

Chapter II. Runge-Kutta and Extrapolation Methods

Numerical methods for ordinary differential equations fall naturally into two classes: those which use *one* starting value at each step (“one-step methods”) and those which are based on *several* values of the solution (“multistep methods” or “multi-value methods”). The present chapter is devoted to the study of one-step methods, while multistep methods are the subject of Chapter III. Both chapters can, to a large extent, be read independently of each other.

We start with the theory of Runge-Kutta methods: the derivation of order conditions with the help of labelled trees, error estimates, convergence proofs, implementation, methods of higher order, dense output. Section II.7 introduces implicit Runge-Kutta methods. More attention will be drawn to these methods in Volume II on stiff differential equations. Two sections then discuss the elegant idea of *extrapolation* (Richardson, Romberg, etc) and its use in obtaining high order codes. The methods presented are then tested and compared on a series of problems. The potential of parallelism is discussed in a separate section. We then turn our attention to an algebraic theory of the composition of methods. This will be the basis for the study of order properties for many general classes of methods in the following chapter. The chapter ends with special methods for second order differential equations $y'' = f(x, y)$, for Hamiltonian systems (symplectic methods) and for problems with delay.

We illustrate the methods of this chapter with an example from Astronomy, the restricted three body problem. One considers two bodies of masses $1 - \mu$ and μ in circular rotation in a plane and a third body of negligible mass moving around in the same plane. The equations are (see e.g., the classical textbook Szebehely 1967)

$$\begin{aligned} y_1'' &= y_1 + 2y_2' - \mu' \frac{y_1 + \mu}{D_1} - \mu \frac{y_1 - \mu'}{D_2}, \\ y_2'' &= y_2 - 2y_1' - \mu' \frac{y_2}{D_1} - \mu \frac{y_2}{D_2}, \\ D_1 &= ((y_1 + \mu)^2 + y_2^2)^{3/2}, \quad D_2 = ((y_1 - \mu')^2 + y_2^2)^{3/2}, \\ \mu &= 0.012277471, \quad \mu' = 1 - \mu. \end{aligned} \tag{0.1}$$

There exist initial values, for example

$$\begin{aligned} y_1(0) &= 0.994, & y_1'(0) &= 0, & y_2(0) &= 0, \\ y_2'(0) &= -2.00158510637908252240537862224, \\ x_{\text{end}} &= 17.0652165601579625588917206249, \end{aligned} \quad (0.2)$$

such that the solution is periodic with period x_{end} . Such periodic solutions have fascinated astronomers and mathematicians for many decades (Poincaré; extensive numerical calculations are due to Sir George Darwin (1898)) and are now often called “Arenstorf orbits” (see Arenstorf (1963) who did numerical computations “on high speed electronic computers”). The problem is C^∞ with the exception of the two singular points $y_1 = -\mu$ and $y_1 = 1 - \mu$, $y_2 = 0$, therefore the Euler polygons of Section I.7 are known to converge to the solution. But are they really numerically useful here? We have chosen 24000 steps of step length $h = x_{\text{end}}/24000$ and plotted the result in Figure 0.1. The result is not very striking.

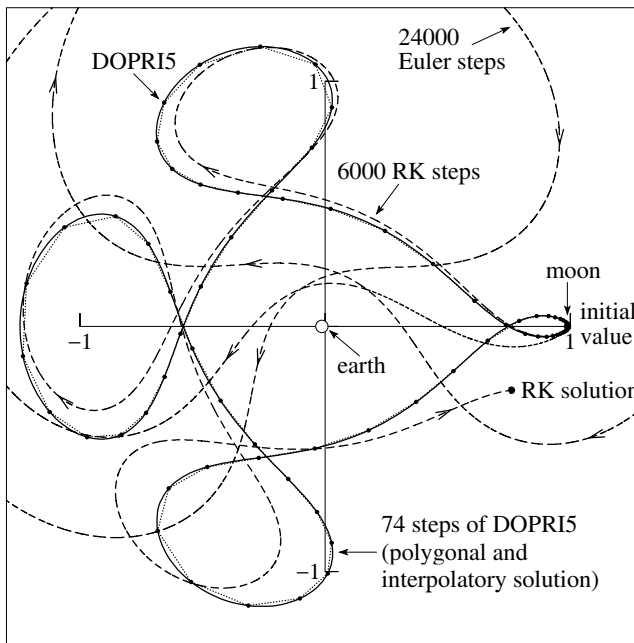


Fig. 0.1. An Arenstorf orbit computed by equidistant Euler, equidistant Runge-Kutta and variable step size Dormand & Prince

The performance of the Runge-Kutta method (left tableau of Table 1.2) is already much better and converges faster to the solution. We have used 6000 steps of step size $x_{\text{end}}/6000$, so that the numerical work becomes equivalent. Clearly, most accuracy is lost in those parts of the orbit which are close to a singularity. Therefore, codes with automatic step size selection, described in Section II.4, perform

much better and the code DOPRI5 (Table 5.2) computes the orbit with a precision of 10^{-3} in 98 steps (74 accepted and 24 rejected). The step size becomes very large in some regions and the graphical representation as polygons connecting the solution points becomes unsatisfactory. The solid line is the interpolatory solution (Section II.6), which is also precise for all intermediate values and useful for many other questions such as delay differential equations, event location or discontinuities in the differential equation.

For still higher precision one needs methods of higher order. For example, the code DOP853 (Section II.5) computes the orbit faster than DOPRI5 for more stringent tolerances, say smaller than about 10^{-6} . The highest possible order is obtained by extrapolation methods (Section II.9) and the code ODEX (with $K_{\max} = 15$) obtains the orbit with a precision of 10^{-30} with about 25000 function evaluations, precisely the same amount of work as for the above Euler solution.

II.1 The First Runge-Kutta Methods

Die numerische Berechnung irgend einer Lösung einer gegebenen Differentialgleichung, deren analytische Lösung man nicht kennt, hat, wie es scheint, die Aufmerksamkeit der Mathematiker bisher wenig in Anspruch genommen . . . (C. Runge 1895)

The Euler method for solving the initial value problem

$$y' = f(x, y), \quad y(x_0) = y_0 \quad (1.1)$$

was described by Euler (1768) in his “*Institutiones Calculi Integralis*” (Sectio Secunda, Caput VII). The method is easy to understand and to implement. We have studied its convergence extensively in Section I.7 and have seen that the global error behaves like Ch , where C is a constant depending on the problem and h is the maximal step size. If one wants a precision of, say, 6 decimals, one would thus need about a million steps, which is not very satisfactory. On the other hand, one knows since the time of Newton that much more accurate methods can be found, if f in (1.1) is independent of y , i.e., if we have a quadrature problem

$$y' = f(x), \quad y(x_0) = y_0 \quad (1.1')$$

with solution

$$y(X) = y_0 + \int_{x_0}^X f(x) dx. \quad (1.2)$$

As an example consider the midpoint rule (or first Gauss formula)

$$\begin{aligned} y(x_0 + h_0) &\approx y_1 = y_0 + h_0 f\left(x_0 + \frac{h_0}{2}\right) \\ y(x_1 + h_1) &\approx y_2 = y_1 + h_1 f\left(x_1 + \frac{h_1}{2}\right) \end{aligned} \quad (1.3')$$

...

$$y(X) \approx Y = y_{n-1} + h_{n-1} f\left(x_{n-1} + \frac{h_{n-1}}{2}\right),$$

where $h_i = x_{i+1} - x_i$ and $x_0, x_1, \dots, x_{n-1}, x_n = X$ is a subdivision of the integration interval. Its global error $y(X) - Y$ is known to be bounded by Ch^2 . Thus for a desired precision of 6 decimals, a thousand steps will usually do, i.e., the method here is a thousand times faster. Therefore Runge (1895) asked whether it would also be possible to extend method (1.3') to problem (1.1). The first step with $h = h_0$ would read

$$y(x_0 + h) \approx y_0 + hf\left(x_0 + \frac{h}{2}, y\left(x_0 + \frac{h}{2}\right)\right), \quad (1.3)$$

but which value should we take for $y(x_0 + h/2)$? In the absence of something better, it is natural to use one small Euler step with step size $h/2$ and obtain from (1.3) ¹

$$\begin{aligned} k_1 &= f(x_0, y_0) \\ k_2 &= f\left(x_0 + \frac{h}{2}, y_0 + \frac{h}{2}k_1\right) \\ y_1 &= y_0 + hk_2. \end{aligned} \tag{1.4}$$

