Chapter 3

Implicit Runge-Kutta methods

Although the family of explicit Runga-Kutta methods is quite rich, they may be ineffective for some (particularly hard) problems. Indeed, we will see that *no* explicit method is suitable for so called stiff problems, which frequently arise in practice, in particular from the spatial discretization of time-dependent partial differential equations. It turns out that implicit methods are much more effective for stiff problems. However, we will see that the price one has to pay for going implicit is very high; function evaluations are replaced by the solution of nonlinear systems!

Example 3.1 To solve the IVP

$$\dot{y}(t) = 500\, y^2(1-y), \quad y(0) = 1/100,$$

we make use of the MATLAB function `ode23s`, which is based on Rosenbrock methods – a variation of implicit Runge-Kutta methods discussed in Section 3.5. For this purpose, we need to define the function as well as its derivative (Jacobian) with respect to y (or an approximation of it):

```
fun = @(t,y) 500*y^2*(1-y); funjac = @(t,y) 1000*y*(1-y) - 500*y^2;
```

Additionally to the usual tolerances for the time stepping procedure, the derivative also needs to be declared in the options:

```
opt = odeset( 'reltol', 0.1, 'abstol', 0.001, 'Jacobian', funjac );
```

Finally, the integration is performed on the interval $[0, 1]$:

```
[t,x] = ode23s(fun, [0,1], 0.01, opt);
plot(t,x);
```

From Figure 3.1, it is clear that `ode23s` requires much less time steps and function evaluations compared to the explicit Runge-Kutta method in `ode23` with the same options. Moreover, `ode23s` appears to be significantly more accurate than `ode23`. $\diamond$

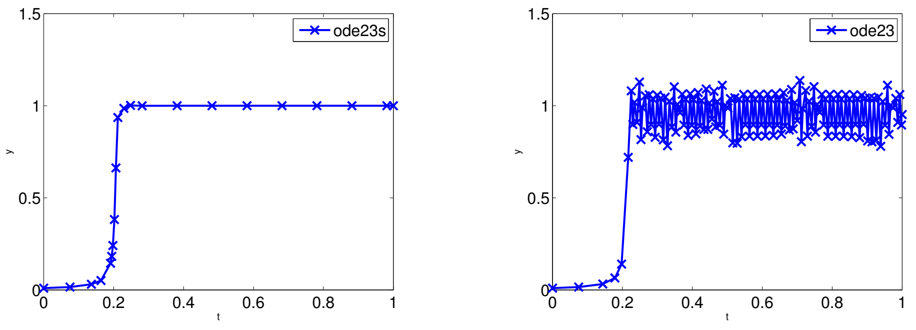

Figure 3.1. Results of `ode23s` and `ode23` applied to Example 3.1.

3.1 Stability concepts

The notion of stability is crucial to understand the limitations of explicit methods for stiff problems. We consider a linear homogeneous IVP

$$\dot{\mathbf{y}}(t) = G\mathbf{y}(t), \qquad \mathbf{y}(t_0) = \mathbf{y}_0, \tag{3.1}$$

for a $d \times d$ matrix G. The IVP (3.1) is called

- **asymptotically stable** if $\|\mathbf{y}(t)\| \to 0$ as $t \to \infty$ for *all* initial values $\mathbf{y}_0 \in \mathbb{R}^d$;
- **stable** if there is a constant C (independent of t and $\mathbf{y}_0$) such that $\|\mathbf{y}(t)\| < C\|\mathbf{y}_0\|$ holds for all $t \geq t_0$ and $\mathbf{y}_0 \in \mathbb{R}^d$;
- **unstable**, otherwise.

Stability has a number of important consequences. For example if we consider an inhomogeneous problem with undergoing a perturbation of the initial value:

$$\begin{aligned} \dot{\mathbf{y}_1}(t) &= C\mathbf{y}_1(t), \qquad & \mathbf{y}_1(t_0) &= \mathbf{y}_0, \\ \dot{\mathbf{y}_2}(t) &= C\mathbf{y}_2(t), \qquad & \mathbf{y}_2(t_0) &= \mathbf{y}_0 + \triangle\mathbf{y}_0. \end{aligned}$$

then stability implies $\|\mathbf{y}_1(t) - \mathbf{y}_2(t)\| \leq C\|\triangle\mathbf{y}_0\|$ and asymptotic stability even implies that $\|\mathbf{y}_1(t) - \mathbf{y}_2(t)\| \to 0$. That is, the impact of a perturbation is bounded or vanishes in the long time, respectively.

Let us define the **matrix exponential**

$$e^B := \sum_{n=0}^{\infty} \frac{1}{n!} B^n, \tag{3.2}$$

which is absolutely convergent for any $B \in \mathbb{C}^{d \times d}$. Then the solution of (3.1) is given by

$$\mathbf{y}(t) = e^{G(t-t_0)}\mathbf{y}_0.$$

Hence, asymptotic stability is equivalent to $\|e^{Gt}\| \to 0$ as $t \to \infty$. Stability is equivalent to $\|e^{Gt}\| \le C$ for all $t > 0$. The following theorem gives a complete characterization of (asymptotic) stability in terms of the eigenvalues of G. Note that an eigenvalue λ of G is called **semi-simple** if its algebraic and geometric multiplicities are equal or, equivalently, if all blocks in the Jordan canonical form of G associated with λ are 1×1.

Theorem 3.2 *The IVP* (3.1) *is asymptotically stable if and only if all eigenvalues λ of G satisfy* $\mathsf{Re}(\lambda) < 0$.

