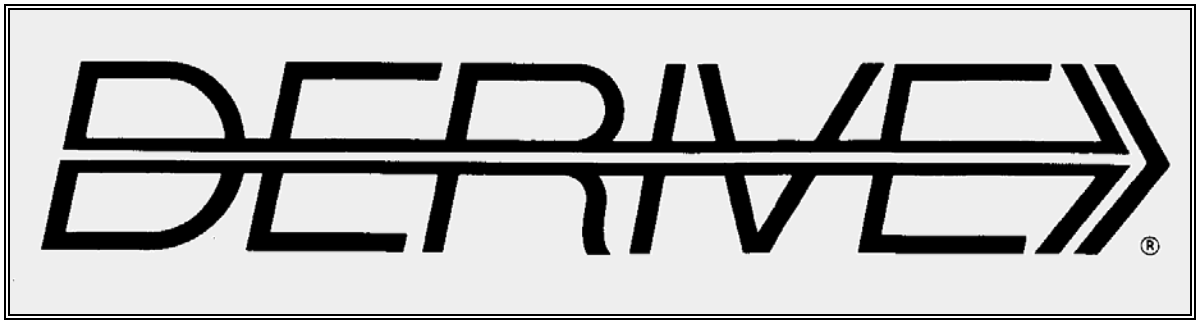
THE DERIVE - NEWSLETTER #56

THE BULLETIN OF THE



USER GROUP + CAS-TI

Contents:

December 2004

DERIVE & CAS-TI User Group Meeting 2004

User Group Meeting 2004 was announced for Saturday, 17 July, during TIME 2004 Conference in Montréal, Quebec, Canada.

DUG-members who attended the conference followed the announcement and joined the meeting. We spent two interesting and inspiring hours together.

The editor and president of the DUG gave a report about the activities of the DUG so far:

The assembly accepted and appreciated the editor´s proposal to have no membership dues for 2004 and the following years due to the fact, that the newsletter will not be printed and shipped by snail mail, but can be downloaded from the DUG-homepage.

Electronic form of the DUG publications meets the wishes and suggestions of many members. About 50 new members joined the User Group in 2004. Welcome to them all.

Since founding the DUG in 1991 we have published **54 newsletters**

containing

more than 2200 pages
386 contributions submitted by
149 authors from
27 countries.

We had **592 requests and answers in the User Forums** and presented
239 CAS-related books on the book shelf.

Among others Johann Wiesenbauer provided 27 Titbits from Algebra and Number Theory which could fill a very special book on this topic. We were very happy that we could express our gratefulness to Johann personally at the meeting.

We found that the Derive-Users have been very productive in delivering papers, the CAS-TI-Users are hesitating – with a few exceptions. So we have to encourage this group to go to public with their findings in the future.

Ideas and intentions for the future are:

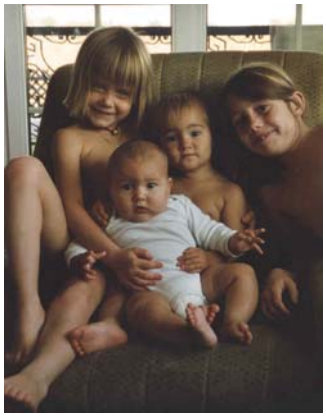
- Continuing publication of revised versions of Newsletters from 1991 - ..... This service is very much appreciated by the members and is an exciting task for the editor.
- Providing an extended index containing all articles and contributors. Later we would like to add short abstracts (2 sentences) and links between related contributions.

(continued on page 3)

Dear DUG-Members,

First of all I´d wish to welcome our 50 new members who joined DUG during 2004. We hope that you will find many information and inspiring ideas in our publications and we would be very pleased if some of you would submit one or the other contribution for publication.

As I reported at the DUG-Meeting in summer we would like to encourage our handheld users to share their experiences with us. I had many positive reactions on the fact that the Newsletter can be downloaded from our website and that I revise the old newsletters from our first years. I can say that this is a very interesting task to recognize how things have changed within a few years. I am sure that we will discover many new qualities of the many contributions from the early nineties. I was relly moved by revising DNL#4, when I found again a Call for Papers for our first DERIVE Spring School 1992 in Krems. Those were the days when it all has begun. In 1991 the Austrian government purchased the nationwide DERIVE licence for our secondary schools and we are very happy that some weeks ago our government renewed the licence contract for DERIVE 6.10.



This issue contains two fine articles from Giora & Nurit (Quadratic Approximation) and Karsten Schmidt who fulfills his promise to present examples for application of Moore-Penrose Matrices. We have a Danish contribution which deals with Quadratic Optimization and shows that this topic can be treated even in secondary school.

Bjoern´s article was written before times of Derive´s slider bar and I tried to include this valueable tool. I left his article in Danish and added some English comments.

Just recently I received a mail from our very productive member Don Phillips who sent a very impressive paper on "Two stage least squares", another mail from Josef Lechner containing two articels for publication and one mail from Canada announcing a paper about "Diophant-ive Polynomials". Lorenz Kopp did not only send his fine tool for generating tree diagrams but also a bundle of simulations for random experiments with very interesting graphic representations.

Please take also notice of the extended report on the DUG-meeting 2004 in Montreal.

I´ll take the occasion to thank Walter Wegscheider and Benjamin Kaineder for putting the DNLs and other files on the website.

Finally I wish you and your families a Merry Christmas Time and a Happy, Healthy and Successful New Year 2005.

With best regards from my grandchildren and from Noor and Josef

Download all *DNL-DERIVE*- and TI-files from

http://www.austromath.ac.at/dug/

http://www.bk-teachware.com/main.asp?session=375059

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

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

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.

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

Next issue: March 2005
Deadline 15 February 2005

Preview: Contributions waiting to be published

Finite continued fractions St. Welke, GER
Some simulations of Random Experiments, J. Böhm, AUT & L. Kopp, GER
Wonderful World of Pedal Curves, J. Böhm
Another Task for End Examination, J. Lechner, AUT
Tools for 3D-Problems, P. Lüke-Rosendahl, GER
ANOVA with *DERIVE* & *TI*, M. R. Phillips, USA
Hill-Encription, J. Böhm
CAD-Design with *DERIVE* and the *TI*, J. Böhm
Avoiding Convolution and Transforming Methods, M. Lesmes-Acosta, COL
Farey Sequences on the *TI,* M. Lesmes-Acosta, COL
Simulating a Graphing Calculator in *DERIVE*, J. Böhm, AUT
Henon & Co, J. Böhm
Pringles, B. Grabinger, GER
Challenges from Fermat, Bj. Felsager, DEN
Actuarial Mathematics, M. R. Phillips, USA
Are all Bodies falling equally fast, J. Lechner, AUT
Modelling Traffic Density, Th. Himmelbauer, AUT
Do you know this? Cabri & CAS on PC and Handheld, W. Wegscheider, AUT
Two Stage Least Squares, M. R. Phillips, USA

and Setif, FRA; Vermeylen, BEL; Leinbach, USA; Koller, AUT,
Keunecke, GER, ...and others

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

- Describing the many treasures which are more or less hidden among the User contributed files. Most of the users don´t know about the contents and how to use.
- It could be useful to deal with one or the other files of the MATH-folder, too.

At this occasion the editor expressed his thanks to Theresa Shelby, Albert Rich, David Stoutemyer and Bernhard Kutzler for their great cooperation.

The immediate contact between the users and to the responsible people is one of the great advantages of Derive and the CAS-TIs and – I am sure – a major part of their success.

Speaking about thanks we must not forget Noor Böhm, who has done all the administration work since 1991.

The report closed with an outlook on future contributions.

Then Bernhard Kutzler reported that he had audited the finances of the DUG and had found that everything was ok.

There was one proposal for the management committee for the next period, which was accepted with one voice:

Josef Böhm
Bärbel Barzel
Noor Böhm
Josef Lechner
Bernhard Kutzler
Walter Klinger

Johann Wiesenbauer suggested that due to the fact that the publications are now on the web we could pose a "Problem of the Month" to be tackled by our members. So we will put again "Challenge" into the next newsletter“ and wait for the responds. If there is need in more "problems" we can start with them at any time, if there is not, it would be lost time and efforts to bring them on the web.

Lana Moore from TI announced cooperation with TI by promoting the User Group and the Newsletter via the TI-Derive-Homepage. Many thanks for this valueable support.

Thanks also to all friends who came together and honored the meeting with their presence.
The next meeting was announced for the next DERIVE Conference which will take place in 2006 in Dresden, Germany.

The meeting was closed after 2 hours.

Total Differential with DERIVE and the TIs

Don Pillips

Thanks for DNL #55! It was great as usual. I really liked how you did my article on TVM comparing my routines with yours and the TI-89.

I've expanded my routines for actuarial math. So, if you're planning on publishing it at some future date, please wait for the "new, expanded, greater" version.

Also, I've noticed, as I'm sure many have, that Derive does not compute total differentials. I've attached a file which corrects that. Maybe Albert, et.al., will include this functionality in a future version of Derive.

Sincerely,
Don Phillips

Example 1: Find the first order total differential for y = x^2.

#2: TotalDifferential(y = x^2)

#3: dy = 2·x·dx

Example 2: Find the first and second order total differentials of z = 4x^3y^2.

#4: TotalDifferential(z = 4·x^3·y^2)

#5: dz = 12·x^2·y^2·dx + 8·x^3·y·dy

#6: TotalDifferential(z = 4·x^3·y^2, 2)

#7: d2z = 24·x·y^2·dx + 48·x^2·y·dx·dy + 8·x^3·dy^2

Example 3: Let z = √(x^2+y^2). Use a total differential to approximate the change in z as (x,y) varies from the point (3,4) to the point (3.04,3.98).

#8: TotalDifferential(z = √(x^2 + y^2))

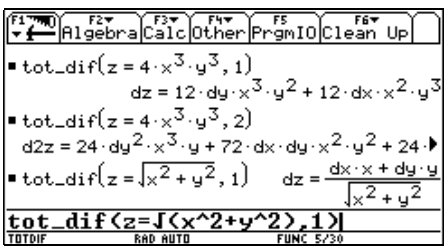
#9: $$dz = \frac{x \cdot dx}{\sqrt{x^2 + y^2}} + \frac{y \cdot dy}{\sqrt{x^2 + y^2}}$$

#10: $$\text{SUBST}\left[dz = \frac{x \cdot dx}{\sqrt{x^2 + y^2}} + \frac{y \cdot dy}{\sqrt{x^2 + y^2}}, [x, y, dx, dy], [3, 4, 0.04, -0.02]\right]$$

#11: dz = 0.008

Josef,

Attached is the total differential program for the TI-89.
It takes two arguments: the equation and the degree of differentiation. I just wish there was some way to put a default value in a TI-89 function or program.
Regards, Don.



You can find the TI-89, the TI-92 and the V200 grouped file among the files for download. Josef

Note from Josef Lechner and Ove Kroll concerning working with angles in DERIVE 6:

Setting Mode Degree/Radian has no effect on the input of angles:

#1: Angle := Degree

#2: TAN(45)

simplified

#3: TAN(45)

approximated

#4: 1.619775190

but tan(45°) gives simplified and approximated:

#5: TAN(45°) = 1

Now the same in Mode Radian:

#6: Angle := Radian

#7: TAN(45)

#8: 1.619775190

#9: TAN(45°) = 1

Setting the Mode has effect for the output of results:

#10: Angle := Degree

#11: SOLVE(TAN(α) = 1, α, Real) = (α = 225° ∨ α = (−135)° ∨ α = 45°)

#12: NSOLVE(TAN(α) = 1, α, Real) = (α = −5.497787147)

#13: NSOLVE(TAN(α°) = 1, α, Real) = (α = 45)

#14: Angle := Radian

#15: Precision := Exact

#16: Notation := Rational

#17: $\text{SOLVE}(\tan(\alpha) = 1, \alpha, \text{Real}) = \left(\alpha = \frac{5 \cdot \pi}{4} \vee \alpha = -\frac{3 \cdot \pi}{4} \vee \alpha = \frac{\pi}{4}\right)$

#18: NSOLVE(TAN(α) = 1, α, Real) = (α = −5.497787147)

#19: $\frac{\text{NSOLUTIONS}(\tan(\alpha) = 1, \alpha, \text{Real})}{1°}$

#20: [−315]

#21: ATAN(1)

#22: 0.7853981633

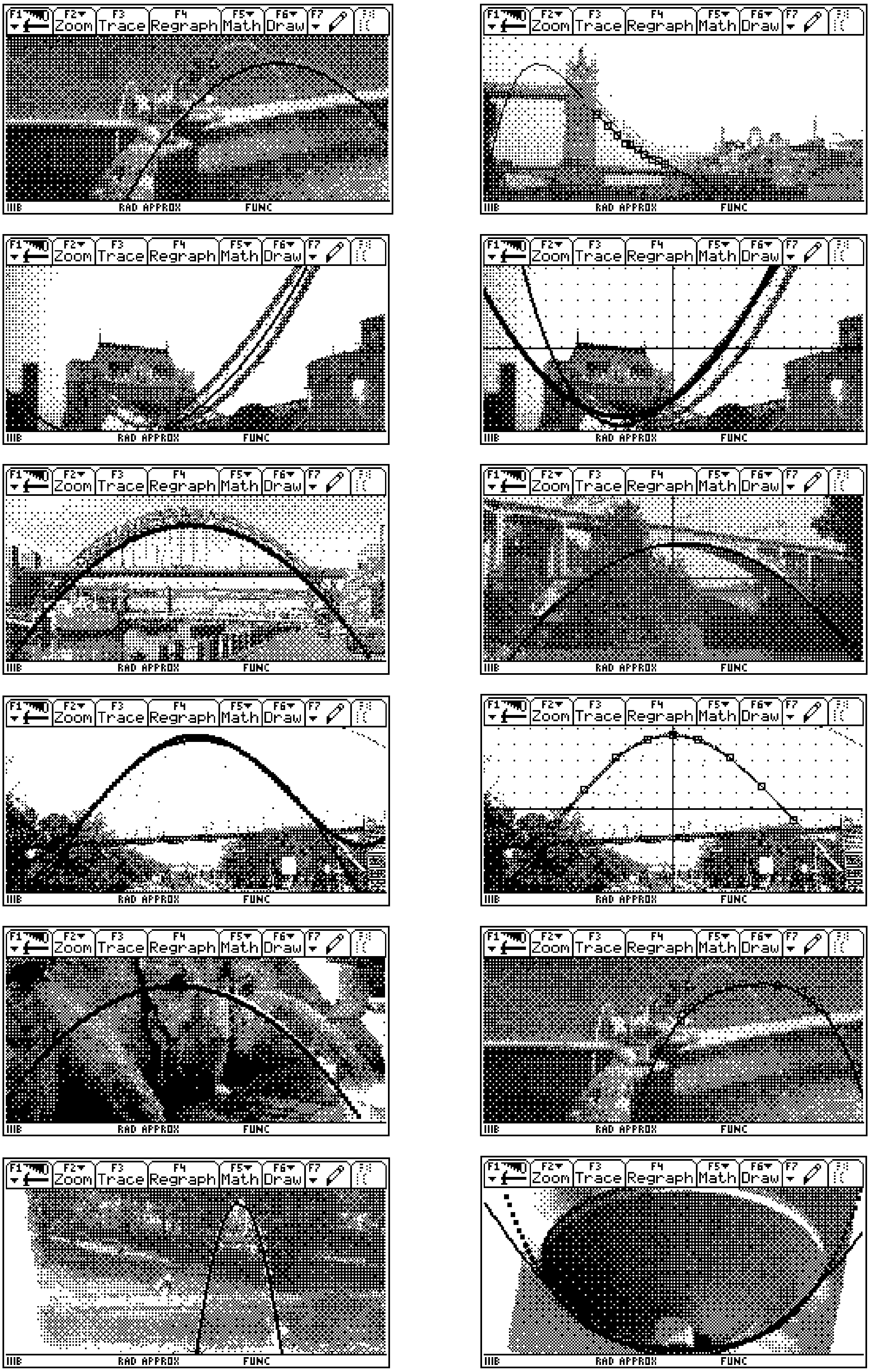
#23: $\frac{\text{ATAN}(1)}{1°} = 45$

But in DERIVE 6 you can use the ARC-functions:

#24: ARCTAN(1) = 45

See Albert Rich´s comment on page 14

Tania Koller and Students (IIIb) Model Reality on the Voyage 200



See also page 22

Quadratic Approximations for Integration

Giora Mann
Levinsky College of Education, Israel

Nurit Zehavi
Weizmann Institute of Science, Israel

Abstract

The fundamental theorem of Calculus states that the definite integral over an interval is the difference of the antiderivative at the two endpoints of the integration interval. However, we need to know the antiderivative, but in too many interesting cases (for example in computing the length of curves) the antiderivative does not exist, or we are unable to find it.

In this paper we present a didactical sequence in which we start by showing that a quadratic approximation of the function to be integrated can give a "good" approximation of the definite integral. Furthermore, this approximation depends only on three values of the integrand (no antidervative is needed). Refining the approximation opens the road to Simpson's rule for computing numerically definite integrals.

Introduction

The motivation for the didactic sequence presented in this paper originated from comments of curious high-school students while learning to compute definite integrals: "It is interesting that the definite integral for a given function depends only on the values of the primitive function at the endpoints of the integration interval; the area under the graph is not affected by the behavior of the primitive function and its derivative inside the interval of integration." This intimate connection between the definite integral and the given function is expressed in the fundamental theorem of Calculus: For $F'(x) = f(x)$, $\int_a^b f(x) = F(b) - F(a)$.