One might of course be surprised that we propose an Euler step for the computation of k_2 , just half a page after preaching its inefficiency. The crucial point is, however, that k_2 is multiplied by h in the third expression and therefore its error becomes less important. To be more precise, we compute the Taylor expansion of y_1 in (1.4) as a function of h ,

$$\begin{aligned} y_1 &= y_0 + hf\left(x_0 + \frac{h}{2}, y_0 + \frac{h}{2}f_0\right) \\ &= y_0 + hf(x_0, y_0) + \frac{h^2}{2}(f_x + f_y f)(x_0, y_0) \\ &\quad + \frac{h^3}{8}(f_{xx} + 2f_{xy}f + f_{yy}f^2)(x_0, y_0) + \dots \end{aligned} \tag{1.5}$$

This can be compared with the Taylor series of the exact solution, which is obtained from (1.1) by repeated differentiation and replacing y' by f every time it appears (Euler (1768), Problema 86, §656, see also (8.12) of Chap. I)

$$\begin{aligned} y(x_0 + h) &= y_0 + hf(x_0, y_0) + \frac{h^2}{2}(f_x + f_y f)(x_0, y_0) \\ &\quad + \frac{h^3}{6}(f_{xx} + 2f_{xy}f + f_{yy}f^2 + f_y f_x + f_y^2 f)(x_0, y_0) + \dots \end{aligned} \tag{1.6}$$

Subtracting these two equations, we obtain for the error of the first step

$$y(x_0 + h) - y_1 = \frac{h^3}{24}(f_{xx} + 2f_{xy}f + f_{yy}f^2 + 4(f_y f_x + f_y^2 f))(x_0, y_0) + \dots \tag{1.7}$$

When all second partial derivatives of f are bounded, we thus obtain

$$\|y(x_0 + h) - y_1\| \leq Kh^3.$$

In order to obtain an approximation of the solution of (1.1) at the endpoint X , we apply formula (1.4) successively to the intervals (x_0, x_1) , (x_1, x_2) , \dots , (x_{n-1}, X) , very similarly to the application of Euler's method in Section I.7. Again similarly to the convergence proof of Section I.7, it will be shown in Section II.3 that, as in the case (1.1'), the error of the numerical solution is bounded by Ch^2 (h the maximal step size). Method (1.4) is thus an improvement on the Euler method. For high precision computations we need to find still better methods; this will be the main task of what follows.

¹ The analogous extension of the *trapezoidal rule* has been given in an early publication by Coriolis in 1837; see Chapter II.4.2 of the thesis of D. Tournès, Paris VII, 1996.

General Formulation of Runge-Kutta Methods

Runge (1895) and Heun (1900) constructed methods by including additional Euler steps in (1.4). It was Kutta (1901) who then formulated the general scheme of what is now called a Runge-Kutta method:

Definition 1.1. Let s be an integer (the “number of stages”) and $a_{21}, a_{31}, a_{32}, \dots, a_{s1}, a_{s2}, \dots, a_{s,s-1}, b_1, \dots, b_s, c_2, \dots, c_s$ be real coefficients. Then the method

$$\begin{aligned}
 k_1 &= f(x_0, y_0) \\
 k_2 &= f(x_0 + c_2 h, y_0 + h a_{21} k_1) \\
 k_3 &= f(x_0 + c_3 h, y_0 + h(a_{31} k_1 + a_{32} k_2)) \\
 &\dots \\
 k_s &= f(x_0 + c_s h, y_0 + h(a_{s1} k_1 + \dots + a_{s,s-1} k_{s-1})) \\
 y_1 &= y_0 + h(b_1 k_1 + \dots + b_s k_s)
 \end{aligned} \tag{1.8}$$

is called an s -stage explicit Runge-Kutta method (ERK) for (1.1).

Usually, the c_i satisfy the conditions

$$c_2 = a_{21}, \quad c_3 = a_{31} + a_{32}, \quad \dots \quad c_s = a_{s1} + \dots + a_{s,s-1}, \tag{1.9}$$

or briefly,

$$c_i = \sum_{j=1}^{i-1} a_{ij}. \tag{1.9'}$$

These conditions, already assumed by Kutta, express that all points where f is evaluated are first order approximations to the solution. They greatly simplify the derivation of order conditions for high order methods. For low orders, however, these assumptions are not necessary (see Exercise 6).

Definition 1.2. A Runge-Kutta method (1.8) has *order* p if for sufficiently smooth problems (1.1),

$$\|y(x_0 + h) - y_1\| \leq K h^{p+1}, \tag{1.10}$$

i.e., if the Taylor series for the exact solution $y(x_0 + h)$ and for y_1 coincide up to (and including) the term h^p .

With the paper of Butcher (1964b) it became customary to symbolize method (1.8) by the tableau (1.8').

$$\begin{array}{c|cccc}
 0 & & & & \\
 c_2 & a_{21} & & & \\
 c_3 & a_{31} & a_{32} & & \\
 \vdots & \vdots & \vdots & \ddots & \\
 c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} \\
 \hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
 \end{array} \quad (1.8')$$

Examples. The above method of Runge as well as methods of Runge and Heun of order 3 are given in Table 1.1.

Table 1.1. Low order Runge-Kutta methods

$ \begin{array}{c c} 0 & \\ 1/2 & 1/2 \\ \hline & 0 \quad 1 \\ \text{Runge, order 2} \end{array} $		$ \begin{array}{c ccc} 0 & & & \\ 1/2 & 1/2 & & \\ 1 & 0 & 1 & \\ 1 & 0 & 0 & 1 \\ \hline & 1/6 & 2/3 & 0 \quad 1/6 \\ \text{Runge, order 3} \end{array} $				$ \begin{array}{c ccc} 0 & & & \\ 1/3 & 1/3 & & \\ 2/3 & 0 & 2/3 & \\ \hline & 1/4 & 0 & 3/4 \\ \text{Heun, order 3} \end{array} $			
--	--	--	--	--	--	---	--	--	--

Discussion of Methods of Order 4

Von den neueren Verfahren halte ich das folgende von Herrn Kutta angegebene für das beste. (C. Runge 1905)

Our task is now to determine the coefficients of 4-stage Runge-Kutta methods (1.8) in order that they be of order 4. We have seen above what we must do: compute the derivatives of $y_1 = y_1(h)$ for $h = 0$ and compare them with those of the true solution for orders 1, 2, 3, and 4. In theory, with the known rules of differential calculus, this is a completely trivial task and, by the use of (1.9), results in the following conditions:

$$\sum_i b_i = b_1 + b_2 + b_3 + b_4 = 1 \quad (1.11a)$$

$$\sum_i b_i c_i = b_2 c_2 + b_3 c_3 + b_4 c_4 = 1/2 \quad (1.11b)$$

$$\sum_i b_i c_i^2 = b_2 c_2^2 + b_3 c_3^2 + b_4 c_4^2 = 1/3 \quad (1.11c)$$

$$\sum_{i,j} b_i a_{ij} c_j = b_3 a_{32} c_2 + b_4 (a_{42} c_2 + a_{43} c_3) = 1/6 \quad (1.11d)$$

$$\sum_i b_i c_i^3 = b_2 c_2^3 + b_3 c_3^3 + b_4 c_4^3 = 1/4 \quad (1.11e)$$

$$\sum_{i,j} b_i c_i a_{ij} c_j = b_3 c_3 a_{32} c_2 + b_4 c_4 (a_{42} c_2 + a_{43} c_3) = 1/8 \quad (1.11f)$$

$$\sum_{i,j} b_i a_{ij} c_j^2 = b_3 a_{32} c_2^2 + b_4 (a_{42} c_2^2 + a_{43} c_3^2) = 1/12 \quad (1.11g)$$

$$\sum_{i,j,k} b_i a_{ij} a_{jk} c_k = b_4 a_{43} a_{32} c_2 = 1/24. \quad (1.11h)$$

These computations, which are not reproduced in Kutta's paper (they are, however, in Heun 1900), are very tedious. And they grow enormously with higher orders. We shall see in Section II.2 that by using an appropriate notation, they can become very elegant.

Kutta gave the general solution of (1.11) without comment. A clear derivation of the solutions is given in Runge & König (1924), p. 291. We shall follow here the ideas of J.C. Butcher, which make clear the role of the so-called *simplifying assumptions*, and will also apply to higher order cases.

Lemma 1.3. *If*

$$\sum_{i=j+1}^s b_i a_{ij} = b_j (1 - c_j), \quad j = 1, \dots, s, \quad (1.12)$$

then the equations (d), (g), and (h) in (1.11) follow from the others.