The IVP (3.1) *is stable if and only if all eigenvalues λ of G satisfy* $\mathsf{Re}(\lambda) \le 0$ and *if every eigenvalue λ with* $\mathsf{Re}(\lambda) = 0$ *is semi-simple.*

Proof. For simplicity, we assume that G is diagonalizable. (The general case requires to study the Jordan canonical form of A, which is beyond the scope of this lecture.) Then there is an invertible matrix P such that

$$G = P\Lambda P^{-1} \quad \text{with} \quad \Lambda = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_d \end{pmatrix},$$

where $\lambda_1, \dots, \lambda_d$ are the eigenvalues of G. Then

$$\begin{aligned} e^{Gt} &= \sum_{n=0}^{\infty} \frac{1}{n!} G^n t^n = \sum_{n=0}^{\infty} \frac{1}{n!} (P\Lambda P^{-1})^n t^n = \sum_{n=0}^{\infty} \frac{1}{n!} P\Lambda^n P^{-1} t^n \\ &= P\, e^{\Lambda t} P^{-1} = P \begin{pmatrix} e^{\lambda_1 t} & & \\ & \ddots & \\ & & e^{\lambda_d t} \end{pmatrix} P^{-1}. \end{aligned}$$

Using that $e^{\lambda t} \to 0$ converges to zero if and only if $\mathsf{Re}(\lambda) < 0$, it follows that $\|e^{Gt}\| \to 0$ if and only if $\mathsf{Re}(\lambda) < 0$. This proves the first part.

Moreover, setting $C = \|P^{-1}|_2 \|P\|_2$, it follows that

$$\|e^{Gt}\|_2 \le C \|e^{\Lambda t}\|_2 = C \cdot \max_{\lambda \in \{\lambda_1, \dots, \lambda_n\}} |e^{\lambda t}|,$$

where the second factor remains bounded (by 1) as $t \to \infty$ if and only if $\mathsf{Re}(\lambda) \le 0$. This proves the second part, as the semi-simplicity condition is already contained in the diagonalizability assumption. □

To illustrate that the semi-simplicity assumption for critical eigenvalues with $\mathsf{Re}(\lambda) = 0$ is needed in Theorem 3.2, consider the matrix

$$G = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix},$$

which has an eigenvalue 0 that is *not* semi-simple. Then, by definition (3.2), we have

$$e^{Gt} = \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix},$$

which is clearly not bounded as $t \to \infty$.

We now consider the application of the explicit Euler method with step size h to (3.1):

$$\mathbf{y}_{i+1} = \mathbf{y}_i + hG\mathbf{y}_i = (I + hG)\mathbf{y}_i. \tag{3.3}$$

More generally, similar to the proof of Lemma 2.5 it can be shown that the application of an s-stage explicit Runga-Kutta method applied to (3.1) yields

$$\mathbf{y}_{i+1} = S(hG)\mathbf{y}_i \tag{3.4}$$

for a polynomial S of degree at most s.

Both, (3.3) and (3.4) are examples of a **linear discrete-time system** of the form

$$\mathbf{y}_{i+1} = D\mathbf{y}_i \tag{3.5}$$

for some matrix $D \in \mathbb{R}^{d\times d}$ and initial values $\mathbf{y}_0 \in \mathbb{R}^d$. Similar to IVPs, we can define a notion for the discrete case; (3.5) is called

- **asymptotically stable** if $\|\mathbf{y}_k\| \to 0$ as $k \to \infty$ for *all* initial values $\mathbf{y}_0 \in \mathbb{R}^d$;
- **stable** if there is a constant C (independent of t and $\mathbf{y}_0$) such that $\|\mathbf{y}_k\| < C\|\mathbf{y}_0\|$ holds for all $k \geq 0$ and $\mathbf{y}_0 \in \mathbb{R}^d$;
- **unstable**, otherwise.

Since the solution of (3.5) is clearly $\mathbf{y}_i = D^i\mathbf{y}_0$, the stability (3.5) is equivalently characterized by the growth of D^i. Analogous to Theorem 3.2, we have the following eigenvalue characterization.

Theorem 3.3 *The linear discrete-time system* (3.5) *is asymptotically stable if and only if all eigenvalues λ of D satisfy $|\lambda| < 1$.*

The linear discrete-time system (3.5) *is stable if and only if all eigenvalues λ of D satisfy* $|\lambda| \leq 1$ and *if every eigenvalue λ with $|\lambda| = 1$ is semi-simple.*

3.1.1 Absolute stability

Any one-step method for approximating the solution of an IVP can be seen as a discrete-time system. The stability *of the method* is concerned with the following question:

Does the discrete-time system inherit the stability of the IVP?

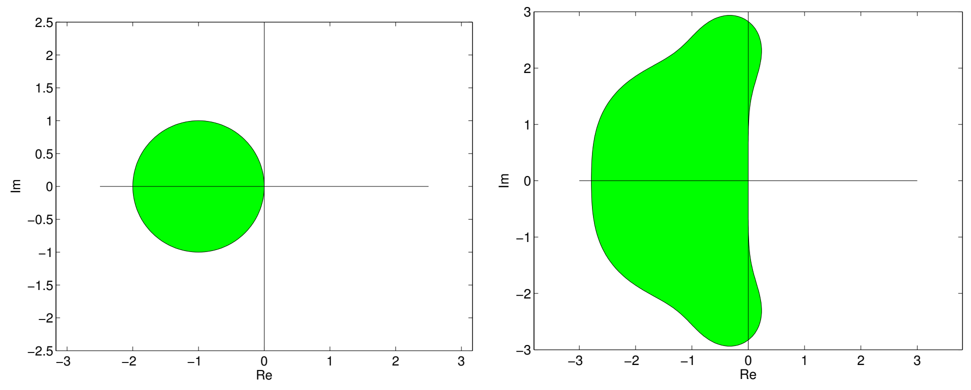

Figure 3.2. The green areas denote the stability regions for the forward Euler method (left plot) and the classical Runga-Kutta method (right plot).

