THE DERIVE - NEWSLETTER #89

ISSN 1990-7079

THE BULLETIN OF THE

DERIVE

USER GROUP

+ CAS-TI

Contents:

March/April 2013

Websites offering Free Online Courses:

Learn. Think. Do.
Advance Your Education with Free College Courses Online: www.udacity.com

Free Online Courses: www.coursera.org

Explore Free Courses from edX Universities: www.edX.org

Learn almost anything for free: www.khanacademy.org

Soziale Plattform für Interaktive Online-Kurse zur Informationstechnologie (deutsch)
www.openhpi.de

Download die Vorträge der Lehrertage an der Universität Wien für viele Jahre:
http://www.oemg.ac.at/DK/Didaktikhefte/index.html

The DERIVE-NEWS archive:

https://www.jiscmail.ac.uk/cgi-bin/webadmin?A0=derive-news

Information from Djordje Kadijevich, Serbia:

Dear Colleagues,

Hope you are fine.
Just to tell you about my new book published by Routledge:

Improving Computer Science Education, suitable for courses on didactics of informatics (didactics of computer science).

More information about this book can, for example, be found at the Publisher web site at

www.routledge.com/books/details/9780415645379/

For on-line orders at this site, you can enter code ERJ60 to receive a 20% discount.
Could you please circulate this information among interested colleagues and students.
Kind regards,
Djordje

Liebe DUG-Mitglieder,

DNL#89 hat einige Zeit zur Fertigstellung benötigt. Er war sehr arbeits- und zeitaufwändig, da die meisten Beiträge erst ins Englische übersetzt werden mussten. Herzlichen Dank an Herrn Erik van Lantschoot, der nicht nur den schönen Beitrag zu einem historischen Bauwerk seines Geburtsorts sondern auch die Übersetzung beisteuerte. Für die anderen Übersetzungen bin ich ganz alleine verantwortlich.

Ich musste einige – sehr interessante – Beiträge zum User Forum für den nächsten DNL aufheben. Alle Anfragen haben ausführliche Beantwortungen nach sich gezogen. Dies würde den Rahmen des DNL bei weitem sprengen.

So hat mich die Frage von Herrn Körner nach dem „Wert" der Polynomdivision sehr lange und ausführlich beschäftigt. Hoffentlich können Sie Teile meiner Antwort auch gebrauchen. Wenn Sie noch weitere Kommentare beitragen wollen, sind Sie herzlich dazu eingeladen.

Kürzlich kam die Meldung, dass die DERIVE-NEWS list eingestellt wird, wenn sich kein neuer Administrator melden würde. Glücklicherweise hat sich Lars Erup (Kanada) dazu bereit erklärt. Herzlichen Dank dafür und viel Erfolg! Auf Seite 47 finden Sie seine Nachricht und die Internetverbindung zu dieser News Group. Bitte nützen Sie diese weitere Kommunikationsmöglichkeit.

Am Ende dieses DNL finden Sie den Folder zur TIME 2014, die im schönen Krems, Niederösterreich, stattfinden wird.

Ich verbleibe mit den besten Grüßen bis zum nächsten Mal

Ihr

Dear DUG Members,

DNL#89 needed some time to be finished. Most of the contributions are translations from German. Many thanks to Erik van Lantschoot. He not only contributed his fine article describing a detail of a historic landmark of his birthplace but also the translation. I am full responsible for all other translations.

I must leave some – very interesting – contributions for our User Forum for the next newsletter. They all require extended answers which would be far beyond the space of this DNL.

Take Klaus Körner's question for the "value" of polynomial long division as an example. I hope that you can use at least parts of my answer. Additional comments would be very welcome.

It was just recently when I received a mail that the DERIVE News list would be discontinued after many years of its existence if nobody from within the news group would volunteer as list administrator. Fortunately Lars Erup (Canada) took over the job. Many thanks for this and much success in the future! You can find his message on page 47 together with the internet connection to the DERIVE News Group. Please use this additional possibility for communication within the DERIVE-CAS community.

At the end of this DNL you can find the information folder for TIME 2014 which will be held in July 2014 in lovely Krems, Lower Austria.

I remain with my best regards
until DNL#90
Yours

The *DERIVE-NEWSLETTER* is the Bulletin of the *DERIVE* & CAS-*TI User Group*. It is published at least four times a year with a contents of 40 pages minimum. The goals of the *DNL* are to enable the exchange of experiences made with *DERIVE*, *TI*-CAS and other CAS as well to create a group to discuss the possibilities of new methodical and didactical manners in teaching mathematics.

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

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

Next issue: June 2013

Preview: Contributions waiting to be published

Some simulations of Random Experiments, J. Böhm, AUT, Lorenz Kopp, GER
Wonderful World of Pedal Curves, J. Böhm, AUT
Tools for 3D-Problems, P. Lüke-Rosendahl, GER
Hill-Encryption, J. Böhm, AUT
Simulating a Graphing Calculator in *DERIVE*, J. Böhm, AUT
Do you know this? Cabri & CAS on PC and Handheld, W. Wegscheider, AUT
An Interesting Problem with a Triangle, Steiner Point, P. Lüke-Rosendahl, GER
Graphics World, Currency Change, P. Charland, CAN
Cubics, Quartics – Interesting features, T. Koller & J. Böhm, AUT
Logos of Companies as an Inspiration for Math Teaching
Exciting Surfaces in the FAZ / Pierre Charland´s Graphics Gallery
BooleanPlots.mth, P. Schofield, UK
Old traditional examples for a CAS – what's new? J. Böhm, AUT
Truth Tables on the TI, M. R. Phillips, USA
Where oh Where is It? (GPS with CAS), C. & P. Leinbach, USA
Embroidery Patterns, H. Ludwig, GER
Mandelbrot and Newton with *DERIVE*, Roman Hašek, CZK
Some Projects with Students, R. Schröder, GER
Dirac Algebra, Clifford Algebra, D. R. Lunsford, USA
Laplace Transforms, ODEs and CAS, G. Picard & Ch. Trottier, CAN
A New Approach to Taylor Series, D. Oertel, GER
Cesar Multiplication, G. Schödl, AUT
Henon & Co; Find your very own Strange Attractor, J. Böhm, AUT
Rational Hooks, J. Lechner, AUT
Simulation of Dynamic Systems with various Tools, J. Böhm, AUT
An APL-like SHAPE function in DERIVE 6, D. R. Lunsford, USA
Recursive Series of Numbers, Polynomials and Functions, D. Halprin, AUS
Inspirations from other sources, J. Böhm, AUT
Teaching Calculus, G. Savard, CAN
and others

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

A Mathematical Model for Snail Shells (4)

Piotr Trebisz, Germany

Der Ansatz, wonach man das Selbstdurchdringungsproblem durch Verwendung krummliniger Koordinatensysteme analytisch lösbar macht, kann natürlich auch auf konische Schneckenhäuser übertragen werden. Konische Schneckenhäuser dieser Art nenne ich „Trebisz-Spiralen – Typ 4".

Es handelt sich dabei aber um mehr als nur um einen mathematischen Trick oder reine Spielerei! Fakt ist dass die sich so ergebenden Schneckenhäuser insgesamt schöner und eleganter sind und – mehr noch als ihre Gegenstücke vom Typ 2 – aus einem Guss erscheinen, weil bei ihnen tatsächlich sämtliche geometrischen Eigenschaften durch die logarithmische Spirale bestimmt werden.

Im finalen 4. Teil meiner Beitragsreihe werde ich zeigen wie das korrespondierende krummlinige Koordinatensystem konischer Schneckenhäuser aus der Spiralen-Gleichung hergeleitet wird. Die Vorgehensweise ist dabei analog zu Trebisz-Spiralen von Typ 3.

Ausgangspunkt unserer Überlegungen ist die Gleichung einer konischen logarithmischen Spirale mit den 3 Parametern *b* (Basis), *t* (Länge) und *m* (Proportionalitätsfaktor) und ihre Tangente.

Gesucht ist nun eine Anti-Spirale mit der Basis *a*, die die normale Spirale in den Schnittpunkten immer senkrecht schneidet. Ihr Proportionalitätsfaktor *m* ist identisch, so ist gewährleistet dass sie sich auf derselben Kegeloberfläche windet. (Siehe auch meinen Kommentar auf Seite 11, Josef)

Um Platz zu sparen sind Piotrs Kommentare nur in (meiner) englischen Fassung, Josef.

The concept solving the interpenetrating problem by using curvilinear coordinate systems can be transferred to conical snail shells, of course. I will call these snail shells "*Type 4 Trebisz-Spirals*".

This is more than a mathematical trick or pure playing around. Fact is that the resulting snail shells are much more beautiful and elegant. They appear all of a piece – even more than the respective type 2 models do because all their geometric attributes are determined by the logarithmic spiral.

In this 4^{th} – and final – contribution I will show how the corresponding curvilinear coordinate system of conical snail shells can be derived from the equation of the spiral. The procedure is the same as demonstrated for the Type 3 Trebisz Spirals.

Important settings for the state variables are:

```
[Precision := Exact, PrecisionDigits := 10]

[Notation := Rational, NotationDigits := 10]

[Branch := Principal, Exponential := Auto, Logarithm := Auto]

[Trigonometry := Auto, Trigpower := Auto, Angle := Radian]

[InputMode := Word, CaseMode := Sensitive]

[b :∊ Real (1, ∞), m :∊ Real (0, ∞), t :∊ Real [0, ∞)]

[θ :∊ Real, φ :∊ Real [-π, +π], a :∊ Real (0, 1), x :∊ Real [0, ∞)]

[R₁ :∊ Real (0, ∞), R₂ :∊ Real (0, ∞), k :∊ Integer]
```

Starting point for our considerations is the equation of a conical logarithmic spiral defined by three parameters: *b* (base), *t* (length), and *m* (proportionality factor). An anti-spiral (base *a*) is needed which intersects the given spiral under 90°. Its prop factor is the same (*m*) which makes sure that both spirals are winding on the same conic surface. They intersect for $t = 1$.

```
SPIRALE(b, m, t) := [t, m·t·COS(2·π·LOG(t, b)), m·t·SIN(2·π·LOG(t, b))]

                           d
SPIRALE_TANGENTE(b, m, t) := — SPIRALE(b, m, t)
                           dt

ANTI_SPIRALE(a, m, t) := [t, m·t·COS(2·π·LOG(t, a)), m·t·SIN(2·π·LOG(t, a))]

                                d
ANTI_SPIRALE_TANGENTE(a, m, t) := — ANTI_SPIRALE(a, m, t)
                                dt

[    SPIRALE(b, m, 1)     ]   [ [1, m, 0] ]
[                         ] = [           ]
[ ANTI_SPIRALE(a, m, 1)   ]   [ [1, m, 0] ]
```

Base *a* of the anti-spiral must be such chosen that both spirals intersect under 90°. This condition is given in the next equation which is then solved for *a* (*a* = BASIS(b, m)).

```
SPIRALE_TANGENTE(b, m, 1) • ANTI_SPIRALE_TANGENTE(a, m, 1) = 0

BASIS(b, m) := (SOLUTIONS(SPIRALE_TANGENTE(b, m, 1) •

  ANTI_SPIRALE_TANGENTE(a, m, 1) = 0, a, Real))
                                               1

                          2   2      2
                  - 4·π ·m /((m  + 1)·LN(b))
BASIS(b, m) = e
```

We still miss the local *y*-axis. It must be perpendicular to the tangent of the anti spiral and in the intersection points of the spirals perpendicular to the tangent of the given spiral, too. For the type 3 shells is this the normal of the plane which contains the planar spiral. Here the normal of the cone surface turns out to be an appropriate local *y*-axis. This normal line is constructed by choosing to every point *t* on the anti-spiral a corresponding point *x* on the *x*-axis such that the line connecting these two points and the tangent of the anti-spiral are perpendicular. The length of the connecting segment depends linearly on *t*.

```
LINIE(a, m, t, x) := ANTI_SPIRALE(a, m, t) - [x, 0, 0]

PUNKT(a, m, t) := (SOLUTIONS(LINIE(a, m, t, x) • ANTI_SPIRALE_TANGENTE(a,

  m, t) = 0, x, Real))
                     1

ANTI_SPIRALE_ABSTAND(a, m, t) := LINIE(a, m, t, PUNKT(a, m, t))

                                           2                    (*)
|ANTI_SPIRALE_ABSTAND(a, m, t)| = m·t·√(m  + 1)
```

We must get rid of this dependency if we will use this line as y-axis in our curvilinear system of coordinates. Now we have got much closer to our goal because we know both axes of the coordinate system.

$$\text{ANTI\_SPIRALE\_NORMALE}(a, m, t) := \frac{\text{ANTI\_SPIRALE\_ABSTAND}(a, m, t)}{t}$$

$$\text{X\_ACHSE}(b, m, t) := \text{ANTI\_SPIRALE}(\text{BASIS}(b, m), m, t)$$

$$\text{Y\_ACHSE}(b, m, t) := \text{ANTI\_SPIRALE\_NORMALE}(\text{BASIS}(b, m), m, t)$$

In these curvilinear coordinates we can define the surface ellipse of the shell with radius R_1 in x-direction and a radius R_2 in y-direction.

$$\text{ELLIPSE}(b, m, R_1, R_2, \phi) := \text{X\_ACHSE}(b, m, 1 + R_1\cdot\text{COS}(\phi)) + \text{Y\_ACHSE}(b, m, 1 + R_1\cdot\text{COS}(\phi))\cdot R_2\cdot\text{SIN}(\phi)$$

The centre of the ellipse is at $t = 1$. How to choose both axes? R_1 should be chosen that the surface of the solid does not interpenetrate, and R_2 in such a way that the ellipse osculates the global x-axis (without intersecting it).

Let's concentrate first on R_1 because it is calculated like the radius of a type 3 shell. The intersection points of both spirals are needed. They both are parameterized wrt to the length and they both have identical x-components. So we need all values for t which give identical y- and z-components. The solutions should be known from the type 3 shells.

$$\text{SOLUTIONS}((\text{SPIRALE}(b, m, t))_2 = (\text{X\_ACHSE}(b, m, t))_2 \wedge \text{SOLVE}((\text{SPIRALE}(b, m, t))_3 = (\text{X\_ACHSE}(b, m, t))_3, t, \text{Real}), t, \text{Real})$$

$$\left[0, 1, b^{-4\cdot\pi^2\cdot m^2/((m^2 + 1)\cdot\text{LN}(b)^2 + 4\cdot\pi^2\cdot m^2)}, b^{4\cdot\pi^2\cdot m^2/((m^2 + 1)\cdot\text{LN}(b)^2 + 4\cdot\pi^2\cdot m^2)}\right]$$

DERIVE returns only 4 special solutions but there are infinite many of them. By inspection it is possible to give a general formula, k is an arbitrary integer.

The difference of two intersections gives the "winding number", i.e. the number of rotations after which the surface is osculating itself.

$$\text{SCHNITT}(b, m, k) := b^{4\cdot\pi^2\cdot m^2\cdot k/((m^2 + 1)\cdot\text{LN}(b)^2 + 4\cdot\pi^2\cdot m^2)}$$

$$\text{WINDUNG}(b, m) := \text{LOG}(\text{SCHNITT}(b, m, k + 1), b) - \text{LOG}(\text{SCHNITT}(b, m, k), b)$$

$$\text{WINDUNG}(b, m) := \frac{4\cdot\pi^2\cdot m^2}{(m^2 + 1)\cdot\text{LN}(b)^2 + 4\cdot\pi^2\cdot m^2}$$

Now it is possible to calculate radius R_1 in direction of the x-axis.

$$\text{RADIUS\_X}(b, m) := (\text{SOLUTIONS}(\text{SCHNITT}(b, m, k)\cdot(1 + R_1) = \text{SCHNITT}(b, m, k + 1)\cdot(1 - R_1), R_1, \text{Real}))_1$$

$$\text{RADIUS\_X}(b, m) := \frac{b^{4\cdot\pi^2\cdot m^2/((m^2 + 1)\cdot\text{LN}(b)^2 + 4\cdot\pi^2\cdot m^2)} - 1}{b^{4\cdot\pi^2\cdot m^2/((m^2 + 1)\cdot\text{LN}(b)^2 + 4\cdot\pi^2\cdot m^2)} + 1}$$

It is not too difficult to find radius R_2 (in direction of the y-axis). As the ellipse shall only osculate the global x-axis we need to know the distance of any arbitrary point on the anti-spiral to this axis along the local y-axis. The attentive reader will remember equation (∗) from above (page 4). This distance has already been calculated. It scaled according to the scale of the local y-axis and we change this by the respective division:

$$|\text{ANTI\_SPIRALE\_ABSTAND}(a, m, t)| = m\cdot t\cdot\sqrt{(m^2 + 1)}$$

$$\frac{|\text{ANTI\_SPIRALE\_ABSTAND}(a, m, t)|}{|\text{ANTI\_SPIRALE\_NORMALE}(a, m, t)|} = t$$

This means that the distance of any point of the anti-spiral from the global x-axis is exact t. Considering this fact we can set up an equation in order to determine R_2. This works as follows:

1. We use a 2-dimensional system of coordinates which is a representative for the local curvilinear system with its x- and y-axis. As a consequence a diagonal with $y = t$ corresponds with the global x-axis which shall be tangent to the ellipse.

2. Analogous to the type 3 shell a circle with radius R_1 is placed for $t = 1$. Is equation is:

$$y = \sqrt{(R_1^2 - (t - 1)^2)}$$

But this is not the requested ellipse because it should have radius R_2 in y-direction. How to achieve this? This is not difficult at all: just scale the circle:

$$y = \sqrt{(R_1^2 - (t - 1)^2)}\cdot\frac{R_2}{R_1}$$

This equation enables finding the intersection points of the ellipse and the diagonal $y = t$.

$$\text{RADIUS\_Y\_TEMP}(b, m, R_2) := \text{SOLUTIONS}\left(\frac{\sqrt{(\text{RADIUS\_X}(b, m)^2 - (t - 1)^2)}}{\text{RADIUS\_X}(b, m)}\cdot R_2 = t, t, \text{Real}\right)$$

Then we choose R_2 in such a way that both intersection points coincide because then – and only then – the global ellipse is osculating the x-axis (without intersecting it).

$$\mathrm{RADIUS\_Y}(b, m) := \left(\mathrm{SOLUTIONS}\left(\left(\mathrm{RADIUS\_Y\_TEMP}(b, m, R_2)\right)_1 = \left(\mathrm{RADIUS\_Y\_TEMP}(b, m, R_2)\right)_2, R_2, \mathrm{Real}\right)\right)_2$$

$$\mathrm{RADIUS\_Y}(b, m) := \frac{2 \cdot b^{2 \cdot \pi^2 \cdot m^2/((m^2 + 1) \cdot \mathrm{LN}(b)^2 + 4 \cdot \pi^2 \cdot m^2)}}{b^{4 \cdot \pi^2 \cdot m^2/((m^2 + 1) \cdot \mathrm{LN}(b)^2 + 4 \cdot \pi^2 \cdot m^2)} + 1}$$

Now we know both radii, R_1 and R_2, We use the surface ellipse for building the surface of the snail shell applying all three kinds of parameterization:

With respect to the length:

$$\mathrm{SchneckenTorusX\_L}(b, m, t, \phi) := t \cdot \mathrm{ROTATE\_X}(2 \cdot \pi \cdot \mathrm{LOG}(t, b)) \bullet \mathrm{ELLIPSE}(b, m, \mathrm{RADIUS\_X}(b, m), \mathrm{RADIUS\_Y}(b, m), \phi)$$

With respect to the rotation number:

$$\mathrm{SchneckenTorusX\_U}(b, m, \theta, \phi) := b^{\theta} \cdot \mathrm{ROTATE\_X}(2 \cdot \pi \cdot \theta) \bullet \mathrm{ELLIPSE}(b, m, \mathrm{RADIUS\_X}(b, m), \mathrm{RADIUS\_Y}(b, m), \phi)$$

With respect to the winding number

$$\mathrm{SchneckenTorusX\_W}(b, m, \theta, \phi) := \mathrm{SchneckenTorusX\_U}(b, m, \theta \cdot \mathrm{WINDUNG}(b, m), \phi)$$

We have not yet reached our final goal because a snail shell with elliptic surface form does not look really beautiful and elegant. It would be much better if the surface ellipse would be circular in the local curvilinear coordinate system. For achieving this both radii, R_1 and R_2, must be equal – scaled with the measures of both axes.

