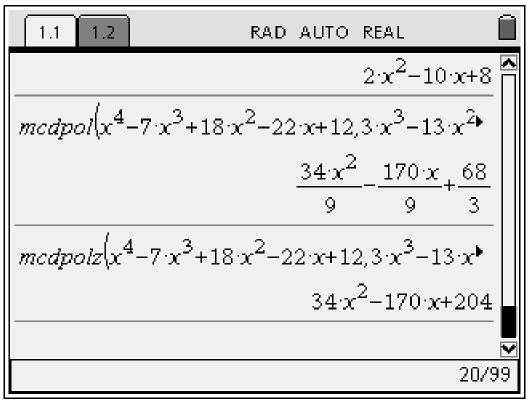
$mcdpol(x^5-5\cdot x^4+4\cdot x^3+x^2-5\cdot x+4,x^4-5\cdot x^3+5$ ▸

$2\cdot x^2-10\cdot x+8$

$mcdpol(x^4-7\cdot x^3+18\cdot x^2-22\cdot x+12,3\cdot x^3-13\cdot x^2$ ▸

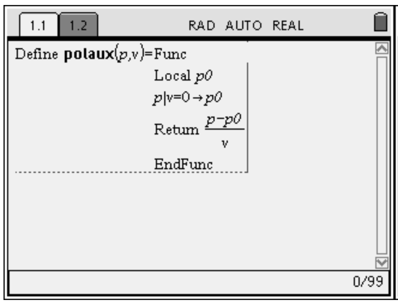
$\frac{34\cdot x^2}{}-\frac{170\cdot x}{}+\frac{68}{}$

3/19

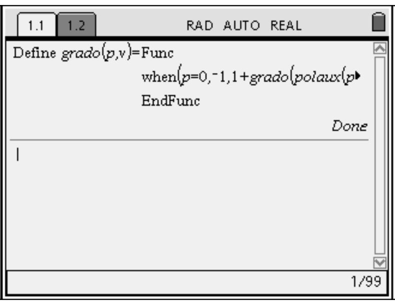


```
2·x^2-10·x+8
mcdpol(x^4-7·x^3+18·x^2-22·x+12,3·x^3-13·x^2▸
                          34·x^2/9 - 170·x/9 + 68/3
mcdpolz(x^4-7·x^3+18·x^2-22·x+12,3·x^3-13·x▸
                          34·x^2-170·x+204
                                              20/99
```

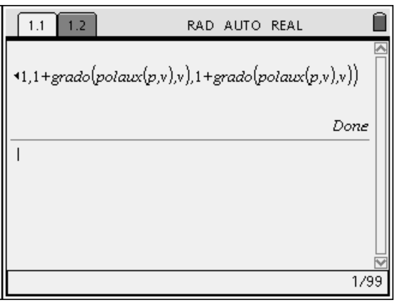
And here are the functions:



```
Define polaux(p,v)=Func
        Local p0
        p|v=0→p0
        Return (p-p0)/v
        EndFunc
                                              0/99
```

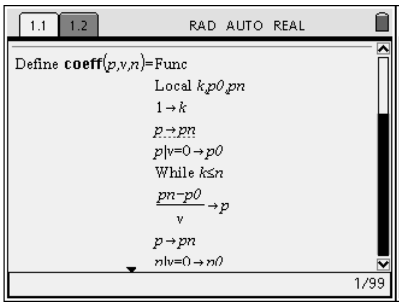


```
Define grado(p,v)=Func
        when(p=0,⁻1,1+grado(polaux(p▸
        EndFunc
                                              Done
                                              1/99
```

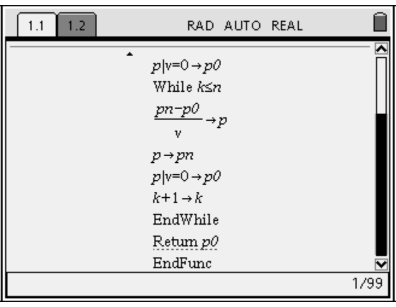


```
◂1,1+grado(polaux(p,v),v),1+grado(polaux(p,v),v))
                                              Done
                                              1/99
```

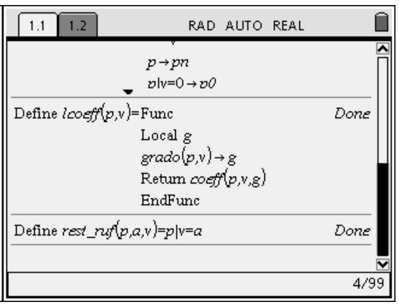
coeff, lcoeff and rest_ruf:



```
Define coeff(p,v,n)=Func
        Local k,p0,pn
        1→k
        p→pn
        p|v=0→p0
        While k≤n
        (pn-p0)/v→p
        p→pn
        p|v=0→p0
                                              1/99
```

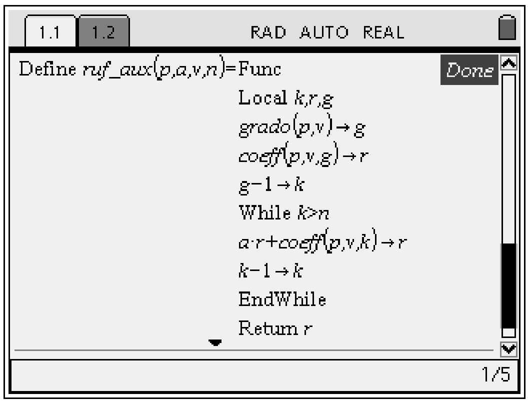


```
        p|v=0→p0
        While k≤n
        (pn-p0)/v→p
        p→pn
        p|v=0→p0
        k+1→k
        EndWhile
        Return p0
        EndFunc
                                              1/99
```

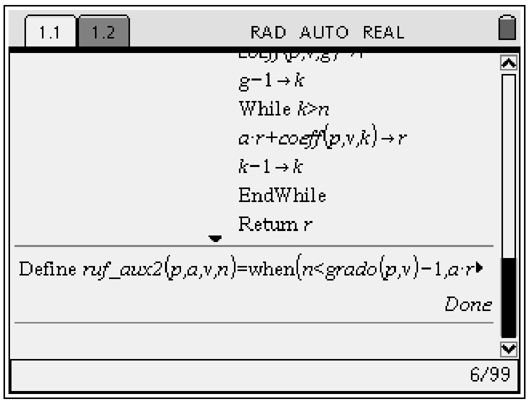


```
        p→pn
        p|v=0→p0
Define lcoeff(p,v)=Func                       Done
        Local g
        grado(p,v)→g
        Return coeff(p,v,g)
        EndFunc
Define rest_ruf(p,a,v)=p|v=a                  Done
                                              4/99
```

For ruf_aux am using first a while-loop instead of a recursive function which is used in ruf_aux2.

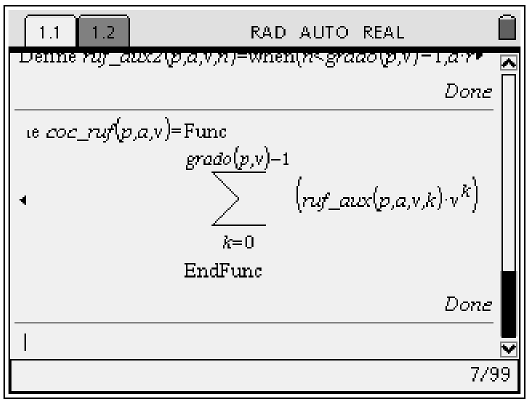


```
Define ruf_aux(p,a,v,n)=Func                  Done
        Local k,r,g
        grado(p,v)→g
        coeff(p,v,g)→r
        g-1→k
        While k>n
        a·r+coeff(p,v,k)→r
        k-1→k
        EndWhile
        Return r
                                              1/5
```

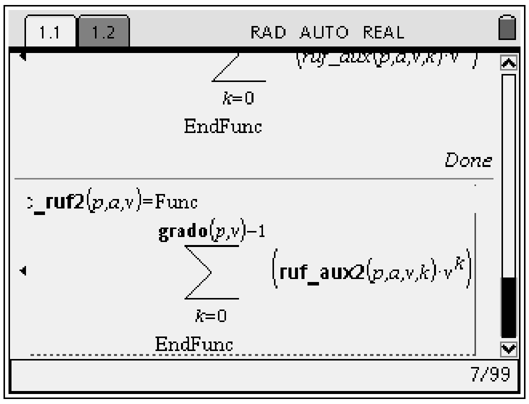


```
        g-1→k
        While k>n
        a·r+coeff(p,v,k)→r
        k-1→k
        EndWhile
        Return r
Define ruf_aux2(p,a,v,n)=when(n<grado(p,v)-1,a·r▸
                                              Done
                                              6/99
```

Compare ruf_aux and ruf_aux2: the while-loop versus the recursive function. Which way to program do you prefer?

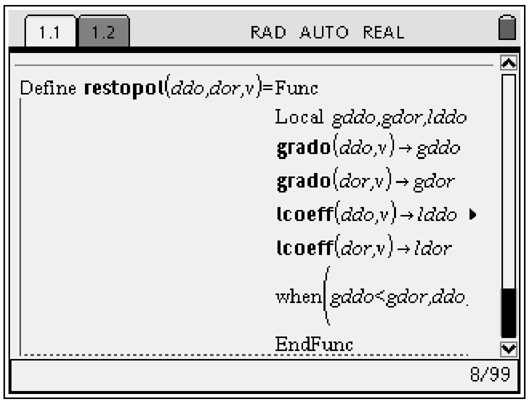


```
Define ruf_aux2(p,a,v,n)=when(n<grado(p,v)-1,a·v▸
                                               Done
ıe coc_ruf(p,a,v)=Func
          grado(p,v)-1
              Σ        (ruf_aux(p,a,v,k)·v^k)
             k=0
          EndFunc
                                               Done
```

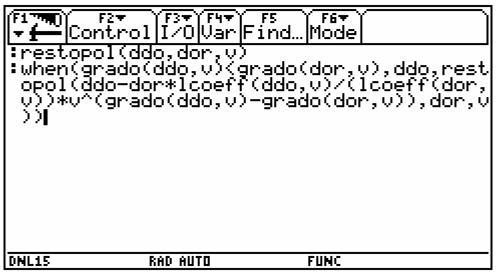


```
                       (ruf_aux(p,a,v,k)·v  )
             k=0
          EndFunc
                                               Done
:_ruf2(p,a,v)=Func
          grado(p,v)-1
              Σ        (ruf_aux2(p,a,v,k)·v^k)
             k=0
          EndFunc
```

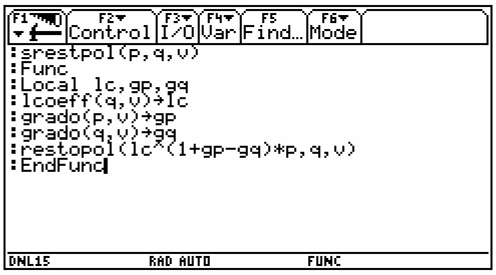
Both programming styles lead to the correct result. restopol on the Nspire and on the Voyage 200:



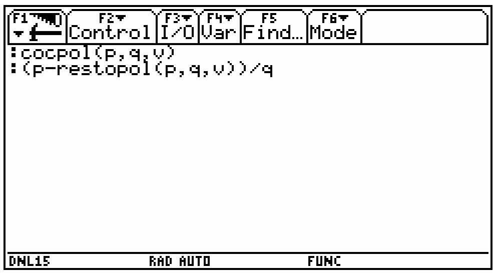
```
Define restopol(ddo,dor,v)=Func
                 Local gddo,gdor,lddo
                 grado(ddo,v)→gddo
                 grado(dor,v)→gdor
                 lcoeff(ddo,v)→lddo ▸
                 lcoeff(dor,v)→ldor
                 when(gddo<gdor,ddo,
                 EndFunc
```



```
:restopol(ddo,dor,v)
:when(grado(ddo,v)<grado(dor,v),ddo,rest
 opol(ddo-dor*lcoeff(ddo,v)/(lcoeff(dor,
 v))*v^(grado(ddo,v)-grado(dor,v)),dor,v
 ))
```

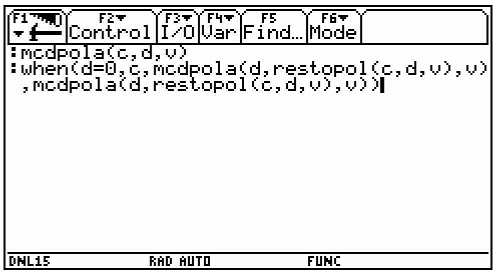


```
:srestpol(p,q,v)
:Func
:Local lc,gp,gq
:lcoeff(q,v)→lc
:grado(p,v)→gp
:grado(q,v)→gq
:restopol(lc^(1+gp-gq)*p,q,v)
:EndFunc
```

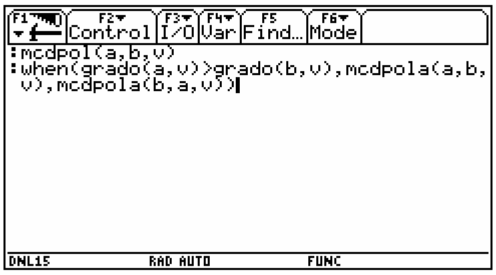


```
:cocpol(p,q,v)
:(p-restopol(p,q,v))/q
```

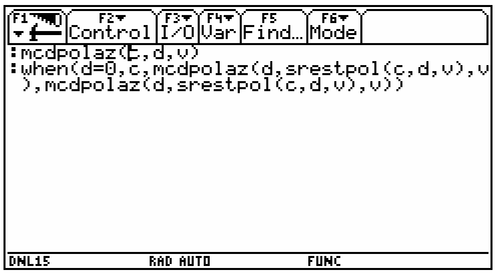
Finally the last group of functions (again on the V 200, because the screen fits better on the page!)