Proof. We demonstrate this for (g):

$$\sum_{i,j} b_i a_{ij} c_j^2 = \sum_j b_j c_j^2 - \sum_j b_j c_j^3 = \frac{1}{3} - \frac{1}{4} = \frac{1}{12}$$

by (c) and (e). Equations (d) and (h) are derived similarly. \square

We shall now show that (1.12) is also *necessary* in our case:

Lemma 1.4. *For $s = 4$, the equations (1.11) and (1.9) imply (1.12).*

The proof of this lemma will be based on the following:

Lemma 1.5. *Let U and V be 3×3 matrices such that*

$$UV = \begin{pmatrix} a & b & 0 \\ c & d & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \det \begin{pmatrix} a & b \\ c & d \end{pmatrix} \neq 0. \quad (1.13)$$

Then either $Ve_3 = 0$ or $U^T e_3 = 0$ where $e_3 = (0, 0, 1)^T$.

Proof of Lemma 1.5. If $\det U \neq 0$, then $UVe_3 = 0$ implies $Ve_3 = 0$. If $\det U = 0$, there exists $x = (x_1, x_2, x_3)^T \neq 0$ such that $U^T x = 0$, and therefore $V^T U^T x = 0$. But (1.13) implies that x must be a multiple of e_3 . \square

Proof of Lemma 1.4. Define

$$d_j = \sum_i b_i a_{ij} - b_j(1 - c_j) \quad \text{for} \quad j = 1, \dots, 4,$$

so that we have to prove $d_j = 0$. We now introduce the matrices

$$U = \begin{pmatrix} b_2 & b_3 & b_4 \\ b_2 c_2 & b_3 c_3 & b_4 c_4 \\ d_2 & d_3 & d_4 \end{pmatrix}, \quad V = \begin{pmatrix} c_2 & c_2^2 & \sum_j a_{2j} c_j - c_2^2/2 \\ c_3 & c_3^2 & \sum_j a_{3j} c_j - c_3^2/2 \\ c_4 & c_4^2 & \sum_j a_{4j} c_j - c_4^2/2 \end{pmatrix}. \quad (1.14)$$

Multiplication of these two matrices, using the conditions of (1.11), gives

$$UV = \begin{pmatrix} 1/2 & 1/3 & 0 \\ 1/3 & 1/4 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad \text{with} \quad \det \begin{pmatrix} 1/2 & 1/3 \\ 1/3 & 1/4 \end{pmatrix} \neq 0.$$

Now the last column of V cannot be zero, since $c_1 = 0$ implies

$$\sum_j a_{2j} c_j - c_2^2/2 = -c_2^2/2 \neq 0$$

by condition (h). Thus $d_2 = d_3 = d_4 = 0$ follows from Lemma 1.5. The last identity $d_1 = 0$ follows from $d_1 + d_2 + d_3 + d_4 = 0$, which is a consequence of (1.11a,b) and (1.9). \square

From Lemmas 1.3 and 1.4 we obtain

Theorem 1.6. *Under the assumption (1.9) the equations (1.11) are equivalent to*

$$b_1 + b_2 + b_3 + b_4 = 1 \quad (1.15a)$$

$$b_2 c_2 + b_3 c_3 + b_4 c_4 = 1/2 \quad (1.15b)$$

$$b_2 c_2^2 + b_3 c_3^2 + b_4 c_4^2 = 1/3 \quad (1.15c)$$

$$b_2 c_2^3 + b_3 c_3^3 + b_4 c_4^3 = 1/4 \quad (1.15e)$$

$$b_3 c_3 a_{32} c_2 + b_4 c_4 (a_{42} c_2 + a_{43} c_3) = 1/8 \quad (1.15f)$$

$$b_3 a_{32} + b_4 a_{42} = b_2(1 - c_2) \quad (1.15i)$$

$$b_4 a_{43} = b_3(1 - c_3) \quad (1.15j)$$

$$0 = b_4(1 - c_4). \quad (1.15k)$$

\square

It follows from (1.15j) and (1.11h) that

$$b_3 b_4 c_2 (1 - c_3) \neq 0. \quad (1.16)$$

In particular this implies $c_4 = 1$ by (1.15k).

Solution of equations (1.15). Equations (a)-(e) and (k) just state that b_i and c_i are the coefficients of a fourth order quadrature formula with $c_1 = 0$ and $c_4 = 1$. We distinguish four cases for this:

1) $c_2 = u$, $c_3 = v$ and $0, u, v, 1$ are all distinct; (1.17)

then (a)-(e) form a regular linear system for b_1, b_2, b_3, b_4 . This system has the solution

$$b_1 = \frac{1 - 2(u + v) + 6uv}{12uv}, \quad b_2 = \frac{2v - 1}{12u(1 - u)(v - u)},$$

$$b_3 = \frac{1 - 2u}{12v(1 - v)(v - u)}, \quad b_4 = \frac{3 - 4(u + v) + 6uv}{12(1 - u)(1 - v)}.$$

Due to (1.16) we have to assume that u, v are such that $b_3 \neq 0$ and $b_4 \neq 0$. The three other cases with double nodes are built upon the Simpson rule:

2) $c_3 = 0$, $c_2 = 1/2$, $b_3 = w \neq 0$, $b_1 = 1/6 - w$, $b_2 = 4/6$, $b_4 = 1/6$;

3) $c_2 = c_3 = 1/2$, $b_1 = 1/6$, $b_3 = w \neq 0$, $b_2 = 4/6 - w$, $b_4 = 1/6$;

4) $c_2 = 1$, $c_3 = 1/2$, $b_4 = w \neq 0$, $b_2 = 1/6 - w$, $b_1 = 1/6$, $b_3 = 4/6$.

Once b_i and c_i are chosen, we obtain a_{43} from (j), and then (f) and (i) form a linear system of two equations for a_{32} and a_{42} . The determinant of this system is

$$\det \begin{pmatrix} b_3 & b_4 \\ b_3 c_3 c_2 & b_4 c_4 c_2 \end{pmatrix} = b_3 b_4 c_2 (c_4 - c_3)$$

which is $\neq 0$ by (1.16). Finally we obtain a_{21} , a_{31} , and a_{41} from (1.9).

Two particular choices of Kutta (1901) have become especially popular: case (3) with $w = 2/6$ and case (1) with $u = 1/3$, $v = 2/3$. They are given in Table 1.2. Both methods generalize classical quadrature rules in keeping the same order. The first is more popular, the second is more precise ("Wir werden diese Näherung als im allgemeinen beste betrachten . . .", Kutta).

Table 1.2. Kutta's methods

0					0				
1/2	1/2				1/3	1/3			
1/2	0	1/2			2/3	-1/3	1		
1	0	0	1		1	1	-1	1	
<hr/>					<hr/>				
	1/6	2/6	2/6	1/6		1/8	3/8	3/8	1/8
"The" Runge-Kutta method					3/8-Rule				

“Optimal” Formulas

Much research has been undertaken, in order to choose the “best” possibilities from the variety of possible 4th order RK-formulas.

The first attempt in this direction was the very popular method of Gill (1951), with the aim of reducing the need for computer storage (“registers”) as much as possible. The first computers in the fifties largely used this method which is therefore of historical interest. Gill observed that most computer storage is needed for the computation of k_3 , where “registers are required to store in some form”

$$y_0 + a_{31}hk_1 + a_{32}hk_2, \quad y_0 + a_{41}hk_1 + a_{42}hk_2, \quad y_0 + b_1hk_1 + b_2hk_2, \quad hk_3.$$

“Clearly, three registers will suffice for the third stage if the quantities to be stored are linearly dependent, i.e., if”

$$\det \begin{pmatrix} 1 & a_{31} & a_{32} \\ 1 & a_{41} & a_{42} \\ 1 & b_1 & b_2 \end{pmatrix} = 0.$$

Gill observed that this condition is satisfied for the methods of type (3) if $w = (1 + \sqrt{0.5})/3$. The resulting method can then be reformulated as follows (“As each quantity is calculated it is stored in the register formerly holding the corresponding quantity of the previous stage, which is no longer required”):

$$\begin{aligned} y &:= \text{initial value}, & k &:= hf(y), & y &:= y + 0.5k, & q &:= k, \\ k &:= hf(y), & y &:= y + (1 - \sqrt{0.5})(k - q), \\ q &:= (2 - \sqrt{2})k + (-2 + 3\sqrt{0.5})q, \\ k &:= hf(y), & y &:= y + (1 + \sqrt{0.5})(k - q), \\ q &:= (2 + \sqrt{2})k + (-2 - 3\sqrt{0.5})q, \\ k &:= hf(y), & y &:= y + \frac{k}{6} - \frac{q}{3}, & (\rightarrow \text{ compute next step}) . \end{aligned} \tag{1.18}$$

Today, in large high-speed computers, this method is no longer used, but could still be of interest for very high dimensional equations.