R_1 and R_2 depend on the base b and the prop factor m. We look for the zero of m in the following equation:

$$0 = \mathrm{RADIUS\_X}(b, m) \cdot \left| \frac{d}{dt} \mathrm{X\_ACHSE}(b, m, t) \right| - \mathrm{RADIUS\_Y}(b, m) \cdot \left| \mathrm{Y\_ACHSE}(b, m, t) \right|$$

After substitution we receive the following equation:

$$0 = \frac{\sqrt{(m^2 + 1)} \cdot \left(b^{4 \cdot \pi^2 \cdot m^2/((m^2 + 1) \cdot \mathrm{LN}(b)^2 + 4 \cdot \pi^2 \cdot m^2)} - 1\right) \cdot \sqrt{((m^2 + 1) \cdot \mathrm{LN}(b)^2 + 4 \cdot \pi^2 \cdot m^2)}}{2 \cdot \pi \cdot m \cdot \left(b^{4 \cdot \pi^2 \cdot m^2/((m^2 + 1) \cdot \mathrm{LN}(b)^2 + 4 \cdot \pi^2 \cdot m^2)} + 1\right)} - \frac{2 \cdot m \cdot b^{2 \cdot \pi^2 \cdot m^2/((m^2 + 1) \cdot \mathrm{LN}(b)^2 + 4 \cdot \pi^2 \cdot m^2)} \cdot \sqrt{(m^2 + 1)}}{b^{4 \cdot \pi^2 \cdot m^2/((m^2 + 1) \cdot \mathrm{LN}(b)^2 + 4 \cdot \pi^2 \cdot m^2)} + 1}$$

which can be further simplified to:

$$0 = \left(b^{4\cdot\pi^2\cdot m^2/((m^2+1)\cdot LN(b)^2+4\cdot\pi^2\cdot m^2)} - 1\right)\cdot\sqrt{(m^2+1)\cdot LN(b)^2 + 4\cdot\pi^2\cdot m^2} - 4\cdot\pi\cdot m^2\cdot b^{2\cdot\pi^2\cdot m^2/((m^2+1)\cdot LN(b)^2+4\cdot\pi^2\cdot m^2)}$$

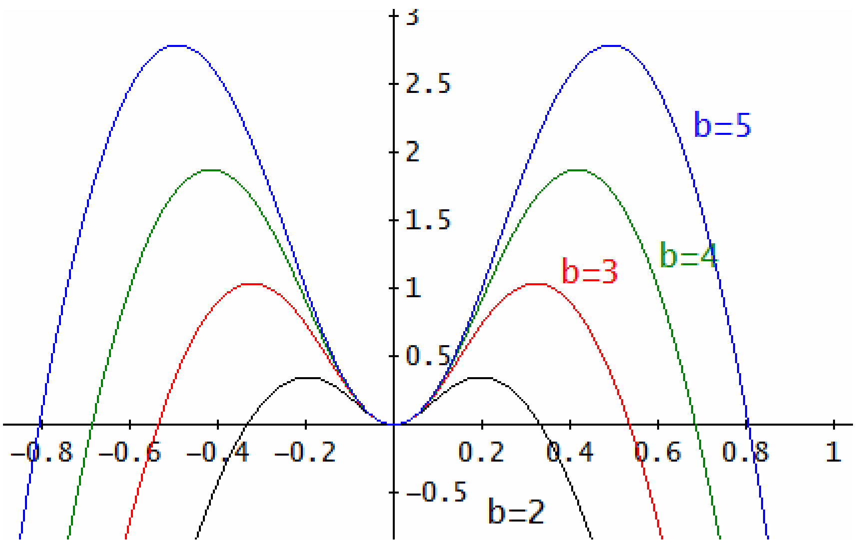
Unfortunately this equation cannot be solved for *m*. We have no other choice: we must apply again the well-tried Newton-Raphson-algorithm for determining *m* numerically.
Function *f*(*m*,*b*) is the equation to be solved. *g*(*m*,*b*) is its first derivative, and *h*(*b*,*m*,*n*) is calculating the next iteration of the algorithm accurate to *n* decimal places.(*)

$$f(b, m) := \left(b^{4\cdot\pi^2\cdot m^2/((m^2+1)\cdot LN(b)^2+4\cdot\pi^2\cdot m^2)} - 1\right)\cdot\sqrt{(m^2+1)\cdot LN(b)^2 + 4\cdot\pi^2\cdot m^2} - 4\cdot\pi\cdot m^2\cdot b^{2\cdot\pi^2\cdot m^2/((m^2+1)\cdot LN(b)^2+4\cdot\pi^2\cdot m^2)}$$

$$g(b, m) := \frac{d}{dm} f(b, m)$$

```
h(b, m, n) :=
  Prog
    δm :∈ Real (0, ∞)
    WRITE(APPEND("m := ", APPROX(STRING(m), n)))
    δm := m - APPROX(f(b, m), 2·n)/APPROX(g(b, m), 2·n)
    RETURN APPROX(δm, n)
```

For finding approximations of the *m*-zeros of *f*(*b*,m) we need a suitable initial value. We will investigate the run of the function.



We see that the function starts increasing and then after passing its maximum point turns to minus infinity.

(*) See our dialogue at the end of Piotr's contribution. Josef

Can we give a general statement about its slope behavior which does not depend on its base *b*?
Yes, we can! First of all, $f(b,m=0) = 0$, and the limit $f(b,m \to \infty) = -\infty$.

$$\left[f(b, 0), \lim_{m \to \infty} f(b, m)\right] = [0, -\infty]$$

This information is useful for finding a suitable initial value m_0 for the approximation algorithm.
It is easy! We start with $m_0 = 1$ and then increase m_0 until $f(b,m_0)$ becomes negative. Then we go backwards to the zero applying the Newton-Raphson-algorithm. An implementation of this algorithm could look as follows:

```
M(b, n) :=
  Prog
    m0 :∈ Real (0, ∞)
    f0 :∈ Real
    V :∈ Vector
    m0 := 1
    Loop
      f0 := APPROX(f(b, m0), 2·n)
      If f0 > 0
         m0 := m0 + 1
         If f0 < 0
            exit
            RETURN m0
    V := ITERATES(h(b, m, n), m, m0)
    RETURN V↓DIM(V)
```

Now we are really done and our award is the presentation of wonderful conical snail shells defined by three parameterizations.

```
SchneckenHausX_U(b, n, θ, φ) := SchneckenTorusX_U(b, M(b, n), θ, φ)

SchneckenHausX_W(b, n, θ, φ) := SchneckenTorusX_W(b, M(b, n), θ, φ)

SchneckenHausX_L(b, n, t, φ) := SchneckenTorusX_L(b, M(b, n), t, φ)
```

(U for "rotation number" (= Umdrehungszahl), W for "winding number" (= Windungszahl)
and L for "Length" (= Länge)
Or in English:

```
SnailShellX_L(b, n, t, φ) := SchneckenTorusX_L(b, M(b, n), t, φ)

SnailShellX_U(b, n, θ, φ) := SchneckenTorusX_U(b, M(b, n), θ, φ)

SnailShellX_W(b, n, θ, φ) := SchneckenTorusX_W(b, M(b, n), θ, φ)
```

Piotr provided a distance function which makes possible to plot three snail shells in one 3D plot:

```
ABSTAND(α, r) := r·[0, COS(α), SIN(α)]

DIST(α, r) := r·[0, COS(α), SIN(α)]
```

$$\text{SchneckenHausX\_L}(2,\ 20,\ t,\ s) + \text{ABSTAND}\left(-\ \frac{5}{6}\cdot\pi,\ 1\right)$$

$$\text{SchneckenHausX\_U}(6,\ 20,\ t,\ s) + \text{ABSTAND}\left(-\ \frac{1}{6}\cdot\pi,\ 1\right)$$

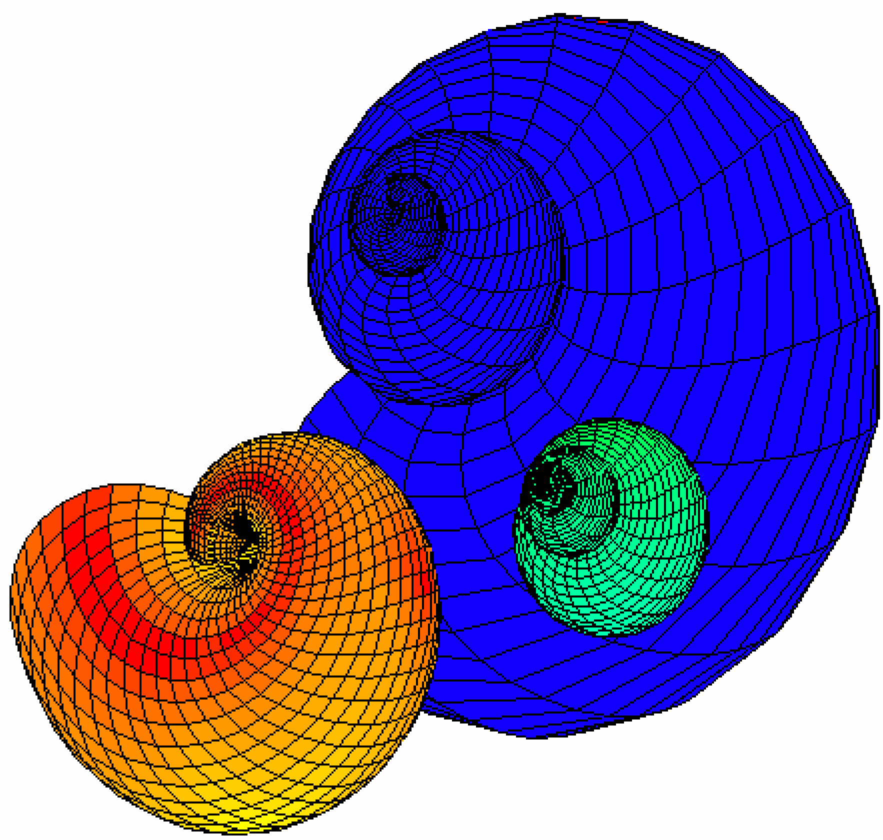
$$\text{SchneckenHausX\_W}(e,\ 20,\ t,\ s) + \text{ABSTAND}\left(+\ \frac{1}{2}\cdot\pi,\ 1\right)$$

Using the English function names:

$$\text{SnailShellX\_L}(2,\ 20,\ t,\ s) + \text{DIST}\left(-\ \frac{5}{6}\cdot\pi,\ 1\right)$$

$$\text{SnailShellX\_U}(6,\ 20,\ t,\ s) + \text{DIST}\left(-\ \frac{1}{6}\cdot\pi,\ 1\right)$$

$$\text{SnailShellX\_W}(e,\ 20,\ t,\ s) + \text{DIST}\left(+\ \frac{1}{2}\cdot\pi,\ 1\right)$$

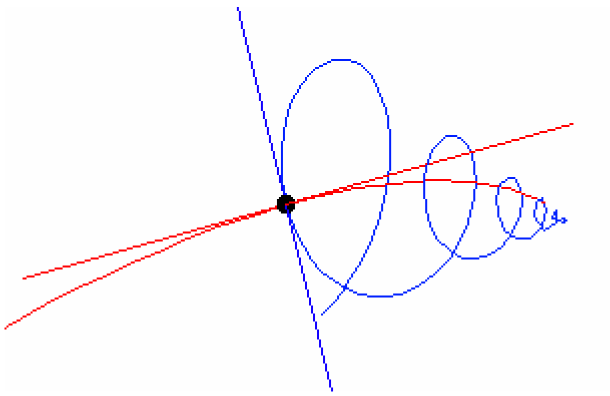


Very strange: One can plot the snails and enjoy their beautiful shapes. After saving the file as a dfw-file (as one is accustomed to) and then reloading plotting again does not work. One can find a strange message in the message bar (created by WRITE in function h(b,m,n). If saved as mth-file you will face no problems at all. Josef. (See also DNL#84 and the following comments.)

Comments and email-dialogue with Piotr Trebisz:

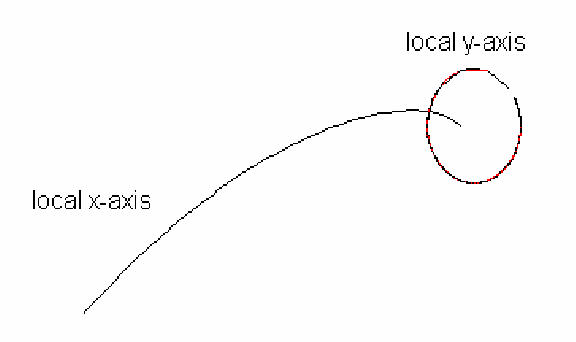
My questions sent to Piotr:
In the first figure I plot the spiral together with its respective anti-spiral and the corresponding tangents in their – trivial – intersection point.

```
SPIRAL(2, 0.4, t)

ANTI_SPIRAL(BASE(2, 0.4), 0.4, t)

[1, 0.4, 0]

[1, 0.4, 0] + t·SPIRAL_TANGENT(2, 0.4, 1)

[1, 0.4, 0] + t·ANTI_SPIRAL_TANGENT(BASE(2, 0.4), 0.4, 1)
```



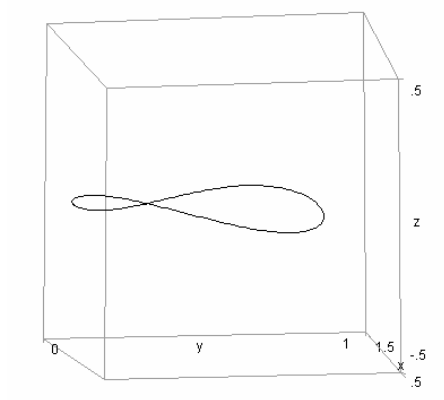
The second figure shows the curvilinear coordinate system. Is this ok?

```
X_AXIS(2, 0.4)

Y_AXIS(2, 0.4)
```



The next figure should show the “surface ellipse”; but it is not an ellipse at all.

```
ELLIPSE(2, 0.4, RADIUS_X(2, 0.4), RADIUS_Y(2, 0.4), φ)
```

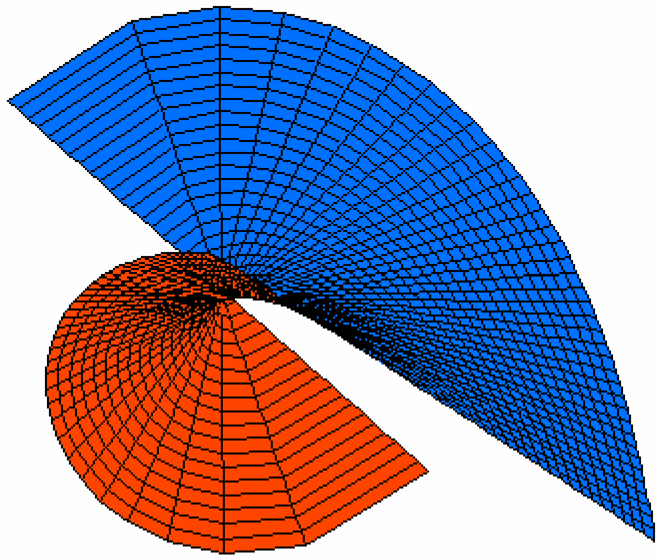


Piotr's answers:

To be honest, I am not very happy with your second figure. Graphs of both axes of the curvilinear coordinate system do not give a meaningful representation and impression of the whole system. I would prefer plotting a plane in this system. The parameter lines give a much better insight of the system.

Example: $b = 2$ and $m = 0.4$

```
X_ACHSE(2, 0.4, t) + s·Y_ACHSE(2, 0.4, t)
```



The surface ellipse is an "ellipse" in the curvilinear coordinate system, not in the Cartesian one! This system is curved and has a torsion. This is the reason for the ellipse to appear as a twisted oval as shown in your third figure.

My reply (and more questions):

You are right with your representation of the coordinate system. I find the plot of the "twisted" ellipse very interesting and I will put it together with the "curved plane" and your comments into the next DNL.

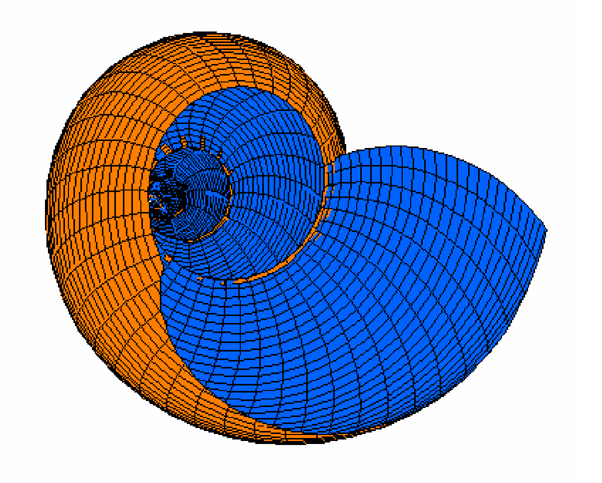
I have two more questions:

(1) I wonder if it is possible to produce a longitudinal intersection of the shells. Is this possible using the curvilinear coordinate system?

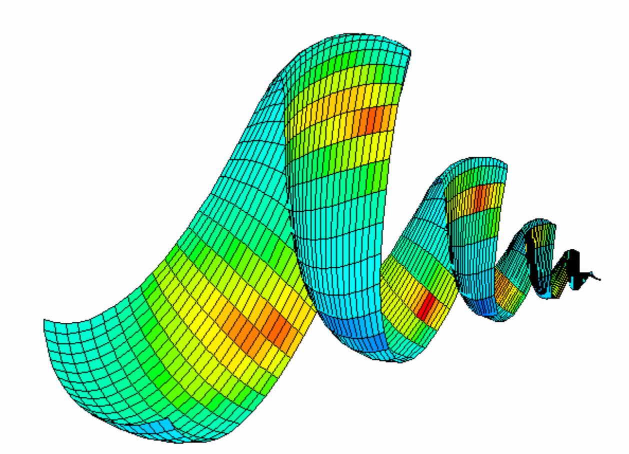
A nice view into the interior of the shell is given by parameter domains $0 \le \phi \le \pi$ ($\pi/2$) and $0 \le t \le 2$ for the length-parameterized shells. I attach two plots.