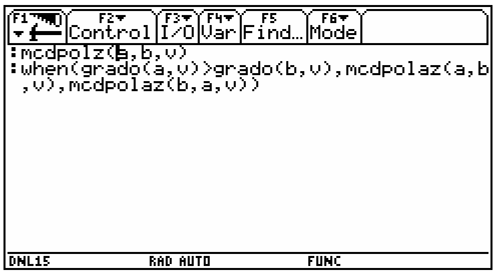
```
:mcdpola(c,d,v)
:when(d=0,c,mcdpola(d,restopol(c,d,v),v)
 ,mcdpola(d,restopol(c,d,v),v))
```



```
:mcdpol(a,b,v)
:when(grado(a,v)>grado(b,v),mcdpola(a,b,
 v),mcdpola(b,a,v))
```



```
:mcdpolaz(c,d,v)
:when(d=0,c,mcdpolaz(d,srestpol(c,d,v),v
 ),mcdpolaz(d,srestpol(c,d,v),v))
```



```
:mcdpolz(a,b,v)
:when(grado(a,v)>grado(b,v),mcdpolaz(a,b
 ,v),mcdpolaz(b,a,v))
```

The following comments are obsolete, because of changing the output of the SOLVE-command and introducing SOLUTIONS. But it might be of historical interest. Josef

Comments on the RHS-function

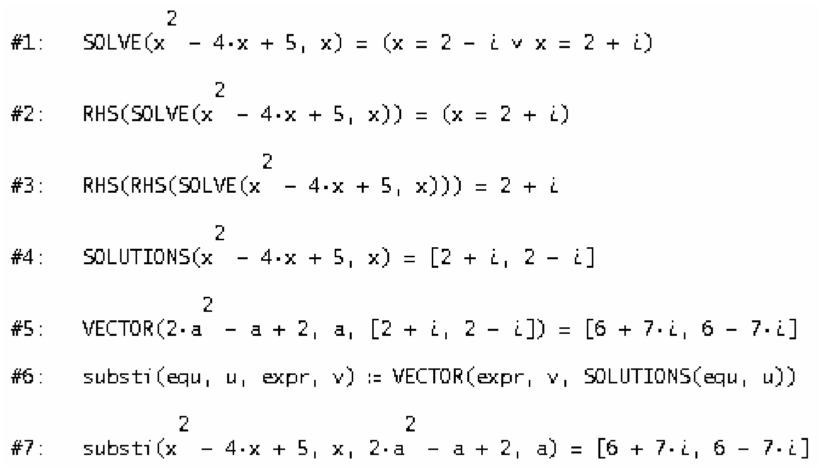
J.Böhm, Würmla, Würmla, Austria

V.Neurath asks in his letter from 15.06.94 to explain the "undocumented" RHS- and LHS-function I used in a file which I´ve sent him. RHS(relation) and LHS(relation) returns the righthand side or the lefthand side of a relation. The two functions are available since version 2.57. In the beginning I didn´t know where to use these functions but another question of V.Neurath and a letter from Dr. Kuenzer brought some ideas. I wanted to write a "DERIVE-program" to investigate a curve.
Many thanks for Dr.Kuenzer´s ideas (Symmetry, ...)

See first an example to illustrate the RHS-function: (This is DERIVE for DOS from 1994!)

```
1:  "Solution of equation equ for variable u:"
2:  SOL(equ, u) := SOLVE(equ, u)
          2
3:  SOL(x  - 4 x + 5, x)
4:  [x = 2 - î, x = 2 + î]
5:  "Using RHS we select the value of the 2nd solution:"
6:  RHS(ELEMENT([x = 2 - î, x = 2 + î], 2))
7:  2 + î
8:  "See the vector of the solutions (without x =)! Now I can use the solutions"
9:  VEC_SOL(equ, u) := VECTOR(RHS(ELEMENT(SOLVE(equ, u), i)), i, 1,
                       DIMENSION(SOLVE(equ, u)))
              2
10: VEC_SOL(x  - 4 x + 5, x)
11: [2 - î, 2 + î]
12: "I want to substitute the solutions of equ for variable v in expression expr:"
    SUBST(equ, u, expr, v) := VECTOR(            lim                   expr, i, 1,
13:                                 v->RHS(ELEMENT(SOLVE(equ, u), i))
                              DIMENSION(SOLVE(equ, u)))
           2                   2
14: SUBST(x  - 4 x + 5, x, 2 a  - a + 2, a)
15: [6 - 7 î, 6 + 7 î]
```

The appearance in modern DERIVE:



```
              2
#1:   SOLVE(x  - 4·x + 5, x) = (x = 2 - ί ∨ x = 2 + ί)

                  2
#2:   RHS(SOLVE(x  - 4·x + 5, x)) = (x = 2 + ί)

                      2
#3:   RHS(RHS(SOLVE(x  - 4·x + 5, x))) = 2 + ί

                  2
#4:   SOLUTIONS(x  - 4·x + 5, x) = [2 + ί, 2 - ί]

                 2
#5:   VECTOR(2·a  - a + 2, a, [2 + ί, 2 - ί]) = [6 + 7·ί, 6 - 7·ί]

#6:   substi(equ, u, expr, v) := VECTOR(expr, v, SOLUTIONS(equ, u))

               2                  2
#7:   substi(x  - 4·x + 5, x, 2·a  - a + 2, a) = [6 + 7·ί, 6 - 7·ί]
```

One of the many reasons I like DERIVE - it is not the most important one - is that it is a powerful tool for my work as schoolteacher. V.Neurath´s and Dr. Kuenzer´s letter made me curious if it could be possible to write a sort of program for "discussing a curve".
Some hours later I saw the product of my efforts and I was satisfied. Here you can see the listing. I think,it doesn´t need any further explication. The function has to be edited within brackets, when it is a rational function then enter **f := [numerator, denominator]**, because I don´t use version 3.xx.

This was the leading paragraph in DNL#15. I reprint the first lines of codes. Then I will proceed with a recent version of the "program" which is not really a program, but a collection of functions. Really programming became true with version 5. Josef

"Discussion of a curve or: how to write a 'program' with DERIVE!"

```
1:  f:=[]
2:  z:=ELEMENT(f,1)
3:  n:=IF(DIMENSION(f)=1,1,ELEMENT(f,2))
4:  [n1:=DIF(n,x,1),n2:=DIF(n,x,2),z1:=DIF(z,x,1),z2:=DIF(z,x,2),f1:=z/n]
5:  "Zeros"
6:  n0:=SOLVE(z,x)
7:  zer:=APPEND([["Zeros:",""],[f1=0,""]],VECTOR([IF(IM(RHS(ELEMENT(n0,i))),
         RHS(ELEMENT(n0,i)),"komplex"),IF(IM(RHS(ELEMENT(n0,i))),0,"!")],i,1,
         DIMENSION(n0)))
```

Discussion of a curve or
How to write a "program" with DERIVE!

#1: [PrecisionDigits := 6, NotationDigits := 6]

#2: f(x) :=

#3: [z(x) := NUMERATOR(FACTOR(f(x), Trivial)), n(x) := DENOMINATOR(FACTOR(f(x), Trivial))]

#4: $\left[\text{f1(x)} := \text{FACTOR}\left(\left(\frac{d}{dx}\right)^1 \text{f(x), Trivial}\right), \text{f2(x)} := \text{FACTOR}\left(\left(\frac{d}{dx}\right)^2 \text{f(x), Trivial}\right)\right]$

#5: $\left[\text{f11} := \frac{d}{dx}\,\text{f(x)},\ \text{f22} := \left(\frac{d}{dx}\right)^2 \text{f(x)}\right]$

Finding the zeros:

#6: n0 := SOLUTIONS(z(x), x, Real)

#7: $\text{zer} := \text{APPEND}\left(\begin{bmatrix} \text{Zeros:} \\ f(x) = 0 \end{bmatrix}, \text{VECTOR}([v, 0], v, n0)\right)$

#8: $f(x) := \dfrac{x^4 - 11\cdot x^3 + 29\cdot x^2 + 35\cdot x - 150}{30}$

#9: $\text{zer} = \begin{bmatrix} \text{Zeros:} & \\ x\cdot(x^3 - 11\cdot x^2 + 29\cdot x + 35) = 150 & \\ -2 & 0 \\ 3 & 0 \\ 5 & 0 \end{bmatrix}$

Finding the local extremal values:

#10: e0 :≡ SOLUTIONS(NUMERATOR(f1(x)) = 0, x, Real)

#11: $e0 = \left[5, \frac{\sqrt{281}}{8} + \frac{13}{8}, \frac{13}{8} - \frac{\sqrt{281}}{8}\right]$

#12: extr :≡ APPEND$\left(\begin{bmatrix} \text{Extremals:} & \\ f'(x) = & f1(x) \end{bmatrix}\right.$, VECTOR([v, f(v), IF(SIGN(SUBST(f22, x, v)) = 1, MIN, MAX, TerrPt)], v, e0)$\left.\right)$

#13: $extr = \begin{bmatrix} \text{Extremals:} & & \\ f'(x) = & \frac{4\cdot x^3 - 33\cdot x^2 + 58\cdot x + 35}{30} & \\ 5 & 0 & \text{MIN} \\ \frac{\sqrt{281}}{8} + \frac{13}{8} & \frac{843\cdot\sqrt{281}}{5120} - \frac{38939}{15360} & \text{MAX} \\ \frac{13}{8} - \frac{\sqrt{281}}{8} & -\frac{843\cdot\sqrt{281}}{5120} - \frac{38939}{15360} & \text{MIN} \end{bmatrix}$

Finding the points of inflection:

#14: w0 :≡ SOLUTIONS(NUMERATOR(f2(x)) = 0, x, Real)

#15: $w0 = \left[\frac{\sqrt{393}}{12} + \frac{11}{4}, \frac{11}{4} - \frac{\sqrt{393}}{12}\right]$

#16: infl :≡ APPEND$\left(\begin{bmatrix} \text{Infl.points:} & & \text{slope} \\ f''(x) = & f2(x) & \end{bmatrix}, \text{VECTOR}\left(\left[v, f(v), \lim_{x\to v} f11\right], v, w0\right)\right)$

#17: $infl = \begin{bmatrix} \text{Infl.points:} & & \text{slope} \\ f''(x) = & \frac{6\cdot x^2 - 33\cdot x + 29}{15} & \\ \frac{\sqrt{393}}{12} + \frac{11}{4} & \frac{5\cdot\sqrt{393}}{64} - \frac{12457}{8640} & \frac{15}{16} - \frac{131\cdot\sqrt{393}}{2160} \\ \frac{11}{4} - \frac{\sqrt{393}}{12} & -\frac{5\cdot\sqrt{393}}{64} - \frac{12457}{8640} & \frac{131\cdot\sqrt{393}}{2160} + \frac{15}{16} \end{bmatrix}$

Finding the poles (vertical asymptotes):

#18: p0 :≡ SOLUTIONS(n(x), x)

#19: pol :≡ APPEND([[Poles:]], IF(DIMENSION(p0) = 0, [[none]], [p0]`))

Finding the endbehaviour:

#20: as_ :≡ QUOTIENT(z(x), n(x))

#21: as :≡ IF(as_ - f(x) = 0, none, as_, as_)

#22: ass :≡ APPEND([[Asymptote:]], [[as]])

Determining the symmetry

#23: sym :≡ APPEND([[Symmetry:]], IF(f(x) - f(-x), [[axial symm.]], IF(f(x) + f(-x), [[centr.symm.]], [[no symm.]], [[no symm.]]), IF(f(x) + f(-x), [[centr.symm.]], [[no symm.]], [[no symm.]])))

Finding the limits for $x \to \pm\infty$:

#24: $\text{lims} := \text{APPEND}\left([[\text{Limits:},]], \left[\left[x \dashrightarrow \infty, \lim_{x\to\infty} f(x)\right]\right], \left[\left[x \dashrightarrow -\infty, \lim_{x\to-\infty} f(x)\right]\right]\right)$

#25: $\text{discuss} := \begin{bmatrix} \text{zer} \\ \text{extr} \\ \text{infl} \\ \text{pol} \\ \text{lims} \\ \text{ass} \\ \text{sym} \end{bmatrix}$

#26: discuss

#27:

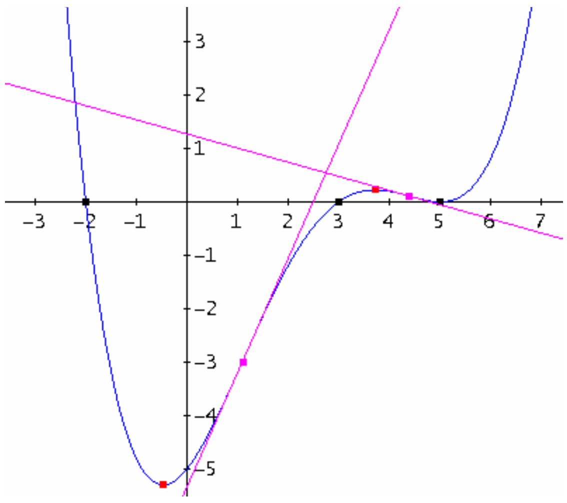
$$\begin{bmatrix}
\begin{bmatrix} & \text{Zeros:} & \\ x\cdot(x^3 - 11\cdot x^2 + 29\cdot x + 35) = 150 & & \\ & -2 & 0 \\ & 3 & 0 \\ & 5 & 0 \end{bmatrix} \\
\begin{bmatrix} \text{Extremals:} & & \\ f'(x) = & \dfrac{4\cdot x^3 - 33\cdot x^2 + 58\cdot x + 35}{30} & \\ 5 & 0 & \text{MIN} \\ \dfrac{\sqrt{281}}{8} + \dfrac{13}{8} & \dfrac{843\cdot\sqrt{281}}{5120} - \dfrac{38939}{15360} & \text{MAX} \\ \dfrac{13}{8} - \dfrac{\sqrt{281}}{8} & -\dfrac{843\cdot\sqrt{281}}{5120} - \dfrac{38939}{15360} & \text{MIN} \end{bmatrix} \\
\begin{bmatrix} \text{Infl.points:} & & \text{slope} \\ f''(x) = & \dfrac{6\cdot x^2 - 33\cdot x + 29}{15} & \\ \dfrac{\sqrt{393}}{12} + \dfrac{11}{4} & \dfrac{5\cdot\sqrt{393}}{64} - \dfrac{12457}{8640} & \dfrac{15}{16} - \dfrac{131\cdot\sqrt{393}}{2160} \\ \dfrac{11}{4} - \dfrac{\sqrt{393}}{12} & -\dfrac{5\cdot\sqrt{393}}{64} - \dfrac{12457}{8640} & \dfrac{131\cdot\sqrt{393}}{2160} + \dfrac{15}{16} \end{bmatrix} \\
\begin{bmatrix} \text{Poles:} \\ \text{none} \end{bmatrix} \\
\begin{bmatrix} \text{Limits:} & \\ x \dashrightarrow \infty & \infty \\ x \dashrightarrow -\infty & \infty \end{bmatrix} \\
\begin{bmatrix} \text{Asymptote:} \\ \text{none} \end{bmatrix} \\
\begin{bmatrix} \text{Symmetry:} \\ \text{no symm.} \end{bmatrix}
\end{bmatrix}$$

You can approximate #27 and then you will receive decimal numbers. Use the results to plot the graph including all critical points.