Other attempts have been made to choose u and v in (1.17), case (1), such that the *error terms* (terms in h^5 , see Section II.3) become as small as possible. We shall discuss this question in Section II.3.

Numerical Example

Zu grosses Gewicht darf man natürlich solchen Beispielen nicht
beilegen . . .
(W. Kutta 1901)

We compare five different choices of 4th order methods on the Van der Pol equation (I.16.2) with $\varepsilon = 1$. As initial values we take $y_1(0) = A$, $y_2(0) = 0$ on the limit cycle and we integrate over one period T (the values of A and T are given in Exercise I.16.1). For a comparison of these methods with lower order ones we have also included the explicit Euler method, Runge's method of order 2 and Heun's method of order 3 (see Table 1.1).

We have applied the methods with several fixed step sizes. The errors of both components and the number of function evaluations (fe) are displayed in logarithmic scales in Fig. 1.1. Whenever the error behaves like $C \cdot h^p = C_1 \cdot (fe)^{-p}$, the curves appear as straight lines with slope $1/p$. We have chosen the scales such that the theoretical slope of the 4th order methods appears to be 45° .

These tests clearly show up the importance of higher order methods. Among the various 4th order methods there is usually no big difference. It is interesting to note that in our example the method with the smallest error in y_1 has the biggest error in y_2 and vice versa.

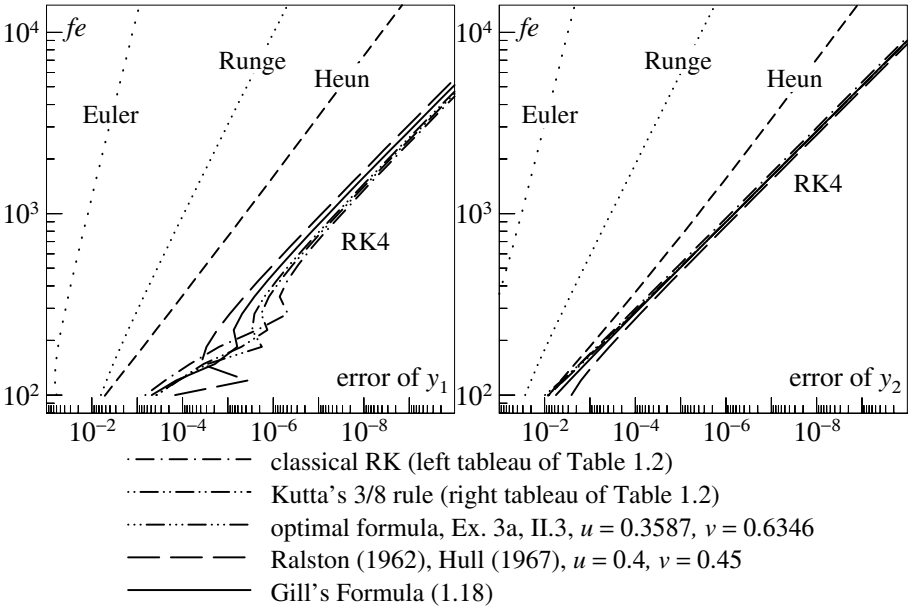


Fig. 1.1. Global errors versus number of function evaluations

Exercises

1. Show that every s -stage explicit RK method of order s , when applied to the problem $y' = \lambda y$ (λ a complex constant), gives

$$y_1 = \left(\sum_{j=0}^s \frac{z^j}{j!} \right) y_0, \quad z = h\lambda.$$

Hint. Show first that y_1/y_0 must be a polynomial in z of degree s and then determine its coefficients by comparing the derivatives of y_1 , with respect to h , to those of the true solution.

2. (Runge 1895, p. 175; see also the introduction to Adams methods in Chap. III.1). The theoretical form of drops of fluids is determined by the differential equation of Laplace (1805)

$$-z = \alpha^2 \frac{(K_1 + K_2)}{2} \quad (1.21)$$

where α is a constant, $(K_1 + K_2)/2$ the mean curvature, and z the height (see Fig. 1.2). If we insert $1/K_1 = r/\sin \varphi$ and $K_2 = d\varphi/ds$, the curvature of the meridian curve, we obtain

$$-2z = \alpha^2 \left(\frac{\sin \varphi}{r} + \frac{d\varphi}{ds} \right), \quad (1.22)$$

where we put $\alpha = 1$. Add

$$\frac{dr}{ds} = \cos \varphi, \quad \frac{dz}{ds} = -\sin \varphi, \quad (1.22')$$

to obtain a system of three differential equations for $\varphi(s)$, $r(s)$, $z(s)$, s being the arc length. Compute and plot different solution curves by the method of Runge (1.4) with initial values $\varphi(0) = 0$, $r(0) = 0$ and $z(0) = z_0$ ($z_0 < 0$ for lying drops; compute also hanging drops with appropriate sign changes in (1.22)). Use different step sizes and compare the results.

Hint. Be careful at the singularity in the beginning: from (1.22) and (1.22') we have for small s that $r = s$, $\varphi = \zeta s$ with $\zeta = -z_0$, hence $(\sin \varphi)/r \rightarrow -z_0$. A more precise analysis gives for small s the expansions ($\zeta = -z_0$)

$$\begin{aligned} \varphi &= \zeta s + \frac{\zeta}{4} s^3 + \left(\frac{\zeta}{48} - \frac{\zeta^3}{120} \right) s^5 + \dots \\ r &= s - \frac{\zeta^2}{6} s^3 + \left(-\frac{\zeta^2}{20} + \frac{\zeta^4}{120} \right) s^5 + \dots \\ z &= -\zeta - \frac{\zeta}{2} s^2 + \left(-\frac{\zeta}{16} + \frac{\zeta^3}{24} \right) s^4 + \left(-\frac{\zeta}{288} + \frac{\zeta^3}{45} - \frac{\zeta^5}{720} \right) s^6 + \dots \end{aligned}$$

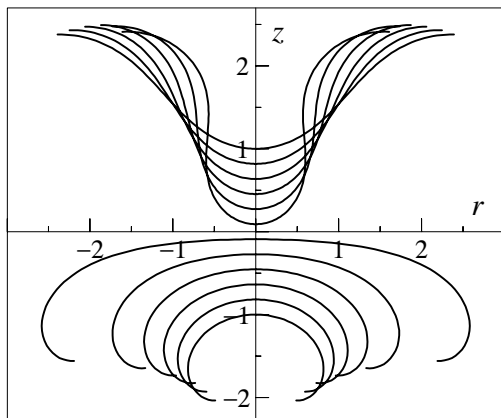


Fig. 1.2. Drops

3. Find the conditions for a 2-stage explicit RK-method to be of order two and determine all such methods (“... wozu eine weitere Erörterung nicht mehr nötig ist”, Kutta).
4. Find all methods of order three with three stages (i.e., solve (1.11;a-d) with $b_4 = 0$).

Result. $c_2 = u$, $c_3 = v$, $a_{32} = v(v-u)/(u(2-3u))$, $b_2 = (2-3v)/(6u(u-v))$, $b_3 = (2-3u)/(6v(v-u))$, $b_1 = 1 - b_2 - b_3$, $a_{31} = c_3 - a_{32}$, $a_{21} = c_2$ (Kutta 1901, p. 438).

5. Construct all methods of order 2 of the form

0				
c_2				
c_3			c_2	
			0	c_3
			0	0
			0	1

Such methods “have the property that the corresponding Runge-Kutta process requires relatively less storage in a computer” (Van der Houwen (1977), §2.7.2). Apply them to $y' = \lambda y$ and compare with Exercise 1.

6. Determine the conditions for order two of the RK methods with two stages which do *not* satisfy the conditions (1.9):

$$\begin{aligned}
 k_1 &= f(x_0 + c_1 h, y_0) \\
 k_2 &= f(x_0 + c_2 h, y_0 + a_{21} h k_1) \\
 y_1 &= y_0 + h(b_1 k_1 + b_2 k_2).
 \end{aligned}$$

Discuss the use of this extra freedom for c_1 and c_2 (Oliver 1975).

II.2 Order Conditions for Runge-Kutta Methods

... I heard a lecture by Merson ...
(J. Butcher's first contact with RK methods)

In this section we shall derive the general structure of the order conditions (Merson 1957, Butcher 1963). The proof has evolved very much in the meantime, mainly under the influence of Butcher's later work, many personal discussions with him, the proof of "Theorem 6" in Hairer & Wanner (1974), and our teaching experience. We shall see in Section II.11 that exactly the same ideas of proof lead to a general theorem of composition of methods (= *B*-series), which gives access to order conditions for a much larger class of methods.