(2) I come back to a question posed in an earlier DNL, too. Why do you not use NSOLUTIONS for finding the zero of $f(b,m)$? Then you could do without the ITERATES- construct.

The interior of a shell ($\phi = s$):



$0 \le t \le 2, 0 \le s \le \pi$



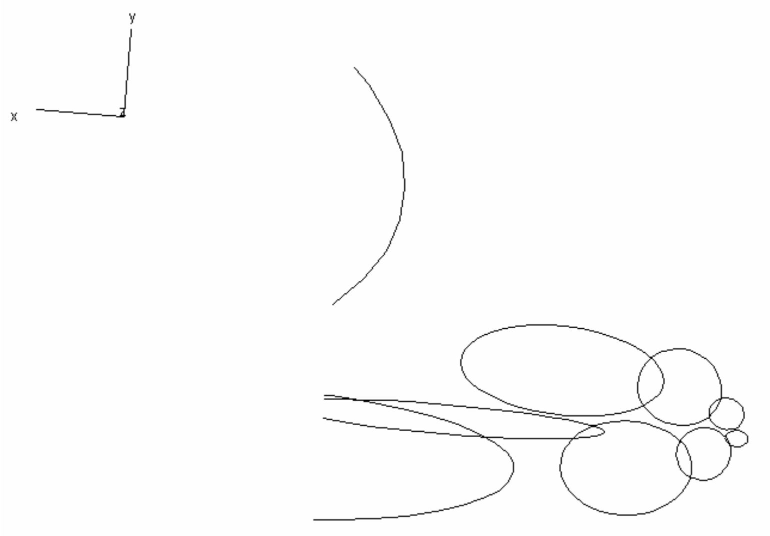
$0 \le t \le 1, 0 \le s \le \pi/2$

Piotr's next answers:

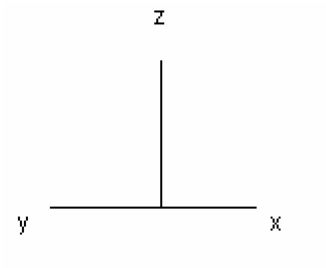
Oh yes, it is possible to produce a longitudinal intersection. Solve the *z*-coordinate for θ or *t*. This solution is say *theta_xy*. Then substitute *theta_xy* for θ or *t* in the *x*- and *y*-coordinate. What remains is the intersection curve of the shell with the *xy*-plane. It is interesting that there are infinite many solutions of form *theta_xy* + *k*/2 (*k* integer). Taking some *k*s will result in a longitudinal intersection.

I followed Piotr's instructions:

```
xy_t := (SOLUTIONS((SchneckenHausX_W_JB(e, 10, t, s))  = 0, t))
                                                     3
                                                              1

                                                                  k
intersect := (SUBST(SchneckenHausX_W_JB(e, 10, t, s), t, xy_t + ———))
                                                                  2
                                                                       [1, 2]

VECTOR(APPEND(intersect, [0]), k, -2, 6)
```



The plot above is a top view (see the axes) and these are the front and side view. It is really the intersection with the *xy*-plane.

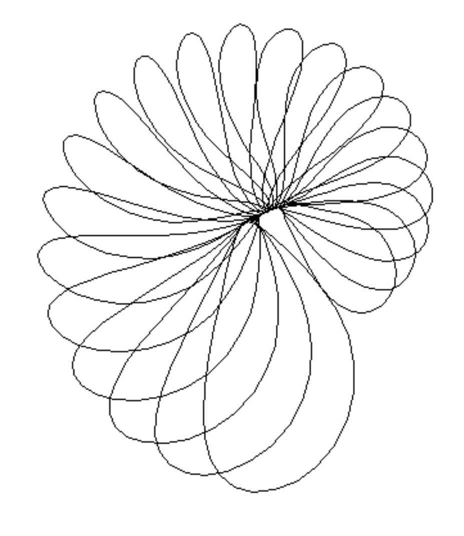


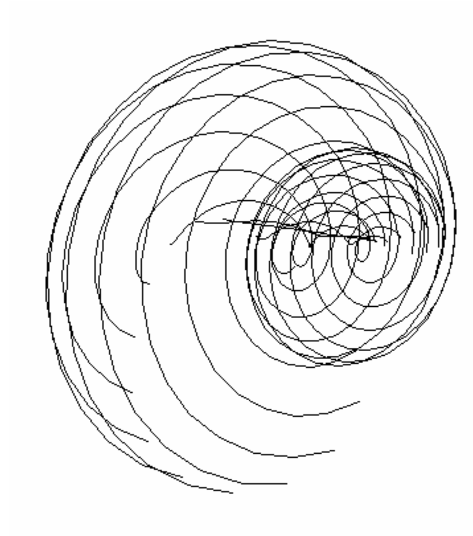
These plots and the next ones as well are produced with …JB-function. I will explain later.

See first some parameter curves of the (b = 2, m = 10) shell.

```
VECTOR(SchneckenHausX_W_JB(2, 10, k, s), k, 1, 2, 0.05)
```

$$\text{VECTOR}\left(\text{SchneckenHausX\_W\_JB}(2,\ 10,\ t,\ s),\ s,\ -\pi,\ \pi,\ \frac{\pi}{8}\right)$$





Piotr:

What concerns function $M(b,n)$ there are two reasons why I did not use `NSOLUTIONS`:

1. I like to catch mathematical problems completely and to solve them by myself. Moreover my function is much faster because I defined the first derivative explicitly in a very compact form. And additionally there is a pretty annotation of the calculation progress (`WRITE`-command).

2. As you found out the initial value 0 for m is a problem, because then thr derivative is 0, too which leads to a division by 0. My opinion is that fixing the lower bound of the solution interval for `NSOLUTIONS` as a small positive number is not a good idea because for small b-values m tends to 0, i.e. for a too big lower bound `NSOLUTIONS` would not find a solution for very small b-values. My short program uses the fact that $f(b, m)$ starts increasing and then switches to decrease. I start with initial value 1 and increase until $f(b, m)$ <0, then I turn and approach the zero from the other side applying the Newton-algorithm. Analysing $f(b, m)$ shows that the zero of $m < 4$, i.e. finding the appropriate initial value does not need long time.

I accept Piotr's explanations, see is my (Josef's) version using NSOLUTIONS.

```
m_(b, n) := APPROX((NSOLUTIONS(f(b, m) = 0, m, 0.1, ∞))  , n)
                                                        1

SchneckenHausX_L_JB(b, n, t, φ) := SchneckenTorusX_L(b, m_(b, n), t, φ)

SchneckenHausX_U_JB(b, n, θ, φ) := SchneckenTorusX_U(b, m_(b, n), θ, φ)

SchneckenHausX_W_JB(b, n, θ, φ) := SchneckenTorusX_W(b, m_(b, n), θ, φ)
```

The advantage of these JB-functions is that one does not face the problems when saving the file as dfw-file and then reloading it – as described on page 10.

Final comment:

Following Piotr's DERIVE code you can see unusual variable names as: R_1, R_2 and m_0.
I wanted to find out how to create these variable names – it is not a subscript produced by R SUB 1 etc.

Please follow my file CODE.dfw:

```
#1:    [R₁ :=, R₂ :=, m₀ :=]

#2:    NAME_TO_CODES(R₁) = [82, 127, 50, 48, 56, 49]

#3:    NAME_TO_CODES(R₂) = [82, 127, 50, 48, 56, 50]
```

I copied the variables in #1 from Piotr's file and tried to find their "codes".

R_1 and R_2 must be written under quotes as argument of NAME_TO_CODES.

```
#4:    CODES_TO_NAME([82, 127, 50, 48, 56, 49]) = R₁
```

R_1 appears as string (i.e. you can see "R_1" when copied into the edit line. 82 is the code of "R". Then the remaining code numbers should be responsible for the subscript "$_1$"?

```
#5:    NAME_TO_CODES(R) = [82]

#6:    CODES_TO_NAME([127, 50, 48, 56, 49]) = ₁

#7:    CODES_TO_NAME([127, 50, 48, 56, 50]) = ₂
```

The codes for "A" to "Z" are 65 to 90; the codes for "a" to "z" are 97 to 122. I try to produce variable names A_1, a_1 and z_{40}.

```
#8:    CODES_TO_NAME([65, 127, 50, 48, 56, 49]) = A₁

#9:    CODES_TO_NAME([65 + 32, 127, 50, 48, 56, 49]) = a₁

#10:   CODES_TO_NAME([122, 127, 50, 48, 56, 52, 127, 50, 48, 56, 48]) = z₄₀
```

Hi, it works. We have to remove the quotes and then we can use the newly created variable names:

```
#11:  [A₁ := 10, a₁ := 20]

#12:  A₁·a₁ = 200

      a  := 30
#13:   1

      A₁·a₁·a  = 200·a
#14:         1        1
```

Please note the difference between a_1 and a_{1} which is the result of a SUB 1.

I can not resist presenting two “non mathematical” snails:

These are two snails which live on trees in order to escape their enemies. I found them in the woods of Madagascar, Josef.

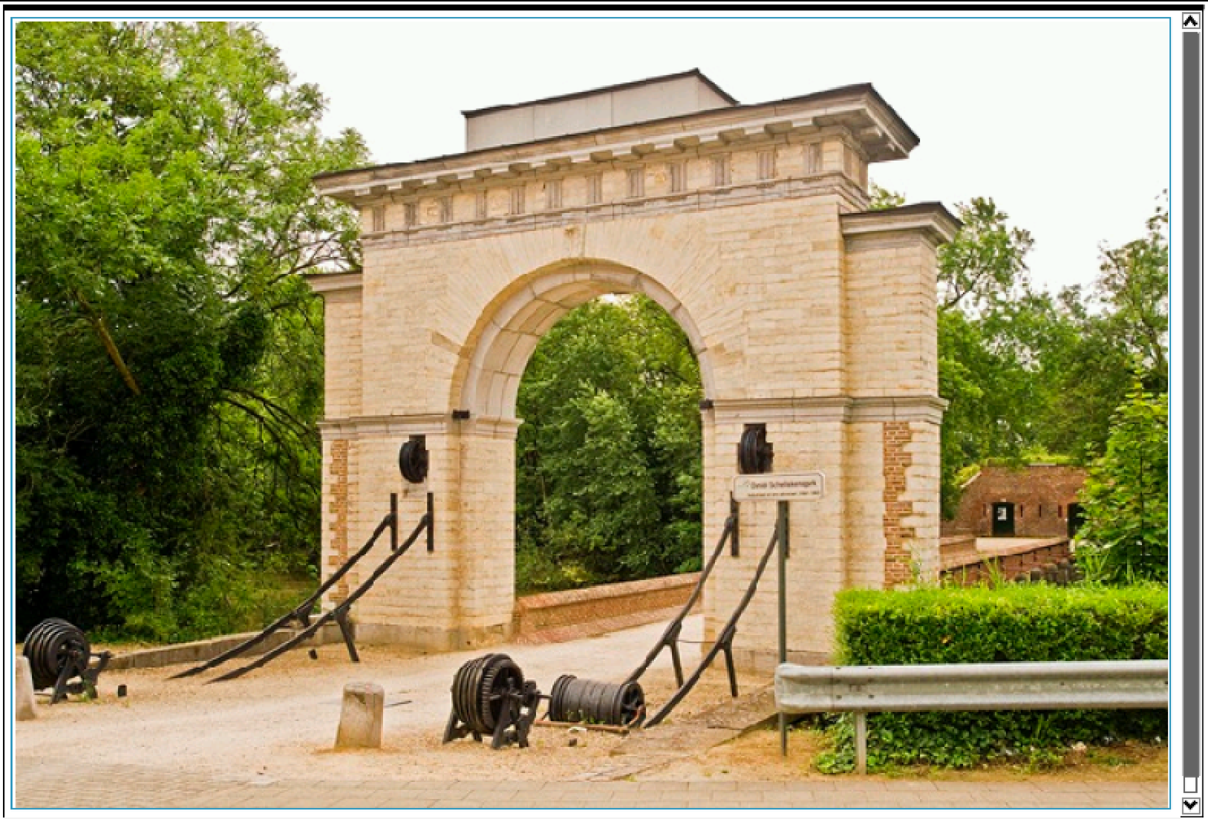
Brussels Gate in Dendermonde, Belgium

Dr.-Ing. E. Lantschoot, Lantschoot_Weisel@web.de

(1) The Program "gate()", meaning the BRUSSELS GATE

Since the end of the Middle Ages, the small Flemish city of Dendermonde has been a military stronghold. Situated at almost equal distance from Ghent, Antwerp and Brussels, it lies at the confluence of two rivers. Its system of defence was as simple as effective: by a controlled flooding of the entire surrounding area, invaders were kept away. Even the French king Louis XIV, in his endeavour of fortifying the north most boundaries of his territory, did not prevail. "Que n´ai-je une armée de canards...", "If I only had an army of ducks..."

A monumental relict of those times is the Brussels gate. It was built in 1822 under Dutch rule, and it featured a drawbridge which, when drawn, isolated the city. Although the drawbridge is now replaced by a fixed section and although the ropes disappeared, the counterweights (in the form of drums), the rails on which they rolled and the winches are still there.



The item of interest to us is the profile of the rails. As it so appears, its shape was dictated by the intention of keeping the bridge and the counterweights in equilibrium. As we will see in the course of our discussion this observation is true, but only to a degree.

From a mathematical point of view it is easier to first consider the theoretical trajectory of **M**, a point representing the center of the axis of the counterweight. This theoretical trajectory is found by applying the Runge-Kutta integration procedure **rk23** (built in in TI-NspireCAS), and we see, - to our disappointment? - that it is not a quadratic curve, as we may have assumed.

Without being aware of Runge-Kutta, the Dutch military engineers deliberately replaced the theoretical trajectory by a theoretical quadratic curve as a surrogate. In doing this, a disequilibrium of forces was created in almost every point of the new trajectory. If this disequilibrium was positive for every point, the bridge would tend to lower itself until the support of the opposed bridgehead was reached. To draw the bridge, a winch would have to be installed and it would have to develop a force at least equal to the disequilibrium force.

The goals and the problems of a **good design** can be summarized as follows:

1) **To compute the theoretical equilibrium trajectory of M**, with the weight of the drum as a parameter.

2) **To compute the surrogate quadratic curve** - we call it the design-curve - with two fixed points and a third one, which influences its shape.

3) **To compute the disequilibrium force**, occasionally rearranging 1) and 2) so that this force is positive, relatively low and relatively constant.

4) **To compute the total work** to be supplied by the winch, and to find out whether the work can be done by one or two humans at each winch.

5) With the data at our disposal **to compute the profile of the wrought iron rails**. We will see that it is also a quadratic, and since a quadratic can be designed by mechanical means, a wooden template could be constructed to serve as an aid in the forgoing process of the rails.

In our programs, we are using row vectors because they fit better in a line of type. If one is more familiar with column vectors, it will help to consult the rules of transposition.

A symbol denoting an initial status ends on **0** (zero). A force either begins or ends with an **f**.

An arm, as used in **crossp(force, arm)** ends on an **a**, e.g. **cga**, center of gravity arm.

Most data in our programs, such as the structural data length, height, weight etc. cannot be modified. Data by which the design can be altered, such as **cga** or **adj**, the adjusting factor of the weight of the drum, can be modified by the operator, but only in section gate1.

We are using three **important rules of statics**:

1) A force in a rope assumes the direction of the rope. If the direction of the rope is changed, e.g. by a deflection sheave, the direction of the force also changes, but its scalar value remains.

2) A point on a curve is at rest, is in equilibrium, if the resultant of all forces acting on that point is perpendicular to the tangent of the curve at that point. To say it otherwise, the point is at rest if the resultant takes the direction of the normal. If the resultant has a direction deviating from the normal, then the projection of the resultant on the tangent is the force which will move the point.

3) A rope is not a solid element, such as the girders of a bridge. It is treated as such as long as it is under stress, but when the rope is relaxed, it transmits no forces anymore. That is the reason why we required that the disequilibrium force should be positive (= traction, stress) for all states of the drum.

SECTIONS GATE1 AND GATE2

The purpose of section gate1 is to present, on the console, all actual data and to enable the operator, if he wishes so, to modify, either the arm of the center of gravity **cga**, or to adjust by **adj** the assumed weight of the drum. The assumed weight of the drum **wd0** is the weight which keeps the flat bridge in equilibrium. Why it can or should be adjusted is explained immediately. If the choices are made, the program proceeds automatically to section 2.

First tool: the method of Runge-Kutta

The initial position of the drum is precisely known: the distance between its center **M** and the deflection sheave **U** (= Umlenkrolle) (here reduced to a point) equals **frr** (free rope) and it hangs on the wall of the gate (distance **M** to wall: **rd**, radius of drum). The final position to the contrary is only defined by the fact that **MU** now equals **lb∗√2 + frr** (rope at the bridge side now completely drawn to the drum side plus the length of the free rope), but it is not known at what final height above the street level. We expected a height of **rd** above null, but that is only wishful thinking; the actual height computed by Runge-Kutta for each of the 20 test points (including the final point **M(20)**) is dictated by the equilibrium of forces. By adjusting (with adj) the weight of the drum, we can, in the next run, either lower or lift the profile of Runge-Kutta. The range of **adj** is approximately between 0.95 and 1.3.

Second tool: the design curve

If instead of implementing the Runge-Kutta curve we decide to replace it by a quadratic curve of our choice (thereby taking the risk of introducing disequilibria of forces), we can specify and we will specify the coordinates of **M(1)** and **M(20)**.

However, it takes three points to design a quadratic curve; we thus add a third point **P3 p** (m) to the right of **M(20)** and **q** (m) above it. In the Graphs-application **p** and **q** can be defined by shifting the sliders, and the shape of the design curve immediately adapts itself to the choice which was made. Caution however! The purpose of the design is not to closely adapt the shape of the design curve to that of Runge-Kutta, but rather to obtain a positive disequilibrium force (see below).

Program **gate1()**. (I don't reprint the program code. What follows is the program output of program gate1,Josef)

```
gate1()
Locked data:
Length of the bridge lb = 3.4
Weight of the half bridge wb = 12000.
Length of free rope frr = 0.9
Radius of drum rd = 0.25

Data for beginner are:
center of gravity arm cga = [-1.7  -0.15]
Adjusting factor adj = 1.

Actual data are:
center of gravity arm cga = [-1.9  -0.4]
Adjusting factor adj = 1.
new cga? (enter & OK/simply OK) [[-1.9,-0.4]]
new adj? (enter & OK/simply OK) 1.
Unadjusted weight of drum wd0 = 9483.55
sol =
[ 0.25     0.487395  0.724791  0.962186  1.19958   1.43698   1.67437   1.91177   2.14916   2.38656   2.62395
 2.53542  1.81528   1.45212   1.17377   0.942654  0.744341  0.571291  0.418849  0.283963  0.164448  0.058665  ·
new design-curve? (y,n) n
                                                                                                        1/1
```

Program **gate2()**

```
Define gate2()=
Prgm
© Computes the true trajectory of M
© Designs a quadratic curve, surrogate for the true trajectory
Local ant,codes,fir,firv,rv,tv
2→lastpr
Lock cga,adj
Unlock wd0: wd0:=force(wb,cga,0): Lock wd0
Disp "Unadjusted weight of drum wd0 = ",wd0
Unlock xb,xe,yb,ye
xb:=rd © Initial location of M, center of the drum axis
xe:=√((√2 · lb+frr)^2-(lb-rd)^2)  © Final location of M
yb:=lb-√(frr^2-rd^2)  © Initial location of M
ye:=rd © Final location of M
Lock xb,xe,yb,ye
© Computes, with Runge-Kutta, the true trajectory of M
DelVar x,y
fir:=force(wb,cga,norm([0  lb]-[x  y])-frr) © Force in rope
© fir as a vector firv, on the drum side (direction has changed)
firv:=fir· unitV([0  lb]-[x  y])
rv:=firv+adj· [0  -wd0] © Resulting vector
tv:=rv· [[0,1][-1,0]] © Turning rv 90° to the left, has direction of the tangent line
yac:=tv[1,2]/tv[1,1] © Inclination of the tangent line
sol:=rk23(yac,x,y,{xb,xe},yb,(xe-rd)/19)
Disp "sol = ",sol
xlst:=mat▶list(sol[1])
ylst:=mat▶list(sol[2])
RequestStr "new design-curve? (y,n)",ant
If ant="y" Then:
© Design-curve
DelVar p,q
codes:=mat▶list(zeros({a· xb^2+b· xb+c-yb,a· xe^2+b· xe+c-ye,a· (xe+p)^2+b· (xe+p)+c-(ye+q)},{a,b,c}))
des:=polyEval(codes,x)
EndIf
EndPrgm
```

Program **gate2** needs function **force** (see Appendix 1).

