Advanced Numerical Analysis

Lecture Notes

Prof. Daniel Kressner
EPFL / SB / MATHICSE / ANCHP

Spring 2015

February 19, 2015

Remarks

- Although not necessary, you may find it helpful to complement these lecture notes with additional literature. These lecture notes are based on material from

  [**DB**] P. Deuflhard and F. Bornemann. *Scientific computing with ordinary differential equations.* Texts in Applied Mathematics, 42. Springer, 2002.

  [**E**] F. Eisenbrand. Lecture notes on Discrete Optimization, 2013. Available from `http://disopt.epfl.ch/Opt2013`.

  [**HLW**] E. Hairer, Ch. Lubich, and G. Wanner. *Geometric numerical integration.* Structure-preserving algorithms for ordinary differential equations. Springer Series in Computational Mathematics, 31. Springer, 2010.

  [**HNW**] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving ordinary differential equations. I.* Nonstiff problems. Second edition. Springer Series in Computational Mathematics, 8. Springer, 1993.

  [**HW**] E. Hairer and G. Wanner. *Solving ordinary differential equations. II.* Stiff and differential-algebraic problems. Second revised edition. Springer Series in Computational Mathematics, 14. Springer-Verlag, Berlin, 2010.

  [**K**] C. T. Kelley. *Iterative Methods for Optimization.* SIAM, 1999.

  [**Kr**] J. Krieger. *Equations Différentielles Ordinaires*, 2014. Lecture notes available from `http://pde.epfl.ch/page-44573-en.html`.

  [**N**] Yu. Nesterov. *Introductory Lectures on Convex Optimization. A Basic Course*, 2004.

  [**NW**] J. Nocedal and S. J. Wright. *Numerical optimization.* Second edition. Springer Series in Operations Research and Financial Engineering. Springer, 2006.

  [**UU**] M. Ulbrich and S. Ulbrich. *Nichtlineare Optimierung*, Birkhäuser Verlag, 2012.

  Please note: None of these books is needed to prepare for the exam. These lecture notes are entirely sufficient.

- Sections or paragraphs marked by a $\star$ contain complementary material that is *not* part of the exam. You may skip reading these sections, but it may be worthwhile not to do so.

- If you have any suggestions or spot mistakes, please send a message to

`daniel.kressner@epfl.ch`
or `michael.steinlechner@epfl.ch`

Chapter 1

Introduction and Review

The first part of this course starts where the course *Analyse Numérique* ended: the numerical solution of ordinary differential equations (ODEs). In this first chapter, we will therefore very briefly review (and slightly extend) some of the material covered in *Analyse Numérique*.

In general, an **initial value problem** *[problème de Cauchy]* takes the form

$$\begin{cases} \mathbf{y}'(t) = f\big(t, \mathbf{y}(t)\big) & t \in I, \\ \mathbf{y}(t_0) = \mathbf{y}_0, \end{cases} \tag{1.1}$$

where:

- $t \in \mathbb{R}$ denotes **time**.
- I is a real interval containing the initial time t_0.
- $\mathbf{y}(t) \in D \subset \mathbb{R}^d$ is the desired solution vector at time t. It contains d time-dependent functions $y_1(t), \ldots, y_d(t)$. Often, $\mathbf{y}(t)$ is called the **state** *[état]* of the ODE. The open set D containing all admissible states is called the **phase space** *[espace des phases]* and $\Omega := I \times D \subset \mathbb{R}^{d+1}$ is called the **augmented phase space**.
- $\mathbf{y}'(t)$ is the entry-wise derivative of $\mathbf{y}$ with respect to time:

$$\mathbf{y}'(t) = \begin{pmatrix} y_1'(t) \\ \vdots \\ y_d'(t) \end{pmatrix} = \begin{pmatrix} \frac{\partial}{\partial t} y_1(t) \\ \vdots \\ \frac{\partial}{\partial t} y_d(t) \end{pmatrix}.$$

- $f : \Omega \to \mathbb{R}^d$ is a continuous function depending on time and state. If there is no direct dependence on time (for example, $f(t, y(t)) = y(t)^2$) then the ODE is called **autonomous** *[équation différentielle autonome]* and we can write $f(\mathbf{y}(t))$ instead of $f(t, \mathbf{y}(t))$.

1.1 Existence and uniqueness of solutions

In the following, we briefly recall classical results on the existence of a solution $\mathbf{y} \in C^1(I, \mathbb{R}^d)$ to the initial value problem (1.1). See Lectures 2 and 3 of [Kr].

Theorem 1.1 (Peano existence theorem) *Consider an initial value problem* (1.1) *with* $f \in C^0(\Omega, \mathbb{R}^d)$ *and initial data* $(t_0, \mathbf{y}_0) \in \Omega$. *Then there is at least one solution* $\mathbf{y} \in C^1([t_0, t_0 + \alpha])$ *for some* $\alpha > 0$.

Uniqueness requires a condition on f that is stronger than continuity.

Definition 1.2 *Consider a mapping* $f \in C(\Omega, \mathbb{R})$ *with* $\Omega = I \times D \subset \mathbb{R} \times \mathbb{R}^d$. *Then* f *is called* **locally Lipschitz continuous** *[localement lipschitzienne] on* Ω *with respect to the state variable if for any compact subset* $K \subset \Omega$ *there exists a Lipschitz constant* L_K *such that*

$$\|f(t, \mathbf{y}) - f(t, \widetilde{\mathbf{y}})\| \leq L_K \|\mathbf{y} - \widetilde{\mathbf{y}}\| \qquad \forall (t, \mathbf{y}), (t, \widetilde{\mathbf{y}}) \in K.$$

In the definition above, $\|\cdot\|$ denotes an arbitrary vector norm.

Theorem 1.3 (Picard–Lindelöf theorem) *Consider an initial value problem* (1.1) *with* $f \in C^0(\Omega, \mathbb{R}^d)$ *locally Lipschitz continuous on* Ω *with respect to the state variable, and* $(t_0, x_0) \in \Omega$. *Then there exists* $\alpha > 0$ *such that the following holds: There exists one and only one solution* $\mathbf{y} \in C^1([t_0, t_0 + \alpha])$ *of the initial value problem on the interval* $[t_0, t_0 + \alpha]$.

Often, solutions can be continued beyond the interval $[t_0, t_0 + \alpha]$ guaranteed by Theorem 1.3. For harmless ODEs, this continuation may extend until $T = \infty$. If the extension of a solution $\mathbf{y}(t)$ breaks down at some point T then because of one of the following two reasons:

1. **Blow up** of the solution: $\lim\limits_{\substack{t \to T \\ t < T}} \|\mathbf{y}(t)\| = \infty$.
2. **Collapse** of the solution: $\lim\limits_{\substack{t \to T \\ t < T}} \operatorname{dist}\big((t, \mathbf{y}(t)), \partial\Omega\big) = 0$.

First, an example for blow up.

Example 1.4 Let $\Omega = \mathbb{R} \times \mathbb{R}$ and consider

$$y' = y^2, \qquad y(0) = 1.$$

Then, by separation of variables, we obtain the solution $y(t) = \frac{1}{1-t}$ on the interval $]-\infty < t < 1$. Hence, the solution blows up at time $T = 1$. Let us apply the MATLAB function `ode23`, which uses an adaptive time stepping strategy, to this IVP:

MATLAB

```
ode23(@blowup,[0,2],1)

function f = blowup(t,y)
f = y.^2;
```

Figure 1.1 shows that the time of blow up is detected quite accurately, and the method actually refuses to continue beyond $T = 1$, delivering the following error message:

```
Warning: Failure at t=1.001616e+00.  Unable to meet integration
tolerances without reducing the step size below the smallest
value allowed (3.552714e-15) at time t.
```

In contrast, the Euler method fails to accurately detect the time of blow up. ◇

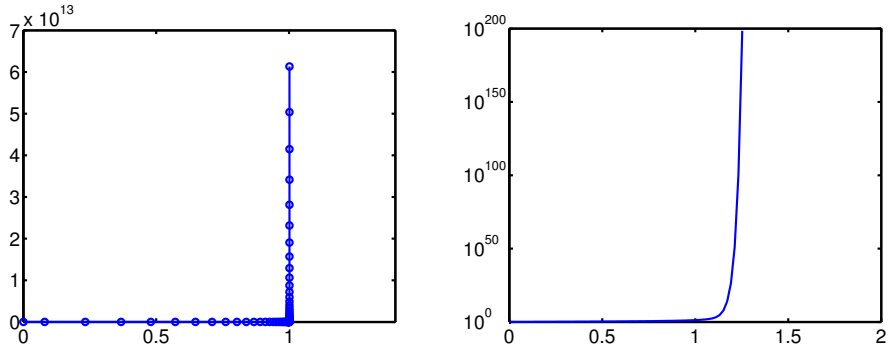


Figure 1.1. *Left plot:* `ode23` *applied to Example 1.4. Right plot: Euler method with step size* $h = 0.02$ *applied to Example 1.4.*

Collapse happens when the solution approaches the boundary of the augmented phase space Ω in finite time. It is important to note that this notion makes most sense when Ω has been maximally chosen, according to the properties of f.