A big advantage is obtained by transforming (1.1) to *autonomous* form by appending x to the dependent variables as

$$\begin{pmatrix} x \\ y \end{pmatrix}' = \begin{pmatrix} 1 \\ f(x, y) \end{pmatrix}. \quad (2.1)$$

The main difficulty in the derivation of the order conditions is to understand the correspondence of the formulas to certain rooted labelled trees; this comes out most naturally if we use well-chosen indices and tensor notation (as in Gill (1951), Henrici (1962), p. 118, Gear (1971), p. 32). As is usual in tensor notation, we denote (in this section) the components of vectors by *superscript* indices which, in order to avoid confusion, we choose as *capitals*. Then (2.1) can be written as

$$(y^J)' = f^J(y^1, \dots, y^n), \quad J = 1, \dots, n. \quad (2.2)$$

We next rewrite the method (1.8) for the autonomous differential equation (2.2). In order to get a better symmetry in all formulas of (1.8), we replace k_i by the argument g_i such that $k_i = f(g_i)$. Then (1.8) becomes

$$\begin{aligned} g_i^J &= y_0^J + \sum_{j=1}^{i-1} a_{ij} h f^J(g_j^1, \dots, g_j^n), \quad i = 1, \dots, s \\ y_1^J &= y_0^J + \sum_{j=1}^s b_j h f^J(g_j^1, \dots, g_j^n). \end{aligned} \quad (2.3)$$

If the system (2.2) originates from (2.1), then, for $J = 1$,

$$g_i^1 = y_0^1 + \sum_{j=1}^{i-1} a_{ij} h = x_0 + c_i h$$

by (1.9). We see that (1.9) becomes a natural condition. If it is satisfied, then for the derivation of order conditions only the autonomous equation (2.2) has to be considered.

As indicated in Section II.1 we have to compare the Taylor series of y_1^J with that of the exact solution. Therefore we compute the derivatives of y_1^J and g_i^J with respect to h at $h = 0$. Due to the similarity of the two formulas, it is sufficient to do this for g_i^J . On the right hand side of (2.3) there appear expressions of the form $h\varphi(h)$, so we make use of Leibniz' formula

$$(h\varphi(h))^{(q)}|_{h=0} = q \cdot (\varphi(h))^{(q-1)}|_{h=0}. \quad (2.4)$$

The reader is now asked to take a deep breath, take five sheets of reversed computer paper, remember the basic rules of differential calculus, and begin the following computations:

$q = 0$: from (2.3)

$$(g_i^J)^{(0)}|_{h=0} = y_0^J. \quad (2.5;0)$$

$q = 1$: from (2.3) and (2.4)

$$(g_i^J)^{(1)}|_{h=0} = \sum_j a_{ij} f^J|_{y=y_0}. \quad (2.5;1)$$

$q = 2$: because of (2.4) we shall need the first derivative of $f^J(g_j)$

$$(f^J(g_j))^{(1)} = \sum_K f_K^J(g_j) \cdot (g_j^K)^{(1)}, \quad (2.6;1)$$

where, as usual, f_K^J denotes $\partial f^J / \partial y^K$. Inserting formula (2.5;1) (with i, j, J replaced by j, k, K) into (2.6;1) we obtain with (2.4)

$$(g_i^J)^{(2)}|_{h=0} = 2 \sum_{j,k} a_{ij} a_{jk} \sum_K f_K^J f^K|_{y=y_0}. \quad (2.5;2)$$

$q = 3$: we differentiate (2.6;1) to obtain

$$(f^J(g_j))^{(2)} = \sum_{K,L} f_{KL}^J(g_j) \cdot (g_j^K)^{(1)} (g_j^L)^{(1)} + \sum_K f_K^J(g_j) (g_j^K)^{(2)}. \quad (2.6;2)$$

The derivatives $(g_j^K)^{(1)}$ and $(g_j^K)^{(2)}$ at $h = 0$ are already available in (2.5;1) and (2.5;2). So we have from (2.3) and (2.4)

$$\begin{aligned} (g_i^J)^{(3)}|_{h=0} &= 3 \sum_{j,k,l} a_{ij} a_{jk} a_{jl} \sum_{K,L} f_{KL}^J f^K f^L|_{y=y_0} \\ &\quad + 3 \cdot 2 \sum_{j,k,l} a_{ij} a_{jk} a_{kl} \sum_{K,L} f_K^J f_L^K f^L|_{y=y_0}. \end{aligned} \quad (2.5;3)$$

The same formula holds for $(y_1^J)^{(3)}|_{h=0}$ with a_{ij} replaced by b_j .

The Derivatives of the True Solution

The derivatives of the correct solution are obtained much more easily just by differentiating equation (2.2): first

$$(y^J)^{(1)} = f^J(y). \quad (2.7;1)$$

Differentiating (2.2) and inserting (2.2) again for the derivatives we get

$$(y^J)^{(2)} = \sum_K f_K^J(y) \cdot (y^K)^{(1)} = \sum_K f_K^J(y) f^K(y). \quad (2.7;2)$$

Differentiating (2.7;2) again we obtain

$$(y^J)^{(3)} = \sum_{K,L} f_{KL}^J(y) f^K(y) f^L(y) + \sum_{K,L} f_K^J(y) f_L^K(y) f^L(y). \quad (2.7;3)$$

Conditions for Order 3

For order 3, the derivatives (2.5;1-3), (with a_{ij} replaced by b_j) must be equal to the derivatives (2.7;1-3), and this for every differential equation. Thus, comparing the corresponding expressions, we obtain:

Theorem 2.1. *The RK method (2.3) (and thus (1.8)) is of order 3 iff*

$$\begin{aligned} \sum_j b_j &= 1, & 2 \sum_{j,k} b_j a_{jk} &= 1, \\ 3 \sum_{j,k,l} b_j a_{jk} a_{jl} &= 1, & 6 \sum_{j,k,l} b_j a_{jk} a_{kl} &= 1. \end{aligned} \quad (2.8)$$

□

Inserting $\sum_k a_{jk} = c_j$ from (1.9), we can simplify these expressions still further and obtain formulas (a)-(d) of (1.11).

Trees and Elementary Differentials

But without a more convenient notation, it would be difficult to find the corresponding expressions . . . This, however, can be at once effected by means of the analytical forms called trees . . .

(A. Cayley 1857)

The continuation of this process, although theoretically clear, soon leads to very complicated formulas. It is therefore advantageous to use a graphical representation: indeed, the indices j, k, l and J, K, L in the terms of (2.5;3) are linked

together as pairs of indices in a_{jk}, a_{jl}, \dots in exactly the same way as upper and lower indices in the expressions f_{KL}^J, f_K^J , namely

$$t_{31} = \begin{array}{c} l \quad k \\ \backslash \quad / \\ j \end{array} \quad \text{and} \quad t_{32} = \begin{array}{c} l \\ / \quad \backslash \\ j \quad k \end{array} \quad (2.9)$$

for the first and second term respectively. We call these objects *labelled trees*, because they are connected graphs (trees) whose vertices are labelled with summation indices. They can also be represented as *mappings*, e.g.,

$$l \mapsto j, \quad k \mapsto j \quad \text{and} \quad l \mapsto k, \quad k \mapsto j \quad (2.9')$$

for the above trees. This mapping indicates to which lower letter the corresponding vertices are attached.

Definition 2.2. Let A be an ordered chain of indices $A = \{j < k < l < m < \dots\}$ and denote by A_q the subset consisting of the first q indices. A (*rooted*) *labelled tree* of order q ($q \geq 1$) is a mapping (the son-father mapping)

$$t : A_q \setminus \{j\} \rightarrow A_q$$

such that $t(z) < z$ for all $z \in A_q \setminus \{j\}$. The set of all labelled trees of order q is denoted by LT_q . We call “ z ” the *son* of “ $t(z)$ ” and “ $t(z)$ ” the *father* of “ z ”. The vertex “ j ”, the forefather of the whole dynasty, is called the *root* of t . The order q of a labelled tree is equal to the number of its vertices and is usually denoted by $q = \varrho(t)$.

Definition 2.3. For a labelled tree $t \in LT_q$ we call

$$F^J(t)(y) = \sum_{K, L, \dots} f_{K, \dots}^J(y) f_{\dots}^K(y) f_{\dots}^L(y) \dots$$

the corresponding *elementary differential*. The summation is over $q-1$ indices K, L, \dots (which correspond to $A_q \setminus \{j\}$) and the summand is a product of q f 's, where the upper index runs through all vertices of t and the lower indices are the corresponding sons. We denote by $F(t)(y)$ the vector $(F^1(t)(y), \dots, F^n(t)(y))$.