We will study this question for the linear IVP (3.1). In this case, we have already seen that Runge-Kutta methods (and this holds for any linear one-step method) can be written as

$$\mathbf{y}_{i+1} = S(hG)\mathbf{y}_i.$$

for some function S, which is typically a polynomial (in the case of explicit Runge-Kutta methods) or a rational function (in the case of implicit Runge-Kutta methods defined below). The question above amounts to investigating whether the eigenvalues of $S(hG)$ have absolute magnitude less than 1. Note that the eigenvalues of $S(hG)$ are given by $S(h\lambda)$ for every eigenvalue λ of G. Let us therefore define the **stability domain** as

$$\mathcal{S} := \{z \in \mathbb{C} : |S(z)| \leq 1\}. \tag{3.6}$$

Assuming that the IVP is stabble, the method is also stable if

$$h\lambda \in \mathcal{S}$$

for every eigenvalue λ of G.[1]

Examples for stability regions are given in Figure 3.2. Note that we have

$$S(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4$$

for the classical Runge-Kutta method (RK4). The best what can happen to is $h\lambda \in \mathcal{S}$ no matter how small (or large) h is chosen. Since we are only interested in $\mathsf{Re}(\lambda) \leq 0$, this ideal case is guaranteed to happen when $\mathbb{C}^- \subset \mathcal{S}$.

[1]Since S has to be analytic in a neighborhood of $\mathcal{S}$, an eigenvalue $S(h\lambda)$ can have modulus one only if $h\lambda \in \partial\mathcal{S}$. The fact that such eigenvalues are semi-simple follows from the fact that a matrix function does not change the block sizes of the Jordan normal form.

Definition 3.4 *A method is called A-stable if its stability region $\mathcal{S}$ satisfies $\mathbb{C}^- \subset \mathcal{S}$, where $\mathbb{C}^-$ denotes the left-half complex plane.*

Figure 3.2 clearly shows that neither the explicit Euler nor the classical Runge-Kutta methods are A-stable. More generally, we have the following negative result.

Lemma 3.5 *The stability domain $\mathcal{S}$ of any explicit Runga-Kutta method is compact.*

Proof. For an explicit Runga-Kutta method, the function S defining $\mathcal{S}$ in (3.6) is a polynomial different of degree at least 1. Since every such polynomial satisfies $|S(z)| \to \infty$ for $|z| \to \infty$, the stability domain must be bounded. □

A compact stability domain necessarily imposes a restriction on the step size. The step size for which the ray $h\lambda$ first leaves $\mathcal{S}$ is called **critical step size**. Under a mild additional assumption[2], the critical step size is given by

$$h_c = \inf\{h > 0 : \, h\lambda \notin \mathcal{S}\}. \tag{3.7}$$

Example 3.6 For the forward Euler method the stability region is a ball of radius 1 centered at -1. If G has only real negative eigenvalues, this implies that the critical step size is given by

$$h_c = \frac{2}{\max\{|\lambda| : \, \lambda \text{ is an eigenvalue of } G\}} \tag{3.8}$$

According to this analysis, the forward Euler method applied to Example 3.1 yields the desired asymptotic behavior if $|1 - 500\,h| \lesssim 1$, that is, $h \lesssim 1/250$. The obtained approximation for $h = 1/100$ is displayed in Figure 3.3; it "explodes" after a short time, demonstrating very clearly the risk of choosing h too large. For $h = 1/200$, this explosion is avoided, but the correct asymptotic behavior is only reflected for $h = 1/250$ or smaller. ◇

The compactness of the stability region imposes severe restrictions on the step size for discretizations of parabolic PDEs. As an illustration, consider the ordinary differential equation

$$\cdot\mathbf{y}(t) = G\mathbf{y}(t), \quad \text{with} \quad G = \frac{1}{h_x^2}\begin{pmatrix} -2 & 1 & & \\ 1 & -2 & \ddots & \\ & \ddots & \ddots & 1 \\ & & -1 & 2 \end{pmatrix},$$

[2]There is the possibility that the ray touches the boundary of the stability domain before, at some $h_- < h_c$. In that case $S(h_-)$ has at least one eigenvalue of magnitude 1. If this eigenvalue is not semi-simple then the critical step size is h_-. For simplicity, we do not consider this pathological case.

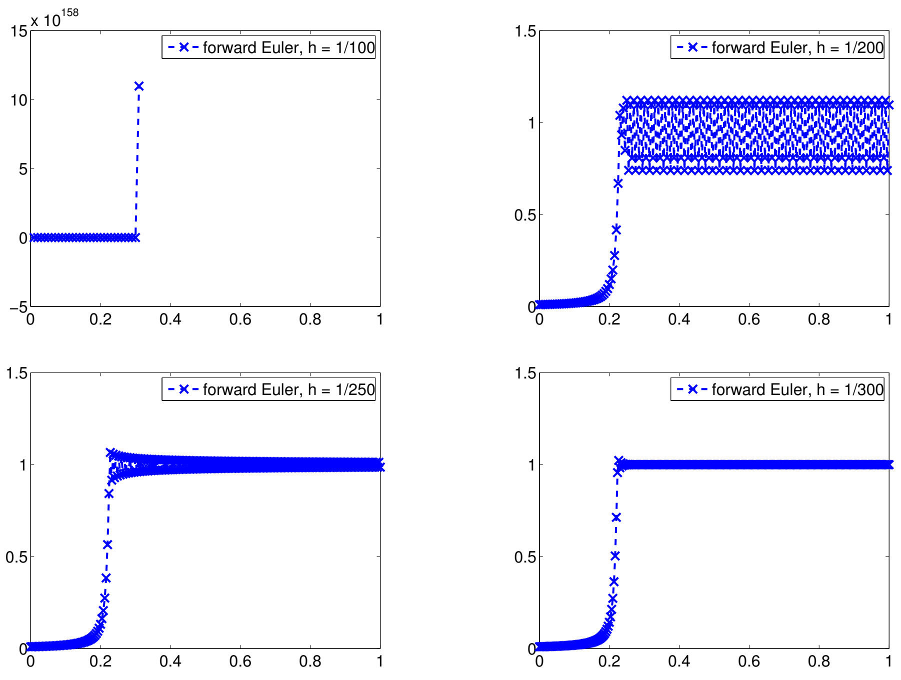

Figure 3.3. Result of forward Euler applied to Example 3.1 with different step sizes h.

