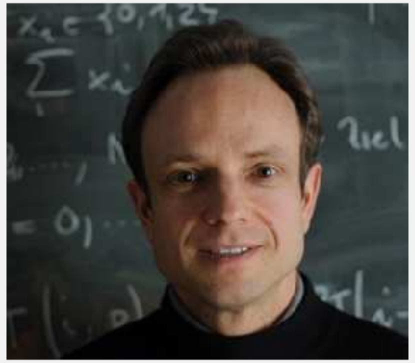
ETH Lecture 401-0674-00L Numerical Methods for Partial Differential Equations

Course Video

Section 7.4: Semi-Implicit Runge-Kutta Methods

Prof. R. Hiptmair, SAM, ETH Zurich

Date: April 7, 2021



Dependencies. [Lecture → Section 7.3]

Duration: minutes

Video and accompanying tablet notes may not match completely!

[Corrections and updates may have been made in tablet notes.]

VII. Single-Step Methods for Stiff Initial-Value Problems

7.4 Semi-Implicit Runge-Kutta Methods

Recall: IRK-SSM for $\dot{y} = f(t, y)$

Definition 7.3.3.1. General Runge-Kutta single step method (*cf.* Def. 6.4.0.9)

For $b_i, a_{ij} \in \mathbb{R}$, $c_i := \sum_{j=1}^{s} a_{ij}$, $i, j = 1, \dots, s$, $s \in \mathbb{N}$, an s-stage Runge-Kutta single step method (RK-SSM) for the IVP (6.1.3.2) is defined by

$$\mathbf{k}_i := \mathbf{f}(t_0 + c_i h, \mathbf{y}_0 + h \sum_{j=1}^{s} a_{ij} \mathbf{k}_j) \;, \quad i = 1, \dots, s \quad , \quad \mathbf{y}_1 := \mathbf{y}_0 + h \sum_{i=1}^{s} b_i \mathbf{k}_i \;.$$

As before, the vectors $\mathbf{k}_i \in \mathbb{R}^N$ are called **increments**.

→ $Ns \times Ns$ non-linear system

Expensive to solve

(e.g. by (simplified) Newton method)

→ Rem. 7.3.3.9

Idea: Use only a fixed *small* number of Newton steps to solve for the $\mathbf{k}_i$, $i = 1, \dots, s$.

Extreme case: use only a single Newton step! Let's try.

Ex. 7.4.0.1 (Semi-implicit Euler single-step method)

▷ Above idea applied to implicit Euler method

ODE : $\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y})$

Impl. Euler: $\mathbf{y}_{k+1}$: $\mathbf{y}_{k+1} = \mathbf{y}_k + h\mathbf{f}(\mathbf{y}_{k+1})$
(timestep h)

$$\Downarrow$$

$$F(\mathbf{y}_{k+1}) := \mathbf{y}_{k+1} - \mathbf{y}_k - h\mathbf{f}(\mathbf{y}_{k+1}) = 0$$

Single Newton step, initial guess $\mathbf{y}_k$:

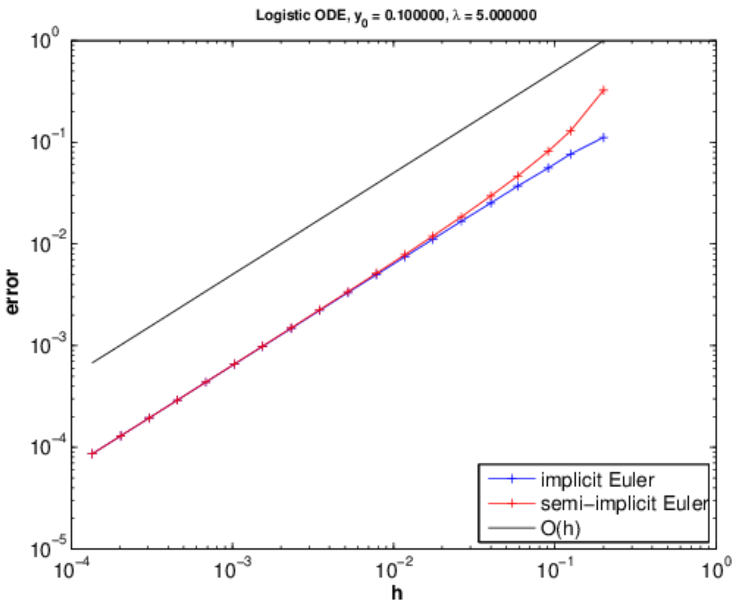
$$\begin{aligned} \mathbf{y}_{k+1} &= \mathbf{y}_k - DF(\mathbf{y}_k)^{-1} F(\mathbf{y}_k) \\ &= \mathbf{y}_k - (I - hD\mathbf{f}(\mathbf{y}_k))^{-1}(-h\mathbf{f}(\mathbf{y}_k)) \end{aligned}$$

$\hat{=}$ semi-implicit Euler method

Exp. 7.4.0.3 (Empiric convergence of semi-implicit Euler method)

IVP : $\dot{y} = \lambda y(1-y)$, $y(0) = 0.1$, $\lambda = 5$

- We run the implicit Euler method (6.2.2.2) and the semi-implicit Euler method (7.4.0.2) with uniform timestep $h = 1/n$, $n \in \{5, 8, 11, 17, 25, 38, 57, 85, 128, 192, 288, 432, 649, 973, 1460, 2189, 3284, 4926, 7389\}$.
- Measured error $\mathrm{err} = \max_{j=1,\dots,n} |y_j - y(t_j)|$

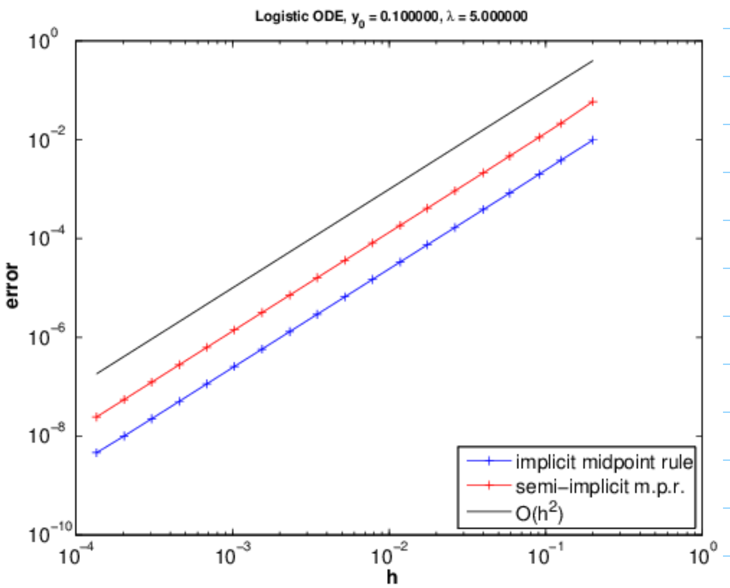


▷ Alg. cvg., rate = 1

Same order & accuracy as impl. Euler method!

③ Exp. 7.4.0.4 (Empiric convergence of semi-implicit midpoint method)

IVP: $\dot{y} = \lambda y(1-y)$, $y(0) = 0.1$, $\lambda = 5$



- Now, implicit midpoint method (6.2.3.3), uniform timesteps $h = 1/n$ as above

  & approximate computation of y_{k+1} by **1 Newton step**, initial guess y_k
- Measured error $\mathrm{err} = \max_{j=1,\dots,n} |y_j - y(t_j)|$

▷ Preserves order = 2 of implicit midpoint method!