The following is a typical problem given to students: Find the area between the graph of the function $f(x) = 0.5x^3 - 2x^2 + 4$ and the x – axis, above it (see Figure 1). We demonstrate the solution and exploration of the problem using *Derive*:

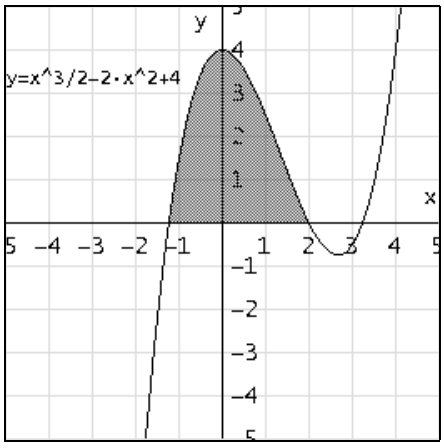


Figure 1.

#1: $f(x) := 0.5 \cdot x^3 - 2 \cdot x^2 + 4$

#2: $\text{SOLVE}(0.5 \cdot x^3 - 2 \cdot x^2 + 4 = 0,\ x)$

#3: $x = 1 - \sqrt{5} \vee x = \sqrt{5} + 1 \vee x = 2$

#4: $\int_{1-\sqrt{5}}^{2} (0.5 \cdot x^3 - 2 \cdot x^2 + 4)\, dx = \frac{5 \cdot \sqrt{5}}{3} + \frac{13}{3}$

#5: $\text{APPROX}\left(\int_{1-\sqrt{5}}^{2} (0.5 \cdot x^3 - 2 \cdot x^2 + 4)\, dx = \frac{5 \cdot \sqrt{5}}{3} + \frac{13}{3}\right) = \text{false}$

Why is *Derive*'s reaction to the Approx command - "false" ?!

Let us approximate the left side first and then the right side:

#6: $8.060113295 = \frac{5 \cdot \sqrt{5}}{3} + \frac{13}{3}$

#7: $8.060113295 = 8.060113295$

So, why "false"?

The conflict is resolved when we change the number of digits:

#8: $\text{NotationDigits} := 12$

#9: $\text{APPROX}\left(\int_{1-\sqrt{5}}^{2} (0.5 \cdot x^3 - 2 \cdot x^2 + 4)\, dx\right) = 8.06011329556$

#10: $\text{APPROX}\left(\frac{5 \cdot \sqrt{5}}{3} + \frac{13}{3}\right) = 8.06011329583$

We see that the software implements different algorithms for approximating integrals and for approximating real numbers. It is worthwhile to learn more about the algorithm for approximating integrals.

A quadratic approximation for a definite integral

Let us look at $g(x) := x^3 - 3x + 6$ and find the area circumscribed by the graph of $g(x)$, $y = 0$, $x = -2$, $x = 1$

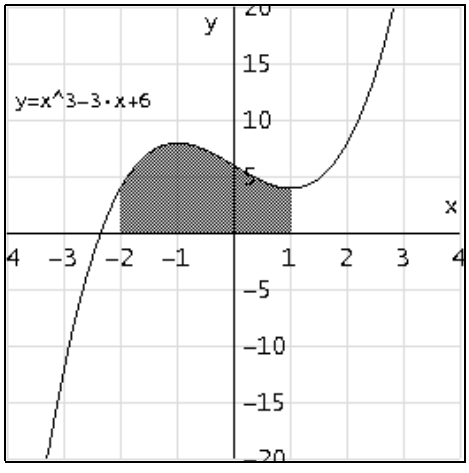


Figure 2.

We want to fit a quadratic function $q(x)$ to $g(x)$:

```
#13:  q(x) := FIT([x, A·x^2 + B·x + C], [ -2    g(-2)   ]
                                        [ -0.5  g(-0.5) ]
                                        [  1    g(1)    ])

#14:  q(x) := -3·x^2/2 - 3·x/2 + 7
```

Figure 3 indicates that (a) the graphs of the two functions in the integration interval are of different types (e.g., only $g(x)$ has an inflection point), and (b) the area under $q(x)$ seems to be the same as the area under $g(x)$.

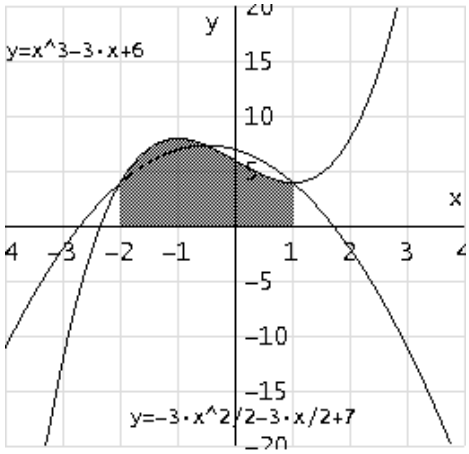


Figure 3.

And indeed,

$$\int_{-2}^{1} g(x) = 18.75 \qquad \int_{-2}^{1} q(x) = 18.75$$

This is true for any definite integral of a polynomial of the third degree. We can show that easily, but having CAS at our disposal, we prefer to present a broader framework. It is enough to deal with the simplest polynomial of n:

```
Pn(x) := A·x^n

Qn(x) := FIT([x, a·x^2 + b·x + c], [ x0 - k   A·(x0 - k)^n ]
                                   [ x0       A·x0^n       ]
                                   [ x0 + k   A·(x0 + k)^n ])

Δ(n) := ∫ from x0 - k to x0 + k Qn(x) dx - ∫ from x0 - k to x0 + k Pn(x) dx
```

Computing the difference for $n = 1 \ldots 7$ yields:

TABLE(Δ(n), n, 1, 7)

1	0
2	0
3	0
4	$\frac{4\cdot A\cdot k^5}{15}$
5	$\frac{4\cdot A\cdot k^5\cdot x0}{3}$
6	$\frac{4\cdot A\cdot k^5\cdot(2\cdot k^2+21\cdot x0^2)}{21}$
7	$\frac{4\cdot A\cdot k^5\cdot x0\cdot(2\cdot k^2+7\cdot x0^2)}{3}$

What conclusions could one make from the table? For n = 1,2 the difference between the integrals of the given polynomial and the quadratic approximation is zero, because the approximation is in fact the same polynomial. For n = 3, the fact that we get zero difference seems peculiar because we have two different integrands. The insight we gain for n = 4 is that the difference depends only on the width of the interval, and it is not affected by the location. For n > 4 the difference depends, as expected, on both the width and location of the interval.

Students may wonder why we need to approximate the definite integral of a cubic polynomial by the integral of a quadratic function. Well, let us try to find the circumference of the braid created between $y = \sin(x)$ and $y = \cos(x)$ from the first intersection point to the second to the right of the origin (Figure 4).

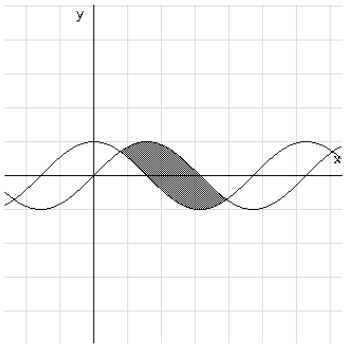


Figure 4.

The length of a curve is expressed by the formula $\int_a^b \sqrt{1+(y')^2}\,dx$.

So the circumference is twice the integral $\int_{\pi/4}^{5\pi/4} \sqrt{1+\cos^2(x)}dx$, which is not easy to compute without a CAS. We can start by approximating such integrals with a quadratic function by computing their coefficients. But this is one of those moments in mathematics when working generally enables us to see surprising results unseen otherwise!

Approximating an integral with (only) three values of the integrand

Assume that $q(x) = Ax^2 + Bx + C$ is a function that meets a given function $f(x)$ at three points: the endpoints and the midpoint of the interval $[a, b]$:

$$f(a) = q(a),\quad f(b) = q(b)\quad f((a+b)/2) = q((a+b)/2)$$

$$f(a) = A\cdot a^2 + B\cdot a + C$$

$$f(b) = A\cdot b^2 + B\cdot b + C$$

$$f\left(\frac{a+b}{2}\right) = A\cdot\left(\frac{a+b}{2}\right)^2 + B\cdot\frac{a+b}{2} + C$$

$$f\left(\frac{a+b}{2}\right) = \frac{A\cdot a^2}{4} + \frac{A\cdot a\cdot b}{2} + \frac{A\cdot b^2}{4} + \frac{B\cdot a}{2} + \frac{B\cdot b}{2} + C$$

$$\int_a^b (A\cdot x^2 + B\cdot x + C)\,dx = \frac{A\cdot(b^3 - a^3)}{3} + \frac{B\cdot(b^2 - a^2)}{2} + C\cdot(b - a)$$

$$\int_a^b (A\cdot x^2 + B\cdot x + C)\,dx = \frac{(b - a)\cdot(2\cdot A\cdot a^2 + 2\cdot A\cdot a\cdot b + 2\cdot A\cdot b^2 + 3\cdot B\cdot a + 3\cdot B\cdot b + 6\cdot C)}{6}$$

From the above we conclude that:

$$\int_a^b q(x) = \frac{b-a}{6}(f(a) + 4f(\frac{a+b}{2}) + f(b))\,.$$

The meaning of the result is that the area under the function $f(x)$ over the interval $[a, b]$ is approximately the area of a rectangle whose length is the length of the integration interval and whose height is the weighted mean (with weights 1, 4, 1) of the three values of the function at a, $(a+b)/2$, and b. As one can see, the quadratic function "disappeared". We used only its existence to obtain the approximation, which depends only on three values of the given function and the interval of integration. Therefore, we reflect again on the fundamental theorem of Calculus that requires the values of the primitive function at the endpoints of the integration interval. Clearly, if we are unable to find a primitive function, we can easily get a reasonable approximation of the integral, using only the values of the integrand at the endpoints and the midpoint of interval.

Toward Simpson's rule

We come back to the approximation of the integral needed for finding the circumference of the braid: $\int_{\pi/4}^{5\pi/4} \sqrt{1+\cos^2(x)}\,dx$. Using the last result for $h(x) := \sqrt{1+\cos^2(x)}$, we approximate $\int_{\pi/4}^{5\pi/4} h(x)dx$ by:

$$\text{APPROX}\left(\frac{1}{6}\cdot\left(\frac{5\cdot\pi}{4} - \frac{\pi}{4}\right)\cdot\left(h\left(\frac{\pi}{4}\right) + 4\cdot h\left(\frac{3\cdot\pi}{4}\right) + h\left(\frac{5\cdot\pi}{4}\right)\right)\right) = 3.84764949045$$

$$\text{APPROX}\left(\frac{1}{6}\cdot\left(h\left(\frac{\pi}{4}\right) + 4\cdot h\left(\frac{3\cdot\pi}{4}\right) + h\left(\frac{5\cdot\pi}{4}\right)\right)\right) = 1.22474487139$$

The first result is $\frac{\pi\sqrt{6}}{2}$, and the second $\frac{\sqrt{6}}{2}$. Figure 5 illustrates the results visually.

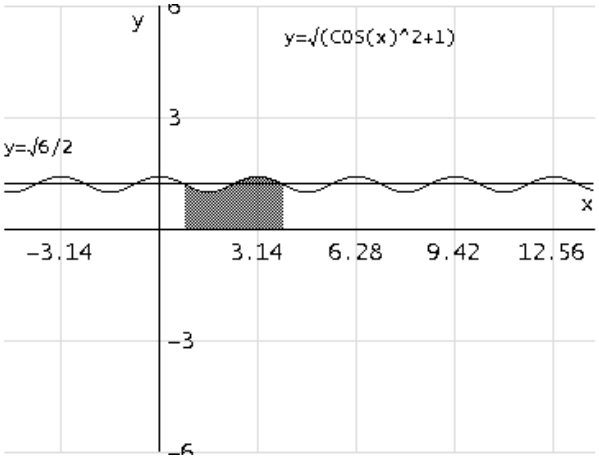


Figure 5.

$$\text{APPROX}\left(\int_{\pi/4}^{5\cdot\pi/4} \sqrt{(1+\text{COS}(x)^2)}\,dx\right) = 3.82019778897$$

Using *Derive*'s numerical integration, we get some control of our result:

Obviously if we want a better approximation, we should divide the integration interval into sub-intervals and approximate the curve by parabolas. This is in fact the idea underlying Simpson's method.

We proceed by dividing the interval into two sub-intervals:

$$\int_{\pi/4}^{5\cdot\pi/4} h(x)\,dx = \int_{\pi/4}^{3\cdot\pi/4} h(x)\,dx + \int_{3\cdot\pi/4}^{5\cdot\pi/4} h(x)\,dx$$

Applying the method, we get for each sub-interval:

$$\int_{\pi/4}^{5\cdot\pi/4} h(x)\,dx = \int_{\pi/4}^{3\cdot\pi/4} h(x)\,dx + \int_{3\cdot\pi/4}^{5\cdot\pi/4} h(x)\,dx$$

$$\int_{\pi/4}^{3\cdot\pi/4} h(x)\,dx = \frac{1}{6}\cdot\left(\frac{3\cdot\pi}{4}-\frac{\pi}{4}\right)\cdot\left(h\left(\frac{\pi}{4}\right)+4\cdot h\left(\frac{\pi}{2}\right)+h\left(\frac{3\cdot\pi}{4}\right)\right)$$

$$\int_{\pi/4}^{3\cdot\pi/4} h(x)\,dx = 1.68847246628$$

$$\int_{3\cdot\pi/4}^{5\cdot\pi/4} h(x)\,dx = \frac{1}{6}\cdot\left(\frac{5\cdot\pi}{4}-\frac{3\cdot\pi}{4}\right)\cdot\left(h\left(\frac{3\cdot\pi}{4}\right)+4\cdot h(\pi)+h\left(\frac{5\cdot\pi}{4}\right)\right)$$

$$\int_{3\cdot\pi/4}^{5\cdot\pi/4} h(x)\,dx = 2.12223589443$$

Combining the above we get the following approximation:

$$\int_{\pi/4}^{5\cdot\pi/4} h(x)\,dx = 1.68847246628 + 2.12223589443$$

$$\int_{\pi/4}^{5\cdot\pi/4} h(x)\,dx = 3.81070836071$$

Doubling the number of intervals yields a better approximation:

$$\int_{\pi/4}^{5\cdot\pi/4} h(x)\,dx = \frac{1}{24}\cdot\left(\frac{5\cdot\pi}{4} - \frac{\pi}{4}\right)\cdot\left(h\left(\frac{\pi}{4}\right) + 4\cdot h\left(\frac{3\cdot\pi}{8}\right) + 2\cdot h\left(\frac{\pi}{2}\right) + 4\cdot h\left(\frac{5\cdot\pi}{8}\right) + 2\cdot h\left(\frac{3\cdot\pi}{4}\right) + 4\cdot h\left(\frac{7\cdot\pi}{8}\right) + 2\cdot h(\pi) + 4\cdot h\left(\frac{9\cdot\pi}{8}\right) + h\left(\frac{5\cdot\pi}{4}\right)\right)$$

$$\int_{\pi/4}^{5\cdot\pi/4} h(x)\,dx = 3.82028240611$$

Comparing the three results that we got by applying the method of quadratic approximation with the numerical approximation by *Derive* shows that the greater the number of sub-intervals, the better the approximation that we obtain:

$$\text{APPROX}(|3.84764949045 - 3.82019778897|) = 0.027451701471694116$$

$$\text{APPROX}(|3.81070836071 - 3.82019778897|) = 0.009489428267303327 2$$

$$\text{APPROX}(|3.82028240611 - 3.82019778897|) = 8.461713328892798 9\cdot 10^{-5}$$

Approximating integrals by *Derive*

This may be a good time to tell students that *Derive* actually uses an adaptation of Simpson's rule to numerically approximate definite integrals.

We previously saw that increasing the number of sub-interval increases the accuracy. Simpson's estimate of $\int_a^b f(x)dx$, when the interval is divided into n sub-intervals of equal length $h = \frac{(b-a)}{2n}$, is the sum:

$$\frac{h}{3}[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \ldots 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)],$$

where the error is at most $\frac{(b-a)M_4h^4}{180}$; M_4 denotes the maximum value of $|f^4(x)|$ for x in [a, b].

Simpson's estimate is exact for polynomials of the form $y = a\,x^3 + b\,x^2 + c\,x + d$, whose fourth derivative is zero. The error in using Simpson's methods for other functions involves the fourth derivative.

We now reflect on the expressions of error for $y = Ax^n$, $n < 5$ that we obtained when we computed the error for polynomials.

For $n < 4$ we got zeros, which agrees with the fact that M^4 is zero. For $n = 4$ we got $\frac{4Ak^5}{15}$.

We substitute in $\frac{(b-a)M_4h^4}{180}$: $M_4 = 4!\cdot A,\ b-a = 2k,\ h = k$ and get:

$$\frac{2k\cdot 4!\cdot A\cdot k^4}{180} = \frac{4A\cdot k^5}{15}.$$

The significance of the last result is that our error for the polynomial $y = Ax^4$ is exactly the accuracy allotted by Simpson's rule.