which arises from the central finite-difference discretization with mesh width h_x of the one-dimensional parabolic PDE

$$\begin{aligned}\frac{\partial}{\partial t}u(t,x) &= \frac{\partial^2}{\partial x^2}u(t,x), \qquad x\in]0,1[\\ u(t,0) &= u(t,1)=0.\end{aligned}$$

There are explicit formulas for the eigenvalues of G, which imply that all eigenvalues are real and negative. Moreover, the smallest eigenvalue satisfies $|\lambda|\sim h_x^{-2}$ Hence, according to (3.8), the critical step size of the forward Euler method applied to this IVP satisfies $h_c\sim h_x^2$. This (highly undesirable) condition is typical when applying explicit methods to discretizations of parabolic PDEs.

3.1.2 The implicit Euler method

Since explicit methods always yield polynomial stability function, there is no hope to obtain an explicit A-stable method. In this section, we will discuss the most basic implicit method, the **implicit Euler method**:

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

In contrast to explicit methods, we need to solve a system of nonlinear equations to determine $\mathbf{y}_1$! This should be done by the Newton method (or some simplified variant), see Section 3.2.1.

The stability function of the implicit Euler method is given by

$$S(z) = \frac{1}{1-z},$$

and hence the stability region is the complement of a disc in the complex plane, see Figure 3.4.

3.1.3 Beyond linear ODEs*

Let a general *autonomous* IVP

$$\dot{\mathbf{y}}(t) = f\big(\mathbf{y}(t)\big), \qquad \mathbf{y}(t_0) = \mathbf{y}_0, \tag{3.9}$$

have a stationary point $\mathbf{y}^\star$ and assume that $\mathbf{y}(t) \to \mathbf{y}^\star$ as $t \to \infty$ for every $\mathbf{y}_0$ sufficiently close to $\mathbf{y}^\star$. Then the linearization around this stationary point takes the form

$$\dot{\mathbf{y}}(t) = f'(\mathbf{u}^\star)\cdot \mathbf{y}(t), \qquad \mathbf{y}(t_0) = \mathbf{y}_0,$$

where all eigenvalues of the derivative $f'(\mathbf{u}^\star) \in \mathbb{R}^{d\times d}$ have negative real part. A Runge-Kutta method with step size h applied to (3.9) converges to $\mathbf{y}^\star$ for sufficiently close initial values $\mathbf{y}_0$ if λh is in the stability region for every eigenvalue λ of $f'(\mathbf{u}^\star)$.

A number of crimes are committed when considering the linearization only. The concept of B-stability is better suited for nonlinear ODEs; see Chapter IV.12 in [HW].

3.2 General form of implicit Runge-Kutta methods

The implicit Euler method discussed above belongs to a whole class of implicit Runge-Kutta methods. These are obtained as a generalization of the explicit Runge-Kutta methods, by giving up the requirement that we can compute the stages by means of forward substitution.

Definition 3.7 *Suppose that* $\mathbf{k}_1,\ldots,\mathbf{k}_s \in \mathbb{R}^d$ *satisfy the following nonlinear equations:*

$$\begin{aligned}
\mathbf{k}_1 &= f(t_0 + c_1 h, \mathbf{y}_0 + h\sum_{\ell=1}^{s} a_{1\ell}\mathbf{k}_\ell) \\
&\vdots \\
\mathbf{k}_s &= f\Big(t_0 + c_s h, \mathbf{y}_0 + h\sum_{\ell=1}^{s} a_{s\ell}\mathbf{k}_\ell\Big),
\end{aligned}$$

for given coefficients $a_{i\ell}, c_i \in \mathbb{R}$. *Then*

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

is one step of the s**-stage implicit Runge-Kutta method**.

Again, the coefficients defining an implicit Runge-Kutta method can be organized compactly in a Butcher tableau:

$$\begin{array}{c|c} \mathbf{c} & A \\ \hline & \mathbf{b}^T \end{array} := \begin{array}{c|cccc} c_1 & a_{11} & \cdots & \cdots & a_{1,s} \\ c_2 & a_{21} & \cdots & \cdots & a_{2,s} \\ \vdots & \vdots & & & \vdots \\ c_s & a_{s1} & \cdots & \cdots & a_{ss} \\ \hline & b_1 & \cdots & \cdots & b_s \end{array}$$

Note that in contrast to explicit methods, the matrix A is now a general matrix and not required to be strictly lower triangular. The three simplest examples of implicit Runge-Kutta methods:

implicit Euler method	implicit midpoint rule	implicit trapezoidal rule
$\begin{array}{c\|c} 1 & 1 \\ \hline & 1 \end{array}$	$\begin{array}{c\|c} 1/2 & 1/2 \\ \hline & 1 \end{array}$	$\begin{array}{c\|cc} 0 & 0 & 0 \\ 1 & 1/2 & 1/2 \\ \hline 0 & 1/2 & 1/2 \end{array}$

The implicit trapezoidal rule has a particularly nice stability region, see Figure 3.4.