Example 1.5 Let $\Omega = \mathbb{R}\times]-\infty, 0[$ and consider

$$y' = -y^{-1/2}, \qquad y(0) = 1.$$

Up to $t < 2/3$ the unique solution is given by $y(t) = (1 - 3t/2)^{2/3}$. At $t = 2/3$, however, the trajectory runs into the boundary of the phase space and collapses due to the singularity of f at 0. ◇

In the previous example, the collapse could be easily detected as the solution approaches $-\infty$ as $t \to 2/3$. A more subtle situation occurs when the solution remains bounded even when approaching the boundary of the phase space.

Example 1.6 Let $\Omega = \mathbb{R}\times]-\infty, 0[$ and consider

$$y' = \sin(1/y) - 2, \qquad y(0) = 1.$$

It can be shown that the solution remains bounded. The function $\sin(1/y)$ is continuously differentiable on the phase space, but cannot be extended continuously to a larger domain. It turns out that the solution collapses at time $T \approx 0.76741$. Figure 1.2 shows that both `ode23` and the Euler method fail to notice the collapse.

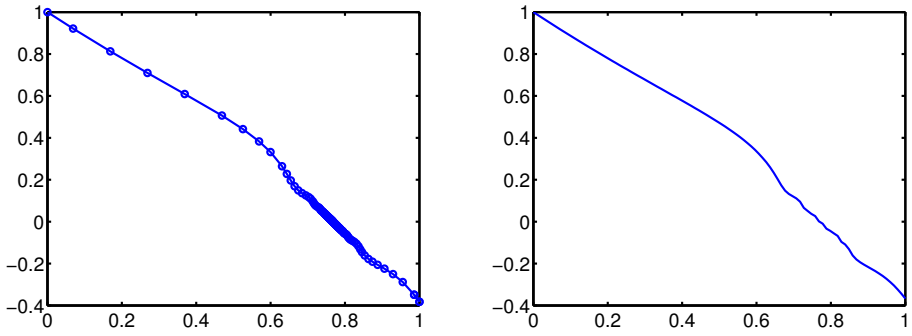


Figure 1.2. *Left plot:* `ode23` *applied to Example 1.6. Right plot: Euler method with step size* $h = 0.01$ *applied to Example 1.6.*

(At least, the very small steps taken by `ode23` when approaching T indicate a problematic behavior.) In fact, the highly oscillatory behavior causing the collapse of the solution is "averaged out" by the numerical methods. This is a good example where mathematical analysis cannot be replaced by numerics. ◇

1.2 Simple one-step methods

The **(forward) Euler method** *[méthode d'Euler (explicite/progressive)]* is the simplest numerical method for solving an IVP. Given a small step size h, the idea is to replace $\dot{\mathbf{y}}(t)$ in the differential equation $\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t))$ by the forward difference. At $t = t_0$, this yields

$$\frac{1}{h}(\mathbf{y}_1 - \mathbf{y}_0) = f(t_0, \mathbf{y}_0),$$

where $t_1 = t_0 + h$ and $\mathbf{y}_1 \approx \mathbf{y}(t_1)$. Rearranging this equation gives

$$\mathbf{y}_1 = \mathbf{y}_0 + hf(t_0, \mathbf{y}_0). \tag{1.2}$$

Repeating this procedure at $t_1, t_2, \ldots$ gives the following algorithm:

1. Choose step size $h = (T - t_0)/N$.
2. for $j = 0, 1, \ldots, N-1$ do

$$t_{j+1} = t_j + h, \qquad \mathbf{y}_{j+1} = \mathbf{y}_j + hf(t_j, \mathbf{y}_j). \tag{1.3}$$

For one-step methods, it is sufficient to specify the first step (1.2); the iteration (1.3) immediately follows.

The **backward Euler method** *[méthode d'Euler (implicite/retrograde)]* is obtained in a similar way as the forward Euler method, but using the backward difference instead of the forward difference. At $t = t_1$, this yields

$$\frac{1}{h}\big(\mathbf{y}_1 - \mathbf{y}_0\big) = f(t_1, \mathbf{y}_1),$$

where $t_1 = t_0 + h$ and $\mathbf{y}_1 \approx \mathbf{y}(t_1)$. Rearranging this equation gives

$$\mathbf{y}_1 = \mathbf{y}_0 + hf(t_1, \mathbf{y}_1).$$

In contrast to forward Euler, we need to solve a system of nonlinear equations to determine $\mathbf{y}_1$! The solution of such nonlinear systems will be discussed in some detail for more general implicit methods in Chapter 3.

As a final for a simple one-step method, we mention the **explicit trapezoidal method**

$$\mathbf{y}_1 = \mathbf{y}_0 + \frac{h}{2}\big(f(t_0, \mathbf{y}_0) + f(t_1, \mathbf{y}_0 + hf(t_0, \mathbf{y}_0))\big). \tag{1.4}$$

1.3 Consistency of one-step methods

To discuss the **local error** committed by a single step of the methods introduced above, it suffices to consider the first step:

$$\mathbf{e}(h) = \mathbf{y}(t_0 + h) - \mathbf{y}_1. \tag{1.5}$$

(Note that $\mathbf{e}(h)$ corresponds to the quantity ε_1 in *Analyse Numérique*.) As a minimal requirement, $\mathbf{e}(h)$ should converge faster to zero than h, that is

$$\mathbf{e}(h) = o(h).$$

A method satisfying this property is called **consistent**. More generally, we have the following definition.

Definition 1.7 *A method has (consistency)* **order** p *if* $\|\mathbf{e}(h)\| = O(h^{p+1})$ *holds for all sufficiently smooth* f.

It is crucial to have the exponent $p+1$ (and not p) in the definition of consistency. The underlying rationale is that we lose one power in the eventual convergence result because we have to perform $O(1/h)$ steps to reach the endpoint of an interval of constant length.

In principle, it is a simple matter to determine the order of a method by comparing the Taylor expansions of $\mathbf{y}(t_0+h)$ and $\mathbf{y}_1$ as functions of h. Assuming that f is sufficiently smooth, we obtain for the forward Euler method that

$$\begin{aligned}\mathbf{y}(t_0+h) &= \mathbf{y}(t_0) + h\mathbf{y}'(t_0) + O(h^2) = \mathbf{y}_0 + h\mathbf{f}(t_0, y(t_0)) + O(h^2),\\ \mathbf{y}_1 &= \mathbf{y}_0 + h\mathbf{f}(t_0, y(t_0)).\end{aligned}$$

Hence, $\|\mathbf{e}(h)\| = O(h^2)$ and therefore the forward Euler method has order 1.

For the backward Euler method, $\mathbf{y}_1(h) = \mathbf{y}_0 + hf(t_0+h, \mathbf{y}_1(h))$. By applying the implicit function theorem, we obtain $\mathbf{y}_1'(0) = f(t_0, \mathbf{y}_1(0)) = f(t_0, \mathbf{y}_0)$. Hence, we obtain the Taylor expansion

$$\mathbf{y}_1 = \mathbf{y}_0 + h\mathbf{f}(t_0, y(t_0)) + O(h^2),$$

which implies that the backward Euler method has order 1 as well.