Concluding remarks

The didactical sequence we described above utilizes CAS to make numerical integration by CAS less mysterious. We start by a 'didactic moment' in which students obtain two different results in using the software to approximate definite integrals. Next they realize that for a quadratic approximation, the coefficients of the quadratic function are not needed; the approximation is produced by using only three values of the integrand. At this stage the road to Simpson's method is open; moreover when we mention the accuracy of the method the CAS can be used to get 'some feeling' of the key indicator of the error.

Albert Rich´s comment on Angular Mode Settings

Q: When calling on trig functions, how do I enter angles in degrees?

A: In Derive 6, the ° operator is used to enter an angle in degrees. The ° operator can be entered by clicking on it on the math symbol toolbar, pressing Ctrl+O, or by typing deg on the expression entry line. For example, SIN(45°) simplifies to SQRT(2)/2. Unlike earlier versions of Derive, selecting Degree in the Angular Unit field of the Simplification tab of the Options > Mode Settings command only effects the display of angles, not how angles are entered.

Q: In approximate mode, how do I get the inverse trig functions to return angles in degrees instead of radians?

A: In approximate mode, the built-in inverse trig functions (e.g. ASIN, ACOS, ATAN, etc.) always return angles in radians, even in Degree mode. For example, in Degree mode ATAN(1) simplifies to 45° but approximates to 0.7853981633. To always get angles returned in degrees use the inverse trig functions (e.g. ARCSIN, ARCCOS, ARCTAN, etc.) defined in MiscellaneousFunctions.mth instead of the built-in functions. For example, ARCTAN(1) simplifies and approximates to 45.

==========

Note the new inverse trig functions: ARCSIN, ARCCOS, etc.

Applications of the Moore-Penrose Inverse of a Matrix

Karsten Schmidt, FH Schmalkalden, Germany, kschmidt@fh-sm.de

Introduction

After giving an introduction to the Moore-Penrose inverse of a matrix, and its computation in *DERIVE*, in DNL #50 (Schmidt 2003), this paper deals with two important applications of the Moore-Penrose inverse. One is a method for solving a system of linear equations, and the other is the computation of the Ordinary Least Squares estimator in linear regression. Some familiarity with matrix algebra as well as basic understanding of the Moore-Penrose inverse of a matrix (as provided in DNL #50) is required.

Computation and Properties of the Moore-Penrose Inverse

In order to facilitate working with this paper, the definition and *DERIVE* functions for the computation of the Moore-Penrose inverse of a matrix are repeated from DNL #50:
For any $m \times n$ -matrix $\boldsymbol{A}$ there exists a unique matrix with properties related to those of the inverse of a nonsingular matrix. This is the Moore-Penrose inverse, denoted by $\boldsymbol{A}^+$, which satisfies the four conditions (the transpose of $\boldsymbol{A}$ is denoted by $\boldsymbol{A}'$)

$$\boldsymbol{A}\boldsymbol{A}^+\boldsymbol{A} = \boldsymbol{A} \tag{1}$$

$$\boldsymbol{A}^+\boldsymbol{A}\boldsymbol{A}^+ = \boldsymbol{A}^+ \tag{2}$$

$$\left(\boldsymbol{A}^+\boldsymbol{A}\right)' = \boldsymbol{A}^+\boldsymbol{A} \tag{3}$$

$$\left(\boldsymbol{A}\boldsymbol{A}^+\right)' = \boldsymbol{A}\boldsymbol{A}^+ \tag{4}$$

Conditions (3) and (4) require both $\boldsymbol{A}^+\boldsymbol{A}$ and $\boldsymbol{A}\boldsymbol{A}^+$ to be symmetric matrices. Note that $\boldsymbol{A}^+$ is an $n \times m$ -matrix, i.e. the dimension of $\boldsymbol{A}^+$ is equal to the dimension of $\boldsymbol{A}'$.
The Moore-Penrose inverse of a matrix can be computed in *DERIVE* with the following two functions:

```
MPIV(a) :=
  If DIM(a') = 1
     If (a'·a)↓1↓1 = 0
        0·a'
        a'/(a'·a)↓1↓1
     "This is not a column vector!"

MPI(A, APLUS, aj, dt, c, bt, J) :=
  Prog
    APLUS := MPIV(A COL [1])
    J := 2
    Loop
      If J > DIM(A')
         RETURN APLUS
      aj := A COL [J]
      dt := aj'·APLUS'·APLUS
      c := (IDENTITY_MATRIX(DIM(A)) - A COL [1, ..., J - 1]·APLUS)·aj
      bt := MPIV(c) + (1 - MPIV(c)·c)/(1 + dt·aj)·dt
      APLUS := APPEND(APLUS - APLUS·aj·bt, bt)
      J :+ 1
```

`MPIV` computes the Moore-Penrose inverse of a vector and `MPI` the Moore-Penrose inverse of a matrix (or vector). Note that `MPIV` requires a column vector passed as parameter, which has to be declared in *DERIVE* as a matrix with one column. Note also that `MPIV` and, therefore, `MPI`, via calling `MPIV` repeatedly, might not be able to compute the Moore-Penrose inverse since it might be impossible to determine if $\boldsymbol{a} = \boldsymbol{0}$, when $\boldsymbol{a}$ has nonnumeric entries. Both functions, along with a couple more from the next section, are provided in the utility file **MP.mth**.

Among the many properties that hold for the Moore-Penrose inverse the following three will be useful later in this paper ($\boldsymbol{I}$ denotes the identity matrix):

$$\left(\boldsymbol{A}'\boldsymbol{A}\right)^{+}\boldsymbol{A}' = \boldsymbol{A}^{+} \tag{5}$$

$$\boldsymbol{A}'\boldsymbol{A}\boldsymbol{A}^{+} = \boldsymbol{A}' \tag{6}$$

$$\operatorname{rank}\left(\underset{m\times n}{\boldsymbol{A}}\right) = n \quad \Leftrightarrow \quad \boldsymbol{A}^{+} = \left(\boldsymbol{A}'\boldsymbol{A}\right)^{-1}\boldsymbol{A}' \quad \text{and} \quad \boldsymbol{A}^{+}\boldsymbol{A} = \underset{n\times n}{\boldsymbol{I}} \tag{7}$$

Application to Systems of Linear Equations

We consider a system of linear equations (SLE)

$$\underset{m\times n}{\boldsymbol{A}}\ \underset{n\times 1}{\boldsymbol{x}} = \underset{m\times 1}{\boldsymbol{b}}$$

where $\boldsymbol{A}$ is the known coefficient matrix, $\boldsymbol{b}$ a vector of known constants, and $\boldsymbol{x}$ a vector of unknown variables.

The Moore-Penrose inverse of $\boldsymbol{A}$ can be applied to such a system

- to check if it is *consistent* or *inconsistent*, i.e. to find out if it has solutions or not, and
- if it is consistent, to provide the general solution, which may consist of either one unique or an infinite number of solutions.

A system of linear equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ is consistent if and only if

$$\boldsymbol{A}\boldsymbol{A}^{+}\boldsymbol{b} = \boldsymbol{b} \tag{8}$$

As an example, consider an SLE defined by

$$\underset{2\times 2}{\boldsymbol{A}} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix};\ \underset{2\times 1}{\boldsymbol{b}} = \begin{pmatrix} 5 \\ 10 \end{pmatrix} \tag{9}$$

The Moore-Penrose inverse of $\boldsymbol{A}$ is

$$\boldsymbol{A}^{+} = \begin{pmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{pmatrix}$$

Since $\boldsymbol{A}$ is nonsingular ($\det\left(\boldsymbol{A}\right) = -\frac{1}{2} \neq 0$) we have $\boldsymbol{A}^{+} = \boldsymbol{A}^{-1}$. Hence

$$\boldsymbol{A}\boldsymbol{A}^{+}\boldsymbol{b} = \underbrace{\boldsymbol{A}\boldsymbol{A}^{-1}}_{\boldsymbol{I}}\boldsymbol{b} = \boldsymbol{b}$$

for any vector $\boldsymbol{b}$. System (9), like any other system with a nonsingular coefficient matrix $\boldsymbol{A}$, is therefore consistent according to (8).

As another example, consider

$$\underset{2\times 2}{\boldsymbol{A}} = \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix};\ \underset{2\times 1}{\boldsymbol{b}} = \begin{pmatrix} 5 \\ 10 \end{pmatrix} \tag{10}$$

This time $\boldsymbol{A}$ is singular ($\det(\boldsymbol{A})=0$), its inverse $\boldsymbol{A}^{-1}$ does not exist. Computing the Moore-Penrose inverse

$$\boldsymbol{A}^{+}=\begin{pmatrix}\frac{1}{25} & \frac{2}{25}\\ \frac{2}{25} & \frac{4}{25}\end{pmatrix}$$

is nevertheless possible and we find that condition (8) is satisfied for system (10):

$$\boldsymbol{A}\boldsymbol{A}^{+}\boldsymbol{b}=\begin{pmatrix}1 & 2\\ 2 & 4\end{pmatrix}\begin{pmatrix}\frac{1}{25} & \frac{2}{25}\\ \frac{2}{25} & \frac{4}{25}\end{pmatrix}\begin{pmatrix}5\\ 10\end{pmatrix}=\begin{pmatrix}\frac{1}{5} & \frac{2}{5}\\ \frac{2}{5} & \frac{4}{5}\end{pmatrix}\begin{pmatrix}5\\ 10\end{pmatrix}=\begin{pmatrix}5\\ 10\end{pmatrix}=\boldsymbol{b}$$

As a third example, look at

$$\underset{2\times 2}{\boldsymbol{A}}=\begin{pmatrix}1 & 2\\ 2 & 4\end{pmatrix};\ \underset{2\times 1}{\boldsymbol{b}}=\begin{pmatrix}5\\ 15\end{pmatrix} \tag{11}$$

This time we find that condition (8) is not satisfied:

$$\boldsymbol{A}\boldsymbol{A}^{+}\boldsymbol{b}=\begin{pmatrix}\frac{1}{5} & \frac{2}{5}\\ \frac{2}{5} & \frac{4}{5}\end{pmatrix}\begin{pmatrix}5\\ 15\end{pmatrix}=\begin{pmatrix}7\\ 14\end{pmatrix}\neq\boldsymbol{b}$$

System (11) is therefore inconsistent.

```
      CHECKSLE(A, b) :=
        If A·MPI(A)·b = b
#1:        "consistent"
           "NO SOLUTIONS!"

#2:   A := [ 1  2 ]
           [ 3  4 ]

#3:   b := [  5 ]
           [ 10 ]

#4:                    CHECKSLE(A, b) = consistent

#5:   A := [ 1  2 ]
           [ 2  4 ]

#6:                    CHECKSLE(A, b) = consistent

#7:   b := [  5 ]
           [ 15 ]

#8:                 CHECKSLE(A, b) = NO SOLUTIONS!

#9:   b := [  5 ]
           [ λ ]

#10:     CHECKSLE(A, b) = IF(λ = 10, consistent, NO SOLUTIONS!)
```

The function `CHECKSLE` in the above screenshot checks if a system of linear equations is consistent or not, and prints the result on the screen. Since there is no *unknown* clause in the `IF`-expression, the entire (simplified) `IF`-expression is returned, which can obviously be fairly informative in cases such as the last SLE (consisting of matrix $\boldsymbol{A}$ in #5 and vector $\boldsymbol{b}$ in #9).

If a system of linear equations $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ is consistent, its general solution is given by

$$\boldsymbol{x}=\boldsymbol{A}^{+}\boldsymbol{b}+\left(\underset{n\times n}{\boldsymbol{I}}-\boldsymbol{A}^{+}\boldsymbol{A}\right)\underset{n\times 1}{\boldsymbol{z}} \tag{12}$$

where $\boldsymbol{z}$ is an arbitrary vector.

Note that since the vector $z \in \mathbb{R}^n$ in (12) is arbitrary, we might simply choose $z = 0$. Consequently, one (possibly unique) solution of a consistent system of linear equations $Ax = b$ is always given by

$$x = A^+ b$$

The following function SOLVESLE either solves a system of linear equations $Ax = b$ where the matrix A and the vector b have been passed as parameters, or displays a message if the system is inconsistent.

```
z := VECTOR(VECTOR(APPEND(z, J), i, 1), J, 1, DIM(A'))

SOLVESLE(A, b) :=
  If A • MPI(A) • b = b
     MPI(A) • b + (IDENTITY_MATRIX(DIMENSION(A')) – MPI(A) • A) • z
     "No solution(s)!"
```

We now want to compute the solution(s) of the above three systems. We start with system (9). Since A is a nonsingular matrix, we have $A^+ = A^{-1}$, and (12) simplifies to

$$\begin{aligned} x &= A^+ b + \left(I - A^+ A\right) z \\ &= A^{-1} b + \left(I - \underbrace{A^{-1} A}_{I}\right) z \\ &= A^{-1} b \end{aligned}$$

for any choice of z. Obviously, the general solution (12) simplifies to a unique solution if A is nonsingular. Hence

$$x = A^{-1} b = \begin{pmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{pmatrix} \begin{pmatrix} 5 \\ 10 \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{5}{2} \end{pmatrix}$$

For system (10) we get

$$x = A^+ b + \left(I - A^+ A\right) z = \begin{pmatrix} \frac{4}{5} z_1 - \frac{2}{5} z_2 + 1 \\ -\frac{2}{5} z_1 + \frac{1}{5} z_2 + 2 \end{pmatrix}$$

In this case there is an infinite number of solutions. For example, by choosing $z = 0$ we get

$$x = A^+ b = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

```
#1:  A := [ 1  2 ]
          [ 3  4 ]

#2:  b := [  5 ]
          [ 10 ]

#3:                    SOLVESLE(A, b) = [  0 ]
                                        [  5 ]
                                        [ --- ]
                                        [  2 ]

#4:  A := [ 1  2 ]
          [ 2  4 ]

#5:                    SOLVESLE(A, b) = [ 4·z1   2·z2          ]
                                        [ ---- - ---- + 1       ]
                                        [  5      5             ]
                                        [                       ]
                                        [ 2·z1    z2            ]
                                        [ - ---- + ---- + 2     ]
                                        [  5      5             ]

#6:                                     [ 1 ]
                                        [   ]
                                        [ 2 ]

#7:  b := [  5 ]
          [ 15 ]

#8:                    SOLVESLE(A, b) = No solution(s)!
```

The above screenshot shows the capability of the function SOLVESLE to handle all three possible scenarios in considering a system of linear equations: a unique solution, an infinite number of solutions, and the case that no solution exists.

Linear Regression and the Moore-Penrose Inverse

We consider the (multiple) linear regression model

$$\underset{N\times 1}{\boldsymbol{y}} = \underset{N\times K}{\boldsymbol{X}}\ \underset{K\times 1}{\boldsymbol{\beta}} + \underset{N\times 1}{\boldsymbol{u}} \qquad (13)$$

where $\boldsymbol{y}$ is the vector of observations on the dependent variable, $\boldsymbol{X}$ the regressor matrix, $\boldsymbol{\beta}$ a vector of parameters, and $\boldsymbol{u}$ a vector of disturbances.

Denoting an estimator of the unknown parameter vector $\boldsymbol{\beta}$ by $\tilde{\boldsymbol{\beta}}$, we have

$$\tilde{\boldsymbol{y}} = \boldsymbol{X}\tilde{\boldsymbol{\beta}}$$
$$\tilde{\boldsymbol{u}} = \boldsymbol{y} - \tilde{\boldsymbol{y}}$$

where $\tilde{\boldsymbol{y}}$ is the estimate of $\boldsymbol{y}$ using $\tilde{\boldsymbol{\beta}}$, and $\tilde{\boldsymbol{u}}$ is the vector of residuals.

The most popular estimator for $\boldsymbol{\beta}$ is the (Ordinary) Least Squares estimator which minimizes the sum of squared residuals

$$\begin{aligned}\varphi\left(\tilde{\boldsymbol{\beta}}\right) &= \sum_{i=1}^{N}\tilde{u}_i^{\,2}\\ &= \tilde{\boldsymbol{u}}'\tilde{\boldsymbol{u}}\\ &= \left(\boldsymbol{y}-\boldsymbol{X}\tilde{\boldsymbol{\beta}}\right)'\left(\boldsymbol{y}-\boldsymbol{X}\tilde{\boldsymbol{\beta}}\right) \to \min_{\tilde{\boldsymbol{\beta}}}\end{aligned}$$

Note that

$$\begin{aligned}\varphi\left(\tilde{\boldsymbol{\beta}}\right) &= \left(\boldsymbol{y}-\boldsymbol{X}\tilde{\boldsymbol{\beta}}\right)'\left(\boldsymbol{y}-\boldsymbol{X}\tilde{\boldsymbol{\beta}}\right)\\ &= \boldsymbol{y}'\boldsymbol{y} - \boldsymbol{y}'\boldsymbol{X}\tilde{\boldsymbol{\beta}} - \tilde{\boldsymbol{\beta}}'\boldsymbol{X}'\boldsymbol{y} + \tilde{\boldsymbol{\beta}}'\boldsymbol{X}'\boldsymbol{X}\tilde{\boldsymbol{\beta}}\\ &= \tilde{\boldsymbol{\beta}}'\boldsymbol{X}'\boldsymbol{X}\tilde{\boldsymbol{\beta}} - 2\boldsymbol{y}'\boldsymbol{X}\tilde{\boldsymbol{\beta}} + \boldsymbol{y}'\boldsymbol{y}\end{aligned}$$

is a convex function since $\boldsymbol{X}'\boldsymbol{X}$ is a nonnegative definite matrix. Therefore, finding its first derivative

$$\begin{aligned}\frac{\partial\varphi\left(\tilde{\boldsymbol{\beta}}\right)}{\partial\tilde{\boldsymbol{\beta}}} &= \tilde{\boldsymbol{\beta}}'\left(\boldsymbol{X}'\boldsymbol{X} + \left(\boldsymbol{X}'\boldsymbol{X}\right)'\right) - 2\boldsymbol{y}'\boldsymbol{X}\\ &= 2\tilde{\boldsymbol{\beta}}'\boldsymbol{X}'\boldsymbol{X} - 2\boldsymbol{y}'\boldsymbol{X}\end{aligned}$$

and setting it equal to $\mathbf{0}$ is necessary and sufficient to determine the minimum of $\varphi\left(\tilde{\boldsymbol{\beta}}\right)$:

$$2\tilde{\boldsymbol{\beta}}'\boldsymbol{X}'\boldsymbol{X} - 2\boldsymbol{y}'\boldsymbol{X} = \underset{1\times K}{\mathbf{0}} \quad\Leftrightarrow\quad \boldsymbol{X}'\boldsymbol{X}\tilde{\boldsymbol{\beta}} - \boldsymbol{X}'\boldsymbol{y} = \underset{K\times 1}{\mathbf{0}} \quad\Leftrightarrow\quad \boldsymbol{X}'\boldsymbol{X}\tilde{\boldsymbol{\beta}} = \boldsymbol{X}'\boldsymbol{y}$$

The last equation constitutes the so-called *system of normal equations*.