For Definition 3.7 to make sense, we need to study the solvability of the system of nonlinear equations defining the stages. Moreover, some uniqueness property of the solutions should be imposed to yield a sensible step. For this purpose, we will reformulate the system as a system of fixed point equations by introducing the quantities

$$\mathbf{g}_i = \mathbf{y}_0 + h \sum_{\ell=1}^{s} a_{i\ell} \mathbf{k}_\ell, \quad i = 1, \ldots, s.$$

Then the solutions $\mathbf{g}_1, \ldots, \mathbf{g}_s$ of the system of fixed point equations

$$\mathbf{g}_i = \mathbf{y}_0 + h \sum_{\ell=1}^{s} a_{i\ell} f(t_0 + c_\ell h, \mathbf{g}_\ell), \quad i = 1, \ldots, s, \tag{3.10}$$

define the next step as

$$\mathbf{y}_1 = \mathbf{y}_0 + h \sum_{i=1}^{s} b_i f(t_0 + c_i h, \mathbf{g}_i), \tag{3.11}$$

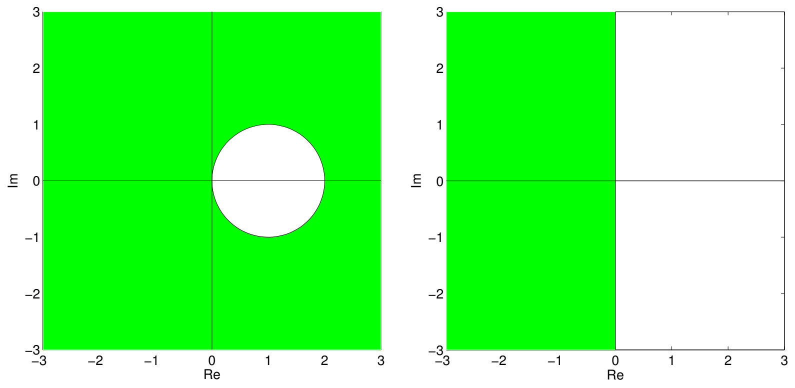

Figure 3.4. Stability regions for the implicit Euler method (left plot) and the implicit trapezoidal rule (right plot).

This is clearly equivalent to Definition 3.7, as can be seen from the relation

$$\mathbf{k}_i = f(t_0 + c_i h, \mathbf{g}_i), \quad i = 1, \dots, s.$$

Theorem 3.8 *Let $f \in C(\Omega, \mathbb{R}^d)$ be Lipschitz continuous with respect to the state on the augmented phase space $\Omega \subset \mathbb{R} \times \mathbb{R}^d$. Then there exists $h_* > 0$ and unique functions $\mathbf{g}_i \in C(]-h_*, h_*[, \mathbb{R}^d)$, such that*

1. $\mathbf{g}_i(0) = \mathbf{y}_0$ *for* $i = 1, \dots, s$;
2. *for all* $0 \leq h < h_*$, *the vectors* $\mathbf{g}_i(h)$ *satisfy the equations* (3.10).

Proof. We only give a sketch of the proof and refer to Theorem 6.28 in [DB] for details. Let us rewrite the system (3.10) as a single equation

$$\mathbf{g} = F(\mathbf{g}), \text{ with } \mathbf{g} = \begin{pmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_s \end{pmatrix}, \quad F(\mathbf{g}) = \begin{pmatrix} \mathbf{y}_0 + h \sum_{\ell=1}^{s} a_{1,\ell} f(t_0 + c_\ell h, \mathbf{g}_\ell) \\ \vdots \\ \mathbf{y}_0 + h \sum_{\ell=1}^{s} a_{s\ell} f(t_0 + c_\ell h, \mathbf{g}_\ell) \end{pmatrix}. \tag{3.12}$$

Then

$$\begin{aligned} \|F(\widehat{\mathbf{g}}) - F(\mathbf{g})\|_\infty &\leq h\|A\|_\infty \max_{1 \leq \ell \leq s} \|f(t_0 + c_\ell h, \widehat{\mathbf{g}}_\ell) - f(t_0 + c_\ell h, \mathbf{g}_\ell)\|_\infty \\ &\leq h\|A\|_\infty L \|\widehat{\mathbf{g}} - \mathbf{g}\|_\infty, \end{aligned}$$

where we used the Lipschitz continuity of f (with Lipschitz constant L). Hence, F

is a contraction provided that

$$h < h_* := \frac{1}{\|A\|_\infty L}. \tag{3.13}$$

By the Banach fixed point theorem, this shows the solvability (3.10). To show the continuous dependence of the solution $\mathbf{g}$ on h (and its uniqueness) requires the application of a parameter-dependent fixed point theorem, which can be found, e.g., in [Dieudonné, J. Foundations of Modern Analysis. 1960]. $\square$

It is interesting to discuss the condition (3.13) on the step size h for a linear ODE $\dot{\mathbf{y}} = G\mathbf{y}$. Then a suitable Lipschitz constant is given by $L = \|G\|_\infty$. Hence, (3.13) becomes $h < \frac{1}{\|A\|_\infty \|G\|_\infty}$, which appears to be a restriction on the step size not better than the restrictions for explicit methods to be stable! However, this only shows that *fixed point iterations are unsuitable* for solving the nonlinear system defining the stages. For solving the nonlinear system, other methods like the Newton method should be used, see also Section 3.2.1.

If we additionally require $f \in C^p(\Omega, \mathbb{R}^d)$ for some $p \geq 1$ in Theorem 3.8 then it can be shown that the vectors $\mathbf{g}_i$ (and therefore also the step $\mathbf{y}_1$) are p times continuously differentiable functions in h. This allows to carry over the discussion of Section 2.2.1 on order conditions to implicit Runge-Kutta methods in a nearly verbatim manner. In particular, an implicit Runge-Kutta method is consistent for all $f \in C(\Omega, \mathbb{R}^d)$ if and only if

$$\mathbf{b}^\mathsf{T}\mathbf{e} = 1.$$

It is invariant under under autonomization if and only if it is consistent and satisfies

$$\mathbf{c} = A\mathbf{e}.$$

The order conditions of Theorem 2.12 also apply, only that $A^{(\beta)}$ is now defined for a general matrix A. Table 2.1 is still valid. The biggest difference is that we now have more coefficients available to design the method and potentially achieve higher order. In fact, the order can be larger than s (but not larger than $2s$). For example, the implicit midpoint rule has order 2.