Graph of the function from above followed by a 2nd example:

#33: $f(x) := \frac{5}{x} - \frac{x}{x^2 - 1}$

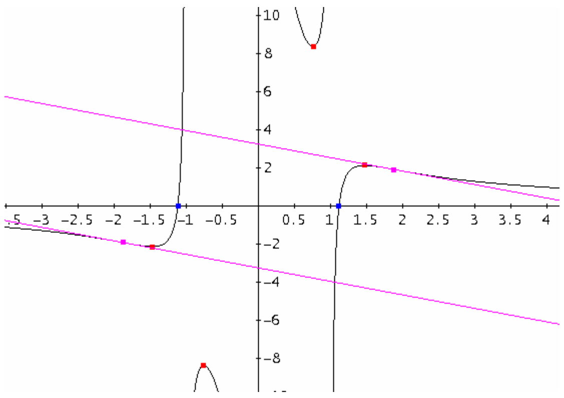
#34: discuss



#36:
$$\left[\begin{array}{c}
\left[\begin{array}{cc} \text{Zeros:} & \\ \frac{x}{x^2-1} - \frac{5}{x} = 0 & \\ 1.11803 & 0 \\ -1.11803 & 0 \end{array}\right] \\
\left[\begin{array}{ccc} \text{Extremals:} & & \\ f'(x) = & -\frac{4\cdot x^4 - 11\cdot x^2 + 5}{x^2\cdot(x^2-1)^2} & \\ 1.47492 & 2.13517 & \text{MAX} \\ -1.47492 & -2.13517 & \text{MIN} \\ 0.758029 & 8.378 & \text{MIN} \\ -0.758029 & -8.378 & \text{MAX} \end{array}\right] \\
\left[\begin{array}{ccc} \text{Infl.points:} & & \text{slope} \\ f''(x) = & \frac{2\cdot(4\cdot x^6 - 18\cdot x^4 + 15\cdot x^2 - 5)}{x^3\cdot(x^2-1)^3} & \\ 1.88163 & 1.91662 & -0.708727 \\ -1.88163 & -1.91662 & -0.708727 \end{array}\right] \\
\left[\begin{array}{c} \text{Poles:} \\ 0 \\ 1 \\ -1 \end{array}\right] \\
\left[\begin{array}{cc} \text{Limits:} & \\ x \text{ --> } \infty & 0 \\ x \text{ --> } -\infty & 0 \end{array}\right] \\
\left[\begin{array}{c} \text{Asymptote:} \\ 0 \end{array}\right] \\
\left[\begin{array}{c} \text{Symmetry:} \\ \text{centr.symm.} \end{array}\right]
\end{array}\right]$$

Try $f(x) := e^{-x/2}\cdot(x^2 - 4)$ or any polynomial or rational function which are usually treated in school.

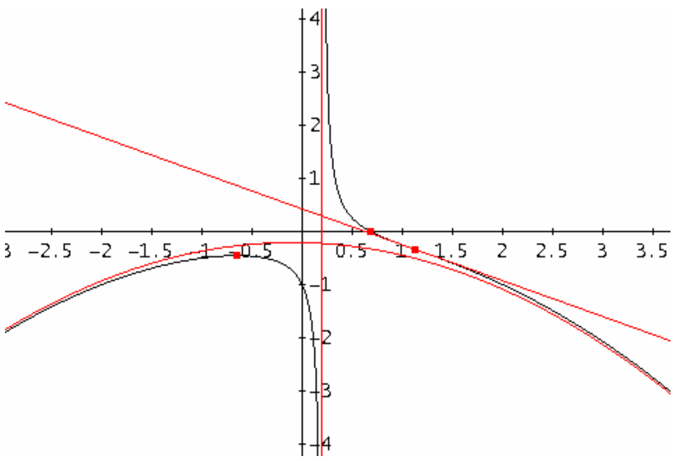
Graph of the function from above and a final example:



#45: $f(x) := \dfrac{x^3 + x - 1}{1 - 5\cdot x}$

#46: discuss

$$\#48:\quad \begin{bmatrix} \begin{bmatrix} \text{Zeros:} & \\ \dfrac{x^3 + x - 1}{5\cdot x - 1} = 0 & \\ 0.682327 & 0 \end{bmatrix} \\ \begin{bmatrix} \text{Extremals:} & & \\ f'(x) = & -\dfrac{10\cdot x^3 - 3\cdot x^2 + 4}{(5\cdot x - 1)^2} & \\ -0.649192 & -0.452859 & \text{MAX} \end{bmatrix} \\ \begin{bmatrix} \text{Infl.points:} & & \text{slope} \\ f''(x) = & \dfrac{2\cdot(25\cdot x^3 - 15\cdot x^2 + 3\cdot x - 20)}{(1 - 5\cdot x)^3} & \\ 1.12521 & -0.335025 & -0.675127 \end{bmatrix} \\ \begin{bmatrix} \text{Poles:} \\ 0.2 \end{bmatrix} \\ \begin{bmatrix} \text{Limits:} & \\ x \dashrightarrow \infty & -\infty \\ x \dashrightarrow -\infty & -\infty \end{bmatrix} \\ \begin{bmatrix} \text{Asymptote:} \\ -0.2\cdot x^2 - 0.04\cdot x - 0.208 \end{bmatrix} \\ \begin{bmatrix} \text{Symmetry:} \\ \text{no symm.} \end{bmatrix} \end{bmatrix}$$



Aufgaben aus der Elektrotechnik - nicht nur für den Unterricht in beruflichen Schulformen 2

H.Scheuermann u. N.Wild, Hofheim, Germany

5. Netzwerkberechnung - einmal anders[8] (Calculation of networks - another way)[8]

Aufgabenstellung:

Berechnung eines beliebigen elektrischen Widerstandsnetzwerkes mit Hilfe eines Näherungsverfahrens. Ein triviales Beispiel soll den Algorithmus erläutern und das iterative Vorgehen demonstrieren:

Problem: Calculation of an arbitrary network of resistors using an approximation method. A trivial example shall explain the algorithm and demonstrate the iterative process:

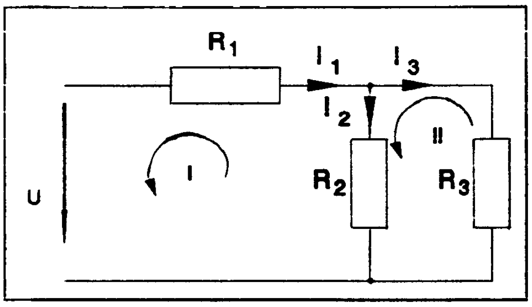


Abb 8: Lineares Widerstandswerk I
Linear network of resistors

Für die Schaltung in Abb.8 gelten die folgenden Werte:

$U = 10V$; $R_1 = 10\Omega$
$R_2 = 20\Omega$; $R_3 = 30\Omega$

Maschengleichungen (Mesh-equations):

(I) $I_1R_1 + I_2R_2 - U = 0$
(II) $I_2R_2 + I_3R_3 = 0$

Im Gegensatz zu den Lösungsverfahren wie z.B. Maschenstromverfahren, Superposition nach Helmholtz, Ersatzspannungs- und Stromquelle usw., wird bei diesem Verfahren zur Netzberechnung nur ein (beliebiger) Pfad vom positiven bis zum negativen Anschlußpol der Spannungsquelle ausgewählt. Alle Ströme dieses Weges erhalten einen beliebigen Startwert ungleich 0, zB 1, alle anderen Ströme den Wert 0.

$$I_1 = I_2 = 1; \quad I_3 = 0$$

Mit diesen Werten sind die Maschengleichungen (I) und (II) natürlich noch nicht erfüllt. Deshalb werden die Gleichungen durch einen additiven Korrekturwert Δ_i ergänzt.

We choose one - optional - path from the positive to the negative connection terminal of the supply point. All the currents along this way get an arbitrary starting value ≠ 0, e.g. 1, the other currents have the value 0. These values don´t meet the equations (I) and (II). So we add a correcting value Δ_i in (I) and (II).)

(I)
$$\left(I_1 + \Delta_1\right)R_1 + \left(I_2 + \Delta_1\right)R_2 - U = 0$$
$$\Delta_1 = \frac{U - I_1R_1 - I_2R_2}{R_1 + R_2} \Rightarrow \begin{array}{l} I_{1\,neu} := I_1 + \Delta_1 \\ I_{2\,neu} := I_2 + \Delta_1 \end{array}$$
(without DERIVE !!!!, editor)

Die zweite Masche wird nun entsprechend korrigiert:

(II)
$$\left(I_2 + \Delta_2\right)R_2 - \left(I_3 - \Delta_2\right)R_3 = 0$$
$$\Delta_2 = \frac{I_3R_3 - I_2R_2}{R_2 + R_3} \Rightarrow \begin{array}{l} I_{2\,neu} := I_{2neu} + \Delta_2 \\ I_{3\,neu} := I_3 - \Delta_2 \end{array}$$

[8] Die Idee zu diesem Verfahren wurde von Klingen [3] veröffentlich. Klingen [3] published the idea of this method.

Die so berechneten Ströme I_{1neu}, I_{2neu}, und I_{3neu} sind eine Verbesserung der ersten Annahme. Ein mehrmaliges Durchlaufen der obigen Prozedur führt schließlich zu immer genaueren Werten.

Zahlenbeispiel:

I_1	I_2	I_3	Δ_1	Δ_2
1	1	0		
0.33	0.33	0	- 0.67	
	0.20	0.13		-0.13
0.42	0.29		0.09	
	0.25	0.17		- 0.04
0.45	0.28		0.02	
	0.27	0.18		- 0.01

Dem Schreiben des DERIVE™ - Listings zur Lösung der Aufgabenstellung stehen einige Probleme entgegen:

- für die Speicherung der gegebenen Widerstandswerte und der gesuchten Ströme steht keine Datenstruktur im Sinne einer Programmiersprache zur Verfügung und
- es gibt keine Schleife, die mehrere Befehle wiederholt.

(There are some problems for a DERIVE-way because DERIVE neither provides any data structure to store the given resistance values and the currents wanted nor a loop to repeat some commands. So I try to work with vector operations.)

Zur Darstellung der Daten in DERIVE scheint die Anwendung von Vektoren bzw. Matrizen angemessen und sinnvoll zu sein. Für die Berechnungen müßten dann die Ausdrücke für Δ_1 und Δ_2 durch Vektoroperationen dargestellt werden.

Die Zähler der Δ_i - Terme sind bis auf das Vorzeichen der Maschengleichungen der Schaltung und lassen sich durch das Skalarprodukt darstellen:

(the numerators of the Δ_i):
$$\vec{R}\cdot\vec{I} = \begin{pmatrix} R_1 \\ R_2 \\ 0 \\ -U \end{pmatrix} \cdot \begin{pmatrix} I_1 \\ I_2 \\ I_3 \\ 1 \end{pmatrix} = R_1 I_1 + R_2 I_2 - U$$

Um die Nenner in gleicher Weise berechnen zu können, wird ein Hilfsvektor $\vec{E}$ definiert, so dass

(the denominators of the Δ_i):
$$\vec{R}\cdot\vec{E} = \begin{pmatrix} R_1 \\ R_2 \\ 0 \\ -U \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = R_1 + R_2$$

ist. Somit lassen sich alle Korrekturwerte Δ durch den Quotienten aus den beiden aufgeführten Skalarprodukten ausdrücken:

$$\Delta = \frac{\vec{R}\cdot\vec{I}}{\vec{R}\cdot\vec{E}} \qquad (*)$$

Die Vektoren $\vec{R}_{(i)}$ repräsentieren so die einzelnen Maschengleichungen der Schaltung. Da in DERIVE Vektoren bevorzugt in Zeilenform dargestellt werden, beschreiben die Matrizen **r_mat** die gesamte Schaltung , **e_mat** die Hilfsvektoren $\vec{E}_{(i)}$ und der Vektor **i_vec** die Startwerte der Ströme.

$$\#1:\quad r\_mat := \begin{array}{c} \begin{array}{cccc} I_1 & I_2 & I_3 & U \end{array} \\ \left[\begin{array}{cccc} 10 & 20 & 0 & -10 \\ 0 & 20 & -30 & 0 \end{array}\right] \end{array} \qquad \begin{array}{l} \text{(the coefficients of (I))} \\ \text{(the coefficients of (II)} \end{array}$$

$$\#2:\quad e\_mat := \left[\begin{array}{cccc} 1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \end{array}\right]$$

$$\#3:\quad i\_vec := [1, 1, 0, 1]$$

Die Korrekturwerte Δ für die i-te Zeile liefert nach Gleichung (*) der Ausdruck:

$$\Delta_i = \frac{\vec{R}_{(i)} \cdot \vec{I}}{\vec{R}_{(i)} \cdot \vec{E}_{(i)}} \qquad \text{(the correcting values for line i)}$$

Damit läßt sich mit DERIVE eine Funktion der Korrekturwerte angeben:

$$\#4:\quad \delta(i\_, n) := \frac{r\_mat_n \cdot i\_}{r\_mat_n \cdot e\_mat_n}$$

Die gesamte Berechnung einer Näherung kann durch die folgende Beziehung beschrieben werden, wobei i die Werte von 1 und 2 (Anzahl der Maschen) durchläuft:

$$\vec{I}_{new} := \vec{I}_{old} - \Delta_i . \vec{E}_{(i)} \qquad (i = 1, 2)$$

Die von DERIVE bereitgestellte Schleifenkonstruktion ITERATE bietet sich zur Berechnung des Ausdrucks an, kann aber nicht verwendet werden, da I_{new} durch zwei Parameter I_{old} und i bestimmt wird, ITERATE aber nur einen Parameter übergeben kann. Da sich jede Iteration auch durch eine Rekursion beschreiben läßt, kann auf die folgende Programmiertechnik zurückgegriffen werden:

```
Funktion I (i_,n)
    wenn n ≤ 2  (if ....)
        dann rufe I (i_ - $\Delta_{(n)}$ . $E_{(n)}$ , n+1)      (then ....)
        sonst setze I(i_,n)  =  i_       (else .....)
```

In DERIVE lautet diese Konstruktion:

```
      I(i_, n) :=
        If n ≤ 2
#5:        I(i_ - δ(i_, n)·e_mat↓n, n + 1)
           i_
```

Mit dem Aufruf I(i_vec,1) liefert die Funktion I dann eine Näherung. Weitere Näherungen lassen sich durch die ITERATES - Funktion generieren:
(Now we have only one parameter i_, so we can use the ITERATES-function:)

```
#6:   ITERATES(I(i_, 1), i_, i_vec, 6)
```

Hierbei gibt der erste Parameter die zu wiederholende Funktion (also I(i_,1)) an, der zweite das Funktionsargument (i_), der dritte den Startwert und der vierte die Anzahl der Wiederholungen.