Under the (usual) assumption that $\operatorname{rank}\left(\boldsymbol{X}\right) = K$, which assures that $\boldsymbol{X}'\boldsymbol{X}$ is nonsingular, we can easily derive the Least Squares estimator from the normal equations

$$\boldsymbol{X}'\boldsymbol{X}\tilde{\boldsymbol{\beta}} = \boldsymbol{X}'\boldsymbol{y} \quad\Leftrightarrow\quad \underbrace{\left(\boldsymbol{X}'\boldsymbol{X}\right)^{-1}\boldsymbol{X}'\boldsymbol{X}}_{\boldsymbol{I}}\tilde{\boldsymbol{\beta}} = \left(\boldsymbol{X}'\boldsymbol{X}\right)^{-1}\boldsymbol{X}'\boldsymbol{y} \quad\Leftrightarrow\quad \tilde{\boldsymbol{\beta}} = \left(\boldsymbol{X}'\boldsymbol{X}\right)^{-1}\boldsymbol{X}'\boldsymbol{y}$$

One might think that the system of normal equations is inconsistent if $\operatorname{rank}\left(\boldsymbol{X}\right) < K$. However, this is not true.

Observe that the system of normal equations is essentially a system of linear equations in the notation of the previous section:

$$\underbrace{\boldsymbol{X}'\boldsymbol{X}}_{\boldsymbol{A}}\underbrace{\tilde{\boldsymbol{\beta}}}_{\boldsymbol{x}} = \underbrace{\boldsymbol{X}'\boldsymbol{y}}_{\boldsymbol{b}}$$

Using properties (5) and (6) of the Moore-Penrose inverse, it can be shown that the system of normal equations is consistent without any rank assumption on $\boldsymbol{X}$:

$$\boldsymbol{A}\boldsymbol{A}^{+}\boldsymbol{b} = \boldsymbol{b} \quad \Rightarrow$$

$$\boldsymbol{X}'\boldsymbol{X}\underbrace{\left(\boldsymbol{X}'\boldsymbol{X}\right)^{+}\boldsymbol{X}'}_{\boldsymbol{X}^{+}}\boldsymbol{y} = \underbrace{\boldsymbol{X}'\boldsymbol{X}\boldsymbol{X}^{+}}_{\boldsymbol{X}'}\boldsymbol{y} = \boldsymbol{X}'\boldsymbol{y}$$

Hence, its general solution is given by

$$\boldsymbol{x} = \boldsymbol{A}^{+}\boldsymbol{b} + \left(\boldsymbol{I} - \boldsymbol{A}^{+}\boldsymbol{A}\right)\boldsymbol{z} \quad \Rightarrow$$

$$\begin{aligned}\tilde{\boldsymbol{\beta}} &= \underbrace{\left(\boldsymbol{X}'\boldsymbol{X}\right)^{+}\boldsymbol{X}'}_{\boldsymbol{X}^{+}}\boldsymbol{y} + \left(\boldsymbol{I} - \underbrace{\left(\boldsymbol{X}'\boldsymbol{X}\right)^{+}\boldsymbol{X}'}_{\boldsymbol{X}^{+}}\boldsymbol{X}\right)\boldsymbol{z} \\ &= \boldsymbol{X}^{+}\boldsymbol{y} + \left(\boldsymbol{I} - \boldsymbol{X}^{+}\boldsymbol{X}\right)\boldsymbol{z}\end{aligned}$$

where $\boldsymbol{z} \in \mathbb{R}^{K}$ is an arbitrary vector.

The number of solutions, however, depends on the rank of the regressor matrix. If $\operatorname{rank}\left(\boldsymbol{X}\right) = K$, it follows from (7) that $\boldsymbol{X}^{+}\boldsymbol{X} = \boldsymbol{I}$, and the general solution simplifies to the unique solution

$$\begin{aligned}\tilde{\boldsymbol{\beta}} &= \boldsymbol{X}^{+}\boldsymbol{y} + \left(\boldsymbol{I} - \underbrace{\boldsymbol{X}^{+}\boldsymbol{X}}_{\boldsymbol{I}}\right)\boldsymbol{z} \\ &= \boldsymbol{X}^{+}\boldsymbol{y}\end{aligned}$$

i.e. the Least Squares estimator is simply the product of the Moore-Penrose inverse of the regressor matrix and the vector of the observations on the dependent variable.

If, however, $\operatorname{rank}\left(\boldsymbol{X}\right) < K$, we have an infinite number of solutions. Therefore, it is not the consistency of the system of normal equations that is guaranteed by assuming $\boldsymbol{X}$ to be of full column rank, but the uniqueness of its solution.

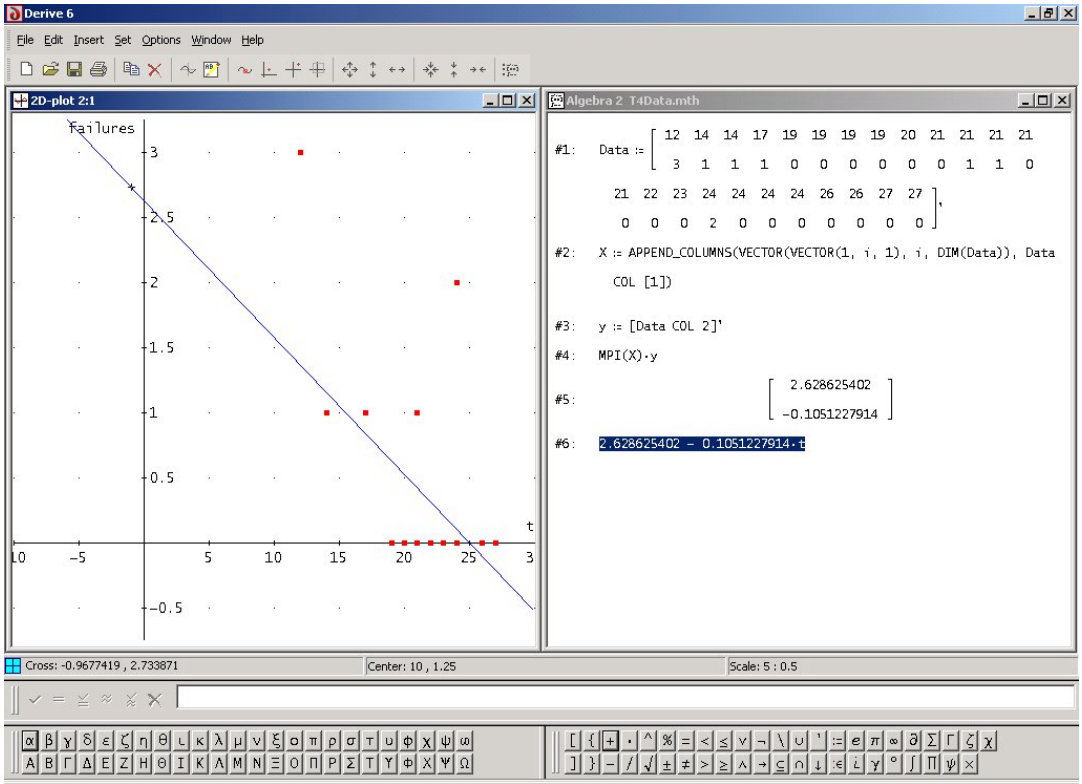
Finally, this straightforward method of computing the Least Squares estimator is demonstrated by means of an example. We want to apply linear regression analysis to predict the number of O-ring failures to be expected when the space shuttle *Challenger* was launched on January 28, 1986.

O-ring failure is when an O-ring, which seals the gaps between the parts of the solid fuel rocket motors, leaks. There had been 24 previous space shuttle launches. During 17 of them no O-ring failure occurred, while during the remaining 7 launches there were between one and three O-ring failures. The table below provides the number of failures and the ambient temperatures before launch, sorted according to temperature.

failures	*t [temp. in °C]*	*temp. in °F*	*failures*	*t [temp. in °C]*	*temp. in °F*
3	12	*53*	0	21	*70*
1	14	*57*	0	21	*70*
1	14	*58*	0	22	*72*
1	17	*63*	0	23	*73*
0	19	*66*	2	24	*75*
0	19	*67*	0	24	*75*
0	19	*67*	0	24	*76*
0	19	*67*	0	24	*76*
0	20	*68*	0	26	*78*
0	21	*69*	0	26	*79*
1	21	*70*	0	27	*80*
1	21	*70*	0	27	*81*

The following screenshot shows an algebra window and a 2D-plot window. The first three expressions in the algebra window are the contents of the file **T4Data.mth**. Expression #1 defines a 24×2-matrix `Data`, which was entered in transposed form to save space (unfortunately, this requires simplification of `Data` prior to plotting the points in the 2D-plot window). In expressions #2 and #3 the data are rearranged according to the definition of the linear regression model (13). `X` denotes the regressor matrix, containing a column of ones (for the *y*-intercept), and a column with the observations on the independent variable (temperature in Celsius), `y` denotes the vector of observations on the dependent variable (number of failures).

Expression #4 is the formula for the computation of the Least Squares estimator using the Moore-Penrose inverse. Approximating #4 yields #5, which is in turn used to define and finally plot the straight line which is the result of the Least Squares estimation.



Considering the relatively high coefficient of determination ($R^2 = 0.3$), and the fact that the slope parameter is statistically significant (at the 1%-level; both values not shown in the screenshot), the above result is fairly reliable.

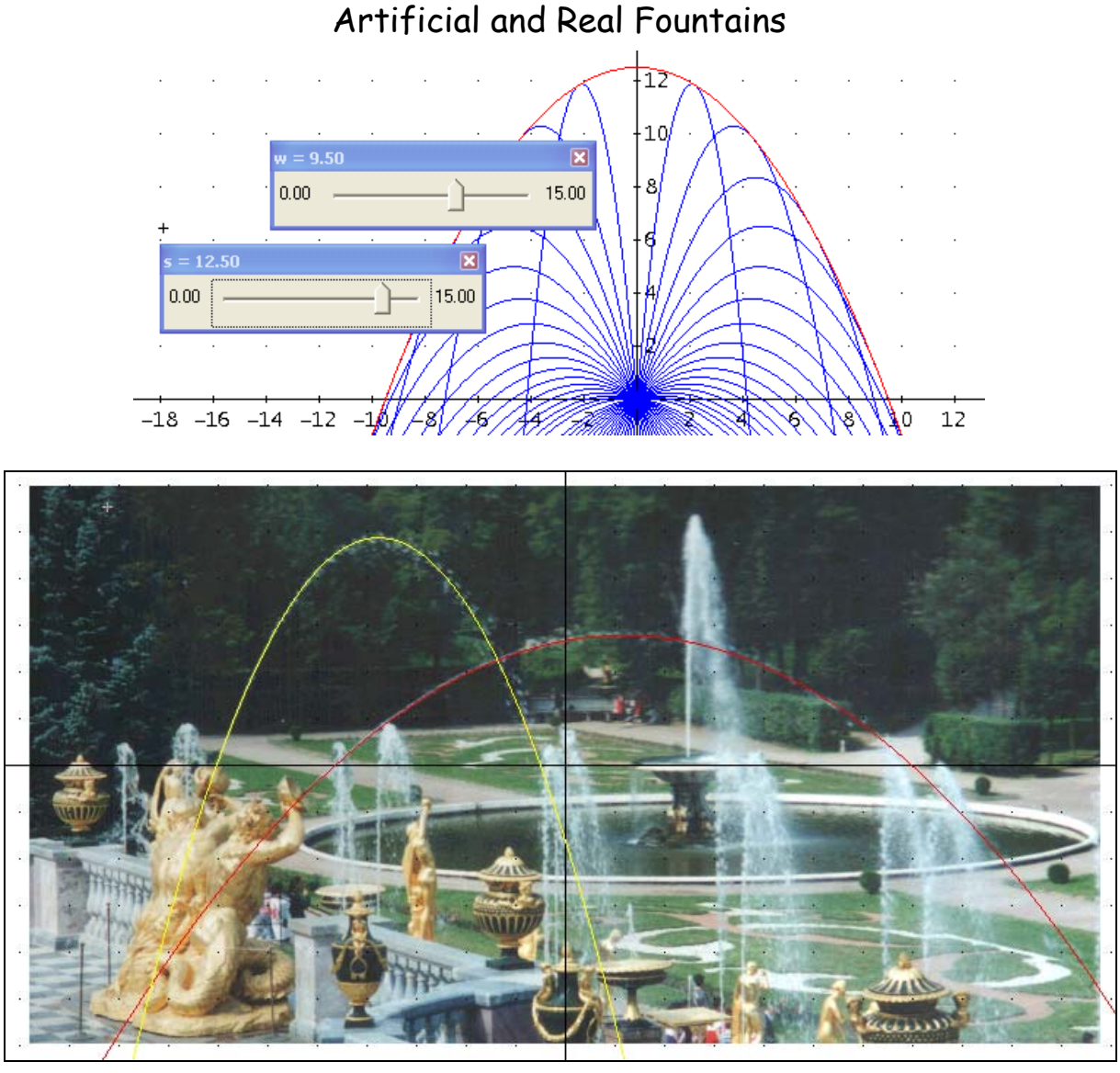
Since the pre-launch ambient temperature on January 28, 1986, was $-1\,°C$ (31°F), the prediction from the above regression would have been

$$failures = 2.629 - 0.105(-1) = 2.734$$

i.e. 2 or 3 O-ring failures were to be expected according to our regression result. Nevertheless, the space shuttle was launched. Less than two minutes into the flight, due to O-ring failure, leaking fuel was ignited by a rocket engine, and *Challenger* exploded.

Reference

Schmidt, K. (2003), An Introduction to the Moore-Penrose Inverse of a Matrix, *The DERIVE-Newsletter #50* (June 2003), 12 – 18.



Picture: Fountain in St. Petersburg, Russia (Tania Koller)

Kvadratisk programmering

Et eksempel på et valgfrit emne i symbolsk matematik

Quadratic Programming

An Example for an Optional Topic in Symbolic Mathematics

Bjoern Felsager, Denmark

Netop fordi symbolske programmer arbejder med vilkårlige udtryk kan man lige så nemt lave kvadratisk programmering, som lineær programmering (eller en hvilken som helst anden form for programmering!). Det er jo alligevel de samme faciliteter, man trækker på! Her vil vi se på nogle eksempler på *kvadratisk programmering* hentet fra en standardlærebog for handelsgymnasiet (Søren Antonius et al.):

> *En produktion af to varer A og B er underlagt betingelser, som kan udtrykkes ved følgende uligheder, hvor x er antal enheder for A og y er antal enheder for B:*
>
> Production of goods A and B underlies some restrictions which can be expressed by the following inequalities for number of units x of A and number of units y for B.
>
> $$2x + 3y \le 240$$
> $$2x + 2y \le 180$$
> $$4x + y \le 240$$
> $$x \ge 0$$
> $$y \ge 0$$

Af de tre første betingelser følger, at de kritiske x-værdier (dvs. skæringen med x-aksen) er 120, 90 og 60, ligesom de kritiske y-værdier er 80, 90 og 240. Altså skærer polygonområdet x-aksen i 60 og y-aksen i 80. Vi starter derfor med at vælge grafrummet

$$-10 \le x \le 70 \quad \text{og} \quad -10 \le y \le 90 \ .$$

Vi indskriver derefter kriterierne

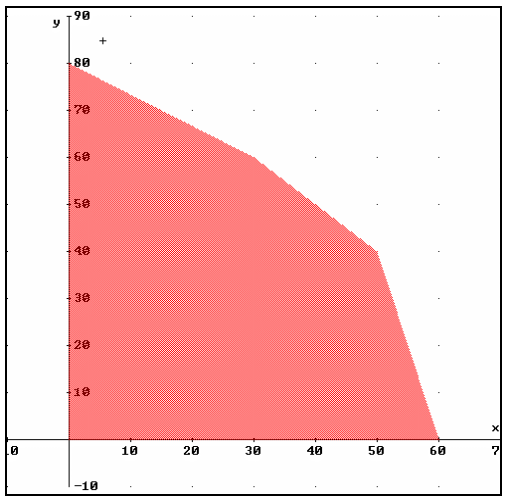
```
2·x + 3·y ≤ 240 ∧ 2·x + 2·y ≤ 180 ∧ 4·x + y ≤ 240 ∧ x ≥ 0 ∧ y ≥ 0
```

og tegner kriterieområdet:

The restrictions describe a region which has its boundaries in critical *x*- and *y*-values.