Finally, we give a compact formula for the stability function of an implicit Runge-Kutta method.

Lemma 3.9 *The stability function of an s-stage implicit Runge-Kutta method is given by*

$$S(z) = 1 + zb^\mathsf{T}(I - zA)^{-1}\mathbf{e},$$

which is a rational function.

Proof. When applying the implicit Runge-Kutta method to the linear IVP $\dot{y}(t) = \lambda y(t)$ we obtain from (3.12) the linear system

$$\mathbf{g} = y_0\mathbf{e} + h\lambda A\mathbf{g}$$

and hence $\mathbf{g} = y_0(I - h\lambda A)^{-1}\mathbf{e}$. According to (3.11) the next step is given by

$$y_1 = y_0 + y_0 h\lambda b^\mathsf{T}(I - h\lambda A)^{-1}\mathbf{e} = \underbrace{(1 + h\lambda b^\mathsf{T}(I - h\lambda A)^{-1}\mathbf{e})}_{=S(h\lambda)} y_0.$$

This shows the first statement. The second statement follows from the fact that the entries of $(I - zA)^{-1}$ are rational functions in z, which is a consequence of $(I - zA)^{-1} = \operatorname{adj}(I - zA)/\det(I - zA)$. □

3.2.1 Solution of the nonlinear system defining the stages

The implementation of an implicit Runge-Kutta method requires the solution of the nonlinear equations (3.10). Since $\mathbf{g}_i - \mathbf{y}_0 = O(h)$ there is the risk of numerical cancellation for small step sizes h. It is therefore preferable to work with the smaller quantities

$$\mathbf{z}_i := \mathbf{g}_i - \mathbf{y}_0.$$

Then (3.10) becomes

$$\mathbf{z}_i = h \sum_{\ell=1}^{s} a_{i\ell} f(t_0 + c_\ell h, \mathbf{y}_0 + \mathbf{z}_\ell), \quad i = 1, \dots, s,$$

which can be written as

$$\mathbf{z} := \begin{pmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_s \end{pmatrix} = (A \otimes I) \begin{pmatrix} hf(t_0 + c_1 h, \mathbf{y}_0 + \mathbf{z}_1) \\ \vdots \\ hf(t_0 + c_s h, \mathbf{y}_0 + \mathbf{z}_s) \end{pmatrix} =: (A \otimes I) F(\mathbf{z}), \tag{3.14}$$

where $\otimes$ denotes the Kronecker product between two matrices. Once $\mathbf{z}_1, \dots, \mathbf{z}_s$ are determined, we could then determine the next step by

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

This seems to suggest that s additional function evaluations are needed. In fact this can be avoided with a small trick. Assuming that A is invertible, it follows from (3.14) that we can write

$$\mathbf{y}_1 = \mathbf{y}_0 + \sum_{i=1}^{s} d_i \mathbf{z}_i,$$

where

$$\begin{pmatrix} d_1, & \dots, & d_s \end{pmatrix} := \mathbf{b}^T A^{-1}.$$

The kth step of the **Newton method applied to the nonlinear system** (3.14) takes the following form:

$$\begin{aligned} \text{Solve linear system} \quad & \big(I - (A \otimes I) F'(\mathbf{z}^k)\big) \triangle \mathbf{z}^k = -\mathbf{z}^k + h(A \otimes I) F(\mathbf{z}^k), \\ \text{Update} \quad & \mathbf{z}^{k+1} = \mathbf{z}^k + \triangle \mathbf{z}^k. \end{aligned} \tag{3.15}$$

Note that each step of (3.15) requires to solve a linear system. In practice, this is usually done via computing a (sparse) LU factorization of the matrix

$$\begin{aligned}
&I - (A \otimes I)F'(\mathbf{z}^k) \\
=&I - h \begin{pmatrix} a_{11}\frac{\partial f}{\partial \mathbf{y}}(t_0 + c_1 h, \mathbf{y}_0 + \mathbf{z}_1^k) & \cdots & a_{1s}\frac{\partial f}{\partial \mathbf{y}}(t_0 + c_1 h, \mathbf{y}_0 + \mathbf{z}_s^k) \\ \vdots & & \vdots \\ a_{s1}\frac{\partial f}{\partial \mathbf{y}}(t_0 + c_s h, \mathbf{y}_0 + \mathbf{z}_1^k) & \cdots & a_{ss}\frac{\partial f}{\partial \mathbf{y}}(t_0 + c_s h, \mathbf{y}_0 + \mathbf{z}_s^k) \end{pmatrix}.
\end{aligned}$$

The factorization of this $sd \times sd$ matrix is often the (by far) most expensive part and needs to be performed in every step of the Newton method (3.15).

We can reduce the cost significantly by, replacing all Jacobians $\frac{\partial f}{\partial \mathbf{y}}(t_0 + c_1 h, \mathbf{y}_0 + \mathbf{z}_s^k)$ with an approximation

$$J \approx \frac{\partial f}{\partial \mathbf{y}}(t_0, \mathbf{y}_0).$$

The resulting **simplified Newton method** takes the form:

$$\begin{array}{rl} \text{Solve linear system} & \big(I - (A \otimes I)J\big)\triangle \mathbf{z}^k = -\mathbf{z}^k + h(A \otimes I)F(\mathbf{z}^k), \\ \text{Update} & \mathbf{z}^{k+1} = \mathbf{z}^k + \triangle \mathbf{z}^k. \end{array} \tag{3.16}$$

Now, the LU factorization of $I - (A \otimes I)J$ needs to be computed only once and can then be reused. Hence, we only need to perform 1 LU factorization in each step of the implicit Runge-Kutta method.