Try: Use linearized increment equations for implicit RK-SSM

$$\mathbf{k}_i := \mathbf{f}\Big(\mathbf{y}_0 + h\sum_{j=1}^{s} a_{ij}\mathbf{k}_j\Big)\,,\quad i = 1,\dots,s$$

Linearisation

$$\mathbf{k}_i = \mathbf{f}(\mathbf{y}_0) + h\mathrm{D}\mathbf{f}(\mathbf{y}_0)\Big(\sum_{j=1}^{s} a_{ij}\mathbf{k}_j\Big)\,,\quad i = 1,\dots,s\,. \tag{7.4.0.5}$$

[Drop $O(h^2)$ terms!]

▷ No change for (affine-) linear ODEs

▷ Stability functions not affected by linearization

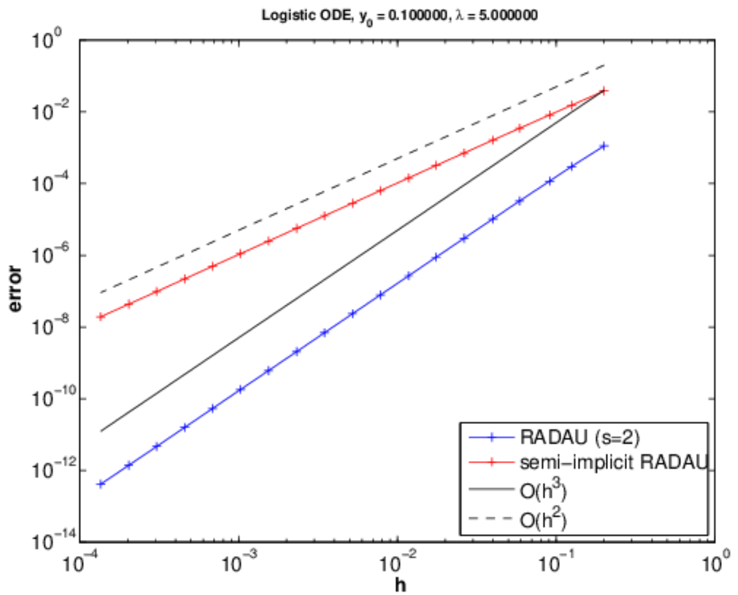
Exp. 7.4.0.6 (Naive semi-implicit Radau method)

IVP: $\dot{y} = \lambda y(1-y)$, $y(0) = 0.1$, $\lambda = 5$

- 2-stage Radau RK-SSM, Butcher scheme

$$\begin{array}{c|cc} \frac{1}{3} & \frac{5}{12} & -\frac{1}{12} \\ 1 & \frac{3}{4} & \frac{1}{4} \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array}\,, \tag{7.4.0.7}$$

  order = 3, see Ex. 7.3.4.21.
- Increments from linearized equations (7.4.0.5)
- We monitor the error through $\mathrm{err} = \max_{j=1,\dots,n} |y_j - y(t_j)|$



▷ Loss of order due to linearization!

(4)

§ 7.4.0.8 (Rosenbrock-Wanner methods)

General formula for ODE $\dot{y} = f(y)$

$$(\mathbf{I} - ha_{ii}\mathbf{J})\mathbf{k}_i = \mathbf{f}(\mathbf{y}_0 + h\sum_{j=1}^{i-1}(a_{ij}+d_{ij})\mathbf{k}_j) - h\mathbf{J}\sum_{j=1}^{i-1} d_{ij}\mathbf{k}_j\,, \quad \mathbf{J} = \mathrm{D}\mathbf{f}(\mathbf{y}_0)\,,$$
$$\mathbf{y}_1 := \mathbf{y}_0 + \sum_{j=1}^{s} b_j\mathbf{k}_j\,. \qquad (7.4.0.9)$$

$\rightarrow$ Matrix "to be inverted":

$\leftrightarrow$ Solve s $N \times N$ LSE

Determine coefficients:

- Solve order conditions $\rightarrow$ achieve desired order
- Ensure A/L-stability

Please answer review questions

After you have watched the video take a short break and then try to answer the

review questions 7.4.0.10

Please avoid consulting the lecture document or tablet notes.

5