SECTION GATE3

This section collects all the choices made in Section 2, and computes all items which are needed for visualizing the design in its present state. Section 3 has computed the vectors **firv[n]**, force in the rope. These vectors have the direction from **M** to **U**, and that is considered to be a positive direction, since it induces stress in the rope. If **M(n)** is not in equilibrium - and in most instances it is not - we must ask ourselves which mechanisms will cause **M(n)** to move. According to rule 2 of the statics, the moving force on **M(n)** is the projection of the resultant on the tangent. In a graphic construction, we depart from the tip of the resultant, take the direction of the normal, cross the tangent, proceed and cross the line **MU**, thus defining a point **V**. The section **MV** defines direction and strength of a vector, which will move **M(n)** when the system is left to its destiny. This vector is the disequilibrium force; if it has the positive direction (from **M** to **V**) it will lift the drum and lower the bridge. Therefore, section 3 will warn the operator when the list of these vectors features a non-positive entry. Then, the design procedure of section 2 must be carried out again, but the following program permits us to visualize our mistakes first.

(I don't present program gate3(). I show the program's output.)

```
gate3()
List of disequil. forces, diseqflst =
{1950.82,2243.91,2234.31,2114.5,1948.46,1763.82,1574.86,1389.98,1214.63,1052.6,906.647,778.854,670.7
All the data needed for controlling the solution are now
in storage. Please proceed with copt(n) and n of your choice
                                                                                  Fertig
```

THE PROGRAM COPT

If one of the 20 test points is selected, **copt(n)** shows the diamond of forces in that point: force in the rope **firv[n]**, weight of the drum **adj∗wd0**, resultant **r** and the disequilibrium force **diseqflst[n]** as a part of force in the rope. In order to complete the picture, one can add the functions **tali[n]** (tangent in **M(n)**) or **noli[n]** (normal line) or even **noli** for the first 16 normal lines. The iron profile (computed in **gate4**) can be added later by entering the sets **xilst** and **yilst**.

```
copt(11)                                   Fertig
diaf                          [2.49378   0.981556]
                              [2.62395   0.855318]
                              [2.62395   -1.04139]
                              [1.86872  -0.308971]
                              [2.62395   0.855318]
                              [1.86872    1.58774]
                              [1.86872  -0.308971]
xdiaflst
 {2.49378,2.62395,2.62395,1.86872,2.62395,1.86872,1.868
```

```
copt                                                        0/2
Define copt(n)=
Prgm
© This program, for a selected point M(n), draws the diamond
© of forces diaf
If lastpr=1 or lastpr=2 Then:  Disp "call gate3 first!": Stop:  EndIf
Local ant,n,weitr,zw1,zw2
 0→zw1:  0→zw2
diaf:=newMat(7,2)
© The point V
diaf[1]:=(unitV(firv[n])·diseqflst[n])/5000+m[n]
© The point M
diaf[2]:=m[n]
© The wd (weight of drum) vector
diaf[3]:=(adj·wd0·[0  -1])/5000.+m[n]
© The force in rope firv vector
diaf[6]:=firv[n]/5000.+m[n]
© Resulting vector
diaf[4]:=diaf[3]+diaf[6]-m[n]
© The point M again
diaf[5]:=diaf[2]
© Closure
diaf[7]:=diaf[4]
xdiaflst:=mat▸list(diafᵀ[1])
ydiaflst:=mat▸list(diafᵀ[2])
EndPrgm
```

SECTION GATE4

This section provides:

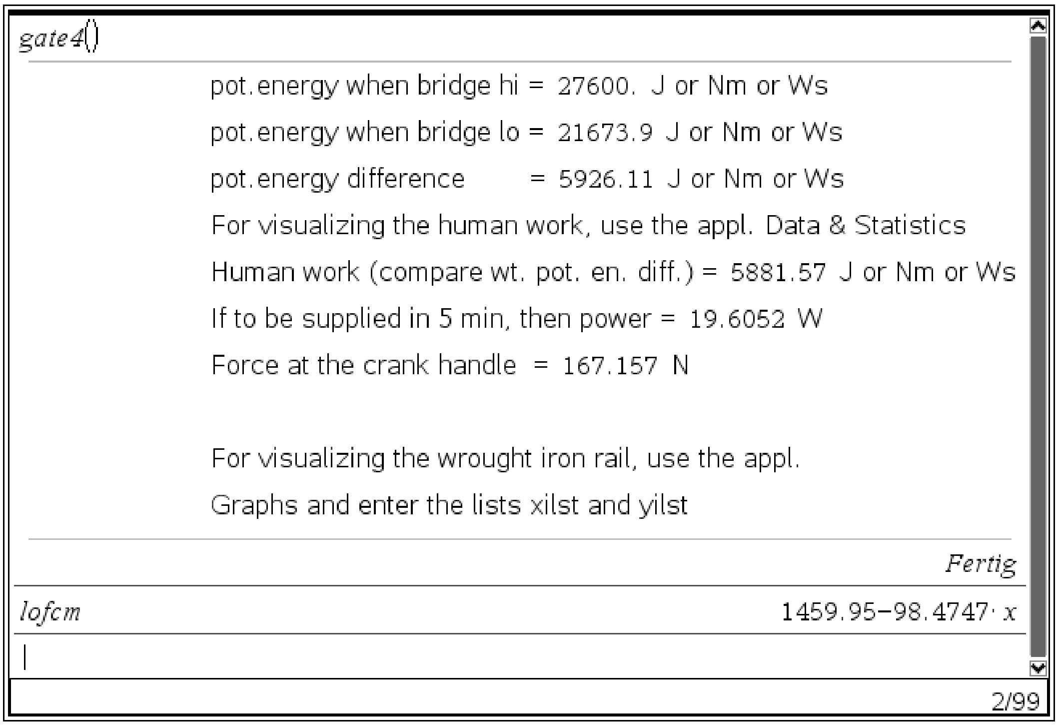
1) **general data**, such as potential energy when bridge high or low, and their difference;

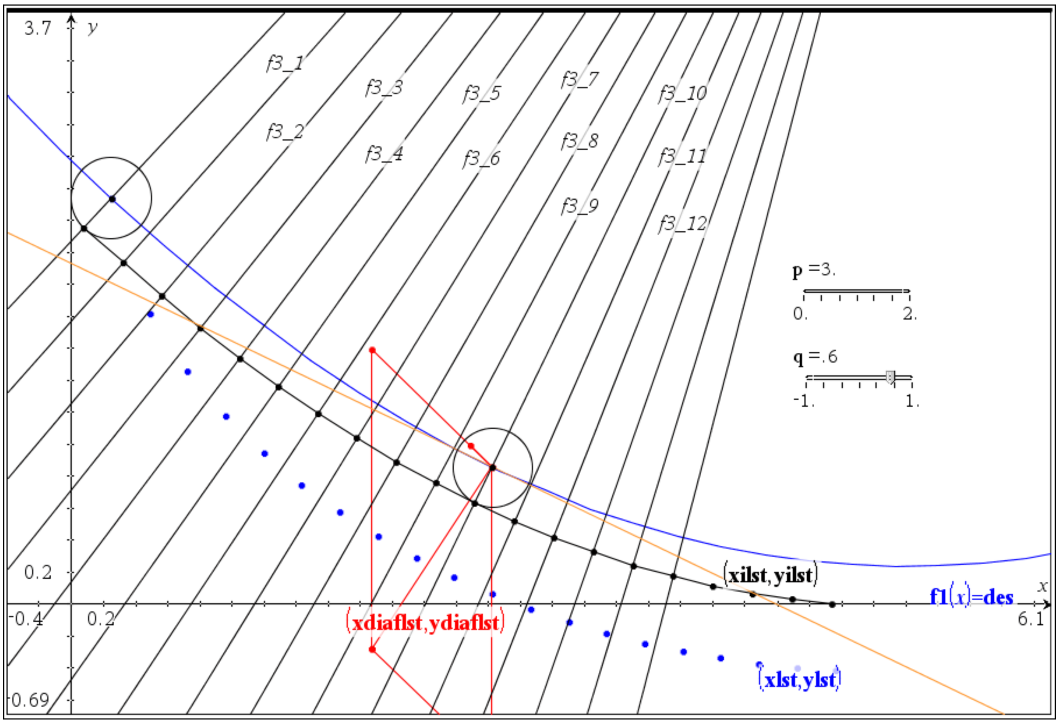
2) **human data**

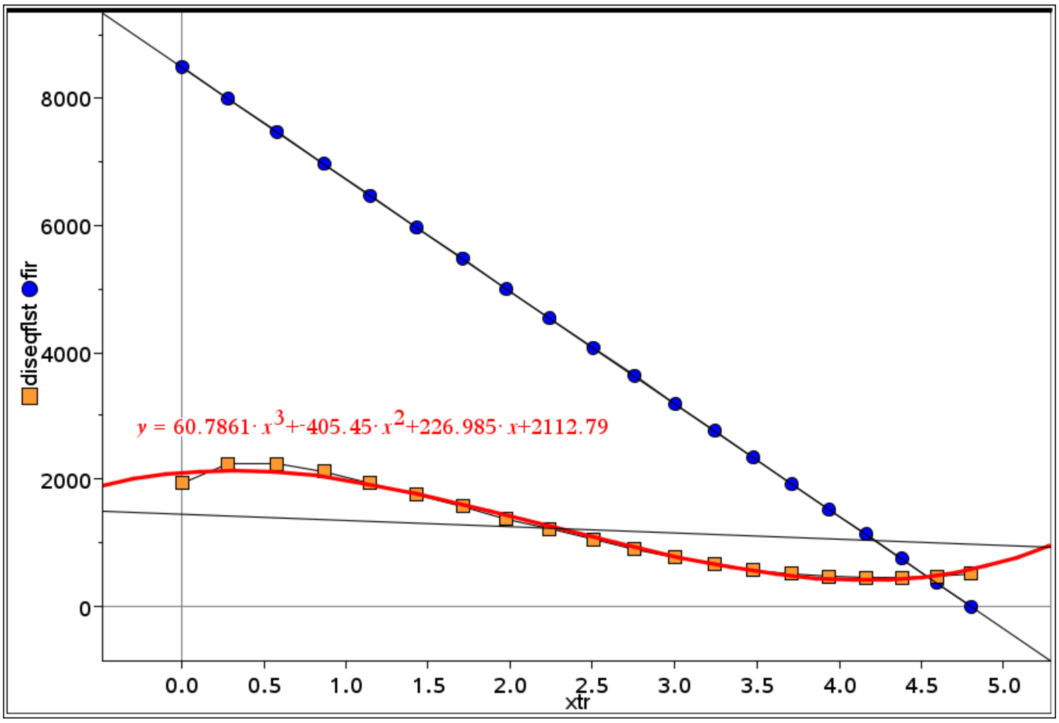
   the above difference must be supplied by human work, and the human work is computed as the integral of diseqflst over the lenght of the rope drawn. Since diseqflst mostly is a list of steadily decreasing forces, it makes sense to supply these forces by a winch with a tapered drum (Spiral of Archimedes), on which an almost constant moment cm can be applied. Finally, it is checked, what (human) force on the crank handle is needed.

3) the profile of the wrought iron rail. Since gate3 already provided the equations of all normal lines, this information can be used to compute a set of points **I(n)** which are **rd** below **M(n)**; If a cubic regression function for the **I(n)** points is computed, it so turns out that the coefficient of x^3 is negligibly small. Consequently, the profile of the iron rail also is a quadratic curve.

Output of program gate4() and respective plots:







(2) DERIVATION of the Function "Force"

The statement of the problem is simple. A bridge with a hub at (0,0) and an end at (-lb,0) should be hauled up by means of a rope which runs over a deflection sheave at (0,lb). The weight of the bridge is **wt** N, and the arm of the gravity center is a vector **[r,s]** (both elements are negative). The problem is to express the force in the rope as a function of the length of rope drawn. It's a horrendous exercise in trigonometric fitness!

Note - Because in this document "force" is treated as a standalone, we converted it to a program and commanded "disp" instead of "return".

The whole computation relies on equating the moment of the weight and the moment of the force in the rope. When moments are considered, one is allowed to lenghten or to shorten an arm, provided the force acting on that arm is reduced or increased correspondingly. Here we lengthen the arm of the gravity center until its first component equals lb. The moment of the weight now is:

mw = (w∗r/lb)*norm([r∗lb/r s∗lb/r])*sin(...)

... being the angle between the weight vector and its arm.

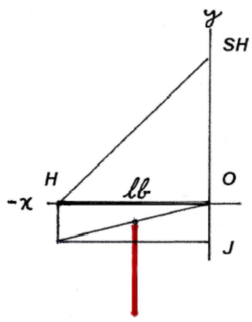


Figure 1

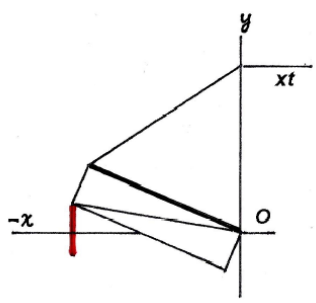


Figure 2

Figure 3 shows an equivalent situation: the arm of the gravity center has been decomposed in two vectors **OH** and **OJ** and the weight vector now acts on the points H and J of the bridge. The first moment is:

$$m1 = (wt*r/lb)*lb*\sin(90°-\theta)$$

After simplification:

$$m1 = wt*r*\cos(\theta)$$

and the length of the other arm, OJ, equals **s∗lb/r**.

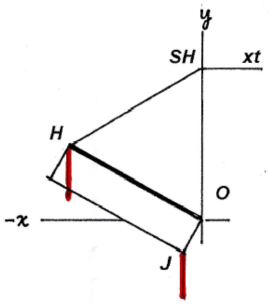


Figure 3

We now pay attention to the vector **OJ** and lengthen it until its length equals **lb**. The moment then becomes:

$$m2 = (wt*r/lb)*(s/r)*(s*lb/r)*(r/s)*\sin(180°-\theta)$$

After simplification:

$$m2 = wt*s*\sin(\theta).$$

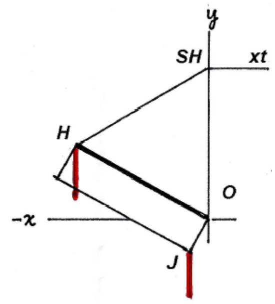


Figure 4

We now consider the scalar variable **fir**, force in the rope. We subdivide the segment H-SH in two equal parts H-M and M-SH and denote the angle H-O-M by **ρ**.

$$mfir = fir*lb*\sin(90°-\rho) = fir*lb*\cos(\rho)$$

and the general equation of the system obviously is

$$mfir+m1+m2 = 0$$

Our attention now goes to the line segment H-SH. If the amount (length) of rope withdrawn is denoted by **xt** , the length of H-SH is

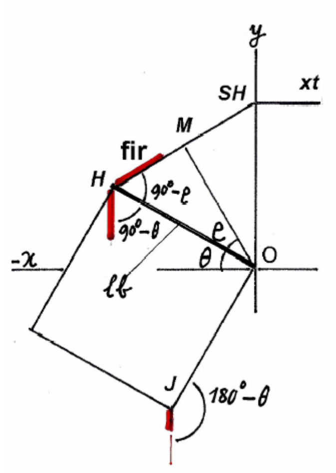


Figure 5

$\sqrt{2}*lb-xt$ and for the rectangular triangle H-O-M we find that the quantity **w**

$$w = (\sqrt{2}*lb-xt)/(2*lb) = \sin((90°-\theta)/2) = \sin(\rho).$$

We now must go back to the trigonometric functions in **m1**, **m2** and **mfir**.

In m1, we have $\sin(90°-\theta) = \sin((90°-\theta)/2) = \sin(2*\rho) = 2*\sin(\rho)*\cos(\rho) = 2*w*\sqrt{(1-w^2)}$.

In m2, we have $\sin(\theta) = \cos(2*\rho) = 1-2*(\sin(\rho)^2) = 1-2*w^2$.

In mfir we have $\cos(\rho) = \sqrt{(1-w^2)}$.

The rest is a little bit of calculus.

The advantage of this derivation is to reveal a detail which goes unnoticed when using crossp´s: if the center of gravity is on the surface of the bridge, <u>the force fir decreases linearly with xt</u>.

(3) The program mechtor()

The gate to Mechelen (background picture on page 3 of the document) is an exact replica of the gate to Brussels. The approach we take her is that of an ordinary but mathematically interested tourist who discovers this contraption, takes a few pictures, notes a few dimensions, and then **tries to find the mathematical equation** of the iron rail by means of the here presented program **mechtor()**.

When looking at page 3, we see a background picture of the Mechelen Gate and a function graph as an overlay; the graph is as left by the last user of the program.

Therefore, we must first explain how the graph was originally designed and what it means. At the very first use of the program, a list of *x*-values **xlst:=seq(n-1,n,1,20)**, the list of numbers 0 to 19, was created. This list was locked. Then a second list **ylst := newlist(20)**, not locked, was defined. Using the Data & Statistics Application the 20 points **(xlst[n], ylst[n])** were displayed as a horizontal row of red points. A movable line **m2(x):=a + b∗x** was added; this line the **footprint** of the rail is supposed to be the vertical projection of the iron rail on the ground.

These raw data are now to be moved to their proper position in the picture (see the respective instructions in the TI-Nspire_SS_Guide, English, pp. 334 and German pp. 374). First of all the origin must be moved until the *y*-axis passes the highest point of the rail. Then the *x*-axis must be stretched or condensed **until xlst[20]** lies exactly under the point where the rail meets the street.

The red points **E1 to E20** with **En = (n–1,0)** are now moved to their positions on the rail by grasping them one after the other with the cursor, and moving them up or down (moving right or left is excluded because **xlst** has been locked).

Note: a point which is grasped loses its color. When the point is on its proper place then click on it again, and it recovers its color. The reason is, that the movement of the cursor is applied to all non colored points. Finally the **footprint** is grasped and dragged to its proper position. Then **mechtor()** was continued and the 4th degree regression curve **fotreg** of the points **E1-E20** was calculated. The white curve displayed on page 3 is **fotreg** (if its equation is not shown then simply click on the curve). This curve, however **is not the solution of our problem, it is only the key for it. fotreg** indeed is only the regression curve approximation of the **photographed** iron rail, and must be corrected for effects of perspective and scaling.