The intersection points of the boundary-lines result in a polygon.

The points on the axes are easy to find. But we need also the intersection points of the restriction lines.



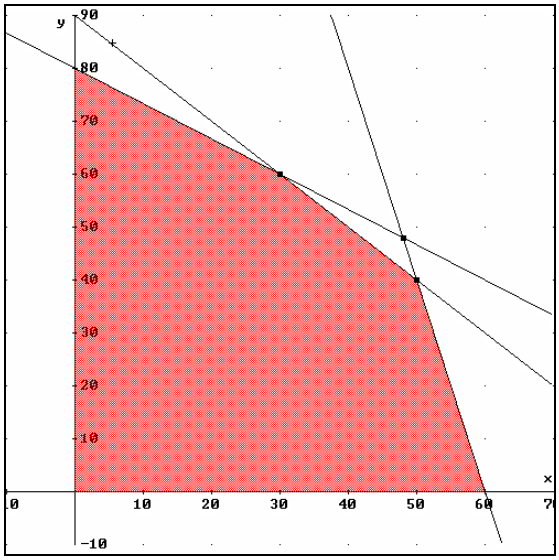
```
[r1 := 2·x + 3·y = 240, r2 := 2·x + 2·y = 180, r3 := 4·x + y = 240]

SOLUTIONS(r1 ∧ r2, [x, y]) = [[30, 60]]

SOLUTIONS(r1 ∧ r3, [x, y]) = [[48, 48]]

SOLUTIONS(r2 ∧ r3, [x, y]) = [[50, 40]]
```

Det kan også betale sig at indskrive ligningerne for radområdet, så vi fx bestemme skæringspunkterne:

Ved at benytte `Solutions` i stedet for `Solve` får vi skæringspunkterne ud på koordinatform, så de kan tegnes direkte:

By using `Solutions` instead of `Solve` we obtain the intersection points in coordinate form and the points are immediately ready for plotting.

Vi har styr på kriterieområdet!



Under passende antagelser bliver den *samlede omsætning* nu givet ved det følgende *kvadratiske udtryk* i x og y:

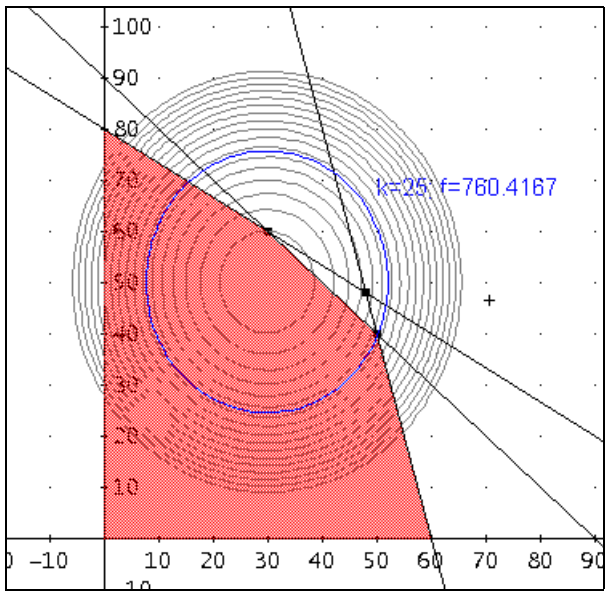
$$f(x,y) = 20x - \frac{x^2}{3} + 25y - \frac{y^2}{4}$$

Det er altså denne funktion vi skal *maksimere* i polygonområdet, så vi ser på nogle *niveaukurver*, fx niveaukurven gennem (25,25), dvs. kurven med ligningen $f(x, y) = f(25, 25)$. Faktisk kan vi lige så godt tegne en familie af niveaukurver $f(x, y) = k$, så på basis af værdien af $f(25,25)$ gætter vi på et passende interval af familieparameteren k.

Under certain circumstances revenue is given by the following quadratic expression:

$$f(x,y) = 20x - \frac{x^2}{3} + 25y - \frac{y^2}{4}$$

We try to find the maximum value for this function without violating the restrictions. We inspect so called level curves (contour curves of the surface *f*), eg the level curve containing point (25,25). This is a curve with equation $f(x,y) = f(25,25)$. But we can also create a whole family of level curves $f(x,y) = k$ with appropriate interval for parameter k.



Now in times of slider bars we can present the family of level curves

#6: $f(x, y) := 20 \cdot x - \frac{1}{3} \cdot x^2 + 25 \cdot y - \frac{1}{4} \cdot y^2$

#7: $f(25, 25) = 760.4166666$

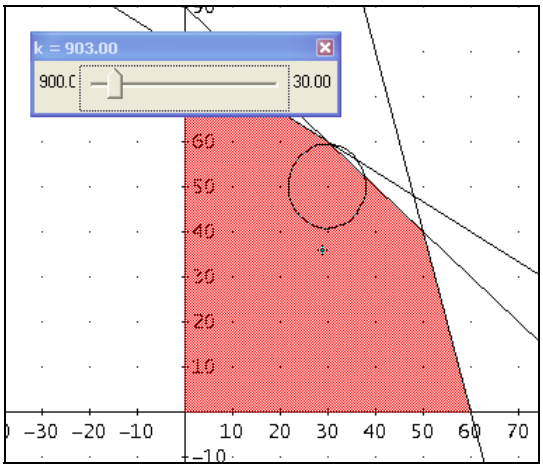
Plot the next curve in blue

#8: $20 \cdot x - \frac{1}{3} \cdot x^2 + 25 \cdot y - \frac{1}{4} \cdot y^2 = f(25, 25)$

Plot the family of curves in grey

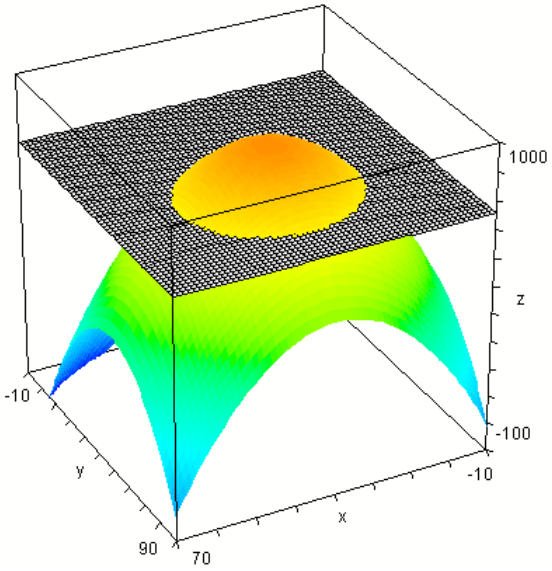
#9: VECTOR(f(x, y) = k, k, 500, 1000, 25)

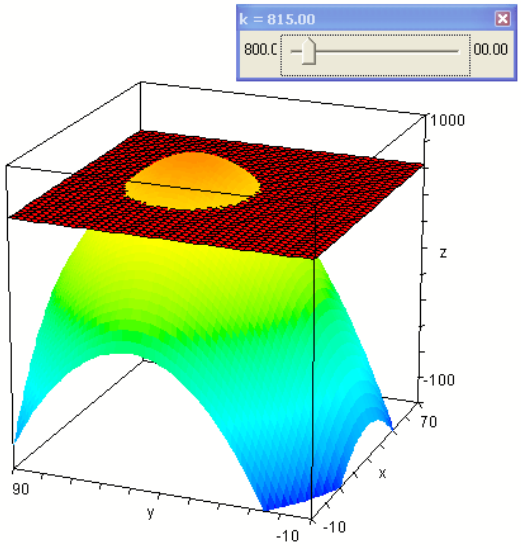
#10: f(x, y) = k



Det kunne godt se ud som om niveaukurverne er ellipser, samt at centrum for disse ellipser ligger inde i kriterieområdet. Vi kan bakke analysen op med et tredimensionalt billede af grafen for omsætnings-funktionen sammen med niveaufladen for $(x, y) = (25,25)$, dvs. $z = 760{,}41666\ldots$

The level curves are ellipses with their center within the feasible region. We can perform a 3D-analysis plotting the revenue surface together with the level plane for $(x,y) = (25,25)$, i.e. $z = 760.416...$ (and we can also introduce a slider bar for level planes $z = k$).





Det er sværere at få kriterieområdet med, fordi kriterieligningerne giver anledning til lodrette planer. Vi må derfor indskrive dem på parameterform (og sætte parameterintervallerne fornuftigt!):

We want to add the boundaries which appear as vertical planes. For plotting them we have to rewrite them in parameter form and set meaningful intervals for the parameters.

Planen med ligningen $2x + 3y = 240$ tegnes altså med parameterfremstillingen

$$\left[x, \frac{2 \cdot (120 - x)}{3}, z\right] \quad 0 \le x \le 70\ ,\ 0 \le z \le 1000$$

$$\text{SOLVE}(r1,\ y) = \left(y = \frac{2\cdot(120 - x)}{3}\right)$$

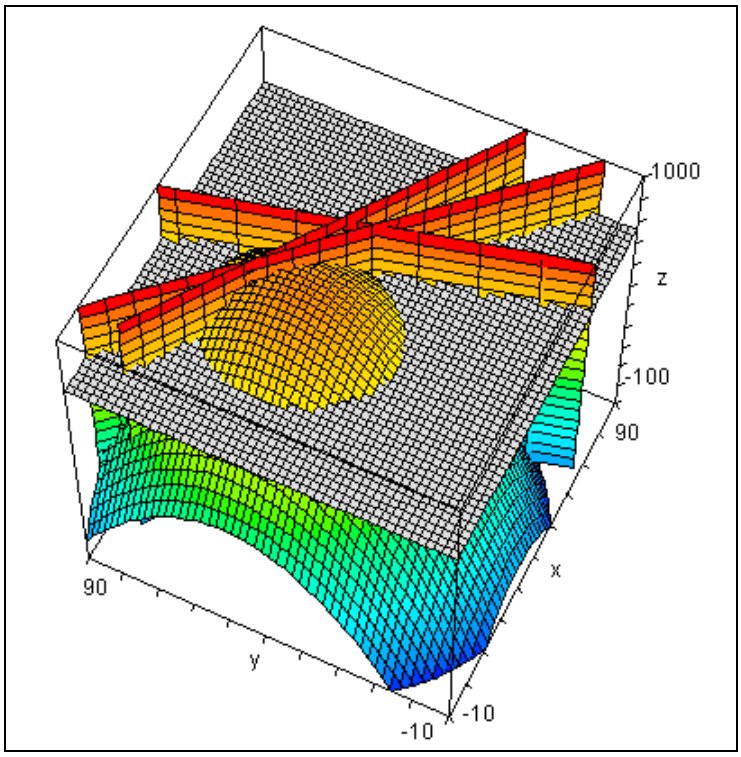
$$\left[x,\ \frac{2\cdot(120 - x)}{3},\ z\right]$$

$$\text{SOLVE}(r2,\ y) = (y = 90 - x)$$

$$[x,\ 90 - x,\ z]$$

$$\text{SOLVE}(r3,\ y) = (y = 240 - 4\cdot x)$$

$$[x,\ 240 - 4\cdot x,\ z]$$



Vi kan beregne toppunktet for det kvadratiske polynomium $f(x, y)$ ved at løse toppunktsligningerne i x henholdsvis y. Det gøres nok nemmest ved at differentiere:

$$\text{DIF}(f(x,y),x) = 0 \quad \text{og} \quad \text{DIF}(f(x,y),y) = 0$$

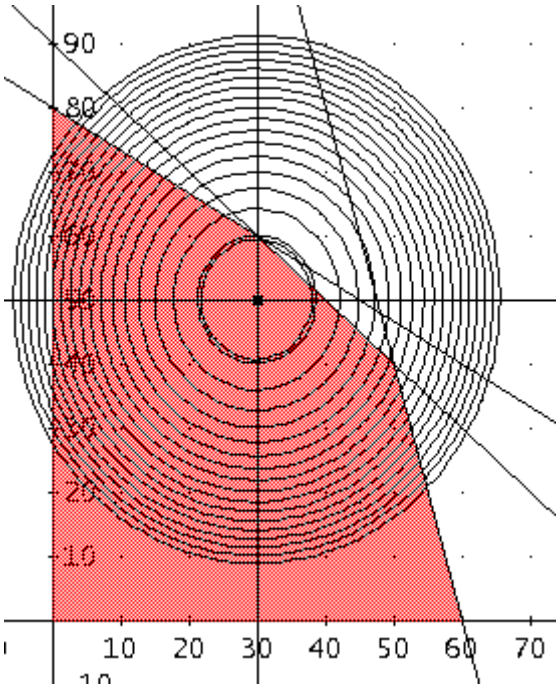
Ved at løse disse to ligninger med hensyn til x og y finder vi altså den optimale produktion:

We can find the top of the surface $f(x,y)$ by means of calculus and solving the respective equations we obtain the optimal production plan.

$$\text{SOLUTIONS}\left(\frac{d}{dx} f(x,\ y) = 0 \wedge \frac{d}{dy} f(x,\ y) = 0,\ [x,\ y]\right)$$

$$[[30,\ 50]]$$

$$f(30,\ 50) = 925$$



Som forventet ligger toppunktet *inde i* kriterieområdet og den maksimale omsætning er altså givet ved $f(30,50) = 925$.

The top of the surface lies *within* the feasible region and the maximum revenue of 925 will be reached by producing x = 30 units of A and y = 50 units of B.

Bemærkning: Vi kan også finde toppunktet mere geometrisk ved at se på skæringen mellem niveaukurverne og en familie af vandrette henholdsvis lodrette linjer. Vi løser altså ligningen for niveaukurverne $f(x, y) = k$ med hensyn til y (henholdsvis x):

Comment: One can find the top point by geometric means only. We solve the general level curve $f(x,y) = k$ for y (and for x):

SOLUTIONS(f(x, y) = k, y)

$$\left[\frac{2\cdot\sqrt{3}\cdot(\sqrt{(-x^2+60\cdot x-3\cdot(k-625))}+25\cdot\sqrt{3})}{3},\ \frac{2\cdot\sqrt{3}\cdot(25\cdot\sqrt{3}-\sqrt{(-x^2+60\cdot x-3\cdot(k-625))})}{3}\right]$$

$$\left[\frac{2\cdot\sqrt{3}\cdot\sqrt{(-x^2+60\cdot x-3\cdot(k-625))}}{3}+50,\ 50-\frac{2\cdot\sqrt{3}\cdot\sqrt{(-x^2+60\cdot x-3\cdot(k-625))}}{3}\right]$$

SOLUTIONS(f(x, y) = k, x)

$$\left[\frac{\sqrt{3}\cdot(\sqrt{(-y^2+100\cdot y-4\cdot(k-300))}+20\cdot\sqrt{3})}{2},\ \frac{\sqrt{3}\cdot(20\cdot\sqrt{3}-\sqrt{(-y^2+100\cdot y-4\cdot(k-300))})}{2}\right]$$

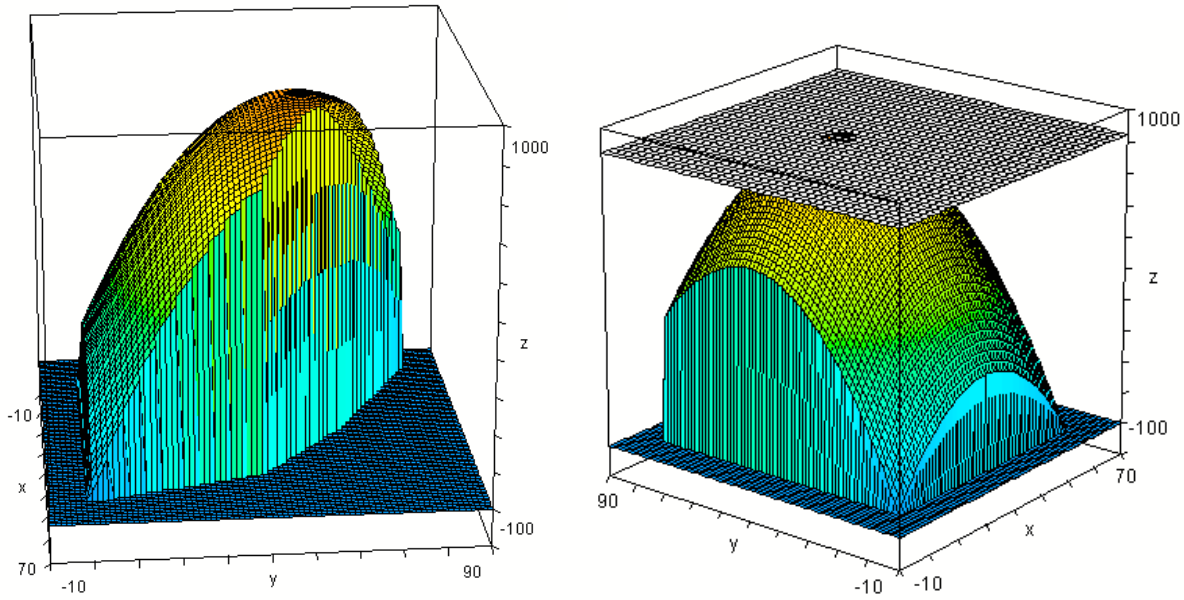
$$\left[\frac{\sqrt{3}\cdot\sqrt{(-y^2+100\cdot y-4\cdot(k-300))}}{2}+30,\ 30-\frac{\sqrt{3}\cdot\sqrt{(-y^2+100\cdot y-4\cdot(k-300))}}{2}\right]$$

Efter en passende `expand` af løsningsudtrykket ses da netop at y-værdierne ligger symmetrisk omkring 50 og tilsvarende fås at x-værdierne ligger symmetrisk omkring 30. Tegner vi linjerne $x = 30$ og $y = 50$ ser vi da også netop, at der er tale om symmetriakserne for ellipserne:

Applying the appropriate `EXPAND`-command on the solutions we recognize symmetry of the curves with respect to x = 30 and y = 50.