The choice

$$\mathbf{z}^0 = \mathbf{0}$$

usually represents a very good starting value, since the exact solution is known to satisfy $\|\mathbf{z}\| = O(h)$. Suggestions for better starting values can be found in Section IV.8 of [HW], which also discusses a suitable stopping criterion for (3.16).

3.2.2 Examples of implicit Runge-Kutta methods

As described in [DB] and [HW], collocation combined with numerical quadrature provides a way to construct high-order implicit Runge-Kutta methods.

Gauss methods. Gauss quadrature yields s-stage implicit Runge-Kutta methods that are A-stable and have order $2s$ (which is optimal). For $s = 1$, the Gauss method coincides with the implicit midpoint rule. For $s = 2$, the Gauss method has the following Butcher tableau:

$$\begin{array}{c|cc} \frac{1}{2} - \frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4} - \frac{\sqrt{3}}{6} \\ \frac{1}{2} + \frac{\sqrt{3}}{6} & \frac{1}{4} + \frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

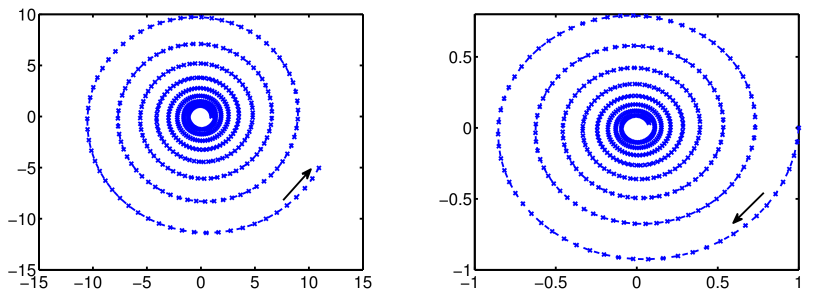

Figure 3.5. *Forward Euler (left plot) and implicit Euler (right plot) applied to (3.17).*

Radau methods. Radau quadrature yields s-stage implicit Runge-Kutta methods that are A-stable and have order $2s-1$. Moreover, Radau methods have a property called L-stability, see Section 3.4 below. For $s = 1$, the Radau method is our good old friend – the implicit Euler method. For $s = 2$ and $s = 3$, the Radau methods have the following Butcher tableaus:

$$\begin{array}{c|cc} \frac{1}{3} & \frac{5}{12} & -\frac{1}{12} \\ 1 & \frac{3}{4} & \frac{1}{4} \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array} \qquad \begin{array}{c|ccc} 0 & \frac{1}{9} & \frac{-1-\sqrt{6}}{18} & \frac{-1+\sqrt{6}}{18} \\ \frac{6-\sqrt{6}}{10} & \frac{1}{9} & \frac{88+7\sqrt{6}}{360} & \frac{88-43\sqrt{6}}{360} \\ \frac{6+\sqrt{6}}{10} & \frac{1}{9} & \frac{88+43\sqrt{6}}{360} & \frac{88-7\sqrt{6}}{360} \\ \hline & \frac{1}{9} & \frac{16+\sqrt{6}}{36} & \frac{16-\sqrt{6}}{36} \end{array}$$

RADAU5, one of the most popular codes for solving DAEs, is based on the 3-stage Radau method.

3.3 The danger of being too stable

Despite all the praise for A-stable methods, they actually bear the danger of turning an unstable or a non-asymptotically stable IVP into an asymptotically stable discrete-time system. This point is nicely illustrated with

$$\dot{\mathbf{y}}(t) = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \mathbf{y}(t), \quad \mathbf{y}(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} =: \mathbf{y}_0, \tag{3.17}$$

which arises, e.g., from the model of a spring pendulum. The solution stays on the unit circle and does not converge to a stationary point. Figure 3.5 displays the approximations on the interval $[0, 10]$ obtained from the forward/implicit Euler methods with step size $h = 1/10$. It can be seen that the forward Euler method does not stay on the unit circle and the solution drifts away as t increases. This is not unexpected: The eigenvalues of the matrix $A = \begin{pmatrix} 0 & 1 \\ -\omega^2 & 0 \end{pmatrix}$ are $\lambda = \pm\omega\mathrm{i}$.

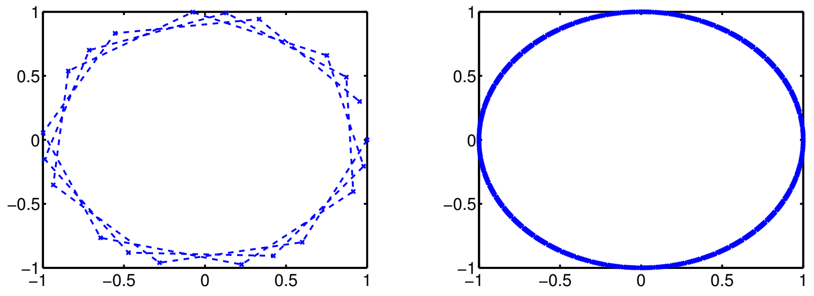

Figure 3.6. *Implicit trapezoidal method with $h = 1$ (left plot) and $h = 0.1$ (right plot) applied to (3.17).*

Hence, no matter how small h is, $h\lambda$ is not in the stability region of the forward Euler method. Unfortunately, the behavior of the implicit Euler method is not much better. The solution does also not stay on the unit circle and approaches zero for long times. Ironically, the problem is now that $h\lambda$ is in the interior of the stability region of the implicit Euler method. Thus, the approximate solution always converges to zero, no matter how small h is.

To avoid the phenomena described above, we need a method for which $h\lambda$ is on the boundary of the stability region. Such a method is given by the trapezoidal method, see Figure 3.4. Indeed, this method produces the exact asymptotic behavior even for relatively large h, see Figure 3.6.

