Prof. Dr. Dirk Langemann & Jannis Marquardt, M.Sc.
Institute for Partial Differential Equations

Technische Universität Braunschweig

Assignment 8

- Programming exercises -

Upload your solution until Mo, 12 June 2023, 03:00 pm.

Programming exercise 8.1 **(7 + 3 = 10 points)**

(a) Write a `Matlab` function file `newton2d` which takes as input a function $F : \mathbb{R}^2 \to \mathbb{R}^2$ and an initial guess $\mathbf{x}_0 \in \mathbb{R}^2$. The `newton2d`-function should return a matrix `X`, where the columns of `X` are the vectors which occur after each application of the Newton iteration rule, i.e. `X=(`$\mathbf{x}_0$`, `$\mathbf{x}_1$`, `$\mathbf{x}_2$`, ...)`. Hence, the last column of `X` represents the computed numerical solution of $F(\mathbf{x}) = \mathbf{0}$. The Newton method is supposed to stop after 50 iterations or when

$$\frac{||\mathbf{x}_{i+1} - \mathbf{x}_i||_2}{||\mathbf{x}_i||_2} \leq 10^{-6},$$

whatever occurs first.

Remark: *You have to compute the inverse of the Jacobian ∇F in the process. Compute each component of ∇F separately with the central difference quotient and $h = 10^{-6}$. Remember that one should avoid a matrix inversion during numerical computations at all costs. Hence, the usage of the `Matlab` command `inv()` is prohibited during the computation of $(\nabla F)^{-1}$.*

(b) Now, use `newton2d` to solve the equation

$$F(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \end{bmatrix} = \mathbf{0} \tag{1}$$

with

$$\begin{aligned} f_1(\mathbf{x}) = f_1(x_1, x_2) &= \exp(-\exp(-(x_1 + x_2))) - x_2(1 + x_1^2), \\ f_2(\mathbf{x}) = f_2(x_1, x_2) &= x_1 \cos(x_2) + x_2 \sin(x_1) - \frac{1}{2} \end{aligned}$$

and the initial guess $\mathbf{x}_0 = (0, 0)$. Plot the waypoints $x_i \in \mathbb{R}^2$ on the path which the Newton method takes in order to get closer to a solution of (1). Use a a suitable clipping of $\mathbb{R}^2$ to do so and connect the single waypoints (vectors) by straight lines. Furthermore, print out the computed numerical solution $\tilde{\mathbf{x}}$ and $F(\tilde{\mathbf{x}})$ in the command line.