Die Ausführung der Zeile #6 liefert (Approximating #6 gives):

$$\#7: \quad \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0.333333 & 0.2 & 0.133333 & 1 \\ 0.422222 & 0.253333 & 0.168888 & 1 \\ 0.445925 & 0.267555 & 0.17837 & 1 \\ 0.452246 & 0.271348 & 0.180898 & 1 \\ 0.453932 & 0.272359 & 0.181573 & 1 \\ 0.454381 & 0.272629 & 0.181752 & 1 \end{bmatrix}$$

Das Lösungsverfahren ist sicherlich nicht so potent, daß es die genannten traditionellen Berechnungsverfahren verdrängen könnte. Dennoch sprechen auch Gründe dafür, dieses Verfahren - insbesondere neben den klassischen - in den Unterricht zu integrieren:

1. Ein überraschender und für die Schule unüblicher, in der Praxis aber durchaus gebräuchlicher Lösungsansatz (Motto: erst vereinfachen wir die Lösung und machen dabei Fehler, die wir später sukzessive wieder korrigieren).
2. Wenn die Schüler schon Kenntnisse und Übung in der Anwendung von DERIVE haben, dann stellt die Aufgabe ein weiteres Beispiel dar, daß Operationen der Vektoralgebra auch zur Lösung von nicht-geometrischen Problemen verwendet werden können.

This method is not so powerful to substitute traditional calculation methods. But from my point of view there are two reasons to integrate this method into classes:
1. A surprising - unusual in school but very usual in practice - approach (at first we simplify the solution risking errors, which we then correct successively).
2. If the students have some skill in working with DERIVE this problem can be a fine example to show that vector algebra operations can be used successfully to solve non geometric problems.

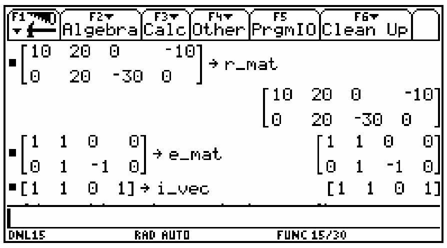
With Derive 5 and higher we are able to collect the procedure into one program and overcome Helmut´s complaints from above. The code line after the "Loop" (i:=[i↓1, …]) is necessary. Try the program without this line to experience the difference. Josef

```
lin_net(r, e, i, n, its, δ1, δ2, k) :=
  Prog
    its := [i]
    k := 0
    Loop
      i := [i↓1, i↓2, i↓3, 1]
      its := APPEND(its, [i])
      If k = n
         RETURN its↓[1, ..., n + 1]
      δ1 := - r↓1·i/(r↓1·e↓1)
      i↓1 := i↓1 + δ1
      i↓2 := i↓2 + δ1
      δ2 := - r↓2·i/(r↓2·e↓2)
      i↓2 := i↓2 + δ2
      i↓3 := i↓3 - δ2
      k :+ 1
```

lin_net(r_mat, e_mat, i_vec, 6)

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0.3333333333 & 0.2 & 0.1333333333 & 1 \\ 0.4222222222 & 0.2533333333 & 0.1688888888 & 1 \\ 0.4459259259 & 0.2675555555 & 0.1783703703 & 1 \\ 0.4522469135 & 0.2713481481 & 0.1808987654 & 1 \\ 0.4539325102 & 0.2723595061 & 0.1815730041 & 1 \\ 0.4543820027 & 0.2726292016 & 0.1817528010 & 1 \end{bmatrix}$$

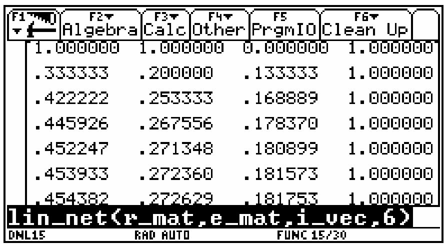
And what about the handheld?


F1 F2▾ Algebra F3▾ Calc F4▾ Other F5 PrgmIO F6▾ Clean Up
DNL15 RAD AUTO FUNC 15/30

$$\blacksquare \begin{bmatrix} 10 & 20 & 0 & -10 \\ 0 & 20 & -30 & 0 \end{bmatrix} \to r\_mat \qquad \begin{bmatrix} 10 & 20 & 0 & -10 \\ 0 & 20 & -30 & 0 \end{bmatrix}$$

$$\blacksquare \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \end{bmatrix} \to e\_mat \qquad \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \end{bmatrix}$$

$$\blacksquare [1\ 1\ 0\ 1] \to i\_vec \qquad [1\ 1\ 0\ 1]$$


F1 F2▾ Algebra F3▾ Calc F4▾ Other F5 PrgmIO F6▾ Clean Up
DNL15 RAD AUTO FUNC 15/30

```
1.000000  1.000000  0.000000  1.000000
.333333   .200000   .133333   1.000000
.422222   .253333   .168889   1.000000
.445926   .267556   .178370   1.000000
.452247   .271348   .180899   1.000000
.453933   .272360   .181573   1.000000
.454382   .272629   .181753   1.000000
lin_net(r_mat,e_mat,i_vec,6)
```

6. Berechnung gemischter passiver Schaltungsnetzwerke (Calculation of mixed passive circuit networks)

Die Berechnung von gemischten Schaltungsnetzwerken hat i.a. zwei Ziele:

1. Umfangreichere Schaltungen auf einfachere Ersatzschaltungen zurückzuführen und
2. bestimmte (Schein-)Widerstandwerte durch die Kombination vorhandener Bauteile zu erzielen.

(The goals are to reduce complex circuits to simpler alternative circuits and to achieve certain impedances combinating the existing components.)

DERIVE bietet schon von der Programmstruktur her geeignete Programmelemente, die eine elegante Lösung der oben genannten Aufgabenstellungen ermöglichen (Funktionsdeklaration, fest implementierte Funktionen, komplexe Zahlen, einfache Programmierungen usw.)

Im Folgenden sollen zur Vorstellung und Erläuterung des Lösungsweges zuerst Gleichstrom-Schaltungen und im weiteren auch Wechselstromschaltungen berechnet werden.

6.1. Berechnung von linearen Schaltungen (Calculation of linear circuits)

Aufgabenstellung:

Der Gesamtwiderstand R_{tot} der in Abb.9 dargestellten Schaltung soll bestimmt werden.
Calculate the total resistance of the circuit in fig.9

Zur Lösung der Aufgabenstellung werden die beiden aus den Kirchhoffschen Gesetzen hergeleiteten Gleichungen „geeignet kombiniert“:

$$\boxed{R_{ser} = R_1 + R_2} \quad (1) \qquad \boxed{R_{par} = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2}}} \quad (2)$$

Für die Schaltung in Abb 9 gilt dann:

$$R_{45} = R_4 + R_5$$

$$R_{3-5} = \frac{1}{\frac{1}{R_3} + \frac{1}{R_{45}}}$$

$$R_{2-5} = R_2 + R_{3-5}$$

$$R_{tot} = R_{ges} = \frac{1}{\frac{1}{R_1} + \frac{1}{R_{2-5}}}$$

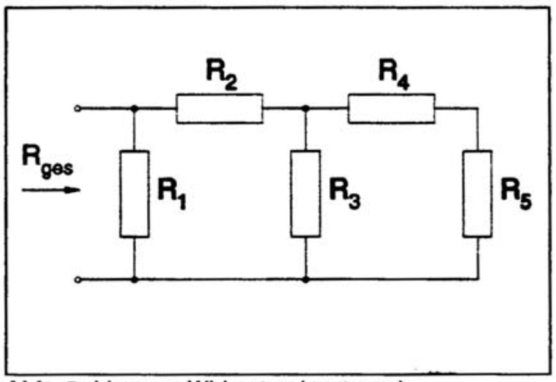


Abb. 9: Lineares Widerstandsnetzwerk

Die Berechnung kann mit Hilfe eines programmierbaren Systems, wie z.B. Taschenrechner, PC mit Programmiersprachen oder mit entsprechender mathematischer Software, durchgeführt werden. Sinnvoll ist dabei, daß nicht zuletzt aus schreibtechnischen Gründen die Berechnungsvorschrift „in einer Zeile“ darstellbar ist. Natürlich kann das Berechnungsverfahren auch wie oben beschrieben unter Verwendung von Variablen bzw. Makro(er)setzungen formuliert werden, jedoch ist dies bezüglich der Rechenzeit und des Speicherbedarfs wenig vorteilhaft. In der herkömmlichen Form, also mit Brüchen geschrieben, wird der Ausdruck für R_{tot} „unhandlich“:

$$R_{tot} = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2 + \frac{1}{\frac{1}{R_3} + \frac{1}{R_4 + R_5}}}}$$

Eine elegantere Darstellung wird durch die Definition von Rechenoperationen für die Reihen- und Parallelschaltung erzielt:

´+ ´ := Reihenschaltung (series conn.) ´//´ := Parallelschaltung (parallel conn.)

Damit folgt für das Beispiel in Abb. 9:

$\mathbf{R_{tot} = R_1 \,/\!/\, (R_2 + (R_3 \,/\!/\, (R_4 + R_5))}$ *Infix - Notation*

Diese Schreibweise entspricht der sogenannten **Infix-Notation** einer zweistelligen Rechenoperation der Algebra, d.h. der Operator steht zwischen den Operanden. Daneben gibt es noch die **Prefix-** und die **Postfix-Notationen**, auch polnische (PN) bzw. umgekehrte polnische Notation (UPN) genannt.[9]

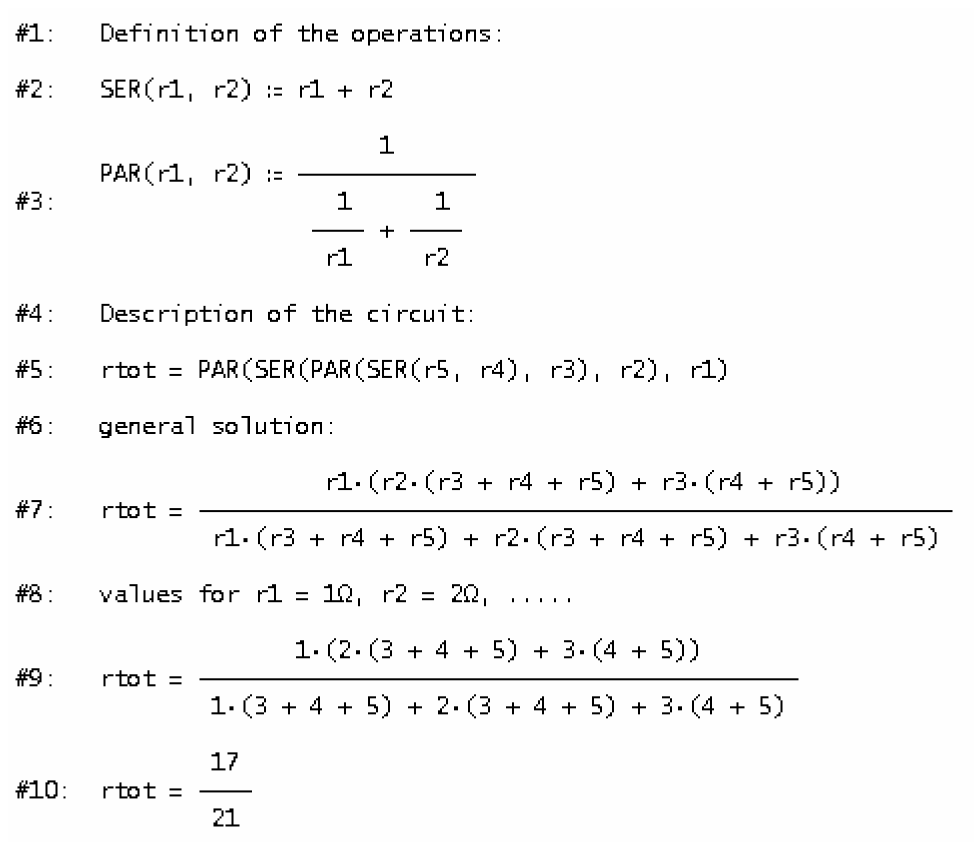
Die Schaltung in Abb.9 kann dann wie folgt in Prefix-Darstellung geschrieben werden:

$\mathbf{R_{tot} = /\!/ ; + ; /\!/ ; + R_5 ; R_4 ; R_3 ; R_2 ; R_1}$ *Prefix - Notation*

In dem benutzten programmierbaren System können nun zwei Funktionen SER(r1,r2) := r1 + r2 und PAR(r1,r2) := 1/(1/r1 + 1/r2) definiert werden. Der Gesamtwiderstand R_{tot} ergibt sich dann aus:

$\mathbf{R_{tot} = PAR(SER(PAR(SER(R_5 , R_4) , R_3) , R_2) , R_1)}$

Für die Berechnung einer Schaltung mit dem Programm DERIVE genügt die Definition der Funktionen SER und PAR, sowie der Term der Schaltung in Prefix - Notation.



```
#1:   Definition of the operations:

#2:   SER(r1, r2) := r1 + r2

                           1
      PAR(r1, r2) := ─────────────
#3:                    1       1
                      ──── + ────
                       r1      r2

#4:   Description of the circuit:

#5:   rtot = PAR(SER(PAR(SER(r5, r4), r3), r2), r1)

#6:   general solution:

                     r1·(r2·(r3 + r4 + r5) + r3·(r4 + r5))
#7:   rtot = ─────────────────────────────────────────────────────
              r1·(r3 + r4 + r5) + r2·(r3 + r4 + r5) + r3·(r4 + r5)

#8:   values for r1 = 1Ω, r2 = 2Ω, .....

                   1·(2·(3 + 4 + 5) + 3·(4 + 5))
#9:   rtot = ─────────────────────────────────────────
              1·(3 + 4 + 5) + 2·(3 + 4 + 5) + 3·(4 + 5)

              17
#10:  rtot = ────
              21
```

[9] Ein einfaches Beispiel verdeutlicht die unterschiedlichen Schreibweisen:
Infix: (3+4) * (5+6)
Postfix: 5; 6; + ; 3; 4; +; *
Prefix: *; +; 3; 4; +; 5; 6
In der Prefixdarstellung wird zuerst der Ausdruck A1: +; 5; 6 und dann A2: +; 3; 4 ausgewertet. Die letzte Operation *; A2; A1 liefert dann das Ergebnis. Diese Arbeitsweise ähnelt stark dem Aufruf von Funktionen - in diesem Fall Funktion + und Funktion * mit je zwei Parametern.