For (reasonable) one-step methods, consistency of order p implies convergence of order p. This has already been discussed in *Analyse Numérique*; we will have a somewhat more elaborate discussion in Section 2.2.2 on this topic.

Chapter 2

Explicit Runge-Kutta methods

The goal of this chapter is to construct explicit one-step methods of higher order. By far, the most popular methods of this type are Runge-Kutta methods, which – as we will see – are behind the MATLAB commands `ode23` and `ode45`.

2.1 A first Runge-Kutta method

The starting point for the development of simple Runge-Kutta methods is the reformulation of the IVP (1.1) on the interval $[t_0, t_0 + h]$ as the integral equation

$$\mathbf{y}(t_0 + h) = \mathbf{y}_0 + \int_{t_0}^{t_0+h} f(t, \mathbf{y}(t)) \, \mathrm{d}t. \tag{2.1}$$

When approximating the integral by a rectangle with area $h \times f(t_0, \mathbf{y}(t_0))$, we obtain

$$\mathbf{y}(t_0 + h) \approx \mathbf{y}_1 = \mathbf{y}_0 + hf(t_0, \mathbf{y}(t_0)),$$

which is the forward Euler method, so nothing new is gained. When using the midpoint rule, we approximate the integral in (2.1) by a rectangle with area $h \times f(t_0 + h/2, \mathbf{y}(t_0 + h/2))$, leading to

$$\mathbf{y}(t_0 + h) \approx \mathbf{y}_0 + hf\left(t_0 + \frac{h}{2}, \mathbf{y}(t_0 + \frac{h}{2})\right). \tag{2.2}$$

This does not look very promising either, as we do not know the value for $\mathbf{y}(t_0 + \frac{h}{2})$. In the absence of a better idea, we approximate it by one step of the forward Euler method with step size $h/2$. Inserted into (2.2), this leads to **Runge's method**:

$$\begin{aligned} \mathbf{k}_1 &= f(t_0, \mathbf{y}_0) \\ \mathbf{k}_2 &= f\left(t_0 + \frac{h}{2}, \mathbf{y}_0 + \frac{h}{2}\mathbf{k}_1\right) \\ \mathbf{y}_1 &= \mathbf{y}_0 + h\mathbf{k}_2. \end{aligned} \tag{2.3}$$

On first sight, it appears contradictory to invoke the forward Euler method for designing a higher-order method. However, the crucial point is that $\mathbf{k}_2$ gets multiplied by h, which weakens the impact of the error made when approximating $\mathbf{y}(t_0 + \frac{h}{2})$ within $\mathbf{k}_2$.

As a warm-up to the error analysis of general Runge-Kutta methods, let us have a look at the Taylor expansion of $\mathbf{y}_1$ in (2.3) as a function of h. When attempting to do so, we have to face the unpleasant situation that we have to expand both arguments of the function f appearing in $\mathbf{k}_2$. We will therefore need to make use of the **multivariate Taylor expansion**.

Remark 2.1 For a function $f \in C^k(\Omega, \mathbb{R}^d)$ with an open set $\Omega \subset \mathbb{R}^m$, the k**th derivative** $f^{(k)}(\mathbf{x})$ at $\mathbf{x} \in \Omega$ applied to $\mathbf{h}_1, \ldots, \mathbf{h}_k \in \mathbb{R}^m$ is defined as

$$f^{(k)}(\mathbf{x}) \cdot (\mathbf{h}_1, \ldots, \mathbf{h}_k) = \sum_{i_1,\ldots,i_k=1}^{m} \frac{\partial^k f(\mathbf{x})}{\partial x_{i_1} \cdots \partial x_{i_k}} h_{1,i_1} \ldots h_{k,i_k}. \tag{2.4}$$

Thus,

$$f^{(k)}(\mathbf{x}) : \underbrace{\mathbb{R}^m \times \cdots \times \mathbb{R}^m}_{k \text{ times}} \to \mathbb{R}^d,$$

is a (symmetric) k-linear map.

For $\mathbf{h}_1 = \cdots = \mathbf{h}_k \equiv \mathbf{h}$, the sum in (2.4) can be represented in more compact form. Let $\alpha = (\alpha_1, \ldots, \alpha_m)$, where each $0 \le \alpha_j \le k$ counts the differentiations with respect to the jth variable x_j. By the usual multi-index notation, we define

$$|\alpha| := \alpha_1 + \cdots + \alpha_m, \quad \frac{\partial^{|\alpha|} f}{\partial \mathbf{x}^\alpha} := \frac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1} \cdots \partial x_m^{\alpha_m}}, \quad \mathbf{h}^\alpha := h_1^{\alpha_1} \ldots h_m^{\alpha_m}.$$

Each α corresponds to $\frac{k!}{\alpha!} = \frac{k!}{\alpha_1! \cdots \alpha_m!}$ terms in the sum of (2.4), and hence (2.4) is equivalent to

$$f^{(k)}(\mathbf{x}) \cdot (\mathbf{h}, \ldots, \mathbf{h}) = \sum_{|\alpha|=k} \frac{k!}{\alpha!} \frac{\partial^{|\alpha|} f(\mathbf{x})}{\partial \mathbf{x}^\alpha} \mathbf{h}^\alpha. \tag{2.5}$$

◇

Theorem 2.2 *Let $f \in C^{p+1}(\Omega, \mathbb{R}^d)$ with an open set Ω and $\mathbf{x} \in \Omega$. Then*

$$f(\mathbf{x} + \mathbf{h}) = \sum_{k=0}^{p} \frac{1}{k!} f^{(k)}(\mathbf{x}) \cdot (\mathbf{h}, \ldots, \mathbf{h}) + O(\|\mathbf{h}\|^{p+1})$$

for all sufficiently small $\mathbf{h} \in \mathbb{R}^m$.

Theorem 2.2 is proven by applying the usual univariate Taylor expansion in each coordinate direction separately and collecting all terms.

Example 2.3 For $d = 1$ and $p = 2$, Theorem 2.2 results in

$$\begin{aligned} f(\mathbf{x}+\mathbf{h}) &= f(\mathbf{x}) + f'(\mathbf{x})\cdot\mathbf{h} + \frac{1}{2}f''(\mathbf{x})\cdot(\mathbf{h},\mathbf{h}) + O(\|\mathbf{h}\|^3) \\ &= f(\mathbf{x}) + \nabla f(\mathbf{x})\cdot\mathbf{h} + \frac{1}{2}\mathbf{h}^\mathsf{T} H(\mathbf{x})\mathbf{h} + O(\|\mathbf{h}\|^3), \end{aligned}$$

where ∇f is the gradient and $H = \left(\frac{\partial^2 f}{\partial x_i \partial x_j}\right)_{i,j=1}^{m}$ is the Hessian *[matrice hessienne]* of f. ◇

For simplicity, we consider the initial value problem (1.1) for an *autonomous* differential equation:

$$\begin{cases} \mathbf{y}'(t) = f\big(\mathbf{y}(t)\big) & t \geq t_0, \\ \mathbf{y}(t_0) = \mathbf{y}_0. \end{cases} \tag{2.6}$$

Let us now recursively construct a third-order Taylor expansion for the exact solution $\mathbf{y}(t_0 + h)$. The first-order Taylor expansion is given by

$$\mathbf{y}(t_0+h) = \mathbf{y}_0 + h\mathbf{y}'(t_0) + O(h^2) = \mathbf{y}_0 + hf(\mathbf{y}_0) + O(h^2).$$