What can we do now, which has not been done already by the last user?

On page 3 – and without opening the program – we can move one or the other point on the rail, thereby observing how **fotreg** changes. We can also move the **footprint**. We call these adjustments “corrections”.

If we would like to know the equation of the rail in metric units, with the origin of measurement **(0,0)** at the intersection of the footprint with the wall then we have to run the program by entering **mechtor()** on the calculator page.

Immediately appears the question “new data or corrections? (nd/c)”. If the answer is “**c**”, then the computation is performed with the most recent data given on page 3; if the answer is “**nd**”, then points **E1 to E20** and the **footprint** as well are set back to some initial format and the whole process bringing everything to the proper place must be done again.

Now we have to explain how the program converts **fotreg** to the final answer **rail_c** which is the equation of the rail in reality i.e. in metric units. The conversion needs four steps:

a) We need to know the real measures. By inspection on the site we find **E1=(0,2.25)** and **E20=(4.82,0)**.

b) The *y*-values of **E1-E20** should not refer to the *x*-axis of the diagram but to **footprint**. The result of this correction is **rail_a**.

c) **rail_a** will now be converted to **rail_b** in such a way that **rail_b=0** for **xlst[20]=4.82**. For reaching this goal the list of coefficients **cof** of **rail_a** are multiplied by list **caf**.

d) The ordinates of **rail_b** will now be multiplied by such a factor that **ylst[1]=2.25**.

The final result – of all this squeezing and stretching) is **rail_c** and it is left to the reader whether the coefficients of x^4 and x^3 are small enough to be neglected, thus supporting the statement that the equation of the rail is a parabola.

mechtor()

new data, or corrections only? (nd,c) c

fotreg = $0.000002\cdot x^4-0.000078\cdot x^3+0.002697\cdot x^2-0.084425\cdot x+0.926316$

rail_a = $0.000002\cdot x^4-0.000078\cdot x^3+0.002697\cdot x^2-0.077395\cdot x+0.772516$

rail_b = $0.000489\cdot x^4-0.004816\cdot x^3+0.042253\cdot x^2-0.306355\cdot x+0.772516$

rail_c = $0.001423\cdot x^4-0.014028\cdot x^3+0.123066\cdot x^2-0.892277\cdot x+2.25$

Fertig

rail|x=0	2.25
rail|x=4.82	0.005709

Comments of the editor:

I do not reprint the program in order to save space. You can find it among the downloadable files together with the picture of the Gate. Thanks to Erik van Lantschoot for providing this.

As the DNL-readers will already know I am an enthusiastic user of background pictures for several reasons. I like this example very much because it is much more than only modelling the picture of the rail; it does an important step further: converting the picture to reality. I studied Geometry and I remember "Photogrammetry" where we had to reconstruct buildings from photographs considering perspective distortions. This gate is a great example what can be done with students (neglecting the perspective because being so close to the object).

I had some very fruitful email exchanges with Erik van Lantschoot and in the last one I asked him why not immediately performing a quadratic regression? (You are invited to do this and then repeat the conversion to reality.) I liked his answer: "I assume that a tourist who has such a powerful tool like TI-Nspire at his disposal will choose the "best" available regression."

My experience as teacher is that most of the students think: "All curves are parabolas."

I found a pretty picture of Dendermonde from earlier times. Both gates mentioned in the contribution can be found there. I sent this picture to EvL. He wrote back:

I lived in the centre of the town where the bridge crosses the Dender River until 1945. This was the year of my end examination at the gymnasium, then I had to leave for starting my studies at university.

41. Porta Mechliniensis, 42. Porta Bruxellensis

You may now find out EvL's age of life – and he is doing mathematics with modern technologies. This is really great, congratulations.

Structures and Rules of the Prime Numbers

(Observations and Conclusions)

Dietmar Oertel, Germany

Preliminary Remarks

This paper deals with the phenomenon of the prime numbers.

My observations with respect to the prime numbers are

- structures in the set of primes can be observed if they are not considered all together in a formula but one after the other
- a regularity in form of a *delta-sequence* with a certain *sequence width* appears which depends on the number of the considered prime numbers
- this *sequence width* related to the given prime numbers and the number of the remaining indivisible numbers can easily be found
- the density of the prime numbers is decreasing
- referring to the number of primes considered equations can be "constructed" which – based on combinatorics – can determine the n+1th prime number if n primes are known
- understanding the delta sequence and its width it becomes clear that there is no regularity in the sequence of the prime numbers when viewing all primes at a whole
- my results match the *Euler Equation* (I learned this in a TV broadcast from 5 September 2011 'hitec' where this equation and the Zeta function were presented).

  http://www.3sat.de/page/?source=/hitec/156496/index.html

My summary of these observations:

It is not difficult to recognize the structure of the prime numbers considering the following aspects:

* prime numbers shall one after the other step by step be considered in an expandable formula
* the principle of combinatorics based on Pascal's Triangle is applied
* only the integer parts (FLOOR-function) of possible fractions appearing in the formulae must be used

My final result is the following formula (MathCad and DERIVE):

$$\mathrm{SPrmFunPN}(v_p, n_{Max}) := \sum_{n=0}^{n_{Max}} \left[\mathrm{floor}(n) + \sum_{k=1}^{\mathrm{länge}(v_p)} \left[(-1)^k \cdot \sum_{j=0}^{\frac{\mathrm{länge}(v_p)!}{k! \cdot (\mathrm{länge}(v_p)-k)!}-1} \mathrm{floor}\left[\frac{n}{\left(\mathrm{K\,Prm\Pi}(v_p, k)\right)_j} \right] \right] \right]$$

$$\mathrm{SPrmFunPN(vPrm,\ nMax)} := \sum_{n=0}^{nMax} \left(\mathrm{FLOOR}(n) + \sum_{k=1}^{\mathrm{DIM(vPrm)}} (-1)^k \cdot \sum_{j=1}^{\mathrm{DIM(vPrm)!/(k! \cdot (DIM(vPrm) - k)!)}} \mathrm{FLOOR}\left(\frac{n}{(\mathrm{KmbNumPrdRev(vPrm,\ k)})_j} \right) \right)$$

(The principle of combinatorics is represented in KPrmΠ, which will be explained later.)

We will see that the next prime number is found when SPrmFunPN(v_{Prm},k) = k + 1 or, defined as a function:

$$\mathrm{PrnNul}(v_p,k) = \mathrm{SPrmFunPN}(v_p,k) - 1 - k.$$

Example:

$v_{\mathrm{Prm}_0} := 2,\ v_{\mathrm{Prm}_1} := 3,\ v_{\mathrm{Prm}_2} := 5;\ \ v_{\mathrm{Prm}}{}^T = (2\ \ 3\ \ 5)$ the next prime number follows if the following conditions are fulfilled:

PrnNul(v_{Prim}, k) = 0 and $k > 0$ leads to $k = 7$ and $v_{\mathrm{Prm}_3} := 7$.

I used two Mathematics programs for my investigations, MathCad 14 and DERIVE 6.10.

DERIVE 6.10:	Advantages;	very fast and adjustable accuracy comfortable building functions
	Disadvantage:	DERIVE has been taken off the market
MathCad 14:	Advantages:	comfortable presentation of functions
	Disadvantage:	restricted accuracy

Regularities and Delta Sequence

As I noted above it is not possible to recognize any regularity in the distances between the prime numbers if their sequence is investigated as a whole. But it is possible performing a step by step approach. Some graphics shall demonstrate this in the following.

Take the natural numbers (without 0) then the numbers indivisible by 2 are the odd numbers and their differences form the “Delta Sequence” {2,2,2,2,2, …}.

Remove the numbers divisible by 2 and 3 (base set {2,3}) from the natural numbers then the remaining numbers are {1,5,7,11,13,17,19,23,25, …} which results in the delta sequence {4,2,4,2,4,2,…}.

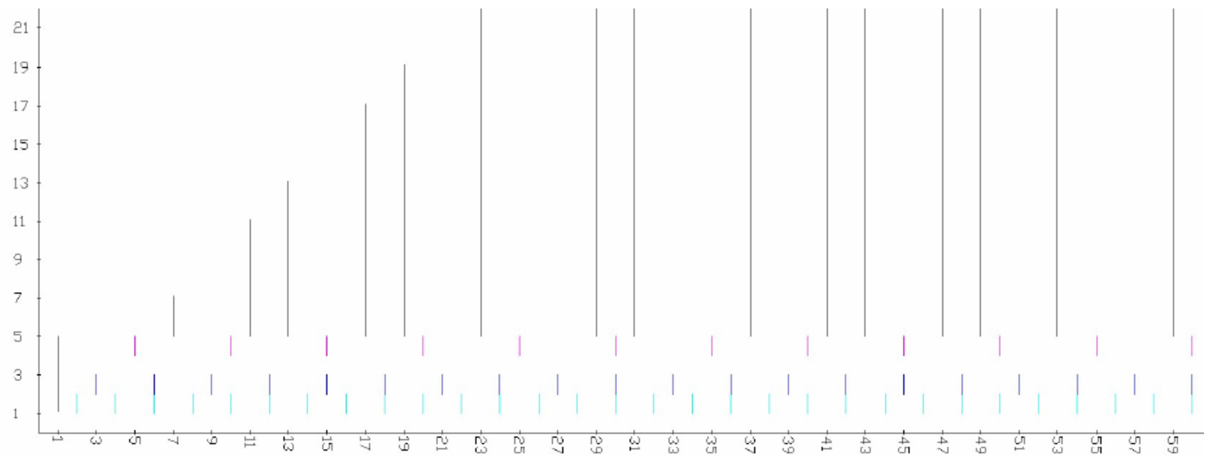
Base set {2, 3, 5} gives a delta sequence {6,4,2,4,2,4,6,2,6,4,2,4,2,4,6,2,6,4,2,4,2,4,6,2, …}

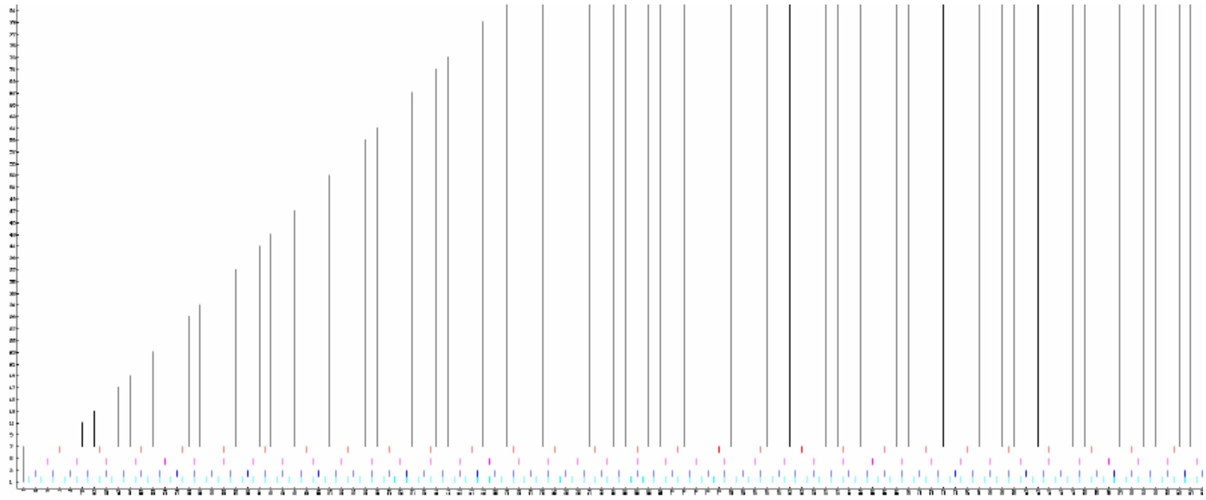
You will recognize the periodicity the sequences.

Will we find this periodicity for bases sets {2,3,5,7} and {2,3,5,7,11} and {p_1,p_2,p_3, …,p_k}, too, with p_i the i-th prime number?

Can we predict the period length of the delta sequence?

See two graphic representations for base sets {2,3,5} and {2,3,5,7}:





Width, Number and Sum

Note (Josef): I wanted to illustrate the delta sequences using DERIVE and defined four functions:

```
        nd(n, k, p, st, i) :=
          Prog
            st := VECTOR(k_, k_, 1, n)
            i := 1
            p := 2
#1:         Loop
              If i > k
                 RETURN st
              st := SELECT(MOD(v, p) ≠ 0, v, st)
              p := NEXT_PRIME(p)
              i :+ 1

        n_d(n, k, p, st, i) :=
          Prog
            st := VECTOR(k_, k_, 1, n)
            i := 1
            p := 2
#2:         Loop
              If i > k
                 RETURN VECTOR([v, 1], v, st)
              st := SELECT(MOD(v, p) ≠ 0, v, st)
              p := NEXT_PRIME(p)
              i :+ 1
```

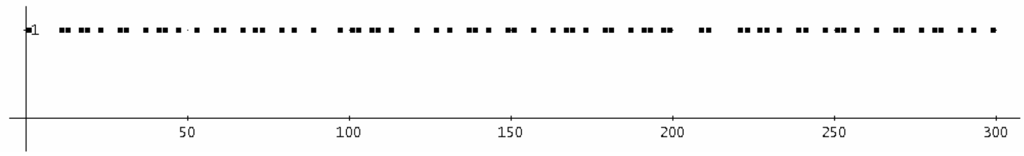
nd(n,k) returns the sequence of all integers $0 < x \le n$ which are indivisible by the first k prime numbers.

n_d(n,k) marks all these integers from **nd(n,k)** as a point $(x,1)$

Examples:

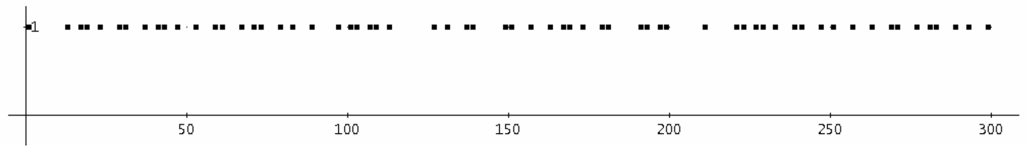
```
#17:  nd(300, 4)

#18:  [1, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73,
        79, 83, 89, 97, 101, 103, 107, 109, 113, 121, 127, 131, 137, 139, 143,
        149, 151, 157, 163, 167, 169, 173, 179, 181, 187, 191, 193, 197, 199,
        209, 211, 221, 223, 227, 229, 233, 239, 241, 247, 251, 253, 257, 263,
        269, 271, 277, 281, 283, 289, 293, 299]
```

#21: n_d(300, 4)



(sequence of all integers ≤ 300 which are indivisible by 2, 3, 5, and 7)

#23: n_d(300, 5)



The periods are not really clear to recognize. Two more functions shall help:

$$\#3:\quad \text{d\_seq}(n, k) := \text{VECTOR}((\text{nd}(n, k))_{i+1} - (\text{nd}(n, k))_{i}, i, 1, \text{DIM}(\text{nd}(n, k)) - 1)$$

$$\#4:\quad \Delta\text{\_seq}(n, k) := \text{VECTOR}\left(\left[i, (\text{n\_d}(n, k))_{i+1,1} - (\text{n\_d}(n, k))_{i,1}\right], i, 1, \text{DIM}(\text{n\_d}(n, k)) - 1\right)$$

d_seq(n,k) gives the delta sequence of **nd(k)**:

d_seq(50, 2)

[4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2]

d_seq(100, 3)

[6, 4, 2, 4, 2, 4, 6, 2, 6, 4, 2, 4, 2, 4, 6, 2, 6, 4, 2, 4, 2, 4, 6, 2, 6]

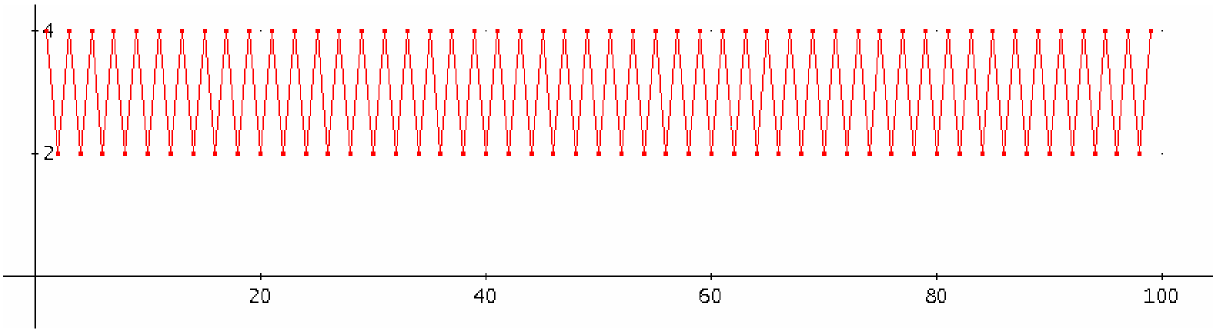
d_seq(300, 4)

[10, 2, 4, 2, 4, 6, 2, 6, 4, 2, 4, 6, 6, 2, 6, 4, 2, 6, 4, 6, 8, 4, 2, 4, 2, 4, 8, 6, 4, 6, 2, 4, 6, 2, 6, 6, 4, 2, 4, 6, 2, 6, 4, 2, 4, 2, 10, 2, 10, 2, 4, 2, 4, 6, 2, 6, 4, 2, 4, 6, 6, 2, 6, 4, 2, 6, 4, 6]

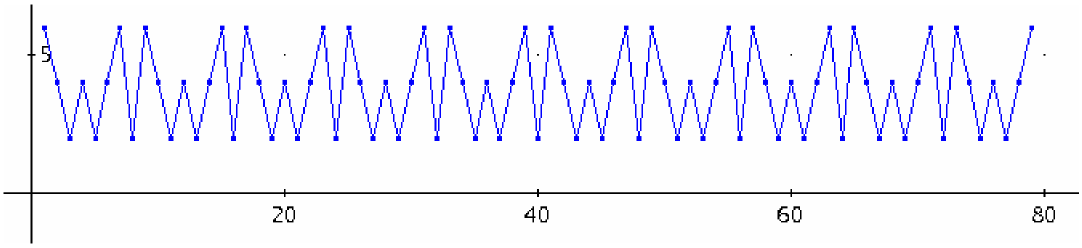
This is a little bit better but still not very illustrative. **Δ_seq(n,k)** plots the points of the sequence one after the other, like (1,10), (2,2), (3,4), (4,2), … for the 4-delta sequence from above. Let's inspect the graphs for k = 2, 3, 4, and 5:

As I noted above it is not possible to recognize any regularity in the distances between the prime numbers if their sequence is investigated as a whole. But it is possible performing a step by step approach. Some graphics shall demonstrate this in the following.