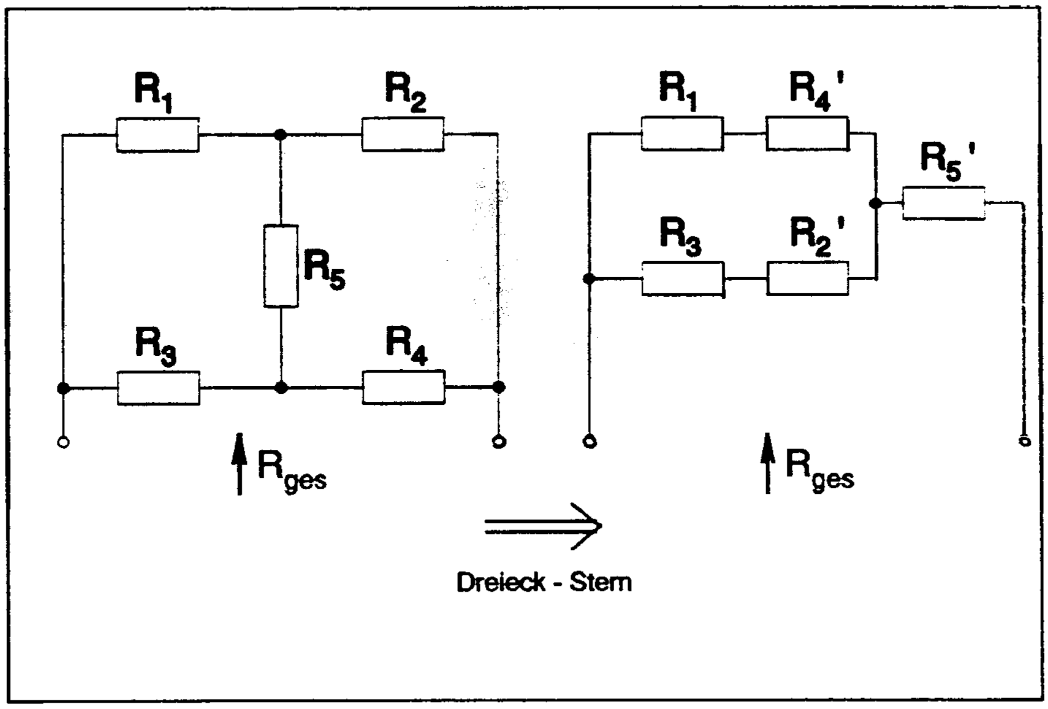


Abb 10: Brückenschaltung (mit Dreieck-Stern-Umwandlung

Das zweite Ziel der Berechnung von gemischten Schaltungen, einen bestimmten Widerstandswert aus der Kombination vorhandener Bauelemente zu erzielen, stellt im gewissen Sinn die Umkehraufgabe dar. Der Gesamtwiderstand ist gegeben und ein einzelner Widerstand ist gesucht. Mit DERIVE ist es kein Problem, die Gleichung für R_{tot} nach einem anderen Widerstand umzustellen. Der Befehl soLve nach R1 liefert, angewandt auf den Ausdruck #5 oder #7, die Zeile #11:

The 2nd aim of calculating mixed circuits - to get a certain resistance value from a combination of existing components - is a sort of reverse task: you know the total resistance and you want to find a particular one. So soLve expression #5 or #7 for R1 to get expression #11:

$$\#11: \quad r1 = \frac{rtot\cdot(r2\cdot(r3 + r4 + r5) + r3\cdot(r4 + r5))}{r2\cdot(r3 + r4 + r5) + r3\cdot(r4 + r5 - rtot) - rtot\cdot(r4 + r5)}$$

Substituting for r2, r3, etc and rtot = 17/21 we obtain the original value for r1 (as we have expected).

$$\#12: \quad r1 = \frac{\frac{17}{21}\cdot(2\cdot(3 + 4 + 5) + 3\cdot(4 + 5))}{2\cdot(3 + 4 + 5) + 3\cdot\left(4 + 5 - \frac{17}{21}\right) - \frac{17}{21}\cdot(4 + 5)}$$

$$\#13: \quad r1 = 1$$

Hat das gegebene Netzwerk Elemente einer Brückenschaltung, so kann das beschriebene Lösungsverfahren auch, aber nicht ohne Modifikation, angewandt werden. Durch Anwendung der Dreieck - Stern - Umformung kann der Gesamtwiderstand R_{tot} der Schaltung laut Abb. 10 nach beschriebenen Lösungsverfahren bestimmt werden.

Mit Hilfe der definierten Funktionen SER und PAR (in den Hintergrund - als Utility file - laden) kann die Gleichung für den Gesamtwiderstand der Brückenschaltung wie folgt geschrieben werden:

Die Funktion TS (expr. #1) transformiert denjenigen Widerstand, der als erster Parameter in TS aufgeführt wird. Z.B. liefert der Aufruf von TS(r2,r1,r3) den Widerstand R2´.

If there are elements of a bridge circuit within the network we can use the algorithm described, too but not without a modification. Applying the Triangle - Star - modification R_{tot} of the circuit in fig. 10 can be determined. Load the functions SER and PAR as Utility files and write the equation for R_{tot}. TS transforms the first resistor of the parameter list, eg. TS(r2,r1,r3) leads to R2´.

#5: $\mathrm{SER}(r1,\ r2) := r1 + r2$

#6: $$\mathrm{PAR}(r1,\ r2) := \frac{1}{\dfrac{1}{r1} + \dfrac{1}{r2}}$$

#7: $$\mathrm{TS}(r1,\ r2,\ r3) := \frac{r2 \cdot r3}{r1 + r2 + r3}$$

#8: $rtot = \mathrm{SER}(\mathrm{PAR}(\mathrm{SER}(r1,\ \mathrm{TS}(r4,\ r2,\ r5)),\ \mathrm{SER}(r3,\ \mathrm{TS}(r2,\ r4,\ r5))),\ \mathrm{TS}(r5,\ r2,\ r4))$

#9: $$rtot = \frac{r1 \cdot (r2 \cdot (r3 + r4) + r3 \cdot (r4 + r5) + r4 \cdot r5) + r2 \cdot (r3 \cdot (r4 + r5) + r4 \cdot r5)}{r1 \cdot (r2 + r4 + r5) + r2 \cdot (r3 + r5) + r3 \cdot (r4 + r5) + r4 \cdot r5}$$

6.2. Berechnung von Wechselstromkreisen - (AC - Circuits)

Aufgabenstellung:

Berechnung von passiven R-C-L-Wechselstromschaltungen unter Anwendung von komplexen Zahlen.
Calculation of passive R-C-L-AC-Circuits using complex numbers.

Die beschriebene Prefix-Notation läßt sich auch auf Berechnung von Wechselstromschaltungen übertragen. Als Beispiel dafür folgt die Vorstellung eines Listings zur Berechnung von gemischten R-C-L-Wechselstromkreisen, die sich auf Reihen- und Parallelschaltungen zurückführen lassen. Da DERIVE die komplexen Zahlen, sowie die auf diesen Zahlenbereich erklärten Rechenoperationen fest implementiert hat, bietet sich bei der Berechnung charakteristischer Größen von Wechselstromkreisen die Anwendung komplexer Zahlen an. Im Rahmen dieses Aufsatzes beschränkt sich die Programmierung auf die Berechnung des „komplexen Scheinwiderstands“[10] Z, dessen Betrag Zabs und des Phasenverschiebungswinkels α.

Die Zeilen #1 bis #4 erklären die (Grund-) Bestimmungsgleichungen für den Scheinwiderstand einer Reihen- bzw. Parallelschaltung aus R und L bzw. C. Dabei ist ZE_ .... ein Element der komplexen Zahlen und wird in Normalform (a + î.b, mit $\hat{\imath}^2 = -1$) dargestellt. Durch diese vier Grundgleichungen lassen sich alle weiteren, aus Reihen- bzw. Parallelschaltungen hervorgegangenen Wechselstromnetzwerke symbolisch beschreiben und berechnen. Die Faktoren „10^{-3}“ bzw. „10^{-6}“ ermöglichen die Eingabe der Induktivität in mH bzw. der Kapazität in μF.

[10] Unter diesem Terminus versteht man den „Scheinwiderstand in symbolischer Darstellung unter Anwendung der komplexen Zahlen“.

#1: $\text{ZE\_SER\_RL}(r, l, f) := r + i\cdot 2\cdot \pi\cdot f\cdot (l\cdot 10^{-3})$

#2: $\text{ZE\_SER\_RC}(r, c, f) := r - \dfrac{i}{2\cdot \pi\cdot f\cdot (c\cdot 10^{-6})}$

#3: $\text{ZE\_PAR\_RL}(r, l, f) := \dfrac{1}{\dfrac{1}{r} - \dfrac{i}{2\cdot \pi\cdot f\cdot (l\cdot 10^{-3})}}$

#4: $\text{ZE\_PAR\_RC}(r, c, f) := \dfrac{1}{\dfrac{1}{r} + i\cdot (2\cdot \pi\cdot f\cdot (c\cdot 10^{-6}))}$

Die folgenden Zeilen #5 und #6 legen die allgemeine Operationsvorschrift fest, wie der komplexe Scheinwiderstand einer Reihen- bzw. Parallelschaltung aus zwei komplexen Scheinwiderständen z1 und z2 gebildet werden kann.

Expr. #5 and #6 provide how to build the complex impedance of a series- and parallel connection from two complex impedances z1 and z2.

#5: $\text{Z\_SER}(z1, z2) := z1 + z2$

#6: $\text{Z\_PAR}(z1, z2) := \dfrac{z1\cdot z2}{z1 + z2}$

So erklärt z.B. der Term `Z_SER(ZE_SER_RL(r1,l.f),ZE_PAR_RC(r2,c,f))` den komplexen Scheinwiderstand der Schaltung bestehend aus R1 in Reihe zu L in Reihe zu R2 ‖ C (vgl. Abb. 11)

Zur Berechnung des Phasenverschiebungswinkels α und des Betrages des Scheinwiderstandes werden die vorhandenen DERIVE-Funktionen PHASE bzw. ABS verwendet. Die Division durch das Symbol „° “ in Zeile #8 bewirkt die Ausgabe von α im Gradmaß. Der komplexe Gesamtscheinwiderstand wird mit Z(x) bezeichnet. Dabei ist x als Platzhalter für die möglichen Parameter eines Wechselstromkreises (zB bei der Darstellung von Ortskurven) erklärt, also:

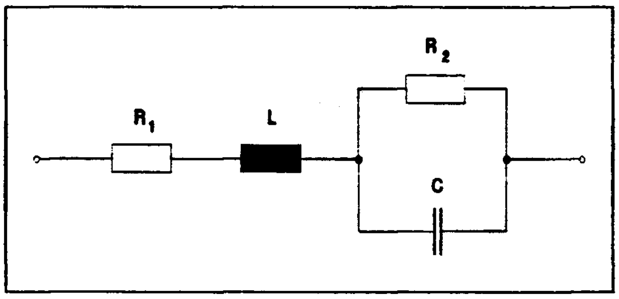


Abb 11: Schaltbild eines R–C–L–Wechselstromkreises

$x \in \{R, C, L, f\}$.

#7: $Z(x) :=$

#8: $\alpha := \dfrac{\text{PHASE}(Z(x))}{1°}$

#9: $\text{zabs} := |Z(x)|$

Diese Programmierung wird zur Lösung (vgl. Abb. 11) der nachfolgenden Aufgabe den weiteren DERIVE - Zeilen vorangestellt oder im „Hintergrund“ aufgerufen.

Aufgabenstellung:

Für den in Abb. 12 dargestellten Wechselstromkreis sind die folgenden Werte gegeben:

R_1 = 25Ω; L = 79,5mH;
R_2 = 12,5Ω; C = 40μF.

a) Man bestimme den komplexen Scheinwiderstand sowie dessen Betrag und Phasenverschiebungswinkel α für f = 50 Hz.

b) Für welche Frequenz f wird die Schaltung reell?

c) Man zeichne die Widerstandsortskurve und parametrisiere diese für 10Hz $\leq$ f $\leq$ 100Hz mit einer Schrittweite von 10Hz

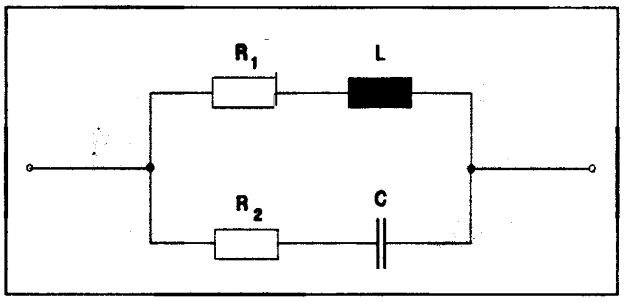
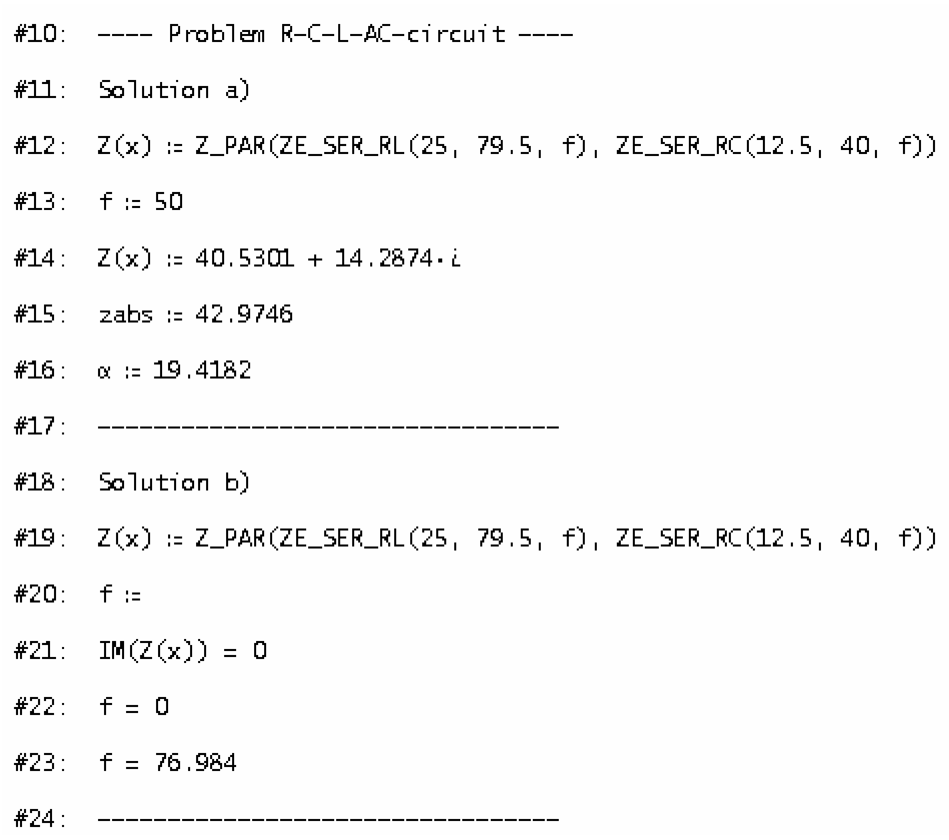