Inserting this into the differential equation (2.6) gives

$$\begin{aligned} \frac{\partial}{\partial h}\mathbf{y}(t_0+h) &= f\big(\mathbf{y}(t_0+h)\big) = f\big(\mathbf{y}_0 + hf(\mathbf{y}_0) + O(h^2)\big) \\ &= f + hf'f + O(h^2), \end{aligned}$$

where we have used the multivariate Taylor expansion from Theorem 2.2 and omitted the argument $\mathbf{y}_0$ to save space. Integration of this relation with respect to h gives the second-order Taylor expansion

$$\mathbf{y}(t_0+h) = \mathbf{y}_0 + hf + \frac{h^2}{2}f'f + O(h^3).$$

Let us repeat this process:

$$\begin{aligned} \frac{\partial}{\partial h}\mathbf{y}(t_0+h) &= f\left(\mathbf{y}_0 + hf + \frac{h^2}{2}f'f + O(h^3)\right) \\ &= f + f'\left(hf + \frac{h^2}{2}f'f\right) + \frac{1}{2}f''(hf, hf) + O(h^3) \\ &= f + hf'f + \frac{h^2}{2}\big(f'f'f + f''(f,f)\big) + O(h^3). \end{aligned}$$

By integration, the third-order Taylor expansion follows:

$$\mathbf{y}(t_0+h) = \mathbf{y}_0 + hf + \frac{h^2}{2}f'f + \frac{h^3}{6}\big(f'f'f + f''(f,f)\big) + O(h^4). \tag{2.7}$$

This will be compared with the Taylor expansion of $\mathbf{y}_1$, produced by applying (2.3) to (2.6), as a function of h:

$$\begin{aligned} \mathbf{y}_1 &= \mathbf{y}_0 + hf\left(\mathbf{y}_0 + \frac{h}{2}f(\mathbf{y}_0)\right) \\ &= \mathbf{y}_0 + hf + \frac{h^2}{2}f'f + \frac{h^3}{8}f''(f,f) + O(h^4). \end{aligned} \tag{2.8}$$

Subtracting this from the expansion (2.7) of the exact solution gives

$$\mathbf{e}(h) = \mathbf{y}(t_0+h) - \mathbf{y}_1 = \frac{h^3}{6}\big(f'f'f + f''(f,f)\big) - \frac{h^3}{8}f''(f,f) + O(h^4)$$

Hence, the local error satisfies $\|\mathbf{e}(h)\| = O(h^3)$, provided that all second partial derivatives of f remain bounded. This shows that the order of Runge's method is 2.

2.2 General form of explicit Runge-Kutta methods

Definition 2.4 *A method of the form*

$$\begin{aligned}
\mathbf{k}_1 &= f(t_0 + c_1 h, \mathbf{y}_0)\\
\mathbf{k}_2 &= f(t_0 + c_2 h, \mathbf{y}_0 + h a_{21}\mathbf{k}_1)\\
\mathbf{k}_3 &= f(t_0 + c_3 h, \mathbf{y}_0 + h a_{31}\mathbf{k}_1 + h a_{32}\mathbf{k}_2)\\
&\vdots\\
\mathbf{k}_s &= f\Big(t_0 + c_s h, \mathbf{y}_0 + h\sum_{\ell=1}^{s-1} a_{s\ell}\mathbf{k}_\ell\Big),\\
\mathbf{y}_1 &= \mathbf{y}_0 + h\sum_{i=1}^{s} b_i \mathbf{k}_i,
\end{aligned}$$

*with all coefficients $a_{i\ell}, b_i, c_i$ being real numbers, is called an s-**stage explicit Runge-Kutta method**.*

The coefficients defining a Runge-Kutta method can be organized compactly in a so-called **Butcher tableau**:

$$\begin{array}{c|c} \mathbf{c} & A \\ \hline & \mathbf{b}^T \end{array} := \begin{array}{c|ccccc} c_1 & 0 & \cdots & \cdots & 0\\ c_2 & a_{21} & \ddots & & \vdots\\ \vdots & \vdots & \ddots & \ddots & \vdots\\ c_s & a_{s1} & \cdots & a_{s,s-1} & 0\\ \hline & b_1 & \cdots & b_{s-1} & b_s \end{array}$$

Here, A is strictly lower triangular $s \times s$ matrix and $\mathbf{b}, \mathbf{c}$ are vectors of length s. All explicit one-step methods we have discussed so far are special cases of the Runge-Kutta method:

Forward Euler: $\begin{array}{c|c} 0 & 0\\ \hline & 1 \end{array}$ Explicit Trapezoidal: $\begin{array}{c|cc} 0 & 0 & 0\\ 1 & 1 & 0\\ \hline 0 & \frac{1}{2} & \frac{1}{2} \end{array}$

Runge's method:

0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0
0	0	1

Another example for an explicit Runge-Kutta method is the **classical Runge-Kutta method** (RK4) defined by the tableau

0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0	0	0
$\frac{1}{2}$	0	$\frac{1}{2}$	0	0
1	0	0	1	0
	$\frac{1}{6}$	$\frac{2}{6}$	$\frac{2}{6}$	$\frac{1}{6}$

According to Definition 2.4, this leads to the following algorithm.

Classical Runge-Kutta method:

1. Choose step size $h = (T - t_0)/N$.

2. for $j = 0, 1, \ldots, N-1$ do
   $\mathbf{k}_1 = f(t_j, \mathbf{y}_j)$
   $\mathbf{k}_2 = f(t_j + h/2, \mathbf{y}_j + h/2\,\mathbf{k}_1)$
   $\mathbf{k}_3 = f(t_j + h/2, \mathbf{y}_j + h/2\,\mathbf{k}_2)$
   $\mathbf{k}_4 = f(t_j + h, \mathbf{y}_j + h\,\mathbf{k}_3)$
   $\mathbf{y}_{j+1} = \mathbf{y}_j + \frac{h}{6}\mathbf{k}_1 + \frac{2h}{6}\mathbf{k}_2 + \frac{2h}{6}\mathbf{k}_3 + \frac{h}{6}\mathbf{k}_4$
   $t_{j+1} = t_j + h$
   end for

What is so special about the classical Runge-Kutta method? When designing an s-stage Runge-Kutta method, one goal could be to attain the maximal possible order. As we will discuss later, there is a certain limit which order can be achieved, but certainly it cannot become larger than s (see Lemma 2.5 below). One could now proceed as in Section 2.1 and compare the Taylor expansion of the exact solution with the Taylor expansion of $\mathbf{y}_1$ produced by (2.4). This appears to be a tedious task for larger values of s, but we will discuss a clever way of organizing it in the the following section. Matching both expansions up to a certain order leads to a system of nonlinear equations in the coefficients defining the Runge-Kutta method. Finding all solutions to this system by hand is a nearly impossible task, except for tiny s. For example, the case $s = 4$ is covered in Section II.1 of [HNW], giving a fairly general parametrization of 4-stage explicit Runge-Kutta methods having order 4.

2.2.1 Order conditions for Runge-Kutta methods

The ultimate goal of this section is to develop an elegant way that avoids the tediousness of Taylor expansions when analyzing the order of a Runge-Kutta method. Before, we will make some basic observations.

Lemma 2.5 *If an s-stage explicit Runge-Kutta method has order p for all $f \in C^\infty(\Omega, \mathbb{R}^d)$, then $p \leq s$.*

Proof. We apply the Runge-Kutta method to the scalar IVP $y'(t) = y(t), y(0) = y_0 = 1$. Then each stage k_i is a polynomial of degree at most $i-1$ in h. Hence, y_1 is a polynomial of degree at most s in h. Consequently, the $(s+1)$th term $\frac{h^{s+1}}{(s+1)!}$ in the Taylor expansion of the exact solution $y(h) = e^h$ cannot be matched by y_1. Hence, the order of the Runge-Kutta method is at most s. ∎

Lemma 2.6 *An explicit Runge-Kutta method is consistent for all $f \in C(\Omega, \mathbb{R}^d)$ if and only if*

$$\sum_{i=1}^{s} b_i = 1. \tag{2.9}$$

Proof. The first term in the expansion of $\mathbf{y}_1$ takes the form

$$\mathbf{y}_1 = \mathbf{y}_0 + h \sum_{i=1}^{s} b_i f(t_0, \mathbf{y}_0) + \cdots$$

This matches the first two terms in the Taylor expansion (2.7) if and only if (2.9) is satisfied. ∎

The analysis greatly simplifies if we only need to consider autonomous IVPs, as we did in Section 2.1. There is a simple trick to transform (1.1) to autonomous form by appending t to the variables:

$$\begin{pmatrix} t \\ \mathbf{y} \end{pmatrix}' = \begin{pmatrix} 1 \\ f(t, \mathbf{y}) \end{pmatrix}, \qquad \begin{pmatrix} t \\ \mathbf{y} \end{pmatrix}_{t=t_0} = \begin{pmatrix} t_0 \\ \mathbf{y}_0 \end{pmatrix}. \tag{2.10}$$

This trick makes perfect sense if we have a correspondence between the Runge-Kutta method applied to (2.10) and applied to (1.1). Ideally, the approximation produced by one step of the Runge-Kutta method applied to (2.10) takes the form

$$\begin{pmatrix} t_0 + h \\ \mathbf{y}_1 \end{pmatrix}.$$

A Runge-Kutta method having this property is called **invariant under autonomization**.

