Technische Universität Chemnitz
Dr. M. Winkler

Chemnitz, 18th October 2018

Numerical Methods for ODEs
Sheet 1

Exercise 1: Transformation to a first-order ODE-system

Show that each initial value problem of the form

$$y^{(m)}(t) = f\big(t, y(t), y'(t), \dots, y^{(m-1)}(t)\big), \tag{1a}$$

$$y^{(k)}(0) = y_{0,k}, \quad k = 0, 1, \dots, m-1 \tag{1b}$$

can be tranformed into a system of first-order ODE's. How can we solve (1) numerically?

Exercise 2: Consistency order of some one-step methods

Consider the one-step methods from Example 3.11. Show the following assertions:

(a) The implicit Euler method has the consistency order 1.

(b) The improved Euler method has the consistency order 2.

(c) The Crank-Nicolson method has the consistency order 2.

Illustrate the first step of these methods for the initial value problem

$$y'(t) = \frac{t^2}{4} + y(t), \quad y(0) = 0$$

for the grid size $h = \frac{1}{2}$ in the corresponding direction field.

Exercise 3: Realization of implicit one-step methods

When realizing the implicit Euler method

$$y_{k+1} = y_k + h\, f(t_{k+1}, y_{k+1}), \quad k = 0, 1, \dots,$$

one has to solve a nonlinear equation system in each iteration.

(a) This system has the form of a fixed-point equation. Discuss the corresponding iteration using Banach's fixed-point theorem. Under which assumptions is the corresponding operator a contraction?

(b) The Newton method is a more efficient method for the solution of nonlinear equation systems. Derive the corresponding Newton system and check convergence criteria for Newton's method.

Homework 1: Implementation of several one-step methods

(a) Implement the one-step methods from Example 3.11. Use the implementation of the explicit Euler method from `euler_expl.m`, available on the Homepage of this lecture, as a template. Use the same Syntax (input and output parameters) for your new methods:

```
[t,y] = my_method(f, time, y0, N)
```

- `f`: Handle to a function $f(t, y)$. In case if $n > 1$ the return value should be a *column* vector.
- `time`: Row vector $[t_0\ T]$ for the time horizon that should be computed.
- `y0`: Initial value y_a as a column vector (if $n > 1$).
- `N`: Number of grid points.
- `t`: The grid $[t_0\ t_1\ \ldots\ t_N]$ related to the numerical solution (required for plotting).
- `y`: The numerically computed solution in the form $[y_0\ y_1\ \ldots\ y_N]^\top$.

(b) Test your implemented methods using the script `convergence_test.m`. To do so, uncomment the commands that are marked with "TODO". The script computes the approximate solutions on several grids $\mathcal{T}_{h_i}$, $i = 1, 2, \ldots$. To proceed with the computation on the next grid press an arbitrary button (the focus of the cursor has to be in the "Command Window" of Matlab). For the realization of the implicit methods use the ideas from Exercise 3.

(c) At the end of the script a table containing the experimentally determined convergence rates is printed and a plot illustrating the discretization errors is plotted in a diagram with logarithmic axes. Interpret these results! Does this confirm our theory from Exercise 2?

Hint: The use of logarithmic axes means that we actually plot the data points $(\log(N_i), \log(\|e_{h_i}\|_{\infty,h}))$ instead of $(N_i, \|e_{h_i}\|_{\infty,h})$ for $i = 1, 2, \ldots$. Since we expect a relation of the form

$$\|e_{h_i}\|_{\infty,h} \approx C\, h_i^p = C\, (1/N_i)^p$$

the error plots are, after taking the logarithm on both sides, linear functions with arguments $\log(N_i)$ and function values $\log(\|e_{h_i}\|)$. With simple logarithmic laws we confirm

$$\log(\|e_{h_i}\|_{\infty,h}) \approx \log(C) - p\, \log(N_i).$$

Obviously, the convergence rate p is responsible for the slope of this function.

Homework 2: Attracting points

Given are n points $\{z_1, \ldots, z_n\}$ in the x-y-plane. The point z_i is attracted by the point z_{i+1}, i. e., there holds the relation

$$\dot{z}_i = z_{i+1} - z_i \qquad i = 1, \ldots, n-1.$$

In a similar way, z_n is attracted by z_1.

Formulate this problem as an initial value problem and solve it numerically with the methods implemented in Homework 1. As initial value choose a random distribution of the points within $[0, 1] \times [0, 1]$ (Matlab: `rand()`). Use different values for n and T and illustrate the solution appropriately.