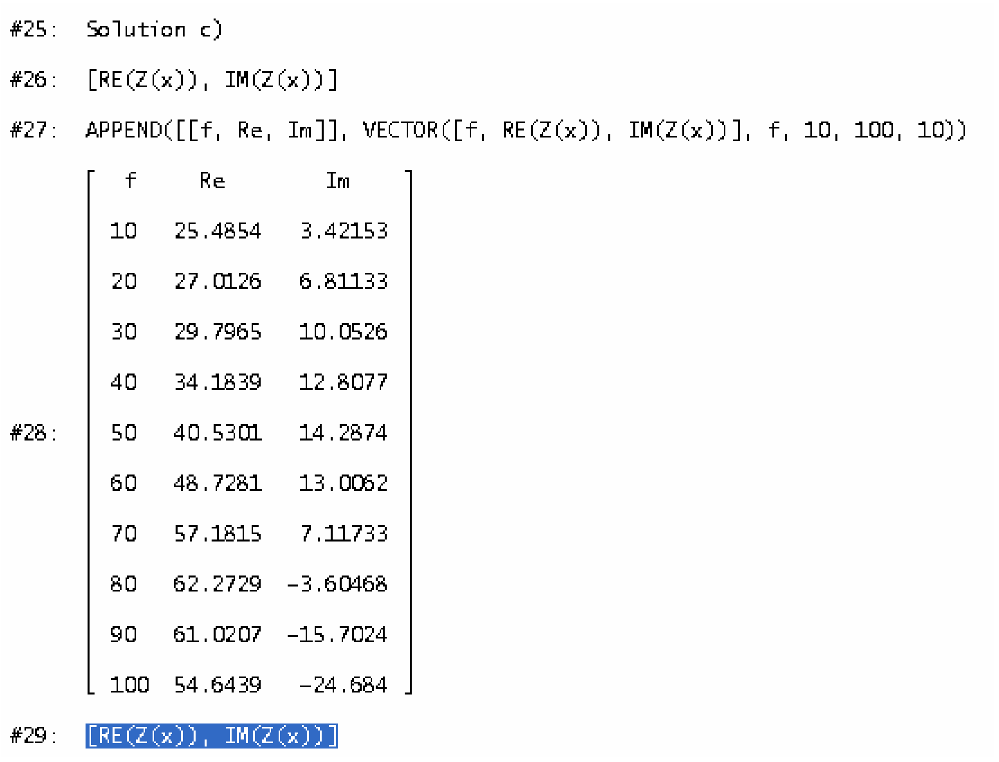
Abb 12: Schaltbild laut Aufgabenstellung

Durch die „Vorarbeiten (Zeilen #1 bis #9) gestaltet sich das Lösungslisting (vgl. Abb.13) zur vorgegebenen Aufgabe recht kurz. Zeile #12 stellt die symbolische Beschreibung der R-L-C- Wechselstromschaltung dar (mit x = f). Nach Festlegung der Frequenz f können durch Vereinfachung der Zeilen #8, #9 und #12 der komplexe Scheinwiderstand, dessen Betrag und Phasenverschiebungswinkel direkt abgerufen werden (vgl. Zeilen #14, #15 und #16). Die Zeile #19 erklärt die Frequenz f (wieder) als Variable. Der Aufruf IM(Z(f)) bestimmt den Imaginärteil der Wechselstromschaltung, der zur Bestimmung des reellen Scheinwiderstandes gleich Null gesetzt wird (Zeile #20, Lösungen stellen die Zeilen #21 und #22 dar). Die Ortskurve wird über die Berechnung der Wertepaare in Zeile #25 geplottet (der Realteil wird auf die 1.Achse, der Imaginärteil auf die 2.Achse abgetragen). der VECTOR-Befehl in Zeile #26 erstellt die Wertetabelle in #27 und ermöglicht die Parametrisierung der Ortskurve laut Aufgabenstellung. `(#30: [RE(Z(x)), IM(Z(x))])`



```
#10:  ---- Problem R-C-L-AC-circuit ----

#11:  Solution a)

#12:  Z(x) := Z_PAR(ZE_SER_RL(25, 79.5, f), ZE_SER_RC(12.5, 40, f))

#13:  f := 50

#14:  Z(x) := 40.5301 + 14.2874·i

#15:  zabs := 42.9746

#16:  α := 19.4182

#17:  --------------------------------

#18:  Solution b)

#19:  Z(x) := Z_PAR(ZE_SER_RL(25, 79.5, f), ZE_SER_RC(12.5, 40, f))

#20:  f :=

#21:  IM(Z(x)) = 0

#22:  f = 0

#23:  f = 76.984

#24:  --------------------------------
```

Abb 13: Lösung zur Aufgabe R-C-L-Wechselstromkreis, Teil 1



```
#25:  Solution c)

#26:  [RE(Z(x)), IM(Z(x))]

#27:  APPEND([[f, Re, Im]], VECTOR([f, RE(Z(x)), IM(Z(x))], f, 10, 100, 10))

      ⎡  f      Re         Im    ⎤
      ⎢                          ⎥
      ⎢ 10    25.4854    3.42153 ⎥
      ⎢                          ⎥
      ⎢ 20    27.0126    6.81133 ⎥
      ⎢                          ⎥
      ⎢ 30    29.7965    10.0526 ⎥
      ⎢                          ⎥
      ⎢ 40    34.1839    12.8077 ⎥
      ⎢                          ⎥
#28:  ⎢ 50    40.5301    14.2874 ⎥
      ⎢                          ⎥
      ⎢ 60    48.7281    13.0062 ⎥
      ⎢                          ⎥
      ⎢ 70    57.1815    7.11733 ⎥
      ⎢                          ⎥
      ⎢ 80    62.2729   -3.60468 ⎥
      ⎢                          ⎥
      ⎢ 90    61.0207   -15.7024 ⎥
      ⎢                          ⎥
      ⎣ 100   54.6439    -24.684 ⎦

#29:  [RE(Z(x)), IM(Z(x))]
```

Abb 13b: Lösung zur Aufgabe R-C-L-Wechselstromkreis

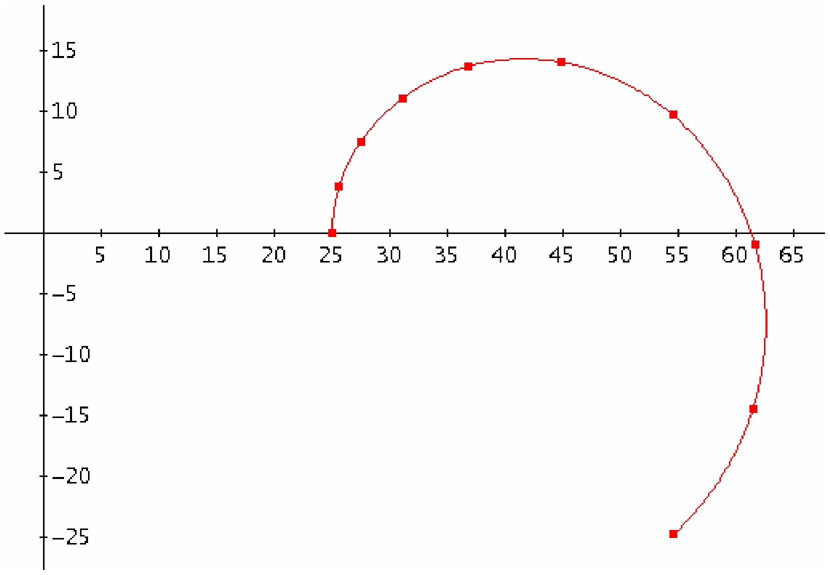


Abb 13c: Darstellung der Ortskurve

Problems to be solved:

a) Find the complex impedance, its absolute value and the angle of phase displacement for f = 50Hz.

b) Which frequence f will make the circuit real?

c) Give the parametric form of the locus and plot it (valuetable in #28).

7 Einschaltvorgänge - Turn - on - processes

Aufgabenstellung:

Gegeben ist die Parallelschaltung aus zwei Reihenschaltungen R_1, C bzw. R_2 und L (vgl. Abb 14). An den Klemmen liegt die Gleichspannung U. Gesucht ist der Zeitpunkt t für den I.C = I.L gilt, mit

R_1 = 100 Ωk, C = 1 nF

R_2 = 20 Ωk, L = 1 H.

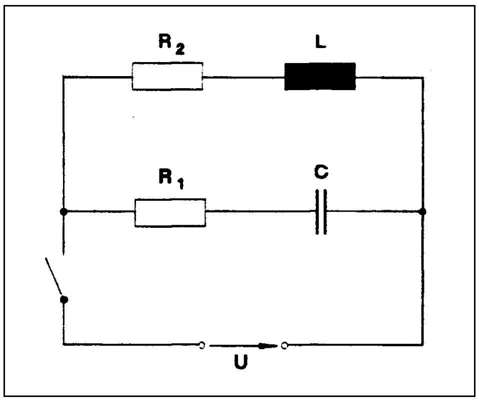


Abb 14: Einschaltvorgang

#1: Wertzuweisungen - Assignment of values

#2: $\left[r1 := 10^{5},\ r2 := 2\cdot10^{4},\ c := 10^{-9},\ l := 1\right]$

#3: Bestimmung der Zeitkonstanten τ1 und τ2 der Reihenschaltungen

#4: Determination of the series connections' constants of time τ1 and τ2

#5: $\left[\tau1 := r1\cdot c,\ \tau2 := \frac{l}{r2}\right]$

#6: Funktionsgleichungen für die Einschaltvorgänge

#7: Equation for the turn-on-processes

#8: $IL(t) := \frac{u}{r2}\cdot(1 - e^{-t/\tau2})$

#9: $IC(t) := \frac{u}{r1}\cdot e^{-t/\tau1}$

#10: Zur Berechnung des Zeitpunktes für Stromgleichheit gilt:

#11: For calculating the moment of equality of current:

#12: IL(t) = IC(t)

#13: $5\cdot10^{-5}\cdot u - 5\cdot10^{-5}\cdot u\cdot e^{-20000\cdot t} = 10^{-5}\cdot u\cdot e^{-10000\cdot t}$

#14: $\frac{1}{20000} - \frac{e^{-20000\cdot t}}{20000} = \frac{e^{-10000\cdot t}}{100000}$

#15: soLve ---> Memory full!!!

#16: Substitute t = -z/10^4

#17: $\frac{1}{20000} - \frac{e^{-20000\cdot(-z/10^{4})}}{20000} = \frac{e^{-10000\cdot(-z/10^{4})}}{100000}$

#18: $\frac{1}{20000} - \frac{e^{2\cdot z}}{20000} = \frac{e^{z}}{100000}$

#19: $z = \text{LN}\left(\frac{\sqrt{101}}{10} - \frac{1}{10}\right)$

#20: $-\frac{\text{LN}\left(\frac{\sqrt{101}}{10} - \frac{1}{10}\right)}{10^{4}}$

#21: $9.983407890\cdot 10^{-6}$

Dieses Beispiel zeigt, dass auch DERIVE (Version 2.52) kein Alleskönner ist. Wichtig erscheint uns, den Schülern auch solche Beispiele vorzustellen[11]. Dies verfolgt die Absicht, den Lernenden zu einer kritischen Haltung gegenüber dem Rechner bzw. der Software anzuregen und damit dem Entstehen einer bequemen und unreflektierten Rechnergläubigkeit entgegenzuwirken.

This example shows that DERIVE (v 2.52) isn´t a jack-of-all-trades. For us it is important to give our students such examples, too. It is our aim to encourage a critical attitude towards the computer and the software.

Later Derive versions have no problems in numerical solving the equation, but is not able to solve the equation in Exact Mode.

$$\text{NSOLVE}\left(\frac{1}{20000} - \frac{e^{-20000\cdot t}}{20000} = \frac{e^{-10000\cdot t}}{100000}, t\right) = (t = 9.983407889\cdot 10^{-6})$$

8. Literatur - Bibliography

[1] Haug,A.: Grundzüge der Elektrotechnik zur Schaltungsberechnung, Hanser Verlag, München 1975

[2] Führer,A./Heidemann,K./Nereter,W.:Grundgebiete der Elektrotechnik, Band 1: Stationäre Vorgän ge, Hanser Verlag, München 1983

[3] Klingen,L.: Zusammenhang und Reichweite modularer Algorithmen in der Schulmathematik. In: Informatik im Unterricht der Sekundarstufe II, Band II (Hrsg.: Bauersfeld,H./Otte,M./Steiner,H.G.), Universität Bielefeld 1977, S 3 - 35

[4] Lipsmeier,A. (Hrsg.): Tabellenbuch - Elektrotechnik - Elektronik, Dümmler Verlag, Bonn 1989

[5] Scheuermann,H.: Anwendungsorientierter Mathematikunterricht des beruflichen Gymnasiums und der Fachoberschule unter besonderer Berücksichtigung von DERIVE. In: Beiträge zum Mathema tikunterricht 1992, S. 391 - 395

[6] Scheuermann,H.: Der belastete Spannungsteiler - Eine Lernsequenz für den Mathematikunterricht der Fachoberschule unter Verwendung von DERIVE. In: Die Berufsbildende Schule, 45. Jg., Heft 5/1993, S. 171 - 176

[11] Diese Aufgabe aus dem Elektrotechnikunterricht wurde von den Schülern mit der Bitte an uns herangetragen, ihren gescheiterten Lösungsversuch mit DERIVE zu untersuchen. Die Möglichkeit, daß das Programm bei dieser Aufgabe „aussteigt“ wurde nur sekundär in Erwägung gezogen. Primär erwarteten sie von uns einen Hinweis auf die „richtige“ Einstellung der Programmparameter bzw. auf die Wahl der lösungsbringenden Tastenkombination.
(Students came to us with the wish to investigate their attempt to find a solution with DERIVE, because they had failed. It was interesting that they didn´t consider DERIVE to fail. They expected us to give them some advice for the correct setting of any “options“ or to show them the right choice of keystrokes.)