Lemma 2.7 *An explicit Runge-Kutta method is invariant under autonomization if and only if it is consistent and satisfies*

$$c_i = \sum_{j=1}^{i-1} a_{ij}, \qquad i = 1, \ldots, s.$$

Proof. The stages of the Runge-Kutta method applied to (2.10) take the form

$$\begin{aligned}\begin{pmatrix}\theta_i\\ \hat{\mathbf{k}}_i\end{pmatrix} &= \begin{pmatrix}1\\ f\Big(t_0 + h\sum_j a_{ij}\theta_j, \mathbf{y}_0 + h\sum_j a_{ij}\hat{\mathbf{k}}_j\Big)\end{pmatrix}\\ &= \begin{pmatrix}1\\ f\Big(t_0 + h\sum_j a_{ij}, \mathbf{y}_0 + h\sum_j a_{ij}\hat{\mathbf{k}}_j\Big).\end{pmatrix}\end{aligned}$$

Clearly, we have $\hat{\mathbf{k}}_i = \hat{k}_i$ for general f if and only if $\sum_j a_{ij} = c_i$. The approximate solution is then given by

$$\begin{pmatrix}t_0 + h\sum_i b_i\theta_i\\ \mathbf{y}_0 + h\sum_i b_i\mathbf{k}_i\end{pmatrix} = \begin{pmatrix}t_0 + h\sum_i b_i\\ \mathbf{y}_1\end{pmatrix}.$$

Hence, invariance holds if we additionally require that $\sum_i b_i = 1$, which – by Lemma 2.6 – is equivalent to the consistency of the Runge-Kutta method. □

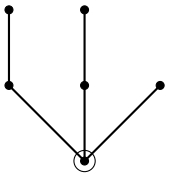
Letting $e = (1, \dots, 1)^{\mathsf{T}} \in \mathbb{R}^s$ denote the vector of all ones, the conditions of Lemma 2.6 and Lemma 2.7 can be compactly written as

$$\mathbf{b}^{\mathsf{T}}\mathbf{e} = 1, \qquad \mathbf{c} = A\mathbf{e}.$$

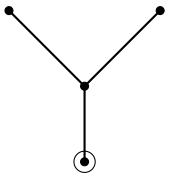
For the rest of this section, we will assume that these conditions are satisfied and only consider autonomous IVPs.

Elementary differentials and rooted trees. The biggest problem when attempting to verify (high) orders of Runge-Kutta methods is to keep track of the terms appearing in the Taylor expansions of the exact and the approximate solutions. As can be seen from (2.7) and (2.8), these terms are elementary differentials of f. Counting the number of the various elementary differentials appearing in the expansion is a combinatorial problem, for which we will use the concept of rooted trees.

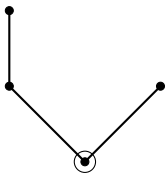
Roughly speaking, an elementary differential is a multilinear form that is fully described by the recursive structure of its arguments, a more precise definition will be given below. To give a specific example, let us consider the differential $f'''(f'f, f'f, f)$. We can deduce the presence of the third derivative from the fact that three arguments need to be provided. To detail the specific structure of the arguments, we can use rooted trees, such as:



for $f'''(f'f, f'f, f)$



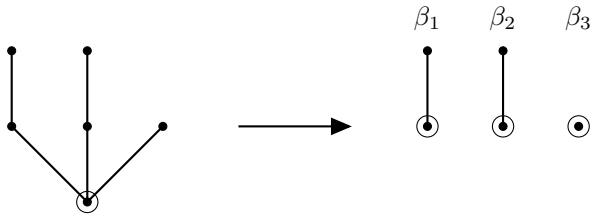
for $f'f''(f, f)$



for $f''(f'f, f)$

Each node of the rooted tree corresponds to a derivative of f at $\mathbf{y}_0$, with the order k of the derivative determined by the number of children. The substitution of the arguments proceeds recursively starting at the root $\odot$. Note that the order of the children is *not* important, due the symmetry of the multilinear form.

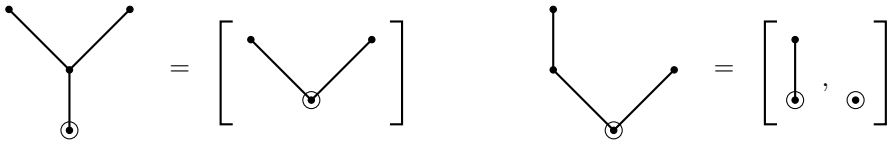
When the root $\odot$ is deleted from a tree β then β decomposes into a forest of k trees $\beta_1, \ldots, \beta_k$, whose roots correspond to the k children of $\odot$. For example:



In the other direction, the tree β can be composed from an unordered tuple of subtrees:

$$\beta = [\beta_1, \ldots, \beta_k], \qquad \#\beta = 1 + \#\beta_1 + \cdots + \#\beta_k,$$

where $\#\beta$ denotes the number of the nodes in a tree β. For creating a tree consisting solely of the root, we use the convention $\odot = []$. Two examples for the composition of trees:



This decomposition can be continued recursively until the subtrees consist of roots only. Two examples:

$$\begin{array}{ll} f'f''(f, f) & \text{corresponds to} \quad [[\odot, \odot]], \\ f''(f'f, f) & \text{corresponds to} \quad [[\odot], \odot]. \end{array}$$

With the notation defined above, we are ready to define elementary differentials.

Definition 2.8 *The* **elementary differential** $f^{(\beta)}(\mathbf{y}_0)$ *corresponding to a rooted tree* $\beta = [\beta_1, \dots, \beta_k]$ *is defined recursively as*

$$f^{(\beta)}(\mathbf{y}_0) = f^{(k)} \cdot \Big(f^{(\beta_1)}(\mathbf{y}_0), \dots, f^{(\beta_k)}(\mathbf{y}_0) \Big).$$

In the definition above, the argument $\mathbf{y}_0$ is given for clarity; we will (again) often omit it in our subsequent considerations.

Remark 2.9 The Hopf algebra of rooted trees does not only play a role in the theory of Runge–Kutta methods, but also in noncommutative geometry and renormalization methods in quantum field theory. Quite remarkably, it was developed first in the context of numerical analysis, by John Butcher. We refer to [Brouder, Ch. Trees, renormalization and differential equations. BIT 44 (2004), no. 3, 425–438] for a very accessible exposition of these connections. ◇

Taylor expansion of the exact solution. Let us recall the Taylor expansion (2.7):

$$\mathbf{y}(t_0 + h) = \mathbf{y}_0 + hf + \frac{h^2}{2} f'f + \frac{h^3}{6} \big(f'f'f + f''(f,f) \big) + O(h^4).$$

A closer inspection suggests that the terms for h^k involve all elementary differentials corresponding to rooted trees with k nodes. Lemma 2.10 below confirms this impression in the general case. For this purpose, we need to introduce two combinatorial quantities. The **factorial of a tree** $\beta = [\beta_1, \dots, \beta_k]$ is recursively defined by

$$\beta! = (\#\beta)\beta_1!\beta_2! \cdots \beta_k!.$$

Moreover, we define recursively

$$\alpha_\beta = \frac{\delta_\beta}{k!} \alpha_{\beta_1} \alpha_{\beta_2} \cdots \alpha_{\beta_k},$$

where δ_β denotes the number of different possibilities for associating an an *ordered* k-tuple with the unordered k-tuple $\beta = [\beta_1, \dots, \beta_k]$.

