Universität Konstanz *SS 10*
Fachbereich Mathematik und Statistik
S. Volkwein, O. Lass, R. Mancini

Übungen zu **Numerik gewöhnlicher Differentialgleichungen**

http://www.math.uni-konstanz.de/numerik/personen/volkwein/teaching/

Sheet 4 **Submission: 04.06.2010, 12:00 o'clock, Box 13**

Exercise 10 (Homework) (4 Points)

Show that the update formula of the *fourth-order Adams-Bashforth method* to solve the problem $y'(t) = f(t, y)$ is given by

$$y_{k+1} = y_k + \frac{h}{24}(55f_k - 59f_{k-1} + 37f_{k-2} - 9f_{k-3}).$$

Exercise 11

The general form of an m-order *Runge-Kutta method* for the solution of the equation $y'(t) = f(t, y)$ can be represented with the Butcher's tableau. Show that the Heun and trapezoidal methods can be viewed as Runge-Kutta methods and write down their Butcher's tableaux. What do you observe regarding the structure of the tableaux?

Exercise 12

Consider the following second-order differential equation

$$y''(t) - 10y'(t) - 11y(t) = 0 \tag{1}$$

with the initial conditions

$$y(0) = 1, \qquad y'(0) = -1.$$

Show that the solution of this initial value problem is $y(t) = e^{-t}$. Then consider the same equation (1) with initial conditions

$$y(0) = 1 + \varepsilon, \qquad y'(0) = -1$$

with ε small. Calculate analytically the solution of the perturbed initial value problem and generate a plot of the results for comparison. What are your considerations about its stability? What is the effect of this observation if a numerical method is applied to this initial value problem?

Program 3

(10 Points)

Implement a function

```
[y,t] = odesolve4(fun,y0,t,epsilon)
```

for solving ordinary differential equations and systems of ordinary differential equations using step size control, where `fun` is a function handle, `y0` an initial condition, `t = [t0,T]` with `t0` the initial time and `T` the end time and `epsilon` a tolerance. The return values are the solution `y` and the time grid `t`.

As a method for solving the ODE we use the Runge-Kutta method of fourth order. We implement the strategy of computing two steps, one with step size h and one with step size $\frac{h}{2}$. Out of the difference we can compute the error estimate $s(h)\doteq\tilde{\delta}_{j,h}$ and the ratio $q(h)$ which then leads to the decision whether to increase or decrease the step size and to recompute or accept the step. For the implementation the flowchart shown in Figure 1 should be used as a guideline. The fixed parameters should be chosen as follows:

$$h_{\min} = 10^{-4}, \quad h_{\max} = 0.5, \quad \alpha_{\min} = 0.2, \quad \alpha_{\max} = 2, \quad \beta = 0.95$$

The step size should be initialized with $h = h_{max}$. The obtained solver should then be tested using `epsilon`$= 10^{-4}$ on the following problems:

- $y'(t) = -y(t), \quad y(0) = 1, \quad t \in [0, 5]$
- $y_1'(t) = \frac{1}{4}y_1(t) - \frac{1}{100}y_1(t)y_2(t), \quad y_2'(t) = -y_1(t) + \frac{1}{100}y_1(t)y_2(t),$
  $y_1(0) = 80, \quad y_2(0) = 30, \quad t \in [0, 12]$
- $y''(t) = 8(1 - y(t)^2)y'(t) - y(t), \quad y(0) = 2, \quad y'(0) = 0, \quad t = [0, 30]$

Generate plots of the solution and the step size in each iteration. Note that the function `odesolve4` should not open any figure. The code should be well documented and a function description/usage should be accessible by the `help` command.

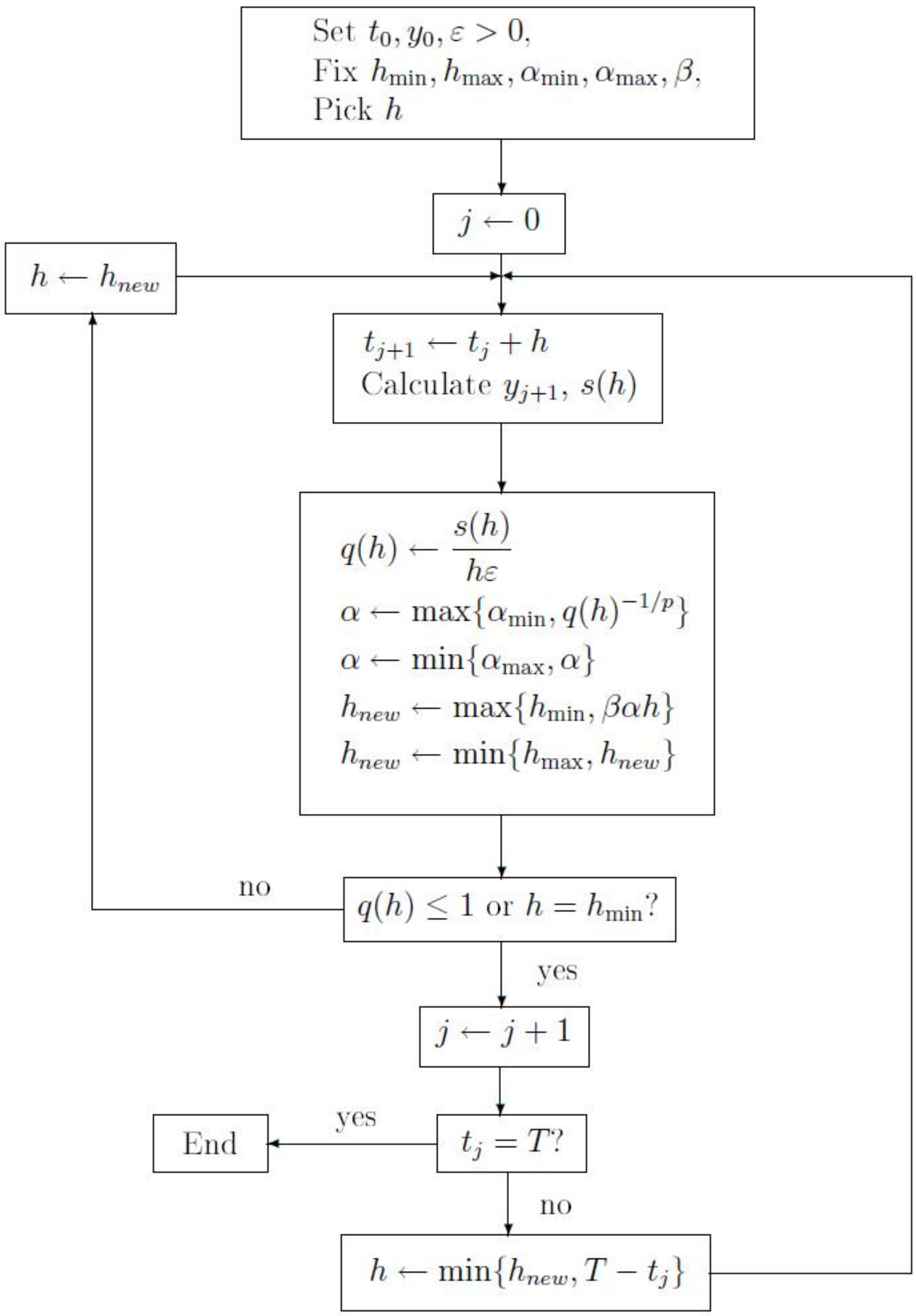

Figure 1: Flowchart for an ODE solver using step size control.