Titbits from Algebra and Number Theory (3)

by Johann Wiesenbauer, Vienna

One of the most important tools of number theory is the so-called Moebius-function (yes, the same Moebius who invented the famous strip!) which is defined on the set of natural numbers in the following way:

$$\mu(n) := \begin{cases} 0 & \text{if } n \text{ is divisible by a square} \\ (-1)^k & \text{if } n \text{ is a product of } k \text{ distinct primes} \end{cases}$$

As for its implementation in DERIVE I have good news and bad news for you. First the bad ones: Unlike some other computer algebra systems there exists no built-in μ-function in DERIVE, and what is more, it is rather cumbersome to write a program for it. Here is the result of all my efforts to do so:

```
LPD(n, s) :=
  If NEXT_PRIME(n - 1) = n
     n
     ITERATE(IF(t^2 > n, n, IF(MOD(n, t) = 0, t, NEXT_PRIME(t))), t, s)
```

(Returns the **l**east **p**rime **d**ivisor of *n* which is greater or equal to *s* > 1. In case, it doesn´t exist, the result will be *n*.)

Examples:

```
LPD(2431, 12) = 13

FACTOR(2431) = 11·13·17
```

$$\mathrm{PFL(n)} := \left(\mathrm{ITERATE}\left(\mathrm{IF}\left(x_2 = 1,\ x,\ \left[\mathrm{APPEND}(x_1,\ [x_3]),\ \frac{x_2}{x_3},\ \mathrm{LPD}\left(\frac{x_2}{x_3},\ x_3\right)\right]\right),\ x,\ [[],\ n,\ \mathrm{LPD}(n,\ 2)]\right)\right)_1$$

(Yields the **l**ist of all **p**rime **f**actors – not necessarily distinct – of *n* in increasing order.)

Examples:

```
PFL(2431) = [11, 13, 17]

PFL(1000) = [2, 2, 2, 5, 5, 5]
```

```
PREMOEBIUS(pl) :=
  If DIMENSION(pl) > 1 ∧ ∏(pl↓k - pl↓(k - 1), k, 2, DIMENSION(pl)) = 0
     0
     (-1)^DIMENSION(pl)
```

(One more auxiliary function, but …)

```
μ(n) := PREMOEBIUS(PFL(n))
```

(… here it is, at last! If somebody comes up with a better solution, I would be dead keen on hearing about it!)

Examples:

```
μ(2431) = -1

μ(1000) = 0

μ(1122) = 1
```

And now the good news: In the forthcoming version 3 of DERIVE all the code above can be replaced by one single line!

```
μ_(n) := ∏(IF(PRIME(k), -1, 0), k, FACTORS(FACTOR(n)))
```

In this short program I have taken advantage of 4(!) new features of DERIVE3, namely:

- The new fast routine `PRIME(k)` replaces the former inferior primality test `NEXT_PRIME(k-1)=k`.
- The menu option `FACTOR` is now also available as a function.
- The output of `FACTOR(n)` is transformed into a vector of the corresponding prime powers of *n* by `FACTORS`. (Unfortunately, the primes of *n* are still not directly accessible.)
- The domain of the variable *k* in the product above may now be given as a vector.

Examples:

```
μ_(2431) = 0

μ_(1000) = 0

μ_(1122) = 0
```

Comparing with the results from above you will note that this routine does not work. So let´s investigate what is wrong and how to adapt Johann´s formula for recent versions!

This was Derive 3

```
FACTOR(1000)
```

$2^3 \cdot 5^3$

```
FACTORS(1000)
[1000]
FACTORS(FACTOR(1000))
```

$[2^3, 5^3]$

This is Derive 6:

FACTOR(1000) = $2^3 \cdot 5^3$

FACTORS(1000) = $\begin{bmatrix} 2 & 3 \\ 5 & 3 \end{bmatrix}$

So I "reinvent" an `old_factors`-function for DERIVE 6

old_factors(n) := VECTOR$\left(v_1^{v_2}, v, \text{FACTORS}(n)\right)$

```
old_factors(1000) = [8, 125]
```

```
μ_(n) := ∏(IF(PRIME(k), -1, 0), k, old_factors(n))

μ_(2431) = -1

μ_(1000) = 0

μ_(1122) = 1
```

This works, but an interesting happens simplifying this function:

```
μ_(n) :≔ ∏(IF(PRIME(k), -1, 0), k, old_factors(n))

μ_(n) :≔
  If PRIME?(n)
     -1
     0

[μ_(2431), μ_(1000), μ_(1122)] = [0, 0, 0]
```

Finally we will not keep secret, that the DERIVE versions from today provide a built-in Moebius-function **MOEBIUS_MU(n)**:

```
[MOEBIUS_MU(2431), MOEBIUS_MU(1000), MOEBIUS_MU(1122)] = [-1, 0, 1]
```

Let´s come back to Johann´s Titbits from 1994.

One of the most important applications of the μ-function is the so-called Moebius inversion formula (cf. [1]). Without going into details I´d like to explain its use in connection with the factorization of polynomials of the form $x^n - 1$. If we define $C_d(x)$ for any natural number d by

$$C_d(x) := \prod_{\delta|d}\left(x^{d/\delta}-1\right)^{\mu(\delta)}$$

(δ runs through all positive divisors of d or, what amounts to the same, through all squarefree positive divisors of d) then it turns out that this rational expression in x can always be simplified to a polynomial with integer coefficients which is called the d^{th} *cyclotomic polynomial*. Moreover, the representation of $x^n - 1$ by a product of irreducible polynomials over the field of rational numbers is given by

$$x^n - 1 = \prod_{d|n} C_d(x).$$

And here is the DERIVE-implementation:

```
ENLARGE(v, a) :≔
  If MOD(v↓DIM(v), a) = 0
     v
     APPEND(v, a·v)
```

(An auxiliary outer operation that is used for a recursive adding of new squarefree divisors to v)

```
PRESFDL(pl) :≔
  If DIM(pl) < 2
     APPEND([1], pl)
     ENLARGE(PRESFDL(VECTOR(pl↓k, k, DIM(pl) - 1)), pl↓DIM(pl))
```

(First we need a <u>l</u>ist of all <u>s</u>quarefree <u>d</u>ivisors of n; this auxiliary function does the main job.)

```
SFDL(n) :≔ PRESFDL(PFL(n))

SFDL(1000) = [1, 2, 5, 10]
```

(That´s it. Any idea, how a routine for **<u>all</u>** divisors of n could look like?)

$$\text{PRECP}(x,\ d,\ v) :=\prod_{k=1}^{\text{DIM}(v)} \left(x^{d/v_k} - 1\right)^{\mu(v_k)}$$

(An auxiliary function which can be omitted as soon as DERIVE3 is available!)

$$\text{CP}(x,\ d) := \text{PRECP}(x,\ d,\ \text{SFDL}(d))$$

(Believe it or not, this function returns the d^{th} cyclotomic polynomial!)

To save space I leave it to the reader to try out these routines. Here are some suggestions:

- Factor $x^n - 1$ both by the built-in FACTOR and by means of cyclotomic polynomials for various values of n and compare!
- Calculate a list of cyclotomic polynomials for small d, say d<100. What can be said about the range of their coeffcients? Now try the value $d = 105$. (What´s so special about 105? It is the first number with three different odd prime factors!) The moral you can draw from this: Beware of rash conclusions!
- Take a Mersenne number $2^n - 1$ for a composite n, say $2^{183} - 1$, and split it up into the factors $C_d(2)$ where d runs through all positive divisors of n. Now look, if the built-in FACTOR of DERIVE can achieve a complete factorization! This procedure can easily be generalized to numbers of the form $a^n - b^n$ with composite n.

$$\text{FACTOR}(x^{12} - 1)$$

$$(x + 1)\cdot(x - 1)\cdot(x^2 + 1)\cdot(x^2 + x + 1)\cdot(x^2 - x + 1)\cdot(x^4 - x^2 + 1)$$

$$\text{VECTOR}(\text{CP}(x,\ d),\ d,\ [1,\ 2,\ 3,\ 4,\ 6,\ 12])$$

$$\left[x - 1,\ x + 1,\ x^2 + x + 1,\ x^2 + 1,\ x^2 - x + 1,\ x^4 - x^2 + 1\right]$$

$$\text{FACTOR}(x^{25} - 1) = (x - 1)\cdot(x^4 + x^3 + x^2 + x + 1)\cdot(x^{20} + x^{15} + x^{10} + x^5 + 1)$$

$$\text{VECTOR}(\text{CP}(x,\ d),\ d,\ [1,\ 5,\ 25])$$

$$\left[x - 1,\ x^4 + x^3 + x^2 + x + 1,\ x^{20} + x^{15} + x^{10} + x^5 + 1\right]$$

For those of you I have lost when dealing with cyclotomic polynomials, I should perhaps mention some other application of the μ-function which is more down to earth. For instance, did you know that one can use it to calculate an approximation of π? All you have to know is that the probability of μ to be nonzero is $6/\pi^2$. This leads to the following DERIVE-function

$$\text{APPROXPI}(n) := \sqrt{\left(\frac{6\cdot n}{\sum_{i=1}^{n} \mu(\text{RANDOM}(1000000))^2}\right)}$$

$$\text{RANDOM}(0) = 1865477040$$

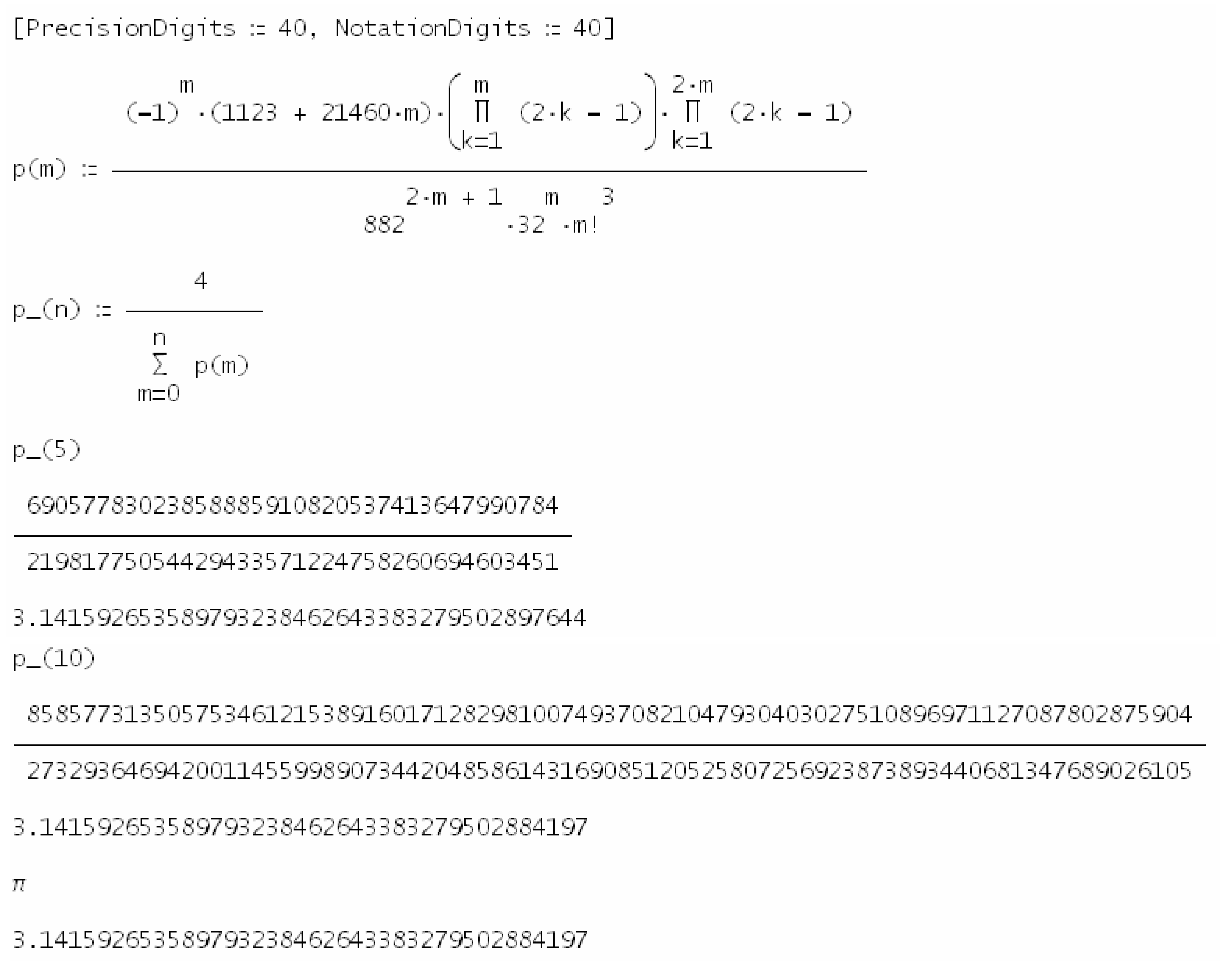
$$\text{APPROXPI}(100) = 3.110855084$$

$$\text{APPROXPI}(200) = 3.188964020$$

Admittedly, it is not as effective as the built-in routine for π which uses according to [2] the following formula of Ramanujan

$$\frac{4}{\pi}=\sum_{m=0}^{\infty}\frac{(-1)^m(1123+21460m)(1\cdot 3\cdots(2m-1))(1\cdot 3\cdots(4m-1))}{882^{2m+1}\cdot 32^m\cdot(m!)^3}$$

but it should even for moderate size of *n*, say *n*=100, easily beat the biblical value 3 of π (cf. 1.kings, ch.7,23). More about μ another time!



```
[PrecisionDigits := 40, NotationDigits := 40]
```

$$p(m) := \frac{(-1)^m\cdot(1123 + 21460\cdot m)\cdot\left(\prod_{k=1}^{m}(2\cdot k - 1)\right)\cdot\prod_{k=1}^{2\cdot m}(2\cdot k - 1)}{882^{2\cdot m + 1}\cdot 32^m\cdot m!^3}$$

$$p\_(n) := \frac{4}{\sum_{m=0}^{n} p(m)}$$

```
p_(5)
```

$$\frac{69057783023858885910820537413647990784}{21981775054429433571224758260694603451}$$

```
3.141592653589793238462643383279502897644
p_(10)
```

$$\frac{8585773135057534612153891601712829810074937082104793040302751089697112708780287590 4}{273293646942001145599890734420485861431690851205258072569238738934406813476890261 05}$$

```
3.141592653589793238462643383279502884197
```

π

```
3.141592653589793238462643383279502884197
```

[1] Nöbauer W.-Wiesenbauer J., Zahlentheorie, Prugg-Verlag, Eisenstadt 1981

[2] Rich A.D. and Stoutemyer D.R., Inside the DERIVE® Computer Algebra System, Int. DERIVE® Journal, 1994, Vol.1, No 1, 3-17.