Lemma 2.10 *Let* $f \in C^p(\Omega, \mathbb{R}^d)$ *with an open set* Ω *and* $\mathbf{y}_0 \in \Omega$. *Then*

$$\mathbf{y}(t_0 + h) = \mathbf{y}_0 + \sum_{\#\beta \le p} \frac{h^{\#\beta}}{\beta!} \alpha_\beta f^{(\beta)}(\mathbf{y}_0) + O(h^{p+1}).$$

Proof. By induction over p. For $p = 0$, the statement trivially holds. We now show the expansion for $p + 1$, assuming that it holds for p. The proof proceeds, as in the analysis of Section 2.1, by inserting the pth order expansion into the

differential equation and using the multivariate Taylor expansion. Setting $\mathbf{h} = \sum_{\#\beta \le p} \frac{h^{\#\beta}}{\beta!} \alpha_\beta f^{(\beta)}$, we obtain

$$\frac{\partial}{\partial h}\mathbf{y}(t_0+h) = f(\mathbf{y}_0+th) = f\left(\mathbf{y}_0 + \mathbf{h} + O(h^{p+1})\right) = \sum_{k=0}^{p} \frac{1}{k!} f^{(k)}(\mathbf{h},\dots,\mathbf{h}) + O(h^{p+1}).$$

Using the multilinearity of $f^{(k)}$ and omitting terms smaller than h^p, we obtain

$$\begin{aligned}
&\frac{1}{k!} f^{(k)}(\mathbf{h},\dots,\mathbf{h}) \\
&= \frac{1}{k!} \sum_{\#\beta_1 \le p} \cdots \sum_{\#\beta_k \le p} \frac{h^{\#\beta_1+\cdots+\#\beta_k}}{\beta_1!\cdots\beta_k!} \alpha_{\beta_1}\cdots\alpha_{\beta_k} f^{(k)}\big(f^{(\beta_1)},\dots,f^{(\beta_k)}\big) \\
&= \frac{1}{k!} \sum_{\#\beta_1+\cdots+\#\beta_k \le p} \frac{h^{\#\beta_1+\cdots+\#\beta_k}}{\beta_1!\cdots\beta_k!} \alpha_{\beta_1}\cdots\alpha_{\beta_k} f^{(k)}\big(f^{(\beta_1)},\dots,f^{(\beta_k)}\big) + O(h^{p+1}) \\
&= \sum_{\substack{\#\beta \le p+1 \\ \beta=[\beta_1,\dots,\beta_k]}} \frac{\#\beta \cdot h^{\#\beta-1}}{\beta!} \underbrace{\frac{\delta_\beta}{k!}\alpha_{\beta_1}\cdots\alpha_{\beta_k}}_{=\alpha_\beta} f^{(\beta)} + O(h^{p+1}),
\end{aligned}$$

where we have used in the last step that the symmetry of $f^{(k)}$ allows to use unordered tuples $[\beta_1,\dots,\beta_k]$ instead of ordered tuples. This introduces the extra factor δ_β, which counts the number of ordered tuples associated with $[\beta_1,\dots,\beta_k]$. To sum up, we obtain

$$\frac{\partial}{\partial h}\mathbf{y}(t_0+h) = \sum_{\#\beta \le p+1} \frac{\#\beta \cdot h^{\#\beta-1}}{\beta!} \alpha_\beta f^{(\beta)} + O(h^{p+1}).$$

Integrating this relation gives

$$\mathbf{y}(t_0+h) = \mathbf{y}_0 + \sum_{\#\beta \le p+1} \frac{h^{\#\beta}}{\beta!} \alpha_\beta f^{(\beta)} + O(h^{p+2}).$$

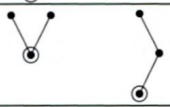
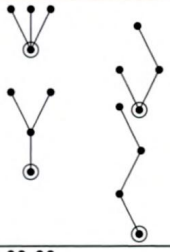
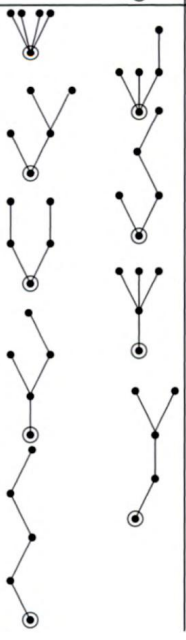
This corresponds exactly to the claimed expansion for $p+1$. □

Examples for the coefficients $\beta!$ and α_β can be found in Table 2.1.

Taylor expansion of the numerical solution. The expansion of the numerical solution $\mathbf{y}_1$ produced by the Runge-Kutta method can also be represented with the help of rooted trees. For a rooted tree $\beta = [\beta_1,\dots,\beta_k]$, we recursively define the vector $A^{(\beta)} \in \mathbb{R}^s$ via its components:

$$A_i^{(\beta)} = \left(A \cdot A^{(\beta_1)}\right)_i \cdots \left(A \cdot A^{(\beta_k)}\right)_i, \qquad i = 1,\dots,s.$$

Table 2.1. *Rooted trees up to order* 5. *Table taken from Page 150 of [DB].*

$\#\beta$	β	rooted tree	$\beta!$	α_β	$f^{(\beta)}$	$\mathcal{A}_i^{(\beta)}$
1	β_{11}		1	1	f	1
2	β_{21}		2	1	$f'f$	c_i
3	β_{31}		3	$\frac{1}{2}$	$f''(f,f)$	c_i^2
	β_{32}		6	1	$f'f'f$	$\sum_j a_{ij}c_j$
4	β_{41}		4	$\frac{1}{6}$	$f'''(f,f,f)$	c_i^3
	β_{42}		8	1	$f''(f'f,f)$	$\sum_j c_i a_{ij}c_j$
	β_{43}		12	$\frac{1}{2}$	$f'f''(f,f)$	$\sum_j a_{ij}c_j^2$
	β_{44}		24	1	$f'f'f'f$	$\sum_{jk} a_{ij}a_{jk}c_k$
5	β_{51}		5	$\frac{1}{24}$	$f^{IV}(f,f,f,f)$	c_i^4
	β_{52}		10	$\frac{1}{2}$	$f'''(f'f,f,f)$	$\sum_j c_i^2 a_{ij}c_j$
	β_{53}		15	$\frac{1}{2}$	$f''(f''(f,f),f)$	$\sum_j c_i a_{ij}c_j^2$
	β_{54}		30	1	$f''(f'f'f,f)$	$\sum_{jk} c_i a_{ij}a_{jk}c_k$
	β_{55}		20	$\frac{1}{2}$	$f''(f'f,f'f)$	$\left(\sum_j a_{ij}c_j\right)^2$
	β_{56}		20	$\frac{1}{6}$	$f'f'''(f,f,f)$	$\sum_j a_{ij}c_j^3$
	β_{57}		40	1	$f'f''(f'f,f)$	$\sum_{jk} a_{ij}c_j a_{jk}c_k$
	β_{58}		60	$\frac{1}{2}$	$f'f'f''(f,f)$	$\sum_{jk} a_{ij}a_{jk}c_k^2$
	β_{59}		120	1	$f'f'f'f'f$	$\sum_{jkl} a_{ij}a_{jk}a_{kl}c_l$

Again, this definition is independent of the order of the children $\beta_1, \ldots, \beta_k$. Note that

$$A^{(\odot)} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \in \mathbb{R}^s.$$

Further examples for $A^{(\beta)}$ can be found in Table 2.1, where the assumed relation $\mathbf{c} = A\mathbf{e}$ has been used extensively to simplify the formulas.