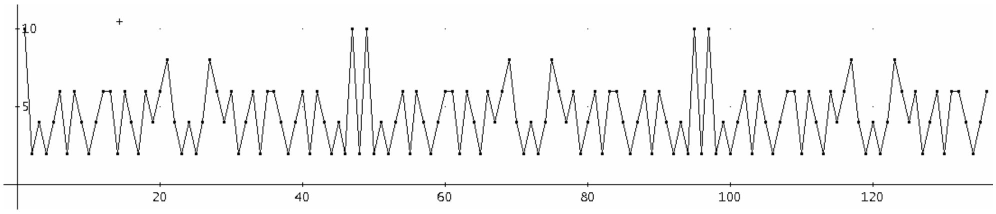
Δ_seq(300, 2)



Δ_seq(300, 3)

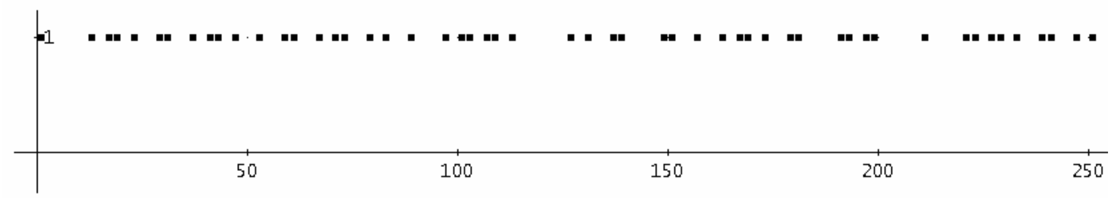


Δ_seq(600, 4)

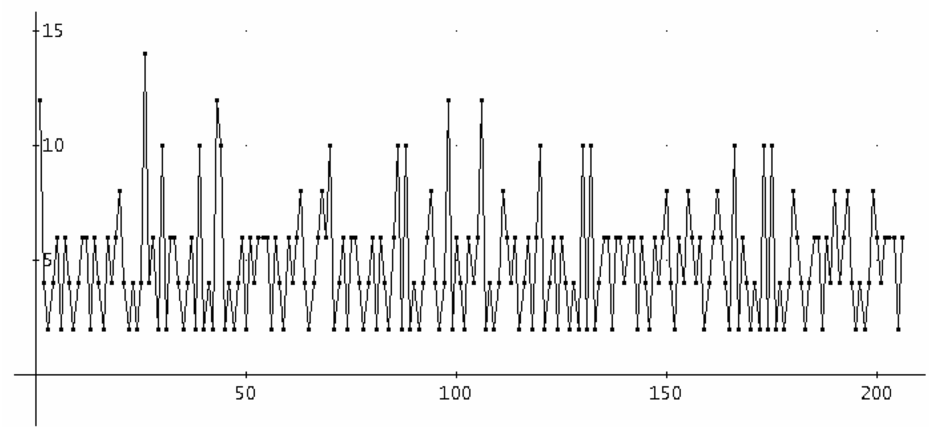


Do you really **see** the periods? Compare with the **nd**-lists and the **n_d**-graphs from above! We will proceed for k = 5:

#11: n_d(300, 5)

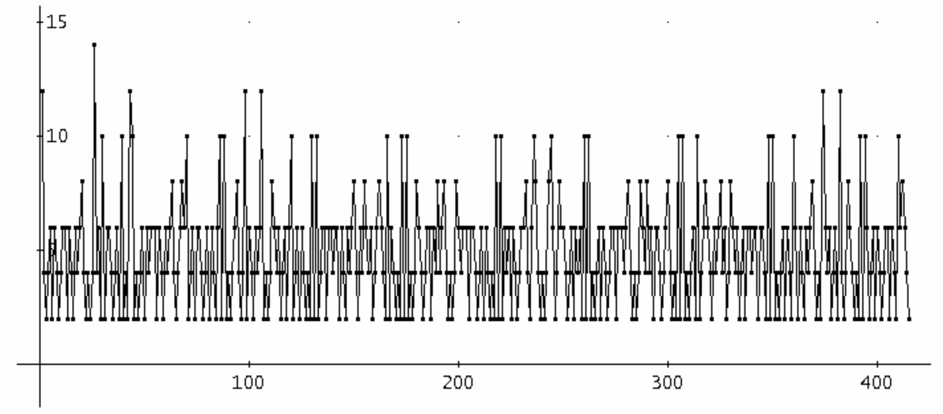


#12: Δ_seq(1000, 5)



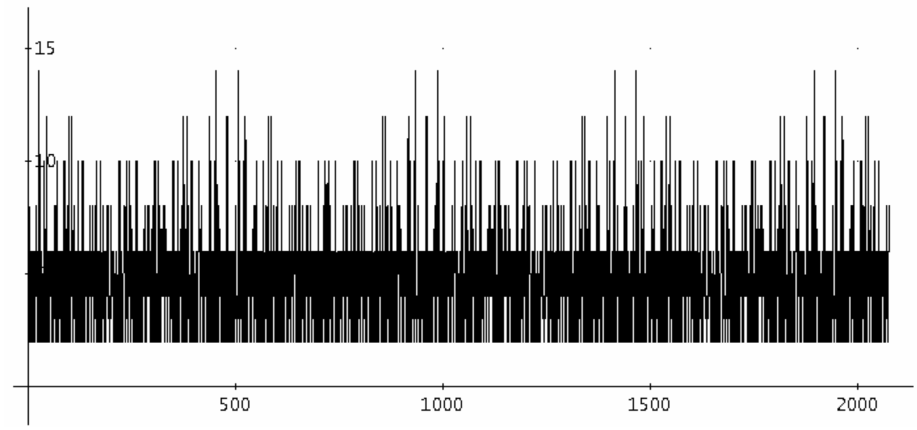
We cannot discover any period. Maybe that we need a larger n? Take n = 2000:

#13: Δ_seq(2000, 5)

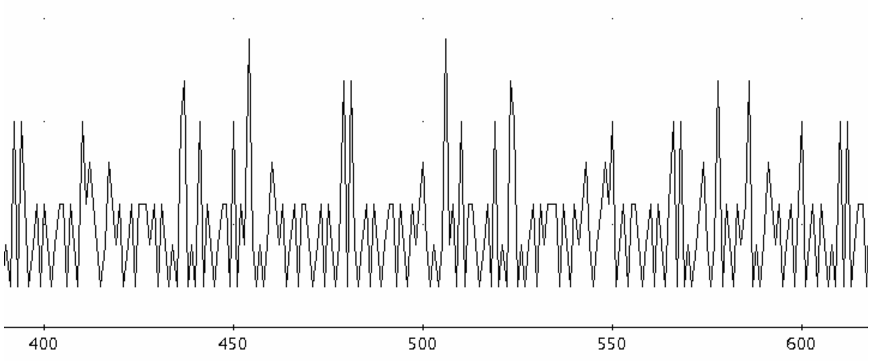


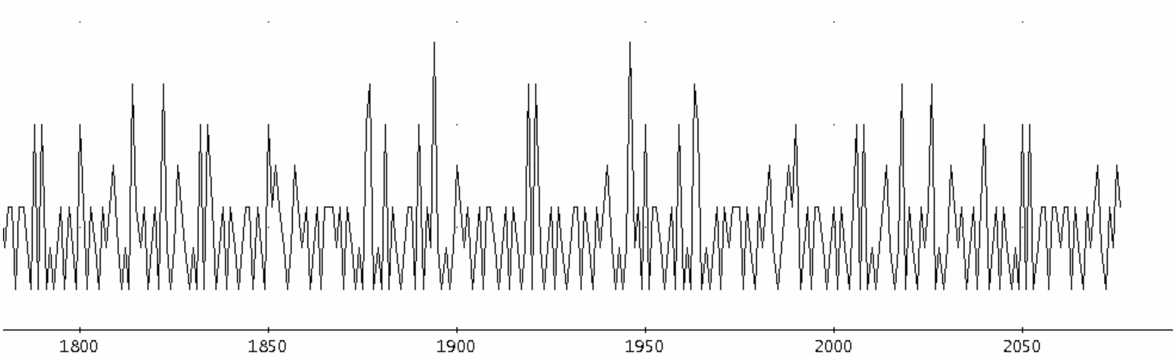
n = 10 000

#14: Δ_seq(10000, 5)



Let’s have a closer look and compare the region with its centre between 450 and 500 and the region with its centre between 1900 and 1950.





Obviously there seems to exist a symmetry.

k = 3, n = 300, period length = 8

```
[6,4,2,4,2,4,6,2,6,4,2,4,2,4,6,2,6,4,2,4,2,4,6,2,6,…]
```

k = 4, n = 500, period length = 48

```
[10,2,4,2,4,6,2,6,4,2,4,6,6,2,6,4,2,6,4,6,8,4,2,4,2,4,8,6,4,6,2,4,6,2,6,6,4,2,4,
6,2,6,4,2,4,2,10,2,10,2,4,2,4,6,2,6,4,2,4,6,6,2,6,4,2,6,4,6,8,4,2,4,2,4,8,6,4,6,
2,4,6,2,6,6,4,2,4,6,2,6,4,2,4,2,10,2,10,2,4,2,4,6,2,6,4,2,4,6,6,2,6,4,2,6,...]
```

k = 5, n = 10 000, period length = ???
According to the graph from above we can assume that it is about 500.

Back to Dietmar's paper:

I will define a **width** as follows:

width(k) is the number of all integers containing the differences of one period, or said with other words, it is the first divisible number after the first set of indivisible numbers by the first k prime numbers. This seems to be a bit complicated. I will make this clearer using a short table:

k	base set	width	number = period length	indivisible integers (including 1)	sum of the indivisibles
1	{2}	2	1	{1}	1
2	{2,3}	6	2	{1,5}	6
3	{2,3,5}	30	8	{1,7,11,13,17,19,23,29}	120
4	{2,3,5,7}	210	48	{1,11,13,17,19,23,29,31,37, 41,43,47,53,59,61,67,71,73, 79,83,89,97,101,103,107, 109,113,121,127,131,137, 139,143,149,151,157,163, 167,169,173,179,181,187, 191,193,197,199,209}	5040
5	{2,3,5,7,11}	2310	480		

General formulae are:

$$\text{width}(k)=\prod_{i=1}^{k} p_i\,,\quad \text{number}(k)=\prod_{i=1}^{k}(p_i-1)$$

According to these conjectures (there are no proofs) the period length for k = 5 should be 480. The plot seems to confirm this. I will check it supported by DERIVE:

```
set1 := (d_seq(3000, 5))
                        [1, ..., 480]

set2 := (d_seq(6000, 5))
                        [481, ..., 960]

set1 - set2
```

The result is a list of 480 zeros. So I should be right.

What concerns the sum of one period it is remarkable that the sums of the first and the last, of the second and the last but one, of the third and the ... are constant and equal the width.

e.g. for $k = 4$: 1 + 209 = 11 + 199 = 13 + 197 = ... = 103 + 107 = 210 (24 times)

This makes it easy to calculate the sum:

$$\text{sumper}(k) = \frac{1}{2} \cdot \text{width}(k) \cdot \text{number}(k) = \frac{1}{2} . \prod_{i=1}^{k} p_i \prod_{i=1}^{k} (p_i - 1).$$

Let's sum this up with DERIVE:

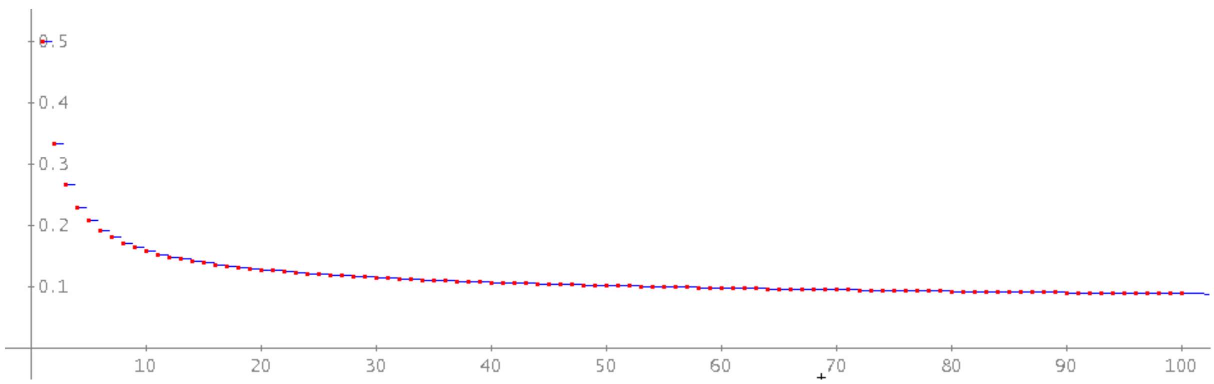
```
width(k) := ∏(VECTOR(NTH_PRIME(i), i, 1, k))

numb(k) := ∏(VECTOR(NTH_PRIME(i) - 1, i, 1, k))

VECTOR(width(k), k, 1, 8) = [2, 6, 30, 210, 2310, 30030, 510510, 9699690]

VECTOR(numb(i), i, 1, 8) = [1, 2, 8, 48, 480, 5760, 92160, 1658880]

            width(k)·numb(k)
sumper(k) := ----------------
                   2

VECTOR(sumper(i), i, 1, 8)

[1, 6, 120, 5040, 554400, 86486400, 23524300800, 8045310873600]
```

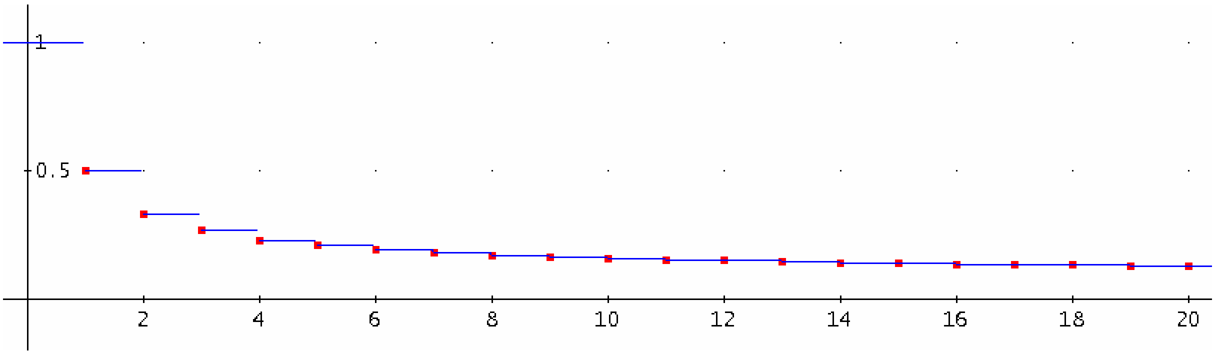
I will define a kind of "prime number density" with respect to the number of the first k prime numbers:

$$\text{my\_pr\_dens}(k) = \frac{\text{numb}(k)}{\text{width}(k)}.$$

I know that not all numbers in numb(k) are prime numbers because only the divisibility by the first k primes is considered but however we can see that this "density" also decreases.

```
my_pr_dens(x)

VECTOR([x, my_pr_dens(x)], x, 1, 100)
```





The plot of **my_pr_dens(x)** appears as a piecewise defined function.

This "density" must not be mixed up with the prime number density function $\pi(x)$ how it is well known in mathematics. $\pi(x)$ gives an approximation of the number of primes $\le x$.
$\pi(x)$ was – as far as I do know – defined by Gauss as follows:

$$\pi(x) = \int_2^x \frac{dt}{\ln t} = \mathrm{Li}(x) \text{ (= Logarithmic Integral).}$$

Using this counting method we can define the density as $\frac{\mathrm{Li}(x)}{x}$.

Dietmar mentions the "Euler equation" which was discovered by Leonhard Euler:

This equation reads as $\lim_{k \to \infty} \prod_{i=1}^{k} \frac{p_i^2}{p_i^2 - 1} = \frac{\pi^2}{6}$.

Again Dietmar Oertel:

Similar to the **numb(k)** function from above I define a function **numb_plus(k)**:

```
numb_plus(k) := ∏(VECTOR(NTH_PRIME(i) + 1, i, 1, k))

VECTOR(numb_plus(i), i, 1, 8)

[3, 12, 72, 576, 6912, 96768, 1741824, 34836480]
```

But unlike to **numb(k)** and **width(k)** I cannot find any meaningful interpretation of these numbers except this one, that **numb_plus** is necessary to connect my functions with the Euler equation:

$$\text{my\_euler\_(k)} := \frac{\text{width(k)}^2}{\text{numb(k)} \cdot \text{numb\_plus(k)}}$$

```
my_euler_(100) = 1.644515221

my_euler_(1000) = 1.644913174
```

It needs a calculation time of 24 seconds for $k = 1000$.

$$\text{my\_euler}(k) := \prod\left(\text{VECTOR}\left(\frac{\text{NTH\_PRIME}(i)^2}{\text{NTH\_PRIME}(i)^2 - 1},\ i,\ k\right)\right)$$

my_euler(10) = 1.63307049

my_euler(100) = 1.644515221

my_euler(1000) = 1.644913174

This procedure needs only 16 seconds for $k = 1000$.