About Cyclotomic Polynomials (JB)

A cyclotomic polynomial of order *d* is a polynomial given by

$$\Phi_d(x)=\prod_{k=1}^{d}(x-\zeta_k)$$

with ζ_i = the primitive d^{th} Root of Unity in $\mathbb{C}$.

A number *r* is an n^{th} Root of Unity if $r^n = 1$ and it is a primitive n^{th} Root of Unity if *n* is the smallest integer of $k = 1, \ldots, n$ for which $r^k = 1$.

Let´s have an example: What is the cyclotomic polynomial of order 6 = $\Phi_6(x)$?

We can use the function from page 9 to find all Roots of Unity of order 6.

$$\#1: \quad \text{NTH\_ROOTS\_OF\_UNITY}(n) := \text{VECTOR}(e^{i\cdot\pi\cdot 2\cdot k/n}, k, 0, n - 1)$$

$$\#2: \quad \text{root6} := \text{NTH\_ROOTS\_OF\_UNITY}(6)$$

$$\#3: \quad \text{root6} := \left[1, \frac{1}{2} + \frac{\sqrt{3}\cdot i}{2}, -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2}, -1, -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}, \frac{1}{2} - \frac{\sqrt{3}\cdot i}{2}\right]$$

$$\#4: \quad \text{VECTOR}(\text{VECTOR}(r^k, r, \text{root6}), k, 6)$$

$$\#5: \quad \begin{bmatrix} 1 & \frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} & -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} & -1 & -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} & \frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} \\ 1 & -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} & -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} & 1 & -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} & -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} & -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} & 1 & -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} & -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} \\ 1 & \frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} & -\frac{1}{2} - \frac{\sqrt{3}\cdot i}{2} & -1 & -\frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} & \frac{1}{2} + \frac{\sqrt{3}\cdot i}{2} \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\#6: \quad (x - \text{root6}_2)\cdot(x - \text{root6}_6) = x^2 - x + 1$$

Only root #2 and root#6 have $k = n = 6$ as smallest integer to give $r^k = r^n = 1$. A table of cyclotomic polynomials shows $\Phi_6(x) = x^2 - x + 1$ and `CP(x,6)` from above gives the same result, of course.

Try to find $\Phi_4(x)$ and $\Phi_5(x)$ in this way. What about $\Phi_9(x)$?

By the way, Johann mentioned the cyclotomic polynomial with $d = 105$, the next ones with a d of interest are with $d = 385, 1365, 1785$. Check it!

Reference: *CRC Concise Encyclopedia of Mathematics*

Paolo Ruffini

Paolo Ruffini was an Italian mathematician and physician. He was born 1765 in Valentano and he died in 1822 in Modena.

Paolo Ruffini was professor for Applied Mathematics and for Practical Medicin at the University of Modena. As a mathematician he contributed for the solution of algebraic equations. In 1799 he published his conjecture that general equations of order 5 cannot be solved algebraically but his proof was uncomplete.

Titbits(3) - then and now

Johann Wiesenbauer, Vienna University of Technology, June 2007

This is a revised version of my Titbits(3) in the DNL #15. Note that if there are two functions with the same name, the first one is usually the old version (sometimes with small notational changes), which is then overwritten by an updated version that makes full use of all the features of DfW 6.10 along with other improvements.

The first function there, namely

```
      lpd(n, s) :=
        If NEXT_PRIME(n - 1) = n
#1:        n
           ITERATE(IF(t^2 > n, n, IF(MOD(n, t) = 0, t, NEXT_PRIME(t))), t, s)
```

was supposed to return the least prime divisor p of n that is not below a given integer bound s >1, but actually the output p is only an ordinary divisor of n (not necessarily prime!) with the property p>=s. In the following this is corrected, although this error had no implications as to the calls in the subsequent functions. After introducing additionally the default case s=2, this function could nowadays look like this:

```
      lpd(n, s := 2, t_) :=
        Prog
          If PRIME?(n)
             RETURN IF(n ≥ s, n)
          t_ := s
#2:       Loop
            If t_^2 > n
               RETURN ?
            If MOD(n, t_) = 0
               RETURN lpd(t_, s)
            t_ :+ 1
```

$$\#3: \quad \text{lpd}(10^{100} + 9) = 3221$$

The "dual" function to lpd(n,s) would be minprime(n,s) (by the way, an only slightly adapted version of the min_prime(n,s) in my recent update of the Titbits(2)) that computes the least prime divisor p <=s of n, if it exists. For the default cases of s both functions lpd(n) and minprime(n) should coincide, returning the least prime divisor of n, if it exists.

```
      minprime(n, s := ∞, t_ := 2, d_ := 4) :=
        Prog
          If PRIME?(n)
             RETURN IF(n ≤ s, n)
          If s = ∞
             s := FLOOR(√n)
          Loop
#4:         If t_ > s
               RETURN ?
            If MOD(n, t_) = 0
               RETURN t_
            If t_ < 5
               t_ :+ 1
               t_ :+ d_
            d_ := 6 - d_
```

#5: $\text{minprime}(10^{100} + 9) = 3221$

The next function pfl(n) was supposed to give a list of all (not necessarily distinct) prime factors of n in increasing order.

#6: $\text{pfl}(n) := \left(\text{ITERATE}\left(\text{IF}\left(x_2 = 1, x, \left[\text{APPEND}(x_1, [x_3]), \frac{x_2}{x_3}, \text{lpd}\left(\frac{x_2}{x_3}, x_3\right)\right]\right), x, [[], n, \text{lpd}(n, 2)]\right)\right)_1$

#7: $\text{pfl}(360) = [2, 2, 2, 3, 3, 5]$

Well, it works, as the example above shows, but meanwhile the much more powerful built-in function factors(n) has become available.

#8: $\text{FACTORS}(360) = \begin{bmatrix} 2 & 3 \\ 3 & 2 \\ 5 & 1 \end{bmatrix}$

What I was actually aiming at in my article then was an implementation of the Moebius μ-function. By definition μ(n)=0, if n is not squarefree, i.e. at least one of its prime divisors has a multiplicity greater than 1, as the prime divisors 2 and 3 for n=360. On the other hand, if n is a product of r distinct prime numbers, then μ(n) = (-1)^r. Hence, my implementation of μ(n) looked like this then

```
      premoebius(pl) :=
        If DIM(pl) > 1 ∧ ∏(pl↓k - pl↓(k - 1), k, 2, DIM(pl)) = 0
#9:        0
           (-1)^DIM(pl)

#10:  μ(n) := premoebius(pfl(n))
```

#11: $[\mu(1), \mu(30), \mu(60)] = [1, -1, 0]$

Oh well, believe it or not, but even this clumsy implementation was a significant progress at the ancient times of Derive 2.xx! Needless to say that since the availibility of the crucial function factors(n) in version 3 of Derive this has been greatly simplified and now all this has become the following oneliner!!

#12: $\text{MOEBIUS\_MU}(n) := \prod(-\ \text{MAX}(2 - \text{v\_}_2, 0), \text{v\_}, \text{FACTORS}(n))$

There are a number of very important applications of Moebius μ-function, and as an example I gave an implementation of the so-called cyclotomic polynomials. Basically the n-th cyclotomic polynomial can be defined as the product of all linear polynomials x - exp(2πi k/n), where k runs through all numbers in 1,2,..,n, which are coprime to n.

Instead of listing all the clumsy routines for cyclotomic polynomials in the original Titbits(3), which are totally outdated by now anyway, let's use Derive to compute Φ24(x) according to this definition

#13: $$\text{APPROX}\left(\text{EXPAND}\left(\prod\left(x - \text{EXP}\left(\frac{2\cdot\pi\cdot i\cdot k}{12}\right), k, \text{SELECT}(\text{GCD}(k, 12) = 1, k, 1, 12)\right)\right)\right) = x^4 - x^2 + 1$$

The result is quite surprising as the coefficients of the resulting polynomial seem to be integers, although this was far from being true for the linear factors we used to build it! Unfortunately we can use this formula only for very small numbers of n, otherwise the rounding errors get too big. Fortunately there is still another more convenient formula though and here is where the Moebius µ.-function comes into play!

#14: $$\text{CYCLOTOMIC}(n, x) := \prod\left((x^{n/d\_} - 1)^{\text{MOEBIUS\_MU}(d\_)}, d\_, \text{DIVISORS}(n)\right)$$

#15: $$\text{CYCLOTOMIC}(12, x) = x^4 - x^2 + 1$$

Now we can compute Φn(x) in no time for much bigger values of n, e.g.

#16: $$\text{CYCLOTOMIC}(95, x) = x^{72} - x^{71} + x^{67} - x^{66} + x^{62} - x^{61} + x^{57} - x^{56} + x^{53} - x^{51} + x^{48} - x^{46} + x^{43} - x^{41} + x^{38} - x^{36} + x^{34} - x^{31} + x^{29} - x^{26} + x^{24} - x^{21} + x^{19} - x^{16} + x^{15} - x^{11} + x^{10} - x^6 + x^5 - x + 1$$

Are the coefficients always in {-1,0,1} as in this example? Would you think that it is possible to write a "oneliner" that will check this for all n in the range 1<=n<=200? Believe me, it is, and I will give the answer below just to give you a chance to try it yourself.

As a last "application" of µ(n) I also gave in my original article a formula that yields more or less good approximations for π deponding on the size of n. It uses the fact that the probabilty for a natural number n to be squarefree is 6/π^2 and the sum in the denominator of the formula below counts exactly the squarefree numbers, doesn't it?

#17: $$\text{approxpi}(n) := \sqrt{\left(\frac{6\cdot n}{\sum_{i=1}^{n} \text{MOEBIUS\_MU}(\text{RANDOM}(10^6)^2)}\right)}$$

#18: approxpi(10000)

#19: $$3.150485409$$

#20: approxpi(100000)

#21: $$3.141662683$$

Well, let's conclude with the solution of my challenge above:

#22: $\mathrm{CYCLOTOMIC}(105, x) = x^{48} + x^{47} + x^{46} - x^{43} - x^{42} - 2\cdot x^{41} - x^{40} - x^{39} +$
$x^{36} + x^{35} + x^{34} + x^{33} + x^{32} + x^{31} - x^{28} - x^{26} - x^{24} - x^{22} - x^{20} + x^{17}$
$+ x^{16} + x^{15} + x^{14} + x^{13} + x^{12} - x^{9} - x^{8} - 2\cdot x^{7} - x^{6} - x^{5} + x^{2} + x + 1$

#23: $\mathrm{CYCLOTOMIC}(165, x) = x^{80} + x^{79} + x^{78} - x^{75} - x^{74} - x^{73} - x^{69} - x^{68} - x^{67}$
$+ x^{65} + 2\cdot x^{64} + 2\cdot x^{63} + x^{62} - x^{60} - x^{59} - x^{58} - x^{54} - x^{53} - x^{52} + x^{50}$
$+ 2\cdot x^{49} + 2\cdot x^{48} + 2\cdot x^{47} + x^{46} - x^{44} - x^{43} - x^{42} - x^{41} - x^{40} - x^{39} -$
$x^{38} - x^{37} - x^{36} + x^{34} + 2\cdot x^{33} + 2\cdot x^{32} + 2\cdot x^{31} + x^{30} - x^{28} - x^{27} - x^{26}$
$- x^{22} - x^{21} - x^{20} + x^{18} + 2\cdot x^{17} + 2\cdot x^{16} + x^{15} - x^{13} - x^{12} - x^{11} - x^{7} -$
$x^{6} - x^{5} + x^{2} + x + 1$

#24: $\mathrm{CYCLOTOMIC}(195, x) = x^{96} + x^{95} + x^{94} - x^{91} - x^{90} - x^{89} - x^{83} - x^{82} + x^{80}$
$+ x^{79} + x^{78} + x^{77} - x^{75} - x^{74} - x^{68} - x^{67} + x^{65} + x^{64} + x^{63} + x^{62} -$
$x^{60} - x^{59} + x^{57} + x^{56} + x^{55} - x^{53} - 2\cdot x^{52} - x^{51} + x^{49} + x^{48} + x^{47} -$
$x^{45} - 2\cdot x^{44} - x^{43} + x^{41} + x^{40} + x^{39} - x^{37} - x^{36} + x^{34} + x^{33} + x^{32} +$
$x^{31} - x^{29} - x^{28} - x^{22} - x^{21} + x^{19} + x^{18} + x^{17} + x^{16} - x^{14} - x^{13} - x^{7} -$
$x^{6} - x^{5} + x^{2} + x + 1$

No, of course, this is not yet the solution, rather sort of a red herring.

But here it comes at last!

```
#25:  SELECT(|∏(SUBST(TERMS(CYCLOTOMIC(n, x)), x, 1))| > 1, n, 200)

#26:                          [105, 165, 195]
```

The spirit of the Plymouth DERIVE conference inspired two of our DUG-members to write poems about DERIVE and its symbol - the little gecko.

Sergey Biryukov is not only a skilled DERIVE user but also a poet. Like a great actor he recited his poem and for all of us who were present this was a great moment:

DERIVE - ing physics at the class
We use the air and the glass
We Plot the rays, we mirror Plot
Light source we represent as Dot.

We use Expand and Simplify
We say hand calculus good-bye
DERIVE we use as tool for mind
Is a unique software kind.

It gives numeric, graphic face
Its vectors handle every space
An analytic face - the main
It is intelligence retrain.

If use DERIVE in mathes need
You will undoubtedly succeed!

Jim Hall wrote a little song, too:

I´M A LITTLE GECKO
(to the tune of ´I´m a Little Tea Pot´)

I´m a little gecko
Slim, not stout.
Here is my tail and
Here is my snout.
If you pull my tail off,
Hear me shout:
´Ha - ha - ha - it will
Grow back out!´