P. Grohs
Ž. Kereta
C. Schillings

Autumn Term 2015

Numerical Analysis II

ETH Zürich
D-MATH

Exam Winter 2015

Problem 1 Gauss-Radau IIA for a DAE of Order 1 [27 Marks]

We consider the following DAE

$$\begin{cases} \dot{x}_1 &= x_1 x_2 x_3 \,, \\ \dot{x}_2 &= x_5 x_3 + x_2 \,, \\ \dot{x}_3 &= -\cos(x_1) x_4 + \pi \,, \\ \dot{x}_4 &= -x_2 + x_4 \,, \\ 0 &= x_1^2 + 3x_3^2 - 4x_5 + x_2 x_4^3 \,. \end{cases} \tag{1.1}$$

with initial conditions

$$x_1(0) = 1 \,, x_2(0) = 1 \,, x_3(0) = 0 \,, x_4(0) = -1 \,, x_5(0) = 0.$$

(1a) ⚀ Check whether the initial conditions are consistent with the algebraic condition.

(1b) ⚀ Show that the DAE (1.1) has index 1 and state the equivalent ODE explicitly.

(1c) ⚂ Apply the order 3 Gauss-Radau IIA method [NUMODE, Sect. 3.4] to (1.1) and explicitly state the system of equations. Furthermore, set up the equations required for solving the given implicit system for the stages of the method by using Newton's algorithm.

HINT: You are not required to take into account the algebraic condition at this point, only set up the Newton system for the k's (stages of the method), i.e., an equation of the form $\mathbf{G}(\mathbf{k}) = 0$ where $\mathbf{G}$ is some function.

(1d) ⚅ Complete MATLAB templates `firstDAE.m` and `Newton.m`, in which the system of equations from subproblem (1c) is solved using `nNewton` iterations of the Newton method, and plot the solution and the error (distance from the manifold).

HINT: In case of problems with writing the implementation of the required Newton method, you may use the corresponding `pcode`. Bear in mind though, that you will not be awarded full marks unless you complete the template `Newton.m` as well. Furthermore, take into account the fact that your code for the Newton method ought to include information about the algebraic condition, which means you will have to adjust the Newton system obtained in (1c). Further instructions will be given in the template.

Problem 2 Exponential Methods [25 Marks]

Consider the general form of the exponential Runge-Kutta method [NUMODE, Def. 3.7.7]

$$\begin{aligned}
\mathbf{k}_i &:= \varphi(\gamma h\mathbf{A})\left(f(\mathbf{u}_i) + h\mathbf{A}\sum_{j=1}^{i-1}\gamma_{ij}\mathbf{k}_j\right), \quad i = 1,\dots,s, \\
\mathbf{u}_i &:= \mathbf{y}_0 + h\sum_{j=1}^{i-1}\alpha_{ij}\mathbf{k}_j \quad i = 1,\dots,s, \\
\Psi^h\mathbf{y}_0 &:= \mathbf{y}_0 + h\sum_{i=1}^{s} b_i\mathbf{k}_i.
\end{aligned} \tag{2.1}$$

for the autonomous differential equation $\dot{\mathbf{y}} = f(\mathbf{y})$, where $\mathbf{A} := Df(\mathbf{y}_0)$ for some $\mathbf{y}_0$, and

$$\varphi(z) = \frac{\exp(z) - 1}{z}.$$

(2a) ⚄ Take $s = 2$. Determine the equations on the parameters $\alpha_{21}, \gamma_{21}, b_1, b_2$ and γ which need to hold so that the method (2.1) is of (consistency) order 3. Furthermore, show that those equations define a quadratic equation for γ, with $\gamma = \frac{1}{2}$ as one solution, and obtain explicit expressions for the parameters with respect to a single parameter $\alpha_{21} = \alpha$, when we take $\gamma = \frac{1}{2}$.

Let us consider now a specific example of the method (2.1). The exponential Euler method [NUMODE, Eq. (3.7.3)] with constant step size is given by

$$\mathbf{y}_{k+1} = \mathbf{y}_k + h\varphi(h\mathbf{A})f(\mathbf{y}_k), \quad k = 0,\dots,N, \quad \mathbf{A} := Df(\mathbf{y}_k), \tag{2.2}$$

where

$$\varphi(z) = \frac{\exp(z) - 1}{z}.$$

(2b) ⚁ Derive the stability function of (2.2).

(2c) ⚂ Complete the MATLAB template

```
function [t, y] = expEM(T, h),
```

which solves the non-linear initial value problem

$$\begin{pmatrix}\dot{y}_1\\ \dot{y}_2\end{pmatrix} = \dot{\boldsymbol{y}} = \begin{pmatrix} -y_2 \\ -\frac{y_2^2}{y_1} + y_1\log(y_1)\end{pmatrix}, \quad \boldsymbol{y}(0) = \begin{pmatrix} e \\ 0\end{pmatrix},$$

with the exponential Euler method (2.2) using a constant step size.

(2d) ⚁ Complete the template `expEMconv.m` and use it to empirically determine the order of convergence of the method (2.2) by computing the error of the method at the end time, i.e., for $t = 6$, with respect to the exact solution

$$\mathbf{y}(t) = (\exp(\cos(t)), \sin(t)\exp(\cos(t)))^\top$$

given on the time interval `T = [0 6]` and for varying time steps `h`$=2^{-2},\dots,2^{-7}$.