There is an additional way to visualize the region of possible solutions:

IF(2·x + 3·y ≤ 240 ∧ 2·x + 2·y ≤ 180 ∧ 4·x + y ≤ 240 ∧ x ≥ 0 ∧ y ≥ 0, f(x, y), 0)



Dette afslutter det første eksempel, som var simpelt, netop fordi toppunktet lå inde i kriterieområdet. Vi følger op med et nyt eksempel, hvor toppunktet ligger udenfor kriterieområdet!

The result of our first example was not so difficult to find because the vertex of the surface is within the boundaries. We proceed with an example with the vertex lying outside of the critical region. We just change the revenue function on one place.

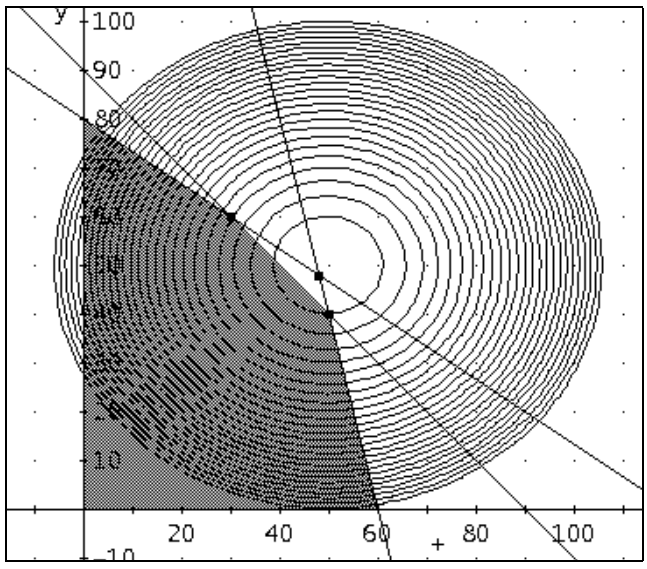
Eksempel 2: Vi prøver så at se på, hvad der sker, hvis vi ændrer omsætningsfunktionen en anelse til

$$f(x,y) = 20x - \frac{x^2}{5} + 25y - \frac{y^2}{4}$$

Det ændrer niveaukurverne, så de nu har centrum udenfor polygonområdet:

$$f1(x, y) := 20 \cdot x - \frac{1}{5} \cdot x^2 + 25 \cdot y - \frac{1}{4} \cdot y^2$$

VECTOR(f1(x, y) = k, k, 500, 1500, 25)



$$\text{SOLUTIONS}\left(\frac{d}{dx}\,f1(x,\,y) = 0 \wedge \frac{d}{dy}\,f1(x,\,y) = 0,\,[x,\,y]\right)$$

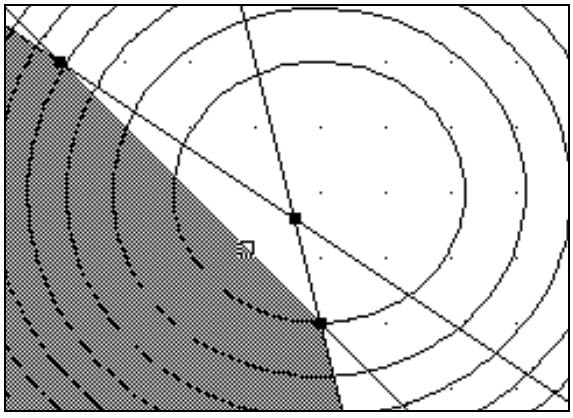
[[50, 50]]

The vertex is outside of the boundaries.

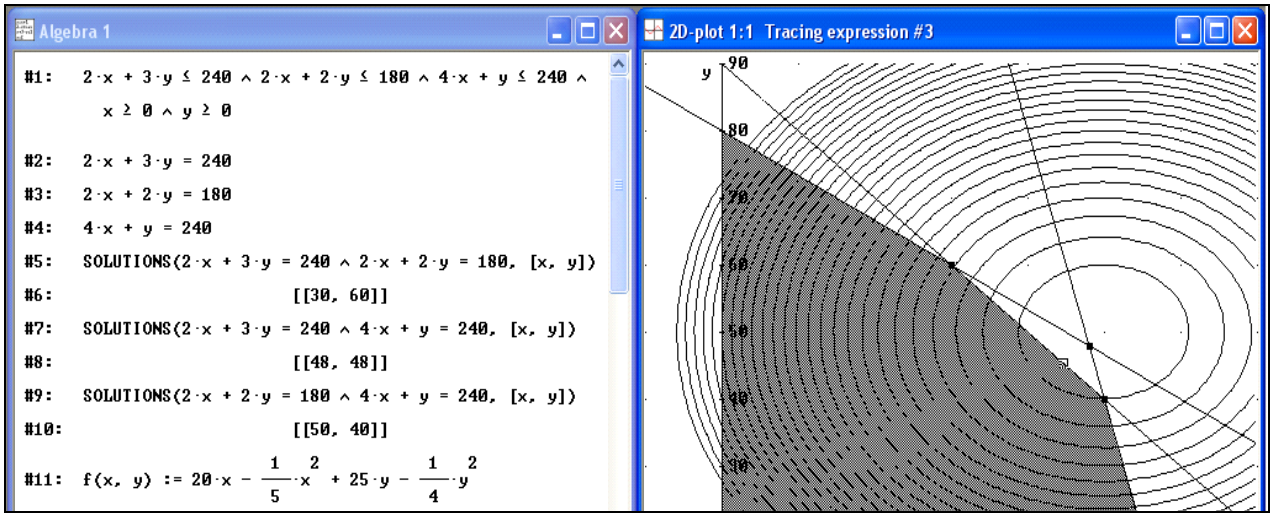
I stedet skal vi derfor bestemme maksimumspunktet, som det punkt på randen af polygonområdet, hvor niveaukurven *netop tangerer* polygonområdet, dvs. hvor niveaukurven slipper kriterieområdet. Først må vi da finde ud af hvilke af begrænsningsfunktionerne der er tale om. En trace viser, at det er den anden af kriteriefunktionerne ($2x + 2y = 180$):

For finding the maximum point we have to find a point on the borderlines of the feasible region (- the circumference of the polygon) which is also a point of an osculating level curve. At first we can try to find an approximative solution by using Trace mode. We trace along the second constraint line and estimate the osculating point of a possible level curve.

One estimate could be (44.3, 45.7).



We proceed by applying implicit differentiating function *f*1(*x*,*y*) which gives the slope in any point. We try to find a level curve which has not only a point with $2x + 2y = 180$ in common, but also the slope in this point.



Randkurven har altså ligningen $y = 90 - x$, og dermed hældningen –1. For at finde hældningen af niveaukurven differentierer vi ligningen for niveaukurven, idet vi opfatter y som en funktion af x. Hældningen for kurven er altså givet ved kommandoen:

`IMP_DIF(f1(x,y),x,y)` (with `x,y` as default settings)

Vi finder da:

$$\text{IMP\_DIF}(f1(x, y)) = \frac{4\cdot(x - 50)}{5\cdot(50 - y)}$$

$$\text{SOLUTIONS}(r2 \wedge \text{IMP\_DIF}(f1(x, y)) = -1, [x, y]) = \left[\left[\frac{400}{9}, \frac{410}{9}\right]\right]$$

$$f1\left(\frac{400}{9}, \frac{410}{9}\right) = \frac{10025}{9}$$

$$f1\left(\frac{400}{9}, \frac{410}{9}\right) = 1113.888888$$

Dermed har vi fundet en oplagt kandidat til den produktion, der giver den maksimale omsætning, som altså viser sig at være:

$$x = 400/9 \text{ og } y = 410/9,$$

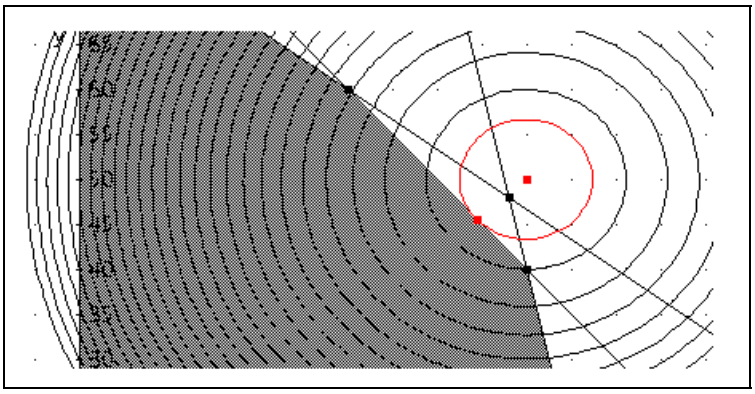
med den maksimale omsætning givet ved

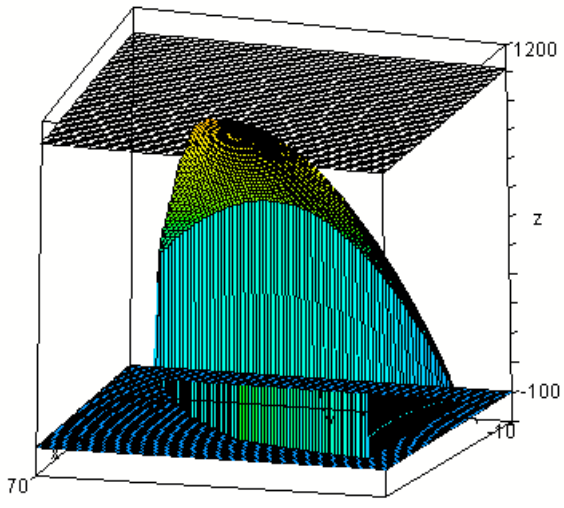
$$f(400/9, 410/9) = 1113.8888\ldots.$$

Vi kan checke løsningen grafisk ved dels at tegne det optimale punkt, dels tegne den tilhørende niveaukurve:

So we found a candidate for the maximum revenue in $x = 400/9$ and $y = 410/9$ with gaining a revenue of 1113.9. We can check this easily by plotting the respective level curve.

$$f1(x, y) = \frac{10025}{9}$$





Øvelser til kvadratisk programmering

Bemærkning: En klassisk introduktion til kvadratisk programmering kan man fx finde i

Mogens Ditlev Hansen: **Matematik - Økonomi – Optimering** (Abacus 1987)

Der er gode diskussioner/eksempler i kapitel 4 samt en del supplerende opgaver, fx

Øvelse 1: Omsætningsfunktionen (kriteriefunktionen) for en virksomhed er givet ved

$$f(x,y) = -3x^2 + 18x - 2y^2 + 28y$$

hvor produktionen er underlagt betingelserne:

$$2x + 5y \le 45$$
$$5x + 2y \le 60$$
$$x \ge 0\ ,\ y \ge 0$$

Tegn produktionsområdet samt en familie af niveaukurver for kriteriefunktionen.
Bestem maksimum for kriteriefunktionen.

Øvelse 2: Omsætningsfunktionen (kriteriefunktionen) for en virksomhed er givet ved

$$f(x,y) = -x^2 + 18x - 3y^2 + 36y$$

hvor produktionen er underlagt betingelserne:

$$x + 3y \le 21$$
$$5x + y \le 35$$
$$x \ge 0\ ,\ y \ge 0$$

Tegn produktionsområdet samt en familie af niveaukurver for kriteriefunktionen.
Bestem maksimum for kriteriefunktionen.

Øvelse 3: Omsætningsfunktionen (kriteriefunktionen) for en virksomhed er givet ved

$$f(x,y) = -10x^2 + 120x - 10y^2 + 220y$$

hvor produktionen er underlagt betingelserne:

$$x + 2y \le 18$$
$$2x + y \le 18$$
$$x \ge 0\ ,\ y \ge 0$$

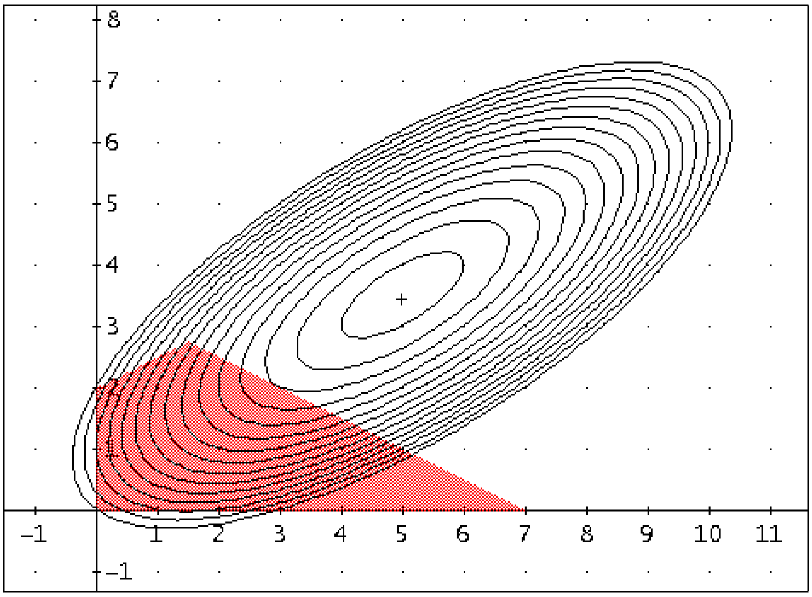
Tegn produktionsområdet samt en familie af niveaukurver for kriteriefunktionen.
Bestem maksimum for kriteriefunktionen.

I like this contribution very much and would like to add two more worked examples (Josef).

Example 1:

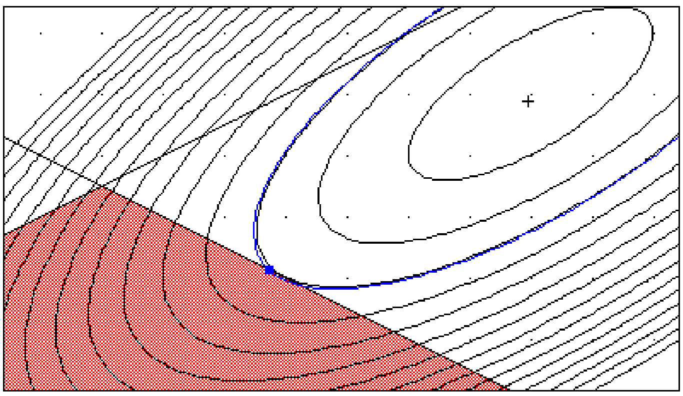
$$f(x,y)=3x+4y-x^2+2xy-2y^2=\text{Maximum}$$
$$x+2y\le 7$$
$$2y-x\le 4$$
$$x,y\ge 0$$

```
x + 2·y ≤ 7 ∧ -x + 2·y ≤ 4 ∧ x ≥ 0 ∧ y ≥ 0

                              2                2
f3(x, y) := 3·x + 4·y - x  + 2·x·y - 2·y

VECTOR(f3(x, y) = k, k, 0, 20, 1)
```



```
SOLUTIONS(x + 2·y - 7 ∧ IMP_DIF(f3(x, y), x, y, 1) = IMP_DIF(x + 2·y - 7, x, y, 1), [x, y])
                                                                        [[3, 2]]

                          2              2
3·x + 4·y - x  + 2·x·y - 2·y  = f3(3, 2)

                                                                        f3(3, 2) = 12
```

Solution curve in blue

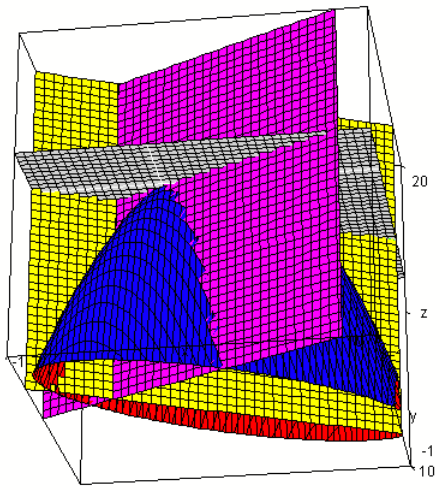


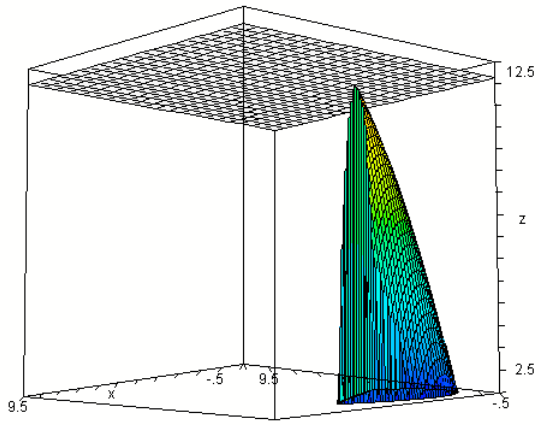
$$\left[x, \frac{7 - x}{2}, z\right]$$

$$\left[x, \frac{4 + x}{2}, z\right]$$

z = f3(3, 2)

IF(x + 2·y ≤ 7 ∧ −x + 2·y ≤ 4 ∧ x ≥ 0 ∧ y ≥ 0, f3(x, y), 0)





Maximum is 12 when for $x = 3$ and $y = 2$.