Interestingly my third procedure is the fastest one:

f_pr(k) := VECTOR(NTH_PRIME(i), i, k)

$$\text{prm\_int\_dens}(\text{vprm}) := \frac{\prod_{n=1}^{\text{DIM}(\text{vprm})} (\text{vprm}_n - 1)}{\prod_{n=1}^{\text{DIM}(\text{vprm})} \text{vprm}_n}$$

$$\text{prm\_int\_dens\_plus}(\text{vprm}) := \frac{\prod_{n=1}^{\text{DIM}(\text{vprm})} (\text{vprm}_n + 1)}{\prod_{n=1}^{\text{DIM}(\text{vprm})} \text{vprm}_n}$$

$$\text{prm\_euler}(\text{vprm}) := \frac{1}{\text{prm\_int\_dens}(\text{vprm}) \cdot \text{prm\_int\_dens\_plus}(\text{vprm})}$$

prm_euler_first(n) := prm_euler(f_pr(n))

prm_euler_first(10) = 1.63307049

prm_euler_first(100) = 1.644515221

prm_euler_first(1000) = 1.644913174

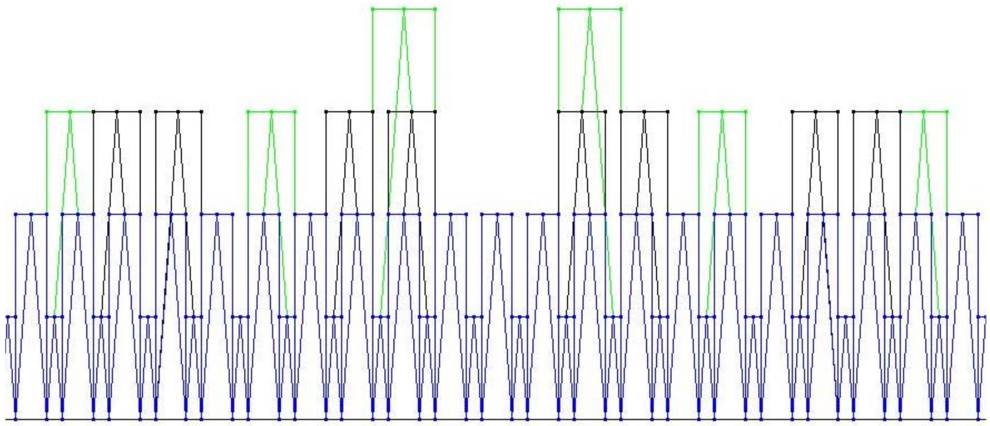
prm_euler_first(10000) = 1.644932811

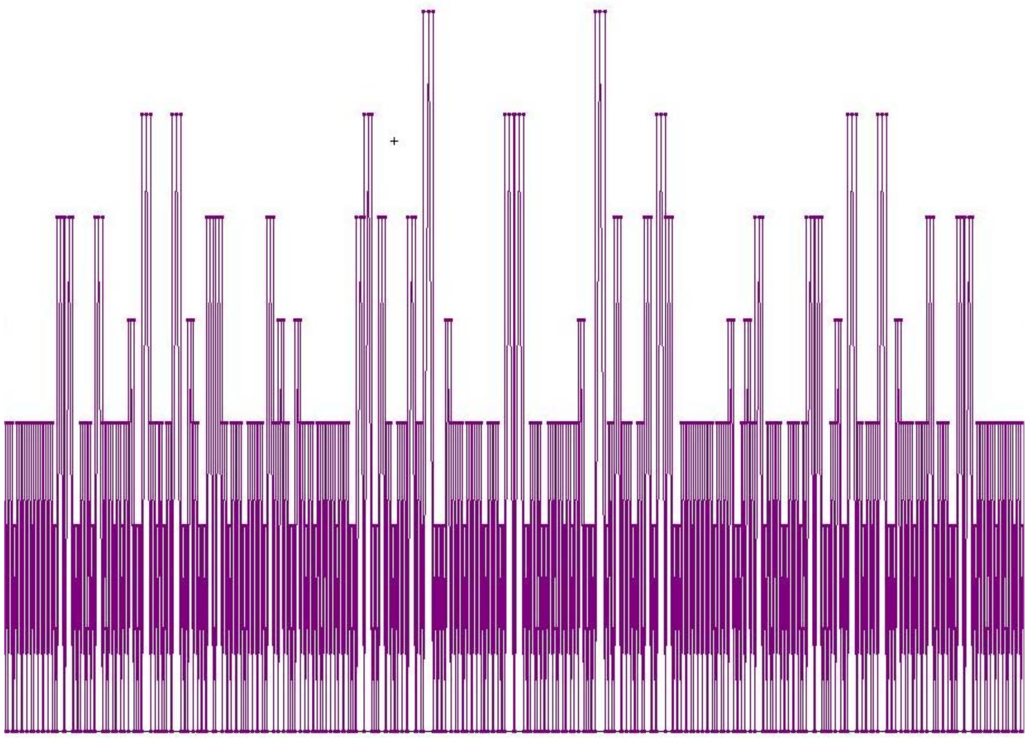
It needs only 8 seconds for $k = 1\,000$, but 986 seconds for $k = 10\,000$.

$$\frac{\pi^2}{6} = 1.644934066$$

Dietmar was inspired by my graphics (pages 33 & 34) that was eager to produce his own illustrations of the Δ-sequences periodicity.

See two of his very pretty plots.





(will be continued)

Links:

http://www.austromath.at/medienvielfalt/materialien/krypto/lernpfad/content/k_prim.htm

http://wwwmath.uni-muenster.de/u/deninger/about/seminare/vorlesungsskript-an-zahltheorie/primzahlen2.pdf

http://arxiv.org/ftp/arxiv/papers/0803/0803.0420.pdf

http://sciblogs.co.nz/guestwork/2010/07/19/unexpected-prime-number-breakthrough/

Dear Josef,

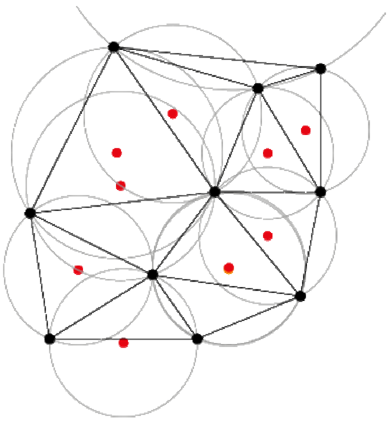
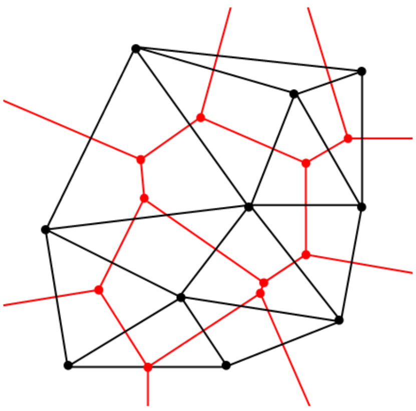
it's been such a long time !!!
I see you are well, always busy. Well done! I also try to keep fit, both physically and mentally.
I remember from old times that you were very keen on 2D graphic representations in Derive.
I'm also a fan of them, and I have recently come across Delaunay triangulations and Voronoi diagrams.
They are truly amazing and beautiful, and I'd like to have a Derive program to generate them, but
I do not quite understand the algorithm. Can you help me? maybe you have developed a program yourself, I
wouldn't be surprised if you did!!

thanks in advance, old friend
Enric Puig, enricpuig02@gmail.com

DNL: I must admit that I had never heard before about Delaunay (Voronoi sounds better for me but it is a very "weak" sound ...). I got some information from WikiPedia:

A **Delaunay triangulation** for a set **P** of points in a plane is a triangulation DT(**P**) such that no point in **P** is inside the circumcircle of any triangle in DT(**P**). Delaunay triangulations maximize the minimum angle of all the angles of the triangles in the triangulation.

The Delaunay triangulation of a discrete point set **P** in general position corresponds to the dual graph of the **Voronoi tessellation** for **P**.

Dear Mr. Böhm,

I read DNL#88 with much interest. I specially liked the contribution about T- and Z-scores. I'd like to add a note from practice. Statistical evaluations like these are made in medicine for osteoporosis patients when measuring their density of bones.

X-raying the lower lumbra vertebrae (untere Lendenwirbel) it is noticed how much of the given ray energy interfuses the observed bone. When much of the energy is absorbed then the bone is dense and stable, otherwise this is a sign for a weak bone with possible danger to be broken.

Due to his age of life the bone density of everyone is decreasing and there is a graphic correlation between the age of a patient and his/her "age-related normal" bone density. Deviations are expressed by T- and Z-score.

At the moment I cannot write more about this but I will inform myself in the next future and then I will give the respective report.

I have two more problems to discuss:

1.) In my materials from my time as student I found calculations for long division of polynomials. These were exercises in the 1st semester. I still know how to perform a long division but I never heard for which purpose it is needed? In the meanwhile I found out that it can be used to reduce the order of a polynomial equation by dividing by a linear factor (*x* – solution) if a solution of the equations is known. This is clear for me. But in my old materials I had to divide a polynomial of order five by a cubic polynomial. I cannot realize any sense in this calculation!

2.) In my job I often had to deal with the problem finding an appropriate function describing physical measuring results. Since the EXCEL era this no longer a problem because after some tries a satisfying fit-function can be found.

But there are physical processes which cannot be described by polynomial functions. In oil hydraulics one can measure spring and damping properties of plain bearings (= Gleitlager) and then present the results in a diagram. These diagrams are used for calculating the resonance behaviour of fast running shafts (turbines or compressors). Because lack of a matching "polynomial approximation" the curves are represented by many nodes for applying linear interpolation between them. This is not so bad but I'd prefer using an approximating function.

In the book "*Bronstein-Semendjajew*" I found a type of curve which seem to fit – at least „optically". It has the form: $y = a + b/x + c/x^2$.

Is it possible to find the coefficients a, b, and c supported by many nodes (x_i/y_i)?

Many thanks in advance and best regards from sunny but cold Nuremberg,

Klaus Körner, koerner.klaus@gmx.de

***DNL*:** Many thanks for your comments on T- and Z-scores. I am quite sure that Guido Herweyers will be very happy reading your kind words.

What concerns your two questions:

(1) I can imagine that students who find long division hard and boring might ask the teacher about its sense. What are possible answers? There are some applications of this algorithm, most of them beyond the students' mathematics knowledge at the time when it is in their curriculum.

- Partial fractions decomposition
- Reduction of the order of a polynomial equation by a linear factor division
- Investigation of the behaviour in infinity of rational functions
- Calculating the proof sum in the CRC polynomial
- One important step in the AES-algorithm (Advanced Encryption Standard)
- GCD of two polynomials (Euclidean algorithm)
- Function value, tangents & more for polynomial functions[1]
- possibly in the theory of Groebner Bases (I am not quite sure)

[1] Strickland-Constable, Charles, "A simple method for finding tangents to polynomial graphs", *Mathematical Gazette* 89, November 2005: 466-467.

I would like to give some examples:

Investigate the behaviour in infinity for the given functions:

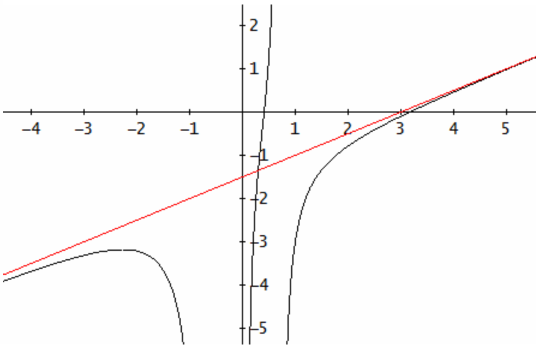
$$f1(x)=\frac{x^4-3x^3-2x+1}{2x^3-x},\quad f2(x)=\frac{x^4-3x^3-2x+1}{2x^2-x},\quad f3(x)=\frac{x^4-3x^3-2x+1}{2x-1}$$

We perform the division. The rational part tends to 0 for $x \to \infty$. The integer part is the equation of the asymptote (curve):

$$\frac{x^4-3\cdot x^3-2\cdot x+1}{2\cdot x^3-x}$$

$$\frac{5\cdot x}{2\cdot(2\cdot x^2-1)}-\frac{7}{2\cdot(2\cdot x^2-1)}+\frac{x}{2}-\frac{1}{x}-\frac{3}{2}$$

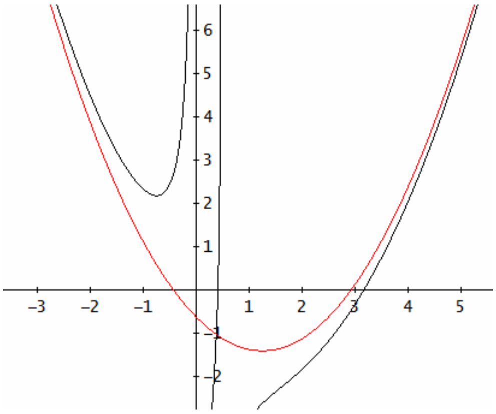
$$\frac{x}{2}-\frac{3}{2}$$



$$\frac{x^4-3\cdot x^3-2\cdot x+1}{2\cdot x^2-x}$$

$$-\frac{5}{8\cdot(2\cdot x-1)}+\frac{x^2}{2}-\frac{5\cdot x}{4}-\frac{1}{x}-\frac{5}{8}$$

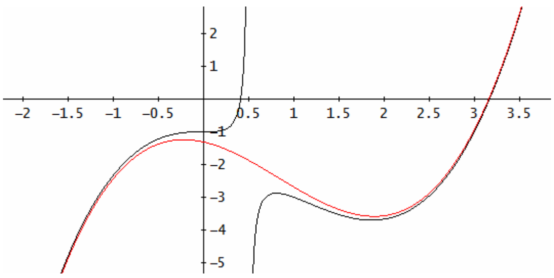
$$\frac{x^2}{2}-\frac{5\cdot x}{4}-\frac{5}{8}$$



$$\frac{x^4-3\cdot x^3-2\cdot x+1}{2\cdot x-1}$$

$$-\frac{5}{16\cdot(2\cdot x-1)}+\frac{x^3}{2}-\frac{5\cdot x^2}{4}-\frac{5\cdot x}{8}-\frac{21}{16}$$

$$\frac{x^3}{2}-\frac{5\cdot x^2}{4}-\frac{5\cdot x}{8}-\frac{21}{16}$$



We have a straight line, a parabola and finally a third order polynomial appearing as asymptotes. In *DERIVE* we can receive the integer part and the remainder of the result immediately.

TI-NspireCAS offers polyQuotient and polyRemainder together with expand and propFrac. The V200 has only the last two functions available.

$$\text{EXPAND}\left(\frac{x^4 + 3\cdot x^3 + 5\cdot x + 6}{x^2 - 5}\right) = \frac{20\cdot x}{x^2 - 5} + \frac{31}{x^2 - 5} + x^2 + 3\cdot x + 5$$

$$\text{QUOTIENT}(x^4 + 3\cdot x^3 + 5\cdot x + 6,\ x^2 - 5) = x^2 + 3\cdot x + 5$$

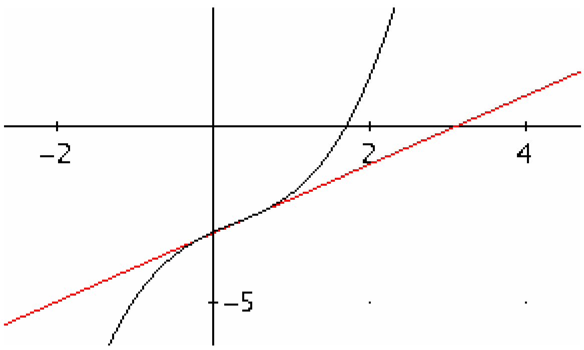
$$\text{REMAINDER}(x^4 + 3\cdot x^3 + 5\cdot x + 6,\ x^2 - 5) = 20\cdot x + 31$$

$\text{polyQuotient}(x^4-3\cdot x^3-2\cdot x+1,2\cdot x^2-x)$	$\frac{x^2}{2}-\frac{5\cdot x}{4}-\frac{5}{8}$
$\text{polyRemainder}(x^4-3\cdot x^3-2\cdot x+1,2\cdot x^2-x)$	$1-\frac{21\cdot x}{8}$
$\text{expand}\left(\frac{x^4-3\cdot x^3-2\cdot x+1}{2\cdot x^2-x}\right)$	$\frac{-5}{8\cdot(2\cdot x-1)}+\frac{x^2}{2}-\frac{5\cdot x}{4}-\frac{1}{x}-\frac{5}{8}$
$\text{propFrac}\left(\frac{x^4-3\cdot x^3-2\cdot x+1}{2\cdot x^2-x}\right)$	$\frac{-(21\cdot x-8)}{8\cdot x\cdot(2\cdot x-1)}+\frac{x^2}{2}-\frac{5\cdot x}{4}-\frac{5}{8}$

This is to tell about the integer part, but there is also a nice feature of the remainder.

Let $f(x)=\frac{x^3}{2}-\frac{2x^2}{5}+x-3$, divide by $\left(x-\frac{1}{2}\right)^2$ and look at the remainder:^

$$\text{REMAINDER}\left(\frac{x^3}{2} - \frac{2\cdot x^2}{5} + x - 3,\ \left(x - \frac{1}{2}\right)^2\right) = \frac{39\cdot x}{40} - \frac{121}{40}$$

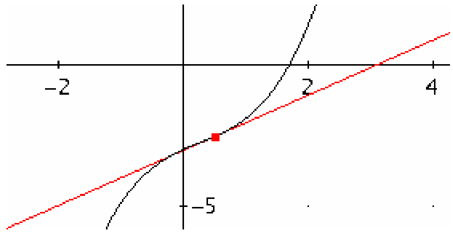


This seems to be the tangent, but in which point? What we do know is that dividing by a linear factor with a zero gives the reduced polynomial and dividing by another linear factor (x – a) gives the function value $f(a)$ as remainder.

Both procedures can be substituted by the Horner algorithm – so it does not need long division.

$$\text{REMAINDER}\left(\frac{x^3}{2}-\frac{2\cdot x^2}{5}+x-3,\ x-\frac{1}{2}\right)=-\frac{203}{80}$$

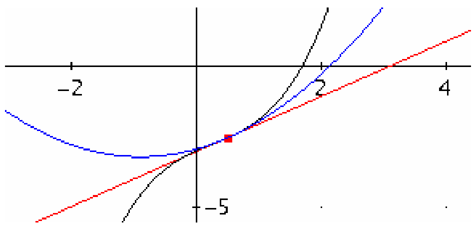
$$\left[\frac{1}{2},\ -\frac{203}{80}\right]$$



Now the point is fixed!

What about proceeding, i.e. let us divide by $(x-a)^3$ and plot the remainder. What do you expect as outcome?

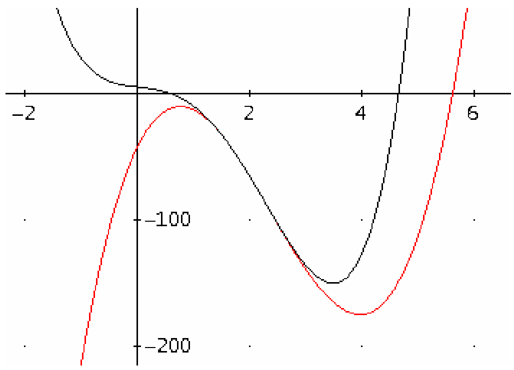
$$\text{REMAINDER}\left(\frac{x^3}{2}-\frac{2\cdot x^2}{5}+x-3,\ \left(x-\frac{1}{2}\right)^3\right)=\frac{7\cdot x^2}{20}+\frac{5\cdot x}{8}-\frac{47}{16}$$



It is the osculating quadratic function – a parabola. One last example to come to an end, let's divide by $(x-a)^4$:

$$\text{REMAINDER}(3\cdot x^4-14\cdot x^3+x^2-5\cdot x+5,\ (x-2)^4)$$

$$10\cdot x^3-71\cdot x^2+91\cdot x-43$$



Do you see what I see?

$$\text{TAYLOR}\left(\frac{x^3}{2}-\frac{2\cdot x^2}{5}+x-3,\ x,\ \frac{1}{2},\ 0\right)=-\frac{203}{80}$$

$$\text{TAYLOR}\left(\frac{x^3}{2}-\frac{2\cdot x^2}{5}+x-3,\ x,\ \frac{1}{2},\ 1\right)=\frac{39\cdot x-121}{40}$$

$$\text{TAYLOR}\left(\frac{x^3}{2}-\frac{2\cdot x^2}{5}+x-3,\ x,\ \frac{1}{2},\ 2\right)=\frac{28\cdot x^2+50\cdot x-235}{80}$$

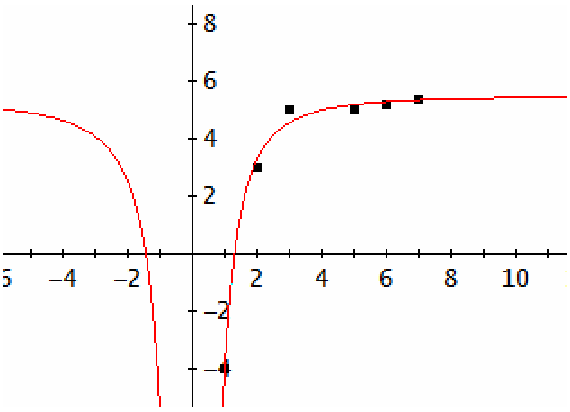
$$\text{TAYLOR}(3\cdot x^4-14\cdot x^3+x^2-5\cdot x+5,\ x,\ 2,\ 3)=10\cdot x^3-71\cdot x^2+91\cdot x-43$$

This does not need any comment.

Possible question for students:

Do you have an idea how to extend Horner's Rule in order to find the remainders from above without applying polynomial divisions?