If the set A_q is written as

$$A_q = \{j_1 < j_2 < \dots < j_q\}, \quad (2.10)$$

then we can write the definition of $F(t)$ as follows:

$$F^{J_1}(t) = \sum_{J_2, \dots, J_q} \prod_{i=1}^q f_{t^{-1}(J_i)}^{J_i}, \quad (2.11)$$

since the sons of an index are its inverse images under the map t .

Examples of elementary differentials are

$$\sum_{K,L} f_{KL}^J f^K f^L \quad \text{and} \quad \sum_{K,L} f_K^J f_L^K f^L$$

for the labelled trees t_{31} and t_{32} above. These expressions appear in formulas (2.5;3) and (2.7;3).

The three labelled trees

$$\begin{array}{ccc} \begin{array}{c} l \\ \diagdown \quad \diagup \\ m \quad j \quad k \end{array} & \begin{array}{c} m \\ \diagdown \quad \diagup \\ l \quad j \quad k \end{array} & \begin{array}{c} m \\ \diagdown \quad \diagup \\ k \quad j \quad l \end{array} \end{array} \quad (2.12)$$

all look topologically alike, moreover the corresponding elementary differentials

$$\sum_{K,L,M} f_{KM}^J f^M f_L^K f^L, \quad \sum_{K,L,M} f_{KL}^J f^L f_M^K f^M, \quad \sum_{K,L,M} f_{LK}^J f^K f_M^L f^M \quad (2.12')$$

are the same, because they just differ by an exchange of the summation indices. Thus we give

Definition 2.4. Two labelled trees t and u are *equivalent*, if they have the same order, say q , and if there exists a permutation $\sigma : A_q \rightarrow A_q$, such that $\sigma(j) = j$ and $t\sigma = \sigma u$ on $A_q \setminus \{j\}$.

This clearly defines an equivalence relation.

Definition 2.5. An equivalence class of q th order labelled trees is called a (*rooted*) *tree of order q* . The set of all trees of order q is denoted by T_q . The *order* of a tree is defined as the order of a representative and is again denoted by $\varrho(t)$. Furthermore we denote by $\alpha(t)$ (for $t \in T_q$) the number of elements in the equivalence class t ; i.e., the number of possible different monotonic labellings of t .

Geometrically, a tree is distinguished from a labelled tree by omitting the labels. Often it is advantageous to include \emptyset , the empty tree, as the only tree of order 0. The only tree of order 1 is denoted by τ . The number of trees of orders $1, 2, \dots, 10$ are given in Table 2.1. Representatives of all trees of order ≤ 5 are shown in Table 2.2.

Table 2.1. Number of trees up to order 10

q	1	2	3	4	5	6	7	8	9	10
card(T_q)	1	1	2	4	9	20	48	115	286	719

Table 2.2. Trees and elementary differentials up to order 5

q	t	graph	$\gamma(t)$	$\alpha(t)$	$F^J(t)(y)$	$\Phi_j(t)$
0	\emptyset	\emptyset	1	1	y^J	
1	τ	$\bullet j$	1	1	f^J	1
2	t_{21}		2	1	$\sum_K f_K^J f^K$	$\sum_k a_{jk}$
3	t_{31}		3	1	$\sum_{K,L} f_{KL}^J f^K f^L$	$\sum_{k,l} a_{jk} a_{jl}$
	t_{32}		6	1	$\sum_{K,L} f_K^J f_L^K f^L$	$\sum_{k,l} a_{jk} a_{kl}$
4	t_{41}		4	1	$\sum_{K,L,M} f_{KLM}^J f^K f^L f^M$	$\sum_{k,l,m} a_{jk} a_{jl} a_{jm}$
	t_{42}		8	3	$\sum_{K,L,M} f_{KM}^J f_L^K f^L f^M$	$\sum_{k,l,m} a_{jk} a_{kl} a_{jm}$
	t_{43}		12	1	$\sum_{K,L,M} f_K^J f_{LM}^K f^L f^M$	$\sum_{k,l,m} a_{jk} a_{kl} a_{km}$
	t_{44}		24	1	$\sum_{K,L,M} f_K^J f_L^K f_M^L f^M$	$\sum_{k,l,m} a_{jk} a_{kl} a_{lm}$
5	t_{51}		5	1	$\sum f_{KLMP}^J f^K f^L f^M f^P$	$\sum a_{jk} a_{jl} a_{jm} a_{jp}$
	t_{52}		10	6	$\sum f_{KMP}^J f_L^K f^L f^M f^P$	$\sum a_{jk} a_{kl} a_{jm} a_{jp}$
	t_{53}		15	4	$\sum f_{KLP}^J f_M^K f^L f^M f^P$	$\sum a_{jk} a_{kl} a_{km} a_{jp}$
	t_{54}		30	4	$\sum f_{KPP}^J f_L^K f_M^L f^M f^P$	$\sum a_{jk} a_{kl} a_{lm} a_{jp}$
	t_{55}		20	3	$\sum f_{KM}^J f_L^K f^L f_M^P f^P$	$\sum a_{jk} a_{kl} a_{jm} a_{mp}$
	t_{56}		20	1	$\sum f_K^J f_{LMP}^K f^L f^M f^P$	$\sum a_{jk} a_{kl} a_{km} a_{kp}$
	t_{57}		40	3	$\sum f_K^J f_{LP}^K f_M^L f^M f^P$	$\sum a_{jk} a_{kl} a_{lm} a_{kp}$
	t_{58}		60	1	$\sum f_K^J f_L^K f_{MP}^L f^M f^P$	$\sum a_{jk} a_{kl} a_{lm} a_{lp}$
	t_{59}		120	1	$\sum f_K^J f_L^K f_M^L f_P^M f^P$	$\sum a_{jk} a_{kl} a_{lm} a_{mp}$

The Taylor Expansion of the True Solution

We can now state the general result for the q th derivative of the true solution:

Theorem 2.6. *The exact solution of (2.2) satisfies*

$$(y)^{(q)}(x_0) = \sum_{t \in LT_q} F(t)(y_0) = \sum_{t \in T_q} \alpha(t) F(t)(y_0). \quad (2.7;q)$$

Proof. The theorem is true for $q = 1, 2, 3$ (see (2.7;1-3) above). For the computation of, say, the 4th derivative, we have to differentiate (2.7;3). This consists of two terms (corresponding to the two trees of (2.9)), each of which contains three factors f, \dots (corresponding to the three nodes of these trees). The differentiation of these by Leibniz' rule and insertion of (2.2) for the derivatives is geometrically just the addition of a new branch with a new summation letter to *each* vertex (Fig. 2.1).

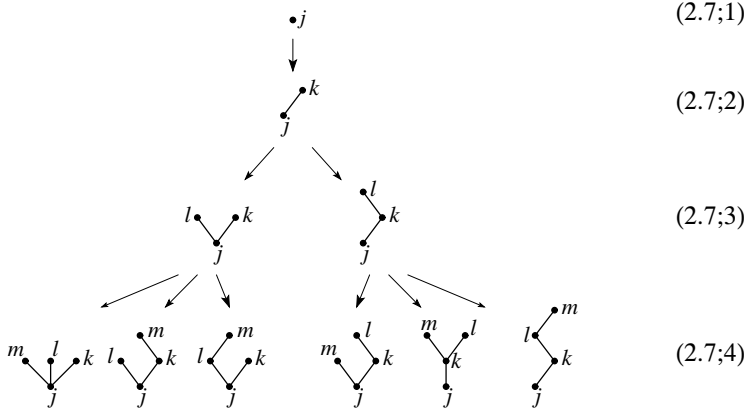


Fig. 2.1. Derivatives of exact solution

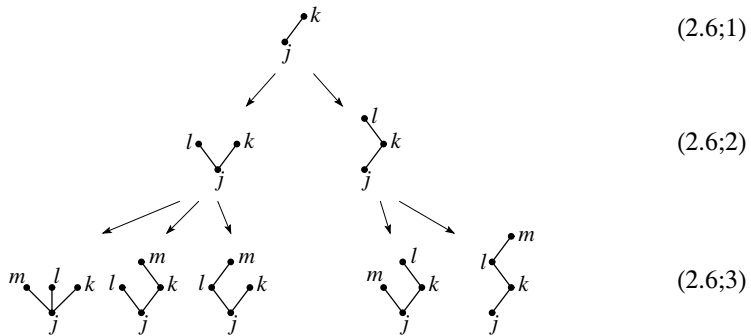
It is clear that by this process *all* labelled trees of order q appear for the q th derivative, each of them *exactly once*.