Lemma 2.11 *Let $f \in C^p(\Omega, \mathbb{R}^d)$ with an open set Ω and $\mathbf{y}_0 \in \Omega$. Then the approximation $\mathbf{y}_1$ obtained by applying the Runge-Kutta method (Definition 2.4) to the autononomous IVP (2.6) satisfies*

$$\mathbf{y}_1 = \mathbf{y}_0 + \sum_{\#\beta \le p} h^{\#\beta} \alpha_\beta \cdot b^{\mathsf{T}} A^{(\beta)} f^{(\beta)}(\mathbf{y}_0) + O(h^{p+1}).$$

Proof. Since

$$\mathbf{y}_1 = \mathbf{y}_0 + h \sum_{i=1}^{s} b_i \mathbf{k}_i,$$

it suffices to show the Taylor expansions

$$\mathbf{k}_i = \sum_{\#\beta \le p} h^{\#\beta - 1} \alpha_\beta A_i^{(\beta)} f^{(\beta)} + O(h^p), \qquad i = 1, \dots, s \tag{2.11}$$

for the stages $\mathbf{k}_i$. The proof of (2.11) proceeds by induction over p. It clearly holds for $p = 0$ and we now aim to derive (2.11) for $p+1$, assuming that it holds for p. We then have

$$\begin{aligned}
\mathbf{k}_i &= f\left(\mathbf{y}_0 + h \sum_{j=1}^{i-1} a_{ij} \mathbf{k}_j\right) \\
&= f\left(\mathbf{y}_0 + \sum_{j=1}^{i-1} a_{ij} \sum_{\#\beta \le p} h^{\#\beta} \alpha_\beta A_i^{(\beta)} f^{(\beta)} + O(h^{p+1})\right) \\
&= f\left(\mathbf{y}_0 + \sum_{\#\beta \le p} h^{\#\beta} \alpha_\beta \Big(A \cdot A^{(\beta)}\Big)_i f^{(\beta)} + O(h^{p+1})\right).
\end{aligned}$$

Setting $\mathbf{h} = \displaystyle\sum_{\#\beta \le p} h^{\#\beta} \alpha_\beta \Big(A \cdot A^{(\beta)}\Big)_i f^{(\beta)}$, the multivariate Taylor expansion combined with the multilinearity of $f^{(k)}$ yield

$$\begin{aligned}
\mathbf{k}_i &= \sum_{k=0}^{p} \frac{1}{k!} f^{(k)} \cdot (\mathbf{h}, \dots, \mathbf{h}) + O(h^{p+1}) \\
&= \sum_{k=0}^{p} \frac{1}{k!} \sum_{\#\beta_1 + \dots + \#\beta_k \le p} h^{\#\beta_1 + \dots + \#\beta_k} \cdot \alpha_{\beta_1} \cdots \alpha_{\beta_k} \Big(A \cdot A^{(\beta_1)}\Big)_i \cdots \Big(A \cdot A^{(\beta_k)}\Big)_i \\
&\qquad\qquad \cdots f^{(k)} \cdot \Big(f^{(\beta_1)}, \dots f^{(\beta_k)}\Big) + O(h^{p+1}) \\
&= \sum_{k=0}^{p} \sum_{\substack{\#\beta \le p+1 \\ \beta = [\beta_1, \dots, \beta_k]}} h^{\#\beta - 1} \cdot \frac{\delta_\beta}{k!} \alpha_{\beta_1} \cdots \alpha_{\beta_k} \cdot A_i^{(\beta)} f^{(\beta)} + O(h^{p+1}) \\
&= \sum_{\#\beta \le p+1} h^{\#\beta - 1} \alpha_\beta \cdot A_i^{(\beta)} f^{(\beta)} + O(h^{p+1}).
\end{aligned}$$

This completes the proof, as the last line corresponds exactly to (2.11) for $p+1$. □

Order conditions. After all these preparations, we can now combine Lemma 2.10 and Lemma 2.11 to obtain a pleasingly elegant necessary and sufficient condition for a Runge-Kutta method having order p.

Theorem 2.12 *A Runge-Kutta method is of order p if and only if*

$$\mathbf{b}^{\mathsf{T}} A^{(\beta)} = \frac{1}{\beta!} \tag{2.12}$$

holds for all rooted trees β of order $\#\beta \le p$.

Proof. The sufficiency of condition (2.12) follows immediately from comparing the expansions in Lemma 2.10 and Lemma 2.11. For showing that (2.12) is also necessary, one needs to show that the elementary differentials for different trees are linearly independent. This is proven, e.g., in Lemma 4.25 of [DB]. □

Example 2.13 For $p = 4$, Theorem 2.12 leads to the following conditions:

$$\begin{array}{rcl} 1 &=& \sum_{i=1}^{s} b_i \\ \hline \frac{1}{2} &=& \sum_i b_i c_i \\ \hline \frac{1}{3} &=& \sum_i b_i c_i^2 \\ \frac{1}{6} &=& \sum_{i,j} b_i a_{ij} c_j \end{array}, \qquad \begin{array}{rcl} \frac{1}{4} &=& \sum_i b_i c_i^3 \\ \hline \frac{1}{8} &=& \sum_{i,j} b_i c_i a_{ij} c_j \\ \frac{1}{12} &=& \sum_{i,j} b_i a_{ij} c_j^2 \\ \frac{1}{24} &=& \sum_{i,j,k} b_i a_{ij} a_{jk} c_k \end{array}$$

It is a simple exercise to verify that RK4 satisfies this conditions and therefore has order 4. ◇

From Example 2.13 one may obtain the misleading impression that the number of conditions grows only slowly with p. In fact, for $p = 10$ one has 1205 conditions and for $p = 20$ one has 20247374 conditions!

2.2.2 Convergence results

By definition, the local error $\mathbf{e}(h) = \mathbf{y}(t_0 + h) - \mathbf{y}_1$ of a Runge-Kutta method of order p satisfies $\|\mathbf{e}(h\| \le Ch^{p+1}$ for some constant C.

We are now concerned with analyzing the **global error**, which is the error of the computed solution after *several* steps: $\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \ldots$. For this purpose, we will write

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h_i \Phi(t_i, \mathbf{y}_i, h_i), \qquad h_i = t_{i+1} - t_i,$$

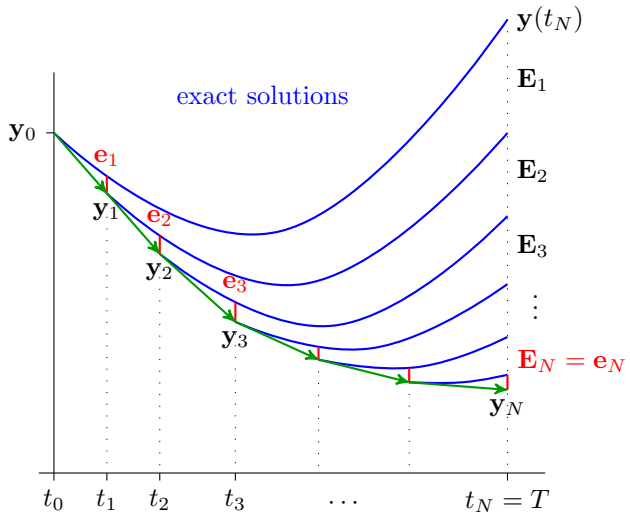


Figure 2.1. *Lady Windemere's fan [l'éventail de Lady Windermere].*

where Φ is the **increment function** of the method. Every one-step method can be written in this form. Our task is to estimate the **global error**

$$\mathbf{E} = \mathbf{y}(T) - \mathbf{y}_N, \quad \text{with} \quad T = t_N. \tag{2.13}$$

This estimate is found by transporting all local errors $\mathbf{e}_i = \mathbf{y}(t_j) - \mathbf{y}_i$, which satisfy

$$\|\mathbf{e}_i\| \le Ch_{i-1}^{p+1}, \tag{2.14}$$

to the end point t_N and adding them up. The principle is illustrated in Figure 2.1.

The following lemma can be used to estimate the magnification of the error during the transport.