Problem 3 Proximal Operator [22 Marks]

Proximal algorithms are a standard tool for solving convex problems. The fundamental operation involves computing the *proximal operator* of a function. They are often solved very quickly as they can admit closed-form solutions. Even though we usually use them for non-smoooth, large scale, or distributed problems, in this problem we will restrict our attention to C^2 functions, thus simplifying computations and allowing us to reinterpret proximal algorithms and relate them to the topics we know more about.

The proximal operator $\operatorname{prox}_f \colon \mathbf{R}^n \to \mathbf{R}^n$ of a function f is defined by

$$\operatorname*{prox}_f(\mathbf{v}) = \operatorname*{argmin}_{\mathbf{x}} \left(f(\mathbf{x}) + \frac{1}{2}\|\mathbf{x} - \mathbf{v}\|^2 \right).$$

(3a) ⚁ Let $f \in C^2$ be convex and $\lambda > 0$. Show that $\mathbf{y} = \operatorname{prox}_{\lambda f}(\mathbf{v})$ if and only if $\mathbf{y} = \mathbf{v} - \lambda \nabla f(\mathbf{y})$.

(3b) ⚂ Show that the minimiser of a convex function $f \in C^2$ is an attractive fixed point of

$$\dot{\mathbf{x}} = -\nabla f(\mathbf{x}). \tag{3.1}$$

(3c) ⚃ Show that for a convex $f \in C^2$ the iterations

$$\mathbf{x}^{k+1} \colon = \operatorname*{prox}_{\lambda f}(\mathbf{x}^k) \tag{3.2}$$

converge to the minimiser of f as $k \to \infty$ for a starting value $\mathbf{x}^0$ that is sufficiently close to the minimiser of f.

HINT: Preceding subproblems might be helpful.

We will now discuss a special case of proximal minimisation. Consider the function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{A}\mathbf{x} - \mathbf{b}^\top \mathbf{x}, \tag{3.3}$$

where $\mathbf{A}$ is an $n \times n$ symmetric positive definite matrix.

(3d) ⚂ Show that for a regular $\mathbf{A}$ the iterations (3.2) converge to the solution of $\mathbf{A}\mathbf{x} = \mathbf{b}$.

(3e) ⚁ Show that

$$\operatorname*{prox}_{\lambda f}(\mathbf{x}) = \mathbf{x} + (\mathbf{A} + \epsilon \mathbf{I})^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}), \quad \text{with } \epsilon = \frac{1}{\lambda}.$$

(3f) ⚁ Consider the matrix

$$\mathbf{A} = \begin{cases} 2, & \text{if } i = j \\ -1, & \text{if } |i - j| = 1\,. \\ 0, & \text{otherwise.} \end{cases}$$

and a vector $\mathbf{b}$ obtained by evaluating the function $g(x) = -2h^2(\cos(x^2) - 2x^2\sin(x^2))$ on an equidistant grid on $[h, \sqrt{\pi} - h]$ of stepsize h. Find the solution of $\mathbf{Ax} = \mathbf{b}$ by using iterations (3.2), and compute the error with respect to the exact solution by completing the template `IterRefine.m`. Matrices `A` and `I`, and the vectors `b` and `xexact` can be found in the folder and loaded.

Problem 4 Singly Diagonal Implicit Runge-Kutta Methods [26 Marks]

For general implicit Runge-Kutta methods, the k_i's cannot be evaluated succesively since they are coupled in the system of implicit equations that is given for their determination. One way to decouple a nonlinear system is to use diagonal implicit Runge-Kutta methods. A special family of those methods are singly diagonal implicit Runge-Kutta methods (SDIRK), in which the matrix of the method is lower triangular and all the diagonal entries are equal. Continuing in this direction we define the following one parameter family of SDIRK-methods

$$\begin{array}{c|cc} \gamma & \gamma & 0 \\ 1-\gamma & 1-2\gamma & \gamma \\ \hline & 1/2 & 1/2 \end{array}$$

Table 4.1: Singly Diagonal Implicit Runge-Kutta

(4a) Determine the values of γ for which the corresponding Runge-Kutta method is of (consistency) order 3.

(4b) Compute the stability function of the Runge-Kutta method (4.1).

(4c) Complete the template `StabilityRegion.m` and plot the stability region of the Runge-Kutta method (4.1). Using the given plots, for which values of γ from problem (4a) (the values with which the method is of order 3) can we conjecture the corresponding Runge-Kutta method to be A-stable?

(4d) Investigate whether the method is A-stable for $\gamma = 1$. Is it L-stable?

HINT: Notice that in order to investigate A-stability it is sufficient to consider only the case $z = \imath y$, for $y \in \mathbb{R}$. Why?

(4e) Consider the ODE

$$\dot{y} = \cos(y^2) + \frac{y}{4}, \quad y(0) = \frac{1}{2}.$$

and compute its numerical solution with the Runge-Kutta method 4.1 on `T = [0 3]` and for `N = 300` To do so compute the stages k_1 and k_2 by first computing k_1 and then k_2, both by using Newton's algorithm, and complete the template

```
function SDIRK(T, N, gamma).
```

References

[NUMODE] Lecture Slides for the course "Numerical Methods for Ordinary Differential Equations", SVN revision # 63606.