(2) To your second question:
It is possible to find the regression line of the requested form. In DNL#63 I presented a generalised regression procedure based on the normal matrix.
DERIVE has this algorithm built in in its FIT-function:

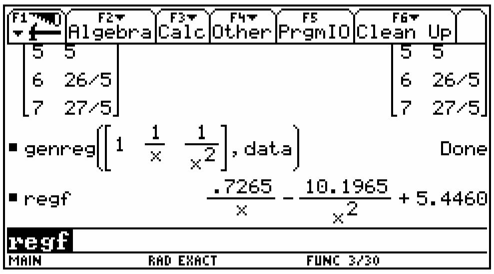


$$\text{data} := \begin{bmatrix} 1 & -4 \\ 2 & 3 \\ 3 & 5 \\ 5 & 5 \\ 6 & \frac{26}{5} \\ 7 & \frac{27}{5} \end{bmatrix}$$

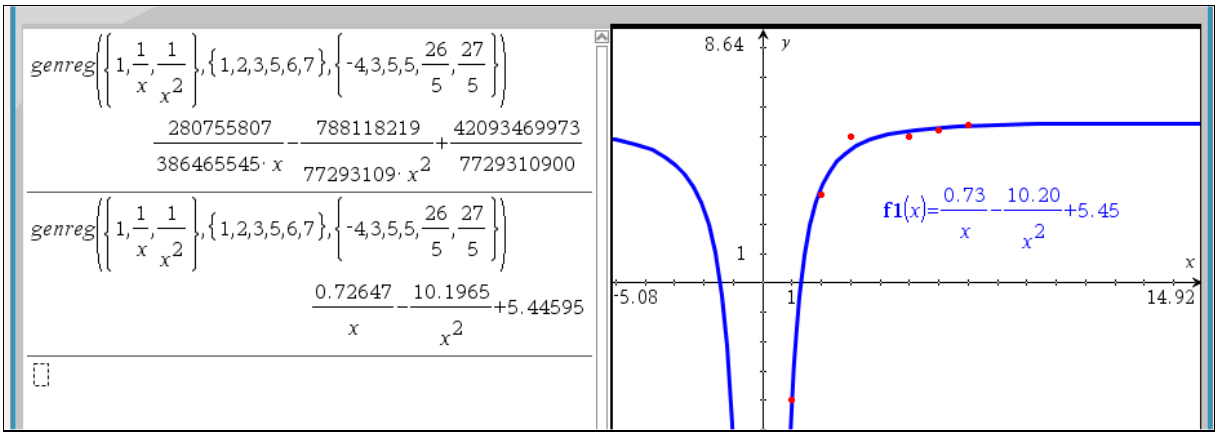
$$\text{FIT}\left(\left[x,\ a + \frac{b}{x} + \frac{c}{x^2}\right],\ \text{data}\right)$$

$$\frac{0.7264704722}{x} - \frac{10.19648749}{x^2} + 5.445953787$$

In DNL#63 I showed a small program performing this regression:



And in DNL#89 I present the Nspire-form.
I am using lists for *x*- and *y*-values because then plotting the scatter diagram is easier.



It might be a nice experience for student to perform the Euclidean algorithm for finding the GCD for polynomials. They can practise long division and then check their manipulations using the respective CAS functions.

Let $a = x^5 - 2x^4 + 2x^2 - 13x + 18$ and $b = x^3 - x^2 + 8x - 20$. Find the GCD(a,b).

I present the manual procedure follwed first by the *DERIVE* treatment and then by the *TI-NspireCAS* screen shot. For obtaining the GCD only the remainders are important. But for controlling the long divisions the students would also need the quotients.

$$a \quad : \quad b = r_0 \quad = \quad q_1$$

$$(x^5 - 2x^4 + 2x^2 - 13x + 18) : (x^3 - x^2 + 8x - 20) = x^2 - x - 9$$

$$21x^2 + 39x - 162 = r_1$$

$$r_0 \quad : \quad r_1 \quad = \quad q_2$$

$$(x^3 - x^2 + 8x - 20) : (21x^2 + 39x - 162) = \frac{x}{21} - \frac{20}{147}$$

$$\frac{1030x}{49} - \frac{2060}{49} = r_2$$

$$r_1 \quad : \quad r_2 \quad = \quad q_3$$

$$(21x^2 + 39x - 162) : \left(\frac{1030x}{49} - \frac{2060}{49}\right) = \frac{21 \cdot 49x}{1030} + \frac{81 \cdot 49}{1030}$$

$$0 = r_3$$

The last remainder ≠ 0 = r_2 which can be factorized: $\frac{1030}{49}(x-2)$. Hence ***x* – 2** is the GCD(a,b). Students can check their operations using *DERIVE* or *TI-NspireCAS* as follows:

#1: $\left[a := x^5 - 2\cdot x^4 + 2\cdot x^2 - 13\cdot x + 18,\ b := x^3 - x^2 + 8\cdot x - 20\right]$

#2: [q1 := QUOTIENT(a, b), r1 := REMAINDER(a, b)]

#3: $\left[q1 := x^2 - x - 9,\ r1 := 21\cdot x^2 + 39\cdot x - 162\right]$

#4: [q2 := QUOTIENT(b, r1), r2 := REMAINDER(b, r1)]

#5: $\left[q2 := \frac{x}{21} - \frac{20}{147},\ r2 := \frac{1030\cdot x}{49} - \frac{2060}{49}\right]$

#6: [q3 := QUOTIENT(r1, r2), r3 := REMAINDER(r1, r2)]

#7: $\left[q3 := \frac{1029\cdot x}{1030} + \frac{3969}{1030},\ r3 := 0\right]$

#8: FACTOR(r2) = $\frac{1030\cdot(x - 2)}{49}$

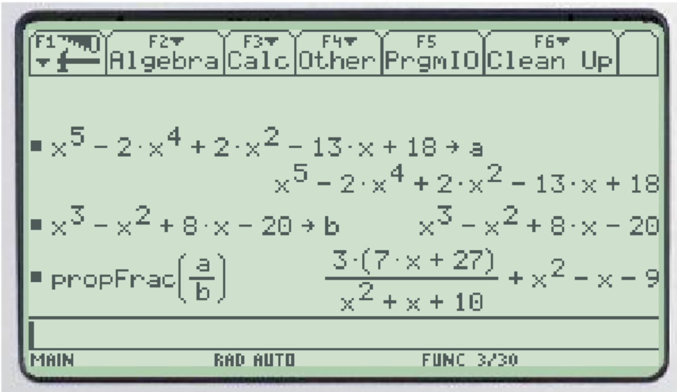
#9: POLY_GCD(a, b) = x – 2

$a:=x^5-2\cdot x^4+2\cdot x^2-13\cdot x+18:b:=x^3-x^2+8\cdot x-20$	$x^3-x^2+8\cdot x-20$
$q1:=\text{polyQuotient}(a,b)$	x^2-x-9
$r1:=\text{polyRemainder}(a,b)$	$21\cdot x^2+39\cdot x-162$
$q2:=\text{polyQuotient}(b,r1)$	$\frac{x}{21}-\frac{20}{147}$
$r2:=\text{polyRemainder}(b,r1)$	$\frac{1030\cdot x}{49}-\frac{2060}{49}$
$q3:=\text{polyQuotient}(r1,r2)$	$\frac{1029\cdot x}{1030}+\frac{3969}{1030}$
$r3:=\text{polyRemainder}(r1,r2)$	0
$\text{factor}(r2)$	$\frac{1030\cdot(x-2)}{49}$

This cannot be done with the V200 / TI-92. The only one function which seems to apply is **propFrac**:

It looks fine - at the first glance. The quotient of the first division is the same as above but the remainder does not match. What happened?

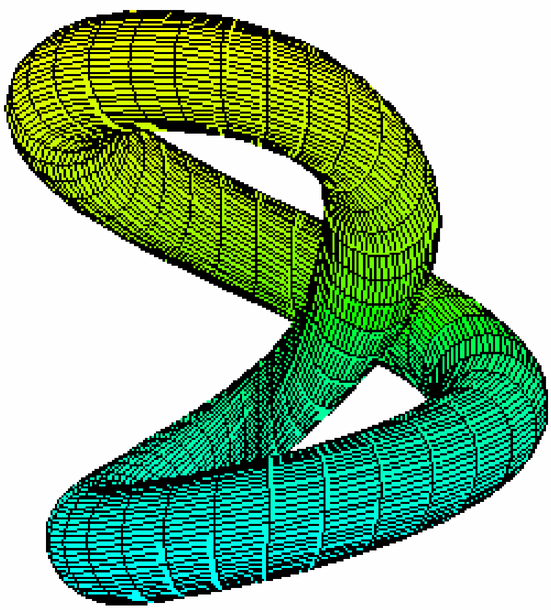
The V200 is too clever! It cancels the common factor (x – 2) so it cannot be detected any longer.



A Pretty Knot

[(3 + SIN(v) + COS(u))·SIN(2·v),
(3 + SIN(v) + COS(u))·COS(2·v),
SIN(u) + 4·COS(v)]

$0 \le u, v \le 2\pi$



Latest News:

In the DERIVE-News JiscMail List circulated the following message:

```
...
If we don't have any volunteer then the list will be closed on Friday 26th
April. When this happens subscribers will not be able to post messages to
the list but archives will be kept available.

Thanks for your co-operation.
JISCMail Helpdesk
```

Fortunately a subscriber offered to maintain the list in the future. His first message to the current subscribers was:

Dear Derivers,

My name is Lars Erup; I am a Satellite Communications Systems Engineer based in Montreal, Canada.

I have agreed to take over the task of acting as Owner for the "Derive-News" list. Let me assure you that I do not have any plans for changing anything, certainly as long as the list is ticking along quietly. Let me also remind you of the current list settings:

- Anyone can subscribe
- Only people subscribed to the list can post messages
- Archives are kept for this list
- This is a public list, with public archives; anyone can find it (& search archives) on the JISCMail website and through search engines

There are currently 172 of us.

Best Regards

Lars

It s great that this list survives and I find it great that we know the person doing the job. I wrote to Lars, offered cooperation and invited him to join the DUG. He answered immediately:

...
In your newsletter, you may want to mention the existence of the list (and where to find it), and encourage people to use it for discussions.
Likewise, it might be a good idea if you post a synopsis of the newsletter to the list whenever a new edition is ready.

These are just a couple of ideas.

Best Regards
Lars

The email address of the list is DERIVE-NEWS@JISCMAIL.AC.UK

I invite you to use this form of communication between DERIVERs from all over the world and subscribe. You can visit the DERIVE-NEWS archive at

https://www.jiscmail.ac.uk/cgi-bin/webadmin?A0=derive-news

Conference Fees

Registration fees for participants

Early Bird (until April 15th, 2014) **€ 300.00**
Late (after April 15th, 2014) **€ 400.00**

Registration fees for participants will cover admission to all presentations, book of abstracts, coffee breaks, welcome reception, excursion and conference dinner.

Registration fees for accompanying persons

Early Bird (until April 15th, 2014) **€ 200.00**
Late (after April 15th, 2014) **€ 300.00**

Registration fees for accompanying persons will cover welcome reception, excursion and conference dinner.

Call for Submissions
Deadline: February 15th, 2014

Format
Short lecture: 25 minutes
(with questions and answers)

Lecture: 55 minutes
(with questions and answers)

Workshop in computer labs: 90 minutes
(with questions and answers)

Topics
Possible topics for the both strands can be found at the conference website.

Time Table

February 15th, 2014	Deadline for submissions
April 1st, 2014	Notification of acceptance
April 15th, 2014	Final day for Early Bird Registration
July 1st–5th, 2014	TIME 2014

Website
www.time2014.org

TIME 2014

Technology and its Integration in Mathematics Education

This conference combines two conferences:

13th
ACDCA Summer Academy

and

11th
Conference for CAS in Education & Research

(Former Int'l TI-Nspire & Derive Conference)

July 1st–5th, 2014

Danube University Krems
Krems, Austria

1st
Announcement & Call for Submissions

The conference is hosted by the Danube University Krems

Conference Co-Chairs

+ Peter Baumgartner, Austria
+ Walter Wegscheider, Austria

Keynote Speakers

+ Peter Baumgartner, Austria
  Danube University Krems
+ Regina Bruder, Germany
  Technical University, Darmstadt
+ Bruno Buchberger, Austria
  RISC Institute, Linz
+ Pavel Pech, Czech Republic
  University of South Bohemia České Budějovice
+ Gilles Picard, Canada
  École de technologie supérieure, Montréal
+ Marlene Torres-Skoumal, USA
  International Baccalaureate Schools

Program Committees

ACDCA Summer Academy

+ Helmut Heugl, Austria (Co-Chair)
+ Tom Reardon, USA (Co-Chair)
+ Hans Stefan Siller, Germany (Co-Chair)

Conference for CAS in Education and Research

+ Michel Beaudin, Canada (Co-Chair)
+ Josef Böhm, Austria (Co-Chair)
+ José Luis Galán, Spain (Co-Chair)

Proceedings

Full paper submission deadline is July 5th, 2014 (Last day of the conference).

Publishing papers will be possible in special issues of the following journals (in case the paper passes the review process):

+ Journal of Symbolic Computation (JSC),
+ International Journal of Mathematical Education in Science and Technology (iJMEST),
+ The International Journal for Technology in Mathematics Education (IJTME).

All other papers will be published on the conference website in appropriate form.

Purpose of the Conference

In 1992 ACDCA (Austrian Center for Didactics of Computer Algebra) started a conference series which has become a driving force in bringing technology, in particular computer algebra systems (CAS), into the classroom.

The conference series comprises two strands: the **ACDCA Summer Academy**, primarily deals with didactical and pedagogical questions which arise from the use of technology for teaching and learning and the **Conference for CAS in Education & Research**, which are geared towards exploring the use of CAS software and symbolic calculators in education and towards using these tools in programming and research.

TIME 2014 offers two additional scopes:

+ "Development of new educational strategies in the information society"
+ "Technology on the way to a final central exam with obligatory use of technology"

Konferenzgebühren

Anmeldegebühr für TeilnehmerInnen

Frühbucher (bis 15. April 2014)	**€ 300,–**
Regulär (ab 15. April 2014)	**€ 400,–**

Die Anmeldegebühr beinhaltet den Eintritt zu allen Vorträgen, das Programmheft mit Abstracts, Kaffeepausen, Welcome-Reception, Exkursion und das Abendbankett.

Anmeldegebühr für LehrerInnen

Frühbucher (bis 15. April 2014)	**€ 20,–**
Regulär (ab 15. April 2014)	**€ 30,–**

Die Anmeldegebühr beinhaltet den Eintritt zu allen Vorträgen am 4. und 5. Juli, das Programmheft mit Abstracts und Kaffeepausen.

Anmeldegebühr für Begleitpersonen

Frühbucher (bis 15. April 2014)	**€ 200,–**
Regulär (ab 15. April 2014)	**€ 300,–**

Die Anmeldegebühr beinhaltet die Teilnahme an der Welcome-Reception, Exkursion und am Abendbankett.

Einladung zur Einreichung

Einreichungen sind bis
15. Februar 2014 möglich.

Vorträge und Workshops

Kurzvortrag: 25 Minuten
(mit Podiumsdiskussion)

Vortrag: 55 Minuten
(mit Podiumsdiskussion)

Workshops im PC-Labor: 90 Minuten

Abgabe der Papers ist am 05. Juli 2014
(letzter Tag der Konferenz).

Zeitplan

15.02.2014	Ende der Einreichfrist
01.04.2014	Benachrichtigung der Akzeptanz
15.04.2014	Ende der Frühbucher-Registrierung
01.-05.07.2014	TIME 2014

Detaillierte Informationen und Programm auf der Homepage der TIME 2014:

www.time2014.org

Austrian Center for Didactics of Computer Algebra

TIME 2014

Technology and its Integration in Mathematics Education

Diese Konferenz besteht aus zwei Teilen:

13th
ACDCA Summer Academy including Teachers' Days

and

11th
Conference for CAS in Education & Research

(früher Int'l TI-Nspire & Derive Conference)

01.–05. Juli 2014

an der Donau-Universität Krems
Krems an der Donau, Österreich

1.
Ankündigung &
Einladung zur Einreichung

Donau Universität Krems

Konferenzvorsitzende

+ Peter Baumgartner, Österreich
+ Walter Wegscheider, Österreich

Keynote-Speakers

+ Peter Baumgartner, Österreich
  Donau-Universität Krems
+ Regina Bruder, Deutschland
  Technische Universität Darmstadt
+ Bruno Buchberger, Österreich
  RISC Institut, Linz
+ Pavel Pech, Tschechische Republik
  Südböhmische Universität České Budějovice
+ Gilles Picard, Kanada
  École de technologie supérieure, Montréal
+ Marlene Torres-Skoumal, USA
  International Baccalaureate Schools

Program Committees

ACDCA Summer Academy

+ Helmut Heugl, Österreich
+ Tom Reardon, USA
+ Hans Stefan Siller, Deutschland

Conference for CAS in Education and Research

+ Michel Beaudin, Kanada
+ Josef Böhm, Österreich
+ José Luis Galán, Spanien

Themen der Konferenz

Im Jahr 1992 wurde die erste Konferenz zum Thema Didaktik des Technologieeinsatzes von ACDCA (Austrian Center for Didactics of Computer Algebra) initiiert. Es war der Beginn der Nutzung von CAS Werkzeugen im Unterricht. Derzeit wird die Konferenz alle zwei Jahre in verschiedenen Teilen der Welt abgehalten und kehrt 2014 nach Österreich zurück.

Diese Konferenzserie besteht aus zwei Teilen: Der **„ACDCA Summer Academy"**, welche sich mit didaktischen Fragen zum Einsatz von Technologien für das Lehren und Lernen beschäftigt, und der **„Conference for CAS in Education & Research"**, bei der es um den Einsatz von CAS in Schulen und Universitäten geht, um die Nutzung solcher Tools zur Programmierung und in der Forschung und um die Analyse aktueller Hard- und Softwareprodukte.

TIME 2014 bietet zwei zusätzliche Schwerpunkte:

+ Die Rolle interaktiver Medien und innovative Bildungstechnologien beim Lehr- und Lernprozess
+ Technologie auf dem Wege zu zentralen Prüfungen mit Technologienutzung

LehrerInnentage

am 4. und 5. Juli 2014 zum Thema
„Technologie auf dem Wege zu zentralen Prüfungen mit Technologienutzung"

Dieser Schwerpunkt berücksichtigt die Tatsache, dass in **Österreich** ab 2018 der Technologieeinsatz bei der zentralen Reifeprüfung verpflichtend ist und daher Überlegungen für den Einsatz im Unterricht schon jetzt hochaktuell sind.

Auch **in den vielen deutschen Bundesländern** wird Technologie beim Zentralabitur eingesetzt und es wird die Verwendung von Computeralgebrasystemen im Curriculum ab Klassenstufe 9/10 verbindlich festgelegt. Vorträge und Workshops befassen sich mit der Rolle der Technologie im Rahmen der Prüfung aber auch mit der Rolle der Technologie im Lernprozess auf dem Wege zur Prüfung.

Dieser Schwerpunkt wird besonders vom österreichischen Bundesministerium für Unterricht, Kunst und Kultur unterstützt.

Die Vorträge werden zum Teil in deutscher Sprache gehalten.

Anmeldungen für die LehrerInnentage bis spätestens 15. April 2014 auf der Webseite!

bm:uk