Lemma 2.14 *Suppose that* $\mathbf{y}(t)$ *and* $\mathbf{v}(t)$ *are solutions to* $\mathbf{y}' = f(t, \mathbf{y})$ *with initial values* $\mathbf{y}(t_0) = \mathbf{y}_0$ *and* $\mathbf{v}(t_0) = \mathbf{v}_0$*, respectively. If*

$$\|f(t, \mathbf{v}) - f(t, \mathbf{y})\| \le L\|\mathbf{v} - \mathbf{y}\| \tag{2.15}$$

then we have the error estimate

$$\|\mathbf{v}(t) - \mathbf{y}(t)\| \le e^{L(t-t_0)} \cdot \|\mathbf{v}_0 - \mathbf{y}_0\|. \tag{2.16}$$

Proof. Integration of $\mathbf{v}' - \mathbf{y}' = f(t, \mathbf{v}) - f(t, \mathbf{y})$ with respect to t gives

$$\mathbf{v} - \mathbf{y} = \mathbf{v}_0 - \mathbf{y}_0 + \int_{t_0}^{t} \big(f(s, \mathbf{v}(s)) - f(s, \mathbf{y}(s))\big)\, \mathrm{d}s.$$

Hence,

$$\|\mathbf{v} - \mathbf{y}\| \le \|\mathbf{v}_0 - \mathbf{y}_0\| + L \int_{t_0}^{t} \|\mathbf{v}(s) - \mathbf{y}(s)\| \, \mathrm{d}s. \tag{2.17}$$

We can apply the (continuous) lemma of Gronwall to this situation (see Lemma 2.15 below), with $u(t) = \|\mathbf{v}(t) - \mathbf{y}(t)\|$, $\rho = \|\mathbf{v}_0 - \mathbf{y}_0\|$, and $w(t) \equiv L$. This immediately gives (2.16). ∎

Lemma 2.15 (Lemma of Gronwall) *Let $u, w \in C([t_0, T])$ be nonnegative functions and $\rho \ge 0$. Then*

$$u(t) \le \rho + \int_{t_0}^{t} u(s)w(s) \, ds \quad \textit{for all } t \in [t_0, T]$$

implies the estimate

$$u(t) \le \rho \exp\left(\int_{t_0}^{t} w(s) \, ds \right) \quad \textit{for all } t \in [t_0, T].$$

Proof. EFY, cf. Ex.1,3(b). ∎

Using Lemma 2.14, it remains to add the transported local errors in order to obtain an estimate for the global error.

Theorem 2.16 *Let U be a neighborhood of $\big\{\big(t, \mathbf{y}(t)\big) : t_0 \le t \le T\big\}$ such that the local error estimate* (2.14) *and the estimate*

$$\left\| \frac{\partial f}{\partial y} \right\| \le L \tag{2.18}$$

hold in U. Then the global error (2.13) *satisfies*

$$\|\mathbf{E}\| \le h^p \frac{C}{L} \left(e^{L(T-t_0)} - 1 \right), \tag{2.19}$$

where $h = \max_i h_i$ is small enough for the numerical solution to remain in U.

Proof. Let $\mathbf{E}_i$ denote the ith local error $\mathbf{e}_i$ transported to the end point, see Figure 2.1. Then Lemma 2.14 applied at time t_i with initial values $\mathbf{y}_i$ vs. $\mathbf{y}(t_i)$ yields

$$\|\mathbf{E}_i\| \le e^{L(T-t_i)} \cdot \|\mathbf{e}_i\| \le C e^{L(T-t_i)} h_{i-1}^{p+1}.$$

Using $h_{i-1}^{p+1} \le h^p h_{i-1}$ yields

$$\|\mathbf{E}\| \le \sum_{i=1}^{N} \|\mathbf{E}_i\| \le C h^p \sum_{i=1}^{N} e^{L(T-t_i)} h_{i-1} \le C h^p \int_{t_0}^{T} e^{L(T-s)} \, \mathrm{d}s.$$

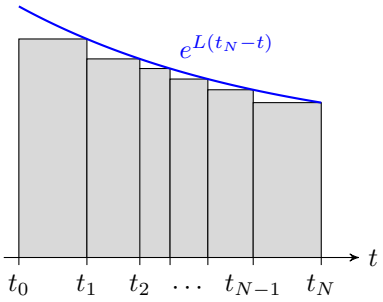


Figure 2.2. *Lower Darboux sum used in the proof of Theorem 2.16.*

In the last inequality, we have used that the sum is actually a lower Darboux sum *[somme de Darboux inférieure]* for the integral on the right-hand side, see Figure 2.2. The elementary fact $\int_{t_0}^{T} e^{L(T-s)}\,\mathrm{d}s = \frac{1}{L}\left(e^{L(T-t_0)}-1\right)$ concludes the proof. □

Clearly, Theorem 2.16 indicates that we may run into severe problems (that is, exponentially growing error) as $T-t_0$ grows. Still, ODEs are routinely integrated over relatively long time, especially in molecular dynamics and astronomy. However, this may require the use of special methods, see [HLW].

Remark 2.17* As the Lipschitz constant L is positive, Theorem 2.16 always implies an exponential growth of the error. This is clearly not very satisfactory for differential equations such as $y' = -y$, where both the solution and the transported error are *damped* as t increases. The guilty party is Lemma 2.14, which is too pessimistic about the effect of transport on the error. We can derive a different estimate as follows. First, note that

$$\begin{aligned}\|\mathbf{v}(t+h)-\mathbf{y}(t+h)\| &= \big\|\mathbf{v}(t)-\mathbf{y}(t)+h\big(f(t,\mathbf{v})-f(t,\mathbf{y})\big)\big\|+O(h^2)\\ &\le \left(\max_{\mathbf{x}\in V}\left\|I+h\frac{\partial f}{\partial \mathbf{y}}(t,\mathbf{x})\right\|\right)\cdot\|\mathbf{v}(t)-\mathbf{y}(t)\|+O(h^2),\end{aligned}$$

where we applied the mean value theorem and V is a connected neighborhood containing both $\mathbf{v}$ and $\mathbf{y}$. Setting $f(t) := \|\mathbf{v}(t)-\mathbf{y}(t)\|$, we therefore obtain

$$\frac{m(t+h)-m(t)}{h} \le \left(\max_{\mathbf{x}\in V}\frac{\left\|I+h\frac{\partial f}{\partial \mathbf{y}}(t,\mathbf{x})\right\|-1}{h}\right) m(t)+O(h). \tag{2.20}$$

As $h\to 0$, the factor on the right-hand side features a quantity known as the logarithmic "norm" of a matrix.

Definition 2.18 *Let $A\in\mathbb{C}^{n\times n}$. Then*

$$\mu(A) := \lim_{\substack{h\to 0\\ h>0}} \frac{\|I+hA\|-1}{h}$$

is called the **logarithmic norm** *[norme logarithmique] of A.*

Of course, Definition 2.18 depends on the choice of matrix norm $\|\cdot\|$. For the matrix 2-norm, it is not hard to show that $\mu(A) = \frac{1}{2}\lambda_{\max}(A + A^*)$.

From (2.20) it now follows that the upper Dini derivative m'_+ satisfies

$$m'_+(t) \leq \max_{\mathbf{x}\in V} \mu\left(\frac{\partial f}{\partial \mathbf{y}}(t, \mathbf{x})\right) m(t).$$

Integrating this inequality gives (2.17) but with $L = \max_{\mathbf{x}\in V} \mu\left(\frac{\partial f}{\partial \mathbf{y}}(t, \mathbf{x})\right)$ replacing the Lipschitz constant. As a consequence, we get an improved variant of Theorem 2.16. The assumption (2.18) is replaced by

$$\mu\left(\frac{\partial f}{\partial \mathbf{y}}\right) \leq L$$

and the global error estimate (2.19) reads as

$$\|\mathbf{E}\| \leq h^p \frac{C'}{L}\left(e^{L(T-t_0)} - 1\right).$$

Here, $C' = C$ for $L > 0$. For $L < 0$, one has to adjust the Darboux sum in the proof of Theorem 2.16 and obtains $C' = Ce^{-Lh}$. ◇