Do you miss a Minimumproblem? Here it is:

Example 2:

$$f(x,y) = \frac{x^2}{2} + \frac{y^2}{2} - x - 2y = \text{Minimum}$$
$$2x + 3y \le 6$$
$$x + 4y \le 5$$
$$x, y \ge 0$$

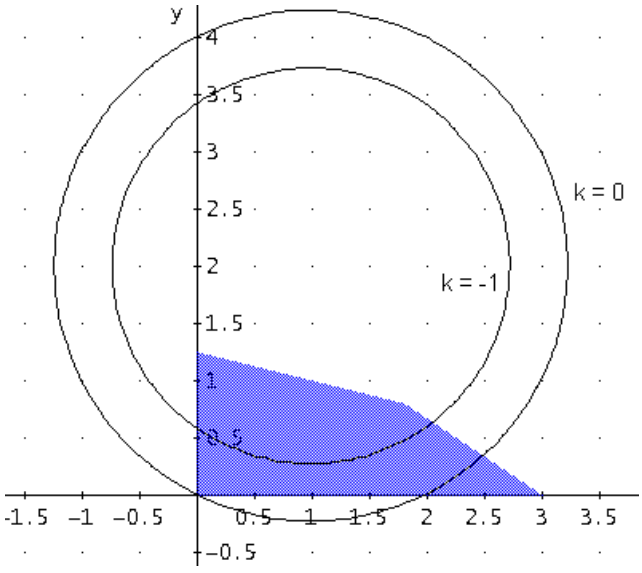
$$f4(x, y) := \frac{x^2}{2} + \frac{y^2}{2} - x - 2\cdot y$$

2·x + 3·y ≤ 6 ∧ x + 4·y ≤ 5 ∧ x ≥ 0 ∧ y ≥ 0

For first information plot two level curves!

f4(x, y) = 0

f4(x, y) = −1

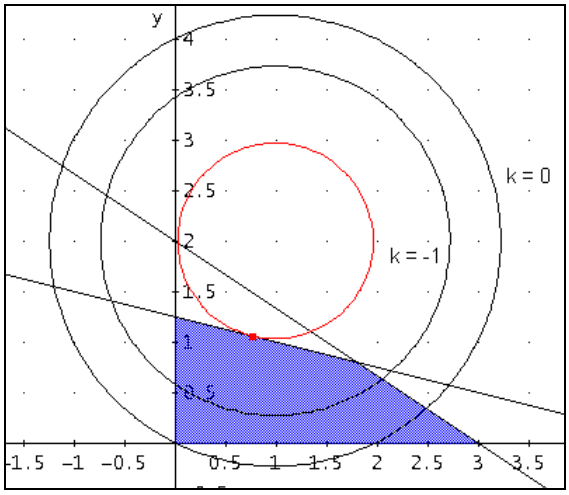


We make the 2nd boundary line to a tangent of one circle of the family of circles with centers in (1,2) which form the level curves. We could do this without calculus, too.

SOLUTIONS(x + 4·y − 5 ∧ IMP_DIF(f4(x, y), x, y, 1) = IMP_DIF(x + 4·y − 5, x, y, 1), [x, y])

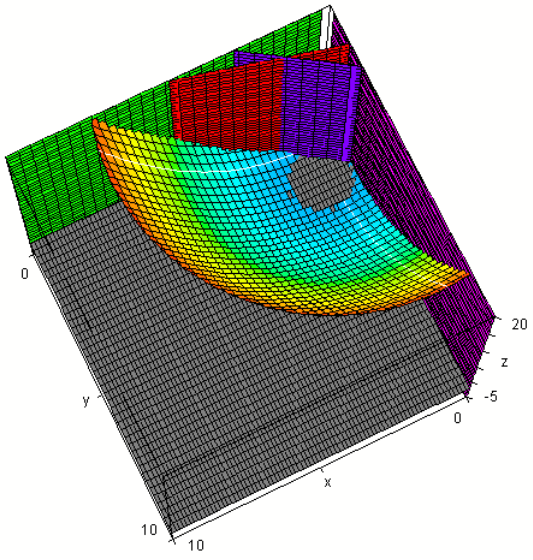
$$\left[\left[\frac{13}{17}, \frac{18}{17}\right]\right]$$

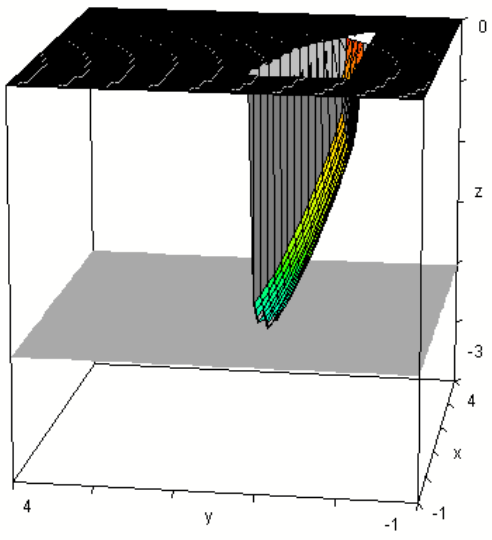
$$f4(x, y) = f4\left(\frac{13}{17}, \frac{18}{17}\right)$$



SOLUTIONS(x + 4·y = 5 ∧ y − 2 = 4·(x − 1), [x, y]) = $\left[\left[\frac{13}{17}, \frac{18}{17}\right]\right]$

$$f4(x, y) = -\frac{69}{34} \quad f4(x, y) = -\frac{69}{34}$$





Minimum = -2.03 for *x* = 0.765 and *y* = 1.059.

Reference: *The Operational Research Problem Solver*, REA, New York 1985

A Tool for Generating Tree Diagrams

Lorenz Kopp, Neumarkt, Germany

Base figure is a circle *kr* with radius *r* and center *M_*.

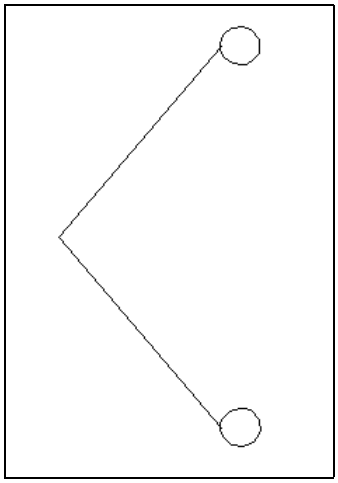
```
r := 0.2

                                2            2    2
kr(m_) := (x - m_ )  + (y - m_ )  = r
                 1            2
```

Meaning of the variables: [*ax*, *ay*] starting point, *dx*, *dy* increase of *x* and *y* to the next circle (1st point right on the top = slope triangle). Radius *r* is considered.

Two Branches: one, two, three or four experiments

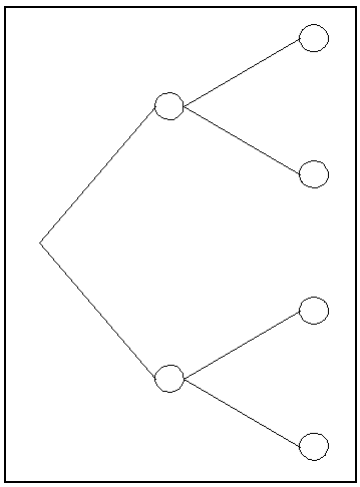
$$\text{tree21}(ax, ay, dx, dy) := \begin{bmatrix} \begin{bmatrix} ax + r & ay \\ ax + dx - r & ay + dy \end{bmatrix} & \text{kr}([ax + dx,\ ay + dy]) \\ \begin{bmatrix} ax + r & ay \\ ax + dx - r & ay - dy \end{bmatrix} & \text{kr}([ax + dx,\ ay - dy]) \end{bmatrix}$$

$$\text{tree22}(ax, ay, dx, dy) := \begin{bmatrix} \text{tree21}(ax, ay, dx, dy) \\ \text{tree21}\left(ax + dx, ay + dy, dx, \frac{dy}{2}\right) \\ \text{tree21}\left(ax + dx, ay - dy, dx, \frac{dy}{2}\right) \end{bmatrix}$$



$$\text{tree23}(ax, ay, dx, dy) := \begin{bmatrix} \text{tree21}(ax, ay, dx, dy) \\ \text{tree22}\left(ax + dx, ay + dy, dx, \frac{dy}{2}\right) \\ \text{tree22}\left(ax + dx, ay - dy, dx, \frac{dy}{2}\right) \end{bmatrix}$$

$$\text{tree24}(ax, ay, dx, dy) := \begin{bmatrix} \text{tree21}(ax, ay, dx, dy) \\ \text{tree23}\left(ax + dx, ay + dy, dx, \frac{dy}{2}\right) \\ \text{tree23}\left(ax + dx, ay - dy, dx, \frac{dy}{2}\right) \end{bmatrix}$$



The first two generations:

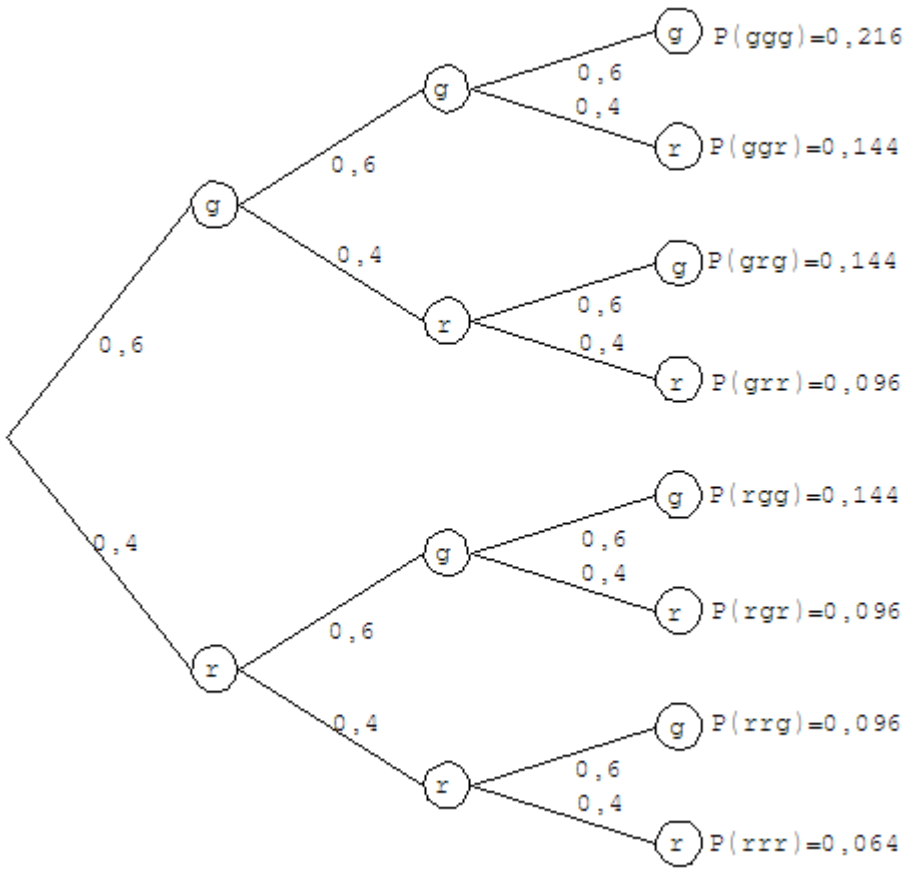
```
tree21(0, 0, 2, 2)

tree22(0, 0, 2, 2)
```

Some hints for plotting the trees:

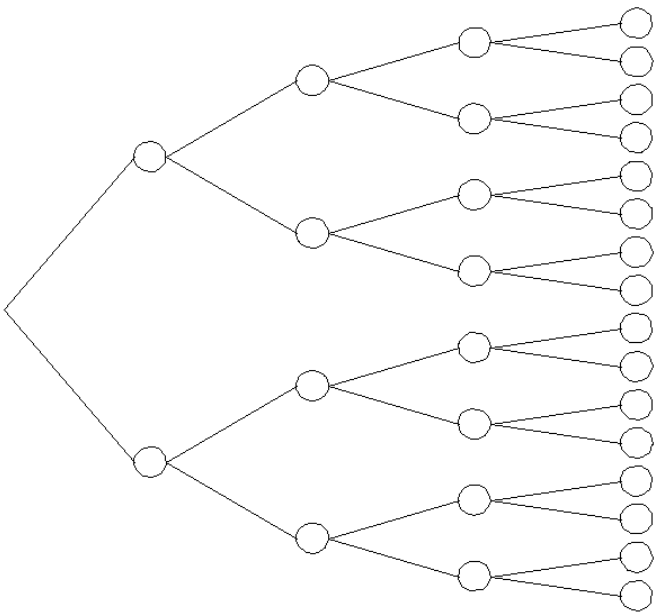
Window Tile Vertically, Plot region $0 \le x \le 8$, $-4 \le y \le 4$, Set Cross 4,0 and Cross on Center. Switch off Axes, Labels, Grids.

My tip: Set Option Display Grids Intervals 14,8 (Josef)

See `tree23(0,0,2,2)` with comments (annotations):



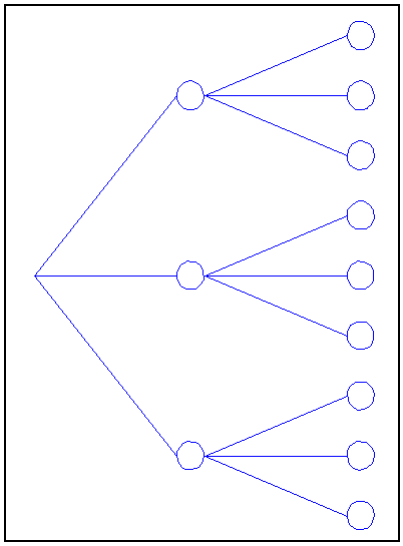
The maximum tree with four generations of branches `tree24(0,0,2,2)`:



We provide some other fundamental types of trees:

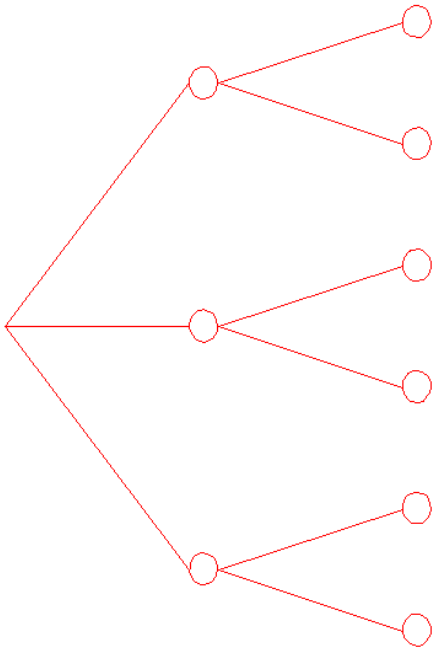
Two generations with three branches.

`tree32(0,0,2.5,2.5)`



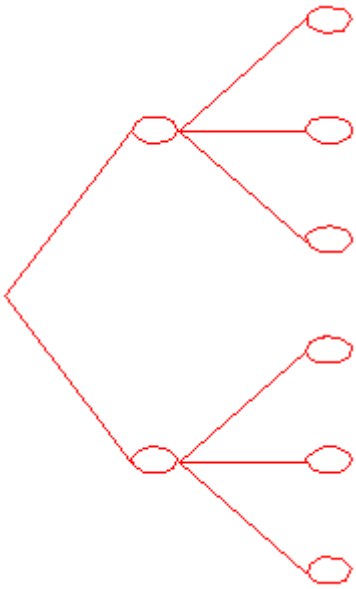
1st generation 3 branches, 2nd generation 2 branches: `tree3121(ax,ay,dx,dy)`

`tree3121(0,0,3,3)`

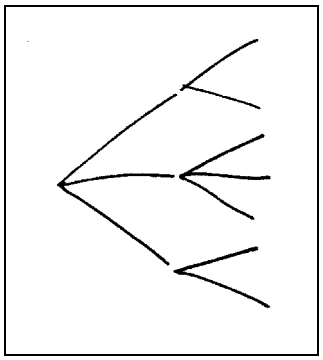


1st generation 2 branches, 2nd generation 3 branches: `tree2131(ax,ay,dx,dy)`

`tree2131(0,0,1.5,2.5)`



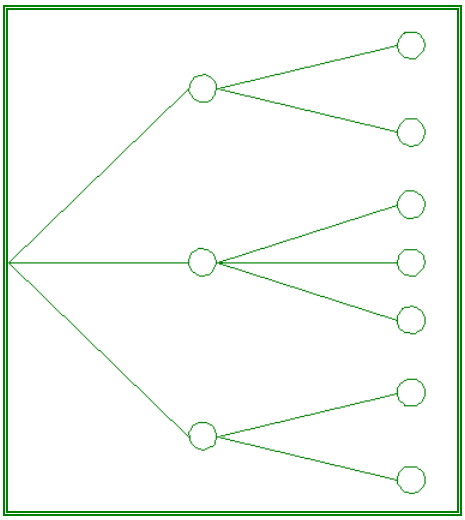
vertically zoomed out



Finally you can use this tool to create your special tree.

