Winfried Auzinger
Dirk Praetorius

Sommersemester 2017
31. Mai / 1. Juni 2017

Übungsaufgaben zur VU Computermathematik

Serie 9

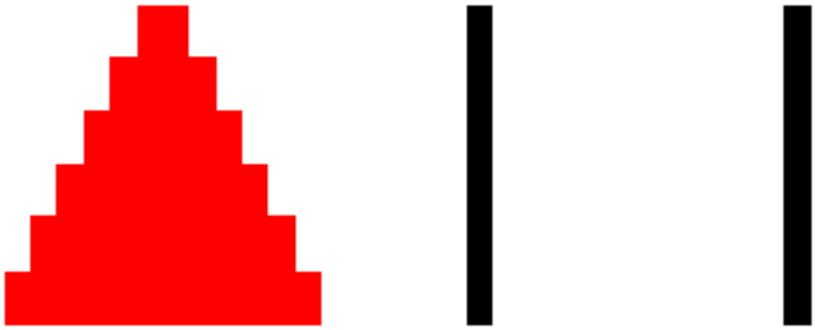
Eine Kollektion verschiedener Problemstellungen, teilweise mit stofflichen Ergänzungen.

Exercise 9.1: *Towers of Hanoi.*

We solve this famous puzzle described at

`https://en.wikipedia.org/wiki/Tower_of_Hanoi`

in Maple based on a recursive approach.



We represent the three rods by three Maple stacks `A,B,C` (see `? stack` and lecture notes, Part III), and use the symbols `1,2,3,...` to represent the disks. After initializing stack `A` with n discs, we solve the problem in the following way:

(i) Move the pile consisting of the upper $n-1$ discs from `A` to `B`.

(ii) Move the single remaining disc from `A` to `C`.

(iii) Move the pile of $n-1$ discs from `B` to `C`.

This results in a recursive solution algorithm, since in steps (i) and (iii), analogous smaller problems are solved.

Realize such a version using a recursive procedure, and display the status of of the three stacks after each step in a simple manner (without using graphics). Try $n = 2, 3, 4, 5, \ldots$.

Exercise 9.2: *Derivation of a 2D quadrature formula.*

Consider the triangle T with vertices $(0,0)$, $(1,0)$, and $(0,1)$. For the numerical approximation of integrals of the form

$$If = \int_T f(x,y)\, d(x,y) = \int_{x=0}^{1} \Big(\int_{y=0}^{1-x} f(x,y)\, dy \Big)\, dx$$

(for real functions $f\colon \mathbb{R}^2 \to \mathbb{R}$) we wish to find a formula Q of the form

$$Qf = \alpha_1 f(0,0) + \alpha_2 f(1,0) + \alpha_3 f(0,1) + \beta_1 f(\tfrac{1}{2},0) + \beta_2 f(0,\tfrac{1}{2}) + \beta_3 f(\tfrac{1}{2},\tfrac{1}{2})$$

which is exact for polynomials $p(x,y)$ of degree ≤ 2, i.e., $Qp = Ip$ for any

$$p(x,y) = c_0 + c_1\,x + c_2\,y + c_{20}\,x^2 + c_{11}\,xy + c_{02}\,y^2$$

This is equivalent to the requirement that $Qp = Ip$ for $p(x,y) = 1$, $x\,y$, $x^2\,xy$ and y^2.

Setup and solve a system of 6 linear equations and use `LinearAlgebra[LinearSolve]` *to solve it, resulting in the desired coefficients α_j and β_j. Check that your result is correct.*

Exercise 9.3: ***Solving a system of polynomial equations.***

Consider the system of four polynomial equations

```
2*x[1]+2*x[2]=1
2*x[3]+x[4]=1
6*x[1]^2*x[3]+3*x[1]^2*x[4]+12*x[1]*x[2]*x[3]+6*x[1]*x[2]*x[4]+12*x[2]^2*x[3]+3*x[2]^2*x[4]=1
12*x[1]*(x[3]^2)+12*x[1]*x[4]*x[3]+3*x[1]*(x[4]^2)+6*x[2]*(x[3]^2)+6*x[2]*x[4]*x[3]+3*x[2]*(x[4]^2)=1
```

in the four variables x_1, x_2, x_3, x_4.