If we group together the terms with identical elementary differentials, we obtain the second expression of (2.7;q). □

Faà di Bruno's Formula

Our next goal will be the computation of the q th derivative of the numerical solution y_1 and of the g_j . For this, we have first to generalize the formulas (2.6;1) (the chain rule) and (2.6;2) for the q th derivative of the composition of two functions. We represent these two formulas graphically in Fig. 2.2.

Formula (2.6;2) consists of two terms; the first term contains three factors, the second contains only two. Here the node "l" is a "dummy" node, not really present in the formula, and just indicates that we have to take the second derivative. The derivation of (2.6;2) will thus lead to *five* terms which we write down for the convenience of the reader (but not for the convenience of the printer ...)

Fig. 2.2. Derivatives of $f^J(g)$

$$\begin{aligned}
 (f^J(g))^{(3)} &= \sum_{K,L,M} f_{KLM}^J(g) \cdot (g^K)^{(1)}(g^L)^{(1)}(g^M)^{(1)} \\
 &+ \sum_{K,L} f_{KL}^J(g) \cdot (g^K)^{(2)}(g^L)^{(1)} + \sum_{K,L} f_{KL}^J(g) \cdot (g^K)^{(1)}(g^L)^{(2)} \\
 &+ \sum_{K,M} f_{KM}^J(g) \cdot (g^K)^{(2)}(g^M)^{(1)} + \sum_K f_K^J(g) \cdot (g^K)^{(3)}.
 \end{aligned} \tag{2.6;3}$$

The corresponding trees are represented in the third line of Fig. 2.2. Each time we differentiate, we have to

- i) differentiate the first factor f_K^J ; i.e., we add a new branch to the root j ;
- ii) increase the derivative numbers of each of the g 's by 1; we represent this by lengthening the corresponding branch.

Each time we add a new label. *All trees which are obtained in this way are those "special" trees which have no ramifications except at the root.*

Definition 2.7. We denote by LS_q the set of *special labelled trees of order q* which have no ramifications except at the root.

Lemma 2.8 (Faà di Bruno's formula). *For $q \geq 1$ we have*

$$(f^J(g))^{(q-1)} = \sum_{u \in LS_q} \sum_{K_1, \dots, K_m} f_{K_1, \dots, K_m}^J(g) \cdot (g^{K_1})^{(\delta_1)} \dots (g^{K_m})^{(\delta_m)} \tag{2.6;q-1}$$

Here, for $u \in LS_q$, m is the number of branches leaving the root and $\delta_1, \dots, \delta_m$ are the numbers of nodes in each of these branches, such that $q = 1 + \delta_1 + \dots + \delta_m$. \square

Remark. The usual multinomial coefficients are absent here, as we use labelled trees.

The Derivatives of the Numerical Solution

It is difficult to keep a cool head when discussing the various derivatives . . .
(S. Gill 1956)

In order to generalize (2.5;1-3), we need the following definitions:

Definition 2.9. Let t be a labelled tree with root j ; we denote by

$$\Phi_j(t) = \sum_{k,l,\dots} a_{jk} a_{\dots} \dots$$

the sum over the $q-1$ remaining indices k, l, \dots (as in Definition 2.3). The summand is a product of $q-1$ a 's, where all fathers stand two by two with their sons as indices. If the set A_q is written as in (2.10), we have

$$\Phi_{j_1}(t) = \sum_{j_2, \dots, j_q} a_{t(j_2), j_2} \dots a_{t(j_q), j_q}. \quad (2.13)$$

Definition 2.10. For $t \in LT_q$ let $\gamma(t)$ be the product of $\varrho(t)$ and all orders of the trees which appear, if the roots, one after another, are removed from t . (See Fig. 2.3 or formula (2.17)).

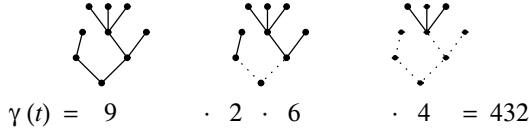


Fig. 2.3. Example for the definition of $\gamma(t)$

The above expressions are of course independent of the labellings, so $\Phi_j(t)$ as well as $\gamma(t)$ also make sense in T_q . Examples are given in Table 2.2.

Theorem 2.11. The derivatives of g_i satisfy

$$g_i^{(q)}|_{h=0} = \sum_{t \in LT_q} \gamma(t) \sum_j a_{ij} \Phi_j(t) F(t)(y_0). \quad (2.5;q)$$

The numerical solution y_1 of (2.3) satisfies

$$\begin{aligned} y_1^{(q)}|_{h=0} &= \sum_{t \in LT_q} \gamma(t) \sum_j b_j \Phi_j(t) F(t)(y_0) \\ &= \sum_{t \in T_q} \alpha(t) \gamma(t) \sum_j b_j \Phi_j(t) F(t)(y_0). \end{aligned} \quad (2.14)$$

Proof. Because of the similarity of y_1 and g_i (see (2.3)) we only have to prove the first equation. We do this by induction on q , in exactly the same way as we obtained (2.5;1-3): we first apply Leibniz' formula (2.4) to obtain

$$(g_i^J)^{(q)}|_{h=0} = q \sum_j a_{ij} (f^J(g_j))^{(q-1)}|_{y=y_0}. \quad (2.15)$$

Next we use Faà di Bruno's formula (Lemma 2.8). Finally we insert for the derivatives $(g_j^{K_s})^{(\delta_s)}$, which appear in (2.6;q-1) with $\delta_s < q$, the induction hypothesis (2.5;1) - (2.5;q-1) and rearrange the sums. This gives

$$\begin{aligned} (g_i^J)^{(q)}|_{h=0} &= q \sum_{u \in LS_q} \sum_{t_1 \in LT_{\delta_1}} \cdots \sum_{t_m \in LT_{\delta_m}} \gamma(t_1) \cdots \gamma(t_m) \cdot \\ &\quad \sum_j a_{ij} \sum_{k_1} a_{jk_1} \Phi_{k_1}(t_1) \cdots \sum_{k_m} a_{jk_m} \Phi_{k_m}(t_m) \cdot \\ &\quad \sum_{K_1, \dots, K_m} f_{K_1, \dots, K_m}^J(y_0) F^{K_1}(t_1)(y_0) \cdots F^{K_m}(t_m)(y_0). \end{aligned} \quad (2.16)$$

The main difficulty is now to understand that to each tuple

$$(u, t_1, \dots, t_m) \quad \text{with} \quad u \in LS_q, \quad t_s \in LT_{\delta_s}$$

there corresponds a labelled tree $t \in LT_q$ such that

$$\gamma(t) = q \cdot \gamma(t_1) \cdots \gamma(t_m) \quad (2.17)$$

$$F^J(t)(y) = \sum_{K_1, \dots, K_m} f_{K_1, \dots, K_m}^J(y) F^{K_1}(t_1)(y) \cdots F^{K_m}(t_m)(y) \quad (2.18)$$

$$\Phi_j(t) = \sum_{k_1, \dots, k_m} a_{jk_1} \cdots a_{jk_m} \Phi_{k_1}(t_1) \cdots \Phi_{k_m}(t_m). \quad (2.19)$$

This labelled tree t is obtained if the branches of u are replaced by the trees t_1, \dots, t_m and the corresponding labels are taken over in a natural way, i.e., in the same order (see Fig. 2.4 for some examples).

In this way, *all* trees $t \in LT_q$ appear exactly *once*. Thus (2.16) becomes (2.5;q) after inserting (2.17), (2.18) and (2.19). \square

The above construction of t can also be used for a recursive definition of trees. We first observe that the equivalence class of t (in Fig. 2.4) depends only on the equivalence classes of t_1, \dots, t_m .

Definition 2.12. We denote by

$$t = [t_1, \dots, t_m] \quad (2.20)$$

the tree, which leaves over the trees t_1, \dots, t_m when its root and the adjacent branches are chopped off (Fig. 2.5).

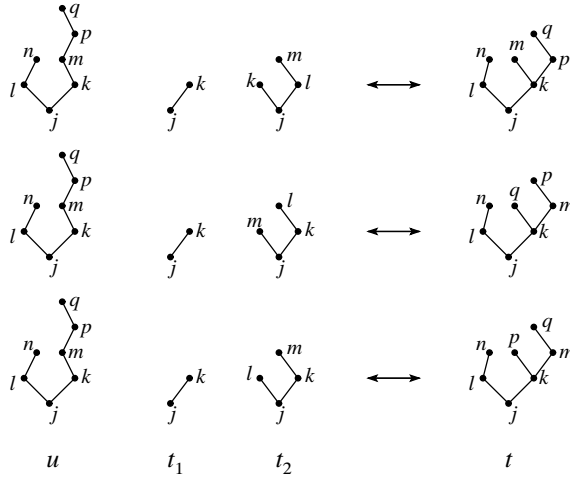


Fig. 2.4. Example for the bijection $(u, t_1, \dots, t_m) \leftrightarrow t$

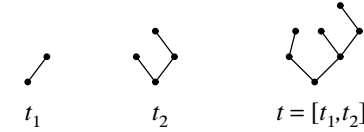


Fig. 2.5. Recursive definition of trees

With (2.20) all trees can be expressed in terms of τ ; e.g., $t_{21} = [\tau]$, $t_{31} = [\tau, \tau]$, $t_{32} = [[\tau]]$, \dots , etc.