3.4 L-stability

Section 3.3 may misleadingly indicate that it is *always* desirable to have the stability region coincide with the left half-plane. In fact, for very stiff problems this is not desirable at all. For a rational stability function S, we have

$$\lim_{x\to-\infty} S(x) = \lim_{x\to\infty} S(x) = \lim_{y\to\infty} S(\mathrm{i}y).$$

If the stability region coincide with the left half-plane, then $|S(\mathrm{i}y)| = 1$ for all $y \in \mathbb{R}$. In turn, $|S(z)|$ is close to 1 whenever z is close to the real axis and has large negative real part. In practice, this means that such a method damps errors only very slowly for stiff problems. To see this, let us consider the IVP

$$\dot{y}(t) = -2000(y(t) - \cos t), \qquad y(0) = 0. \tag{3.18}$$

As can be seen from Figure 3.7, the initial numerical oscillation is damped much more quickly for the implicit Euler method compared to the trapezoidal method. This is due to the following property.

Definition 3.10 *A method is called L-stable if it is A-stable and if, additionally,*

$$\lim_{x\to\infty} R(x) = 0.$$

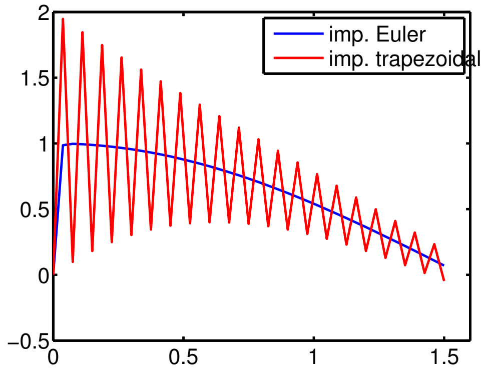

Figure 3.7. *Implicit Euler and implicit trapezoidal method with* $h = 1.5/40$ *applied to (3.18).*

For an implicit Runge-Kutta method defined by $\mathbf{b}, \mathbf{c}, A$ with nonsingular A, we have

$$S(\infty) = 1 - \mathbf{b}^T A^{-1}\mathbf{e},$$

where $\mathbf{e}$ is the vector of all ones. Hence, the method is L-stable if and only if $\mathbf{b}^T A^{-1}\mathbf{e} = 1$. Examples of L-stable methods include the implicit Euler method and the Radau methods mentioned above.

Apart from stiff ODEs, L-stability also plays an important role when solving differential-algebraic equations (DAEs), that is, ODEs with additional algebraic side constraints.

3.5 Rosenbrock methods*

When using the simplified Newton method (3.16), one has to strike a balance between the stopping criterion the inner iterations and h. It would be much simpler if we could just use one iteration and let the accuracy be handled solely by the step size control for the Runge-Kutta method. **Rosenbrock methods** (also called linearly implicit Runge-Kutta methods) rationalize this idea. They can be motivated by considering an implicit Runge-Kutta method with a lower triangular matrix A applied to an autonomous ODE:

$$\begin{aligned} \mathbf{k}_i &= f\Big(\mathbf{y}_0 + h\sum_{\ell=1}^{i-1} a_{i\ell}\mathbf{k}_\ell + a_{ii}\mathbf{k}_i\Big), \quad i = 1,\dots,s, \\ \mathbf{y}_1 &= \mathbf{y}_0 + h\sum_{i=1}^{s} b_i\mathbf{k}_i. \end{aligned}$$

Such a method is also called DIRK (diagonally implicit Runge-Kutta method). Linearizing this formula in order to get rid off the implicit part yields

$$\mathbf{k}_i = f(\mathbf{g}_i) + hf'(\mathbf{g}_i)a_{ii}\mathbf{k}_i,$$

where

$$\mathbf{g}_i = \mathbf{y}_0 + h\sum_{\ell=1}^{i-1} a_{i\ell}\mathbf{k}_\ell$$

for $i = 1, \dots, s$. This is one step of the Newton method. Now we replace $f'(\mathbf{g}_i)$ by $J = f'(\mathbf{y}_0)$ and obtain one step of the simplified Newton method. Some additional freedom is gained by allowing for linear combinations of $J\mathbf{k}_\ell$.

Definition 3.11 *An s-stage Rosenbrock method is defined by*

$$\begin{aligned}\mathbf{k}_i &= f\Big(\mathbf{y}_0 + h\sum_{\ell=1}^{i-1} a_{i\ell}\mathbf{k}_\ell\Big) + hJ\sum_{\ell=1}^{i}\gamma_{i\ell}\mathbf{k}_\ell, \quad i = 1, \dots, s,\\ \mathbf{y}_1 &= \mathbf{y}_0 + h\sum_{i=1}^{s} b_i\mathbf{k}_i,\end{aligned}$$

with coefficients $\alpha_{i\ell}, \gamma_{i\ell}, b_i$ and $J = f'(\mathbf{y}_0)$.

Note that each stage of the Rosenbrock method requires to solve a linear system with the matrix $I - h\gamma_{ii}J$. Naturally, methods with $\gamma_{11} = \cdots = \gamma_{ss} \equiv \gamma$ are of particular interest, because they allow to reuse the LU factorization of $I - h\gamma J$ when having to solve the linear system for each stage. We refer to Section IV.7 in [HW] for the construction of specific Rosenbrock methods.

A particularly simple example of a Rosenbrock method is given by:

$$\begin{aligned}(I - h\gamma J)\mathbf{k}_1 &= f(\mathbf{y}_0), \qquad \gamma = \frac{1}{2+\sqrt{2}},\\ (I - h\gamma J)\mathbf{k}_2 &= f\Big(\mathbf{y}_0 + \frac{1}{2}h\mathbf{k}_1\Big) - h\gamma J\mathbf{k}_1\\ \mathbf{y}_1 &= \mathbf{y}_1 + h\mathbf{k}_2.\end{aligned}$$

A modified variant of this second-order method is behind MATLAB's `ode23s`; see [L. F. Shampine and M. W. Reichelt: The MATLAB ODE suite] for more details.