a) *Use* `solve` *to compute the exact solutions of this system, and evaluate them using* `evalf`*. How many real solutions can you identify?*

b) *Check that all computed solutions are correct.*

Exercise 9.4: ***Source code on external files. Fighting with formatted input.***

a) *Take one of your worksheets which requires no interactive input and export it to a Maple language file (file type* `.mpl`*). Then execute it using* `maple` *(or* `cmaple`*) and redirect the output to a text file.*

Remark: This is useful if longer jobs have to run in batch mode. Depending on your installation, the command line version of Maple is called `maple` or `cmaple`. If this is not in your execution path, check its location.

b) (∗) *Assume that the integer coefficients of a matrix consisting of rows of a given length n stored on a text file. Each line contains one row of the matrix. Design a procedure which expects the name of the text file as its argument and which returns the matrix in form of an object of type* `Matrix`*.*

Hint: Use (e.g.) `? readline` followed by `? sscanf`.

Exercise 9.5: ***Exploring the behavior of a sequence via experiment.***

We consider sequences (x_n) defined by

$$x_n := f(x_{n-1}), \quad n = 1,2,3,\ldots, \qquad \text{with} \quad f(x) = \frac{1}{2}\left(x - \frac{1}{x}\right)$$

starting from an initial x_0. We observe:

- For $x_0 = 0$ we immediately end up with $x_1 = \infty$.
- For $x_0 = \pm 1$ we have $x_1 = 0$ and $x_2 = \infty$.

- For all other $x_0 \in \mathbb{Q}$, the sequence (x_n) is well-defined for all n. (*Why? This is very simple to prove. Note that for $x_0 \in \mathbb{Q}$, all x_n are rational numbers.*)
- For a complex $x_0 \in \mathbb{C}$, $x_0 \notin \mathbb{R}$, the sequence (x_n) is well-defined, with $x_n \notin \mathbb{R}$ for all n.
- For $x_0 \in \mathbb{R}$, the (real) sequence (x_n) cannot be convergent, since the only possible limits are i and $-i$.

a) *Design a procedure which, for given $n \in \mathbb{N}$, produces a plot of the points (n, x_n) for given x_0.*

Hint: For `pointplot`, a recommended set of options is

`style=point`, `axes=boxed`, `symbolsize=20`, and `symbol=solidcircle`

b) *Conjecture:* For all $x_0 \in \mathbb{C}$ with $\operatorname{Im} x_0 > (<)\ 0$, the iteration converges to $\pm i$. We explore this experimentally:

Design a procedure which expects $x_0 \in \mathbb{C}$, $\epsilon > 0$ and $n_{max} \in \mathbb{N}$ as its arguments and which returns the minimal value $n \in \mathbb{N}$ such that $|x_n - (\pm i)| \leq \varepsilon$. If no such $n \leq n_{max}$ is detected, use `error` *to issue an error message including the value of $x_{n_{max}}$. Use* `evalf`.

Play with your procedure, in particular with x_0 very close to 0.

Exercise 9.6: *Simulating the movement of a soap bubble.*

Imagine that a moving disk (see `? plottools[disk]`) represents a soap bubble floating around in the plane.

a) *Use the random number generator (*`? rand`*) and* `plots[display]` *with options*

`axes=none`, `scaling=constrained`, *and* `insequence=true`

to generate an animation of a moving soap bubble.

b) [optional:] *3D version of* **a)**, *using* `? plottools[sphere]`.

Remark: Play with parameters to simulate 'realistic' dynamics.

Exercise 9.7: *Extension of Exercise 9.6 a).*

$(*)$ *Proceed in a similar way as in Exercise 9.6 a), and design a procedure which generates two lists* `b1, b2` *of plot structures representing two different soap bubbles (with different colors). Stop when these bubbles (disks) are overlapping for the first time.*

Then, use `plots[display]` *within a loop to generate a list of plot structures* `b[k]` *comprising* `b1[k]` *and* `b2[k]` *within a single plot, for each* `k=1,2,...`. *Finally, use* `plots[display]` *on* `b` *with option* `insequence=true` *to animate the simultaneous movement of both bubbles. The animation stops when overlap occurs.*

Remark: Just for fun, you may include some special effect at the end of the animation.

