Institute of Scientific Computing
Technical University Braunschweig
Dr. Thilo Moshagen
Dr. Dishi LIu

Summer Term 2018

Advanced Methods for ODEs and DAEs: *Assignment 3*

Exercise 1: **(36 points)**

Consider solving an 1-D Heat equation. I.e. for $0 < x < 1,\ t > 0$ we have

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \tag{1}$$

with initial and boundary conditions:

$$u(x, t = 0) = \sin(x),\ \ u(0, t) = u(1, t) = 0.$$

Let the spacial and temporal domains be discretized by mesh x_m and t_n:

$$x_m = mh,\ \ m = 0, 1, \cdots, M,\ \ h = 1/M;$$

$$t_n = n\kappa,\ \ n = 0, 1, \cdots, N,\ \ \kappa = T_{max}/N;$$

where T_{max} is the maximum time length, and let U_m^n be the solution computed at node (x_m, t_n). We approximate $\frac{\partial^2 u}{\partial x^2}$ by the central difference scheme $(U_{m+1}^n - 2U_m^n + U_{m-1}^n)/h^2$, which transforms the equation (1) into a discretized linear system

$$\frac{d\mathbf{U}^n}{dt} = \mathbf{A}\mathbf{U}^n \tag{2}$$

You can get the matrix $\mathbf{A}$ by the provided downloadable Matlab file.

Write a Python class for s-stage Runge Kutta methods which solves ODE systems as equation (2) (notice that each k_i is a vector now). The class should take the configuration parameters $\mathbf{a} \in \mathbb{R}^{s \times s}$, $\mathbf{c} \in \mathbb{R}^s$ and $\mathbf{b} \in \mathbb{R}^s$ (as would be specified in a Butcher's table) as initializing variables.
And then solves the system (2) by a Gauss-Legendre Runge Kutta method as detailed in the last assignment.