Numerical Analysis of Ordinary Differential Equations

Programming Exercises

Summer Semester 2018

Universität Heidelberg - IWR
Prof. Dr. Guido Kanschat, Dr. Dörte Jando,
Bastian Boll, Pablo Lucero

Exercise Sheet 1

There will be no scores given and accumulated over the semester. Nevertheless, we strongly recommend solving the exercises. If you choose to hand in your solutions they will be annotated according to completeness and mathematical correctness. Please hand in your solution in groups of at least 2 or 3 students.
Important: Please register to one of the exercise groups in **Müsli**.

Problem P1.1 (**Implementation blueprint**)
The practical implementation of methods should follow a well considered blueprint in order to gain portability and ease of debugging. The provided sample code demonstrates how this can be approached.

a) Run the demo codes.

b) Which problem has been solved? Solve the IVP until $t_{\text{end}} = 2$.

c) Familiarize yourself with the demo codes. Describe the structure of each of the implementations and how they provide a framework for code reuse.

Note: You do not need to describe details of how the demonstrated method computes solution steps. It can be treated as a blackbox.

Problem P1.2 (**Experimental Order of Convergence**)
In general, numerical methods are of polynomial order

$$a(h) \to a \quad (h \to 0) \quad \Rightarrow \quad \|a(h) - a\| = \mathcal{O}(h^\alpha)$$

α denotes the polynomial order of the convergence. In practice, we estimate the order using the following formula

$$\alpha = \frac{1}{\log(2)} \log \left(\frac{\|a(h) - a(\frac{h}{2})\|}{\|a(\frac{h}{2}) - a(\frac{h}{4})\|} \right).$$

Use this formula to analyse the convergence order of the implemented method.

Problem P1.3 (Explicit Euler)
Consider the initial value problem

$$u'(t) = f(t, u(t)), \quad u(t_0) = u_0$$

Given any point $(t_n, u(t_n))$ on the graph of the solution, we can use the linear approximation

$$u(t_n + h) \approx u(t_n) + hu'(t_n) = u(t_n) + hf(t_n, u(t_n))$$

for small time steps h. This motivates the so called (explicit) Euler method

$$t_{n+1} = t_n + h, \quad y_{n+1} = y_n + hf(t_n, y_n).$$

It computes approximations $y_n := y(t_n)$ at t_n of the exact solution $u(t_n)$ starting at the initial iterate $(t_0, y_0) = (t_0, u_0)$.
Implement this method by building on one of the provided code blueprints. Test your implementation on the following IVP

$$u'(t) = t\, u(t), \quad u(0) = \pi, \quad t \in [0, 1]$$

with exact solution

$$u(t) = \pi\, e^{t^2/2}$$

Compare the accuracy of your computed approximation $\|y(1) - u(1)\|$ between different step lengths h. How does the gained accuracy compare to the number of required function evaluations?