Assume you would like to have a decision tree according to the sketch!

```
tree00(ax, ay, dx, dy) := [ tree31(ax, ay, dx, dy)
                            tree21(ax + dx, ay + dy, dx, dy/4)
                            tree31(ax + dx, 0, dx, dy/3)
                            tree21(ax + dx, ay - dy, dx, dy/4) ]

tree00(0, 0, 3, 2.5)
```

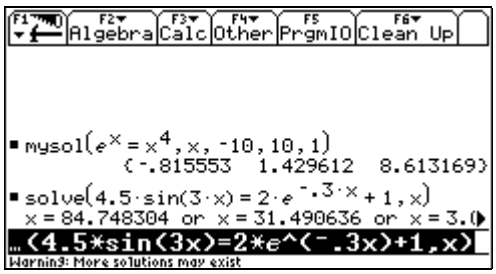


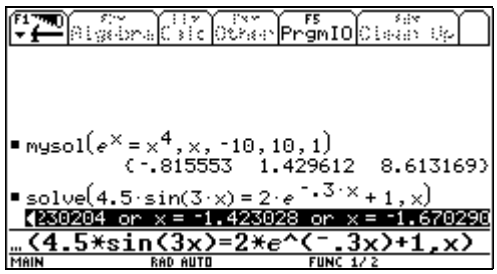
Warning: More solutions may exist

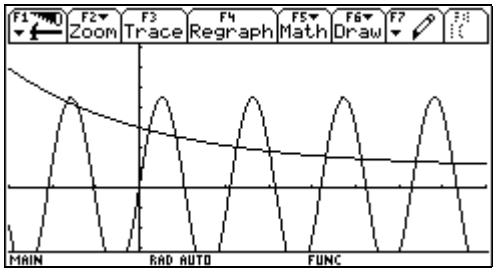
I received an email concerning the various different answers when solving equations using different CAS-calculators, different OS, different modes. In many cases you find a warning: More ..... So you never can be sure, if there are other solutions in addition the shown ones.

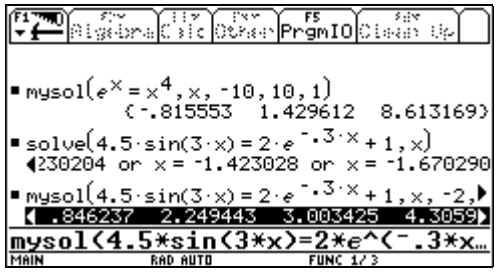
Some time ago I published a DERIVE-tool to overcome this problem for many cases. So I adapted this program for the handheld devices and made of MYSOLUTIONS a function mysol(equation, variable, lower_bound, upper_bound, step). The equation which caused the discussion was $e^x = x^4$.

For my experiments I took another equation which has much more solutions.
Compare the `solve`-output with the `mysol`-output:









Josef

Titbits from Algebra and Number Theory (29)

by Johann Wiesenbauer, Vienna

In the first place, as for the last column of this series, which dealt with Josephus permutations from a programming point of view, I want to apologize to Stefan Welke again for having overlooked that he had already written a treatise on exactly the same topic. In fact, using a new feature of delete( ) in Derive 6, he could achieve an impressive performance, and there was little to add as you can read yourself in the accompanying Derive file to the Titbits(28).

Nevertheless, I hope that the derivation of the nice formula for the "survivor" in the special case s=2 depending on n, was interesting enough. By the way, I doubt, if there is a similarly simple formula for the general case, but I'm convinced that there should be a way to write a very fast program to compute the number of the survivor for general numbers n and s. In view of the fact that I have not yet programmed it myself, it may be a little audacious, but nevertheless I would like to pose this as a challenge for those intrepid readers, who "fear neither death nor devil", when it comes to programming in Derive.

Speaking of a challenge, this reminds me of the nice problem posed by Steven Schonefeld in the ACDC column in the DNL #55, p40. As you can read there, this problem served as an introductory challenge for people who wanted to get hired by Google as engineers. In fact, as you can easily check yourself this problem has been discussed on many newsgroups and many solutions in different languages were presented on this occasion. What follows is my solution of this nice problem in Derive 6.

```
PrimeSearch(c, d := 10, n := 1, s := 1000, k_ := 0, l_, p_, u_ := []) :=
  Prog
    s := APPROX(STRING(c - FLOOR(c)), s + d - 1)
    s := REST(REST(s))
    l_ := DIM(s) - d
    Loop
      If k_ > l_
         RETURN REVERSE(u_)
      p_ := CODES_TO_NAME(NAME_TO_CODES(s↓[1, ..., d]))
      If PRIME?(p_) ∧ DIM(p_) = d
         Prog
           u_ := ADJOIN([k_ + 1, p_], u_)
           n := n - 1
           If n = 0
              RETURN REVERSE(u_)
      s := REST(s)
      k_ :+ 1
```

And here a few remarks as to the programming:

Basically, the program searches for the first n occurrences of d-digit primes in the decimal representation of the positive real number c, which is assumed to be irrational, where only the first s (by default 1000) decimal places after the decimal point are used. Here the primes should have really d digits, i.e. leading zeros are not allowed. Furthermore, blocks may exceed that bound of s digits as long as their start is within this bound. As a nice consequence, always s blocks are checked for primality not depending on the value of d and leading zeros are not allowed, hopefully all in perfect agreement to what Steven demanded in his challenge. Ok, maybe not quite, because I set n:=1 by default, which yields only the first occurrence of a d-digit prime, while Steven seemed to be more interested in all occurrences within the given bound, which you get by setting n:= inf. Furthermore, the routine returns also the locations of the starting digit of the primes, but it's easy to suppress this information, if you are not interested in it, as shown in the examples below.

```
VECTOR(TABLE(PrimeSearch(c, d), d, 1, 10), c, [e, π, √2])
```

1	[[1, 7]]	1	[[4, 5]]	1	[[4, 2]]
2	[[1, 71]]	2	[[2, 41]]	2	[[1, 41]]
3	[[4, 281]]	3	[[7, 653]]	3	[[3, 421]]
4	[[14, 4523]]	4	[[2, 4159]]	4	[[7, 5623]]
5	[[24, 74713]]	5	[[1, 14159]]	5	[[7, 56237]]
6	[[12, 904523]]	6	[[9, 358979]]	6	[[5, 135623]]
7	[[20, 6028747]]	7	[[3, 1592653]]	7	[[6, 3562373]]
8	[[64, 72407663]]	8	[[33, 28841971]]	8	[[3, 42135623]]
9	[[19, 360287471]]	9	[[29, 795028841]]	9	[[4, 213562373]]
10	[[99, 7427466391]]	10	[[4, 5926535897]]	10	[[1, 4142135623]]

```
PrimeSearch(e, 10, ∞)'
```

99	123	149	171	182	201	214	218	254
7427466391	7413596629	6059563073	3490763233	2988075319	1573834187	7021540891	5408914993	6480016847
295	309	313	322	399	438	440	473	505
9920695517	1838606261	6062613313	3845830007	1692836819	4425056953	2505695369	5490598793	1782154249
507	516	536	537	547	577	599	617	645
8215424999	9229576351	9519366803	5193668033	1825288693	8294887933	1730123819	4039701983	4804295311
667	669	687	702	705	709	728	743	781
8194558153	9455815301	1332069811	6181881593	1881593041	5930416903	1934580727	3858942287	4841984443
819	856	870	892	901	921			
1978623209	3140934317	3640546253	8887070167	7683964243	4563549061			

```
(PrimeSearch(e, 10, ∞)) COL 1
[99, 123, 149, 171, 182, 201, 214, 218, 254, 295, 309, 313, 322, 399, 438, 440, 473, 505, 507, 516, 536, 537, 547,
  577, 599, 617, 645, 667, 669, 687, 702, 705, 709, 728, 743, 781, 819, 856, 870, 892, 901, 921]

(PrimeSearch(e, 10, ∞)) COL 2
[7427466391, 7413596629, 6059563073, 3490763233, 2988075319, 1573834187, 7021540891, 5408914993, 6480016847,
  9920695517, 1838606261, 6062613313, 3845830007, 1692836819, 4425056953, 2505695369, 5490598793, 1782154249,
  8215424999, 9229576351, 9519366803, 5193668033, 1825288693, 8294887933, 1730123819, 4039701983, 4804295311,
  8194558153, 9455815301, 1332069811, 6181881593, 1881593041, 5930416903, 1934580727, 3858942287, 4841984443,
  1978623209, 3140934317, 3640546253, 8887070167, 7683964243, 4563549061]

                                        DIM(PrimeSearch(e, 10, ∞)) = 42
```

In particular, these computations show that there are 42 10-digit primes which start not later than the 1000-th digit after the decimal point. Is this in agreement with what we had expected? Sure! After all, due to the prime number theorem the density of primes near a positive real number x should be about 1/ln(x). Hence, we would expect about 1000/(10 ln 10) ≈43.43 10-digit primes for s=1000, which is not too far away from the actual number above, indeed! More generally, the dependency of the number of d-digit primes on s and d can be roughly described by the function

$$\frac{1}{LN(10)} \cdot \frac{s}{d}$$

Closely related to the problem above is the problem of finding the starting location(s) of any string d of digits within a given positive irrational number c, e.g. e, π, $\sqrt{2}$, etc. In fact, it is not very difficult to modify the program above accordingly.

```
StringSearch(c, d, n := 1, s := 1000, d_, k_ := 0, l_, p_, u_ := []) :=
  Prog
    d_ := DIM(d)
    p_ := APPROX(STRING(c - FLOOR(c)), s + d_ - 1)
    s := REST(REST(ITERATE(APPEND(x_, "0"), x_, p_, s + d_ + 1 - DIM(p_))))
    l_ := DIM(s) - d_
    Loop
      If k_ > l_
         RETURN REVERSE(u_)
      p_ := s↓[1, ..., d_]
      If p_ = d
         Prog
           u_ := ADJOIN(k_ + 1, u_)
           n :- 1
           If n = 0
              RETURN REVERSE(u_)
      s := REST(s)
      k_ :+ 1
```

There is a small subtlety, you should be aware of, though: Trailing zeros, which had been cut off by Derive when approximating c-floor(c), must be appended again. This is done by the line

```
s := REST(REST(ITERATE(APPEND(x_, "0"), x_, p_, s + d_ + 1 - DIM(p_))))
```

in the program above. (Note that this cutting off of trailing zeros didn't bother us in the first program, since we were looking for strings representing primes only!)

Well, what about the distribution of 0-9 within say the first 10000 digits of π after the decimal point? Here you are!

```
TABLE(DIM(StringSearch(π, STRING(d), ∞, 10000)), d, 0, 9)'
```

0	1	2	3	4	5	6	7	8	9
968	1026	1021	974	1012	1046	1021	970	948	1014

Now that the year 2004 is coming to an end (hopefully it was a good year for you and your family!), we could ask whether it is contained in the decimal representation of π starting not later than 10000 digits after the decimal point? Wanna bet?

```
StringSearch(pi, "2004", 1, 10000) = [7235]
```

Yes! By the way, this bet hasn't been extremely risky, as the chances of a failure are only about $1/e \approx 36.79\%$, assuming that π is normal, i.e. its decimal places form a "good" pseudorandom sequence. (Most mathematicians believe this to be true, although no rigorous proof is known!)

Let's turn to a completely different topic now, which should have already been part of my talk in Montreal, but it turned out that there was not enough time for it. It deals with the so-called discrete logarithm problem (DLP for short) with many important applications in modern cryptography (Diffie-Hellman key exchange, ELGamal cryptosystem, DSA etc).

In its most general form it can be stated as follows. Given a finite cyclic group G of order n, a generator α of G and an element $\beta \in G$, find the unique integer x, $0 \le x < n$, such that $\alpha^x = \beta$. This integer x is called the discrete logarithm of β to the base α and is denoted by $\log_\alpha \beta$. The most important examples of G are the multiplicative groups of a finite field $\mathbf{F}_q$, where q is either a big prime or a big power of 2 (the "classical" DLP), or groups emerging from the theory of elliptic curves (ECDLP).

Basically, there are algorithms for solving DLP that work in any group G and others, which take advantage of special features of the given group.

A prominent member of the first category is the so-called baby-step giant-step algorithm by Shanks. Using the notations above, this is an algorithm that takes $O(\sqrt{n})$ group operations to compute the solution x of $\alpha^x=\beta$ by carrying out the following steps:

1. Set $n \leftarrow$ ceiling($\sqrt{n}$) and $i \leftarrow 0$.
2. Compute the values α^j, j=0,1,…,n-1 and store them in a list. Furthermore, set $\alpha \leftarrow \alpha^{-n}$.
3. Compare β with all elements in the list above. If $\beta = \alpha^j$ for some j, then return x=in+j.
4. Set $\beta \leftarrow \alpha\,\beta$, $i \leftarrow i+1$ and go to step 3.

In order to implement a simple example let's specify the group G as the prime residue class group mod p for some prime p and let α be an element of order n for this group. Now a program that computes discrete logarithms $\log_\alpha\beta$ in this group could look like this. (Note that the fourth parameter n can be omitted, if it is unknown or α is chosen to be a primitive root of G, as in the example below. In both cases n will be set p-1.)

```
BSGS(α, β, p, n := 0, i_ := 0, j_, l_, n_) :=
  Prog
    If n = 0
       n := p - 1
    n := CEILING(√n)
    l_ := ITERATES(MOD(α·x_, p), x_, 1, n - 1)
    α := POWER_MOD(α, -n, p)
    Loop
      j_ := POSITION(β, l_)
      If NUMBER?(j_)
         RETURN i_·n + j_ - 1
      β := MOD(α·β, p)
      i_ :+ 1

                      10
NEXT_PRIME(RANDOM(10   )) = 3397745833

PRIMITIVE_ROOT(3397745833) = 5

BSGS(5, 123456789, 3397745833) = 62032454      (11.2s)

     62032454
MOD(5        , 3397745833) = 123456789
```

Another method of this general flavour is Pollard's ρ-method for solving the DLP. If you are familiar with Pollard's ρ-method for the integer factoring problem, then you already know the general idea behind it. As for the details, have a look at the following program. (Note that unlike the first program the exact order n of α must be known and cannot be omitted here!)

```
pollard_ρ(α, β, p, n, a_ := 0, b_ := 0, c_ := 0, d_ := 0, x_ := 1, y_ := 1) :=
  Loop
    a_ := MOD(a_ + [a_, 0, 1]↓(MOD(x_, 3) + 1), n)
    b_ := MOD(b_ + [b_, 1, 0]↓(MOD(x_, 3) + 1), n)
    x_ := MOD(x_·[x_, β, α]↓(MOD(x_, 3) + 1), p)
    c_ := MOD(c_ + [c_, 0, 1]↓(MOD(y_, 3) + 1), n)
    d_ := MOD(d_ + [d_, 1, 0]↓(MOD(y_, 3) + 1), n)
    y_ := MOD(y_·[y_, β, α]↓(MOD(y_, 3) + 1), p)
    c_ := MOD(c_ + [c_, 0, 1]↓(MOD(y_, 3) + 1), n)
    d_ := MOD(d_ + [d_, 1, 0]↓(MOD(y_, 3) + 1), n)
    y_ := MOD(y_·[y_, β, α]↓(MOD(y_, 3) + 1), p)
    If x_ = y_
       Prog
         x_ := SOLVE_MOD((b_ - d_)·x - c_ + a_, x, n)
         Loop
           If MOD(α^FIRST(x_), p) = MOD(β, p)
              RETURN FIRST(x_)
           x_ := REST(x_)

pollard_ρ(5, 123456789, 3397745833, 3397745832) = 62032454      (5.89s)
```

Originally, this algorithm was only designed for the case, where n is a prime, although our adaption works also in the case, where n is composite. Nevertheless, in the latter case you should rather use the algorithm by Pohlig-Hellman, which is particularly good, if n is very "smooth" (see example below)!

```
PohligHellman(α, β, p, n, p_, q_, x_) :=
  Prog
    If PRIME?(n)
       RETURN pollard_ρ(α, β, p, n)
    If PRIME_POWER?(n)
       Prog
         p_ := FIRST(FIRST(FACTORS(n)))
         x_ := pollard_ρ(MOD(α^(n/p_), p), MOD(β^(n/p_), p), p, p_)
         β := MOD(β·POWER_MOD(α, -x_, p), p)
         RETURN x_ + p_·PohligHellman(MOD(α^p_, p), β, p, n/p_)
    q_ := VECTOR(u_↓1^u_↓2, u_, FACTORS(n))
    CRT(VECTOR(PohligHellman(MOD(α^(n/r_), p), MOD(β^(n/r_), p), p, r_), r_, q_), q_)

PohligHellman(5, 123456789, 3397745833, 3397745832) = 62032454  (0.02s !!)
```

$$\text{FACTOR}(3397745832) = 2^3 \cdot 3 \cdot 13 \cdot 19 \cdot 307 \cdot 1867$$

Believe me, I would love to go into details, but there is simply no space left! If you are curious though, you will find them in http://www.cacr.math.uwaterloo.ca/hac/ anyway.