Exercise 9.8: ***Simulating the movement of a pendulum.***

We consider the movement of a pendulum, described by its angle of deflection $\varphi = \varphi(t)$ as a function of time t. The governing diffferential equation is

$$\ddot{\varphi}(t) = -\sin(\varphi(t))$$

where $\ddot{\varphi}$ is the second derivative of φ w.r.t. t. Together with initial conditions for φ and $\dot{\varphi}$, e.g.

$$\varphi(0) = 1, \quad \dot{\varphi}(0) = 1$$

the problem has a unique solution $\varphi(t)$ for all t, but the solution cannot be represented in an exact, analytic form. Therefore we investigate some numerical methods. First of all, we introduce the angular velocity $\psi(t) := \dot{\varphi}(t)$ as a separate variable and consider the equivalent system

$$\begin{pmatrix} \dot{\varphi}(t) \\ \dot{\psi}(t) \end{pmatrix} = \begin{pmatrix} \psi(t) \\ -\sin(\varphi(t)) \end{pmatrix} \qquad \text{with initial conditions} \quad \begin{pmatrix} \varphi(0) \\ \psi(0) \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

a) We choose a timestep h and use the simplest numerical scheme in order to compute approximations (φ_n, ψ_n) to the solution $(\varphi(t_n), \psi(t_n))$ at the times $t_n := n\,h$, $n = 0, 1, 2, \ldots$. To this end we replace the derivative $\dot{\varphi}(t_n)$ by the forward difference quotient, i.e.,

$$\dot{\varphi}(t_n) \approx \frac{\varphi_{n+1} - \varphi_n}{h}$$

and analogously for ψ. This leads to the recursion

$$\begin{pmatrix} \varphi_{n+1} \\ \psi_{n+1} \end{pmatrix} = \begin{pmatrix} \varphi_n \\ \psi_n \end{pmatrix} + h \begin{pmatrix} \psi_n \\ -\sin(\varphi_n) \end{pmatrix}, \quad n = 0, 1, 2, \ldots, \quad \text{starting from} \quad \begin{pmatrix} \varphi_0 \\ \psi_0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Implement this method by a simple loop, using `evalf`*, to generate a list of vectors containing the solution values* (φ_n, ψ_n) *at the times* t_n, $n = 0, 1, 2, \ldots$. *Choose the timestep* $h = 0.1$ *and produce a pointplot of the* (φ_n, ψ_n), $n = 0, 1, 2, \ldots$, *for* n *up to* 200 *(i.e.,* $t = 20$*). The solution should behave periodic. What do you observe?*

b) For larger time intervals, the method from **a)** produces a qualitatively incorrect approximation. Here is a simple remedy: We use the modified recursion

$$\begin{pmatrix} \varphi_{n+1} \\ \psi_{n+1} \end{pmatrix} = \begin{pmatrix} \varphi_n \\ \psi_n \end{pmatrix} + h \begin{pmatrix} \psi_n \\ -\sin(\varphi_{n+1}) \end{pmatrix}$$

Repeat the experiment from **a)** *using the modified recursion. What do you observe?*

c) *In addition, apply* `dsolve` *with option* `numeric` *(without further special settings), and use* `plots[odeplot]`*:*

```
sol:=dsolve([D(u)(t)=v(t),D(v)(t)=-sin(u(t)),u(0)=1,v(0)=1],[u(t),v(t)],numeric)
plots[odeplot](sol,[u(t),v(t)],t=0..200,axes=boxed,thickness=2)
```

What do you observe? Also extend the range from $t = 20$ *to* $t = 300$ *and repeat* **a)**, **b)**, *and* **c)**.

Remark: `dsolve/numeric` delivers a procedure, and calls to this procedure can be use to evaluate the numerical approximation at particular particular times t, or it can be passed to `? plots[odeplot]`.