The Order Conditions

Comparing Theorems 2.6 and 2.11 we now obtain:

Theorem 2.13. *A Runge-Kutta method (1.8) is of order p iff*

$$\sum_{j=1}^s b_j \Phi_j(t) = \frac{1}{\gamma(t)} \quad (2.21)$$

for all trees of order $\leq p$.

Proof. While the “if” part is clear from the preceding discussion, the “only if” part needs the fact that the elementary differentials for different trees are actually independent. See Exercises 3 and 4 below. \square

From Table 2.1 we then obtain the following number of order conditions (see Table 2.3). One can thus understand that the construction of higher order Runge Kutta formulas is not an easy task.

Table 2.3. Number of order conditions

order p	1	2	3	4	5	6	7	8	9	10
no. of conditions	1	2	4	8	17	37	85	200	486	1205

Example. For the tree t_{42} of Table 2.2 we have (using (1.9) for the second expression)

$$\sum_{j,k,l,m} b_j a_{jk} a_{jl} a_{km} = \sum_{j,k} b_j a_{jk} c_j c_k = \frac{1}{8},$$

which is (1.11;f). All remaining conditions of (1.11) correspond to the other trees of order ≤ 4 .

Exercises

1. Find all trees of order 6 and order 7.

Hint. Search for all representations of $p-1$ as a sum of positive integers, and then insert all known trees of lower order for each term in the sum. You may also use a computer for general p .

2. (A. Cayley 1857). Denote the number of trees of order q by a_q . Prove that

$$a_1 + a_2 x + a_3 x^2 + a_4 x^3 + \dots = (1-x)^{-a_1} (1-x^2)^{-a_2} (1-x^3)^{-a_3} \dots$$

Compare the result with Table 2.1.

3. Compute the elementary differentials of Table 2.2 for the case of the scalar non-autonomous equation (2.1), i.e., $f^1 = 1$, $f^2 = f(x, y)$. One imagines the complications met by the first authors (Kutta, Nyström, Huřa) in looking for higher order conditions. Observe also that in this case the expressions for t_{54} and t_{57} are the same, so that here Theorem 2.13 is sufficient, but not necessary for order 5.

Hint. For, say, t_{54} we have non-zero derivatives only if $K = L = 2$. Letting M and P run from 1 to 2 we then obtain

$$F^2(t) = (f_x + f f_y)(f_{yx} + f f_{yy}) f_y$$

(see also Butcher 1963a).

4. Show that for every $t \in T_q$ there is a system of differential equations such that $F^1(t)(y_0) = 1$ and $F^1(u)(y_0) = 0$ for all other trees u .

Hint. For t_{54} this system would be

$$y'_1 = y_2 y_5, \quad y'_2 = y_3, \quad y'_3 = y_4, \quad y'_4 = 1, \quad y'_5 = 1$$

with all initial values $= 0$. Understand this and the general formula

$$y'_{\text{father}} = \prod y_{\text{sons}}.$$

5. Kutta (1901) claimed that the scheme given in Table 2.4 is of order 5. Was he correct in his statement? Try to correct these values.

Result. The values for $a_{6j} (j = 1, \dots, 5)$ should read $(6, 36, 10, 8, 0)/75$; the correct values for b_j are $(23, 0, 125, 0, -81, 125)/192$ (Nyström 1925).

Table 2.4. A method of Kutta

0						
$\frac{1}{3}$	$\frac{1}{3}$					
$\frac{2}{5}$	$\frac{4}{25}$	$\frac{6}{25}$				
1	$\frac{1}{4}$	-3	$\frac{15}{4}$			
$\frac{2}{3}$	$\frac{6}{81}$	$\frac{90}{81}$	$-\frac{50}{81}$	$\frac{8}{81}$		
$\frac{4}{5}$	$\frac{7}{30}$	$\frac{18}{30}$	$-\frac{5}{30}$	$\frac{4}{30}$	0	
	$\frac{48}{192}$	0	$\frac{125}{192}$	0	$-\frac{81}{192}$	$\frac{100}{192}$

6. Verify $\sum_{\varrho(t)=p} \alpha(t) = (p-1)!$

7. Prove that a Runge-Kutta method, when applied to a linear system

$$y' = A(x)y + g(x), \quad (2.22)$$

is of order p iff

$$\sum_j b_j c_j^{q-1} = 1/q \quad \text{for } q \leq p$$

$$\sum_{j,k} b_j c_j^{q-1} a_{jk} c_k^{r-1} = 1/((q+r)r) \quad \text{for } q+r \leq p$$

$\sum_{j,k,l} b_j c_j^{q-1} a_{jk} c_k^{r-1} a_{kl} c_l^{s-1} = 1/((q+r+s)(r+s)s) \quad \text{for } q+r+s \leq p$
 ... etc (write (2.22) in autonomous form and investigate which elementary differentials vanish identically; see also Crouzeix 1975).

II.3 Error Estimation and Convergence for RK Methods

Es fehlt indessen noch der Beweis dass diese Näherungs-Verfahren convergent sind oder, was practisch wichtiger ist, es fehlt ein Kriterium, um zu ermitteln, wie klein die Schritte gemacht werden müssen, um eine vorgeschriebene Genauigkeit zu erreichen. (Runge 1905)

Since the work of Lagrange (1797) and, above all, of Cauchy, a numerically established result should be accompanied by a reliable error estimation (“... l’erreur commise sera inférieure à ...”). Lagrange gave the well-known error bounds for the Taylor polynomials and Cauchy derived bounds for the error of the Euler polygons (see Section I.7). A couple of years after the first success of the Runge-Kutta methods, Runge (1905) also required error estimates for these methods.

Rigorous Error Bounds

Runge’s device for obtaining bounds for the error in one step (“local error”) can be described in a few lines (free translation):

“For a method of order p consider the local error

$$e(h) = y(x_0 + h) - y_1 \quad (3.1)$$

and use its Taylor expansion

$$e(h) = e(0) + he'(0) + \dots + \frac{h^p}{p!}e^{(p)}(\theta h) \quad (3.2)$$

with $0 < \theta < 1$ and $e(0) = e'(0) = \dots = e^{(p)}(0) = 0$. Now compute explicitly $e^{(p)}(h)$, which will be of the form

$$e^{(p)}(h) = E_1(h) + hE_2(h), \quad (3.3)$$

where $E_1(h)$ and $E_2(h)$ contain partial derivatives of f up to order $p-1$ and p respectively. Further, because of $e^{(p)}(0) = 0$, we have $E_1(0) = 0$. Thus, if all partial derivatives of f up to order p are bounded, we have $E_1(h) = \mathcal{O}(h)$ and $E_2(h) = \mathcal{O}(1)$. So there is a constant C such that $|e^{(p)}(h)| \leq Ch$ and

$$|e(h)| \leq C \frac{h^{p+1}}{p!}. \quad (3.4)$$

A slightly different approach is adopted by Bieberbach (1923, 1. Abschn., Kap. II, §7), explained in more detail in Bieberbach (1951): we write

$$e(h) = y(x_0 + h) - y_1 = y(x_0 + h) - y_0 - h \sum_{i=1}^s b_i k_i \quad (3.5)$$

and use the Taylor expansions

$$\begin{aligned} y(x_0 + h) &= y_0 + y'(x_0)h + y''(x_0)\frac{h^2}{2!} + \dots + y^{(p+1)}(x_0 + \theta h)\frac{h^{p+1}}{(p+1)!} \\ k_i(h) &= k_i(0) + k'_i(0)h + \dots + k_i^{(p)}(\theta_i h)\frac{h^p}{p!}, \end{aligned} \quad (3.6)$$

where, for vector valued functions, the formula is valid componentwise with possibly different θ 's. The first terms in the h expansion of (3.5) vanish because of the order conditions. Thus we obtain

Theorem 3.1. *If the Runge-Kutta method (1.8) is of order p and if all partial derivatives of $f(x, y)$ up to order p exist (and are continuous), then the local error of (1.8) admits the rigorous bound*

$$\begin{aligned} \|y(x_0 + h) - y_1\| &\leq h^{p+1} \left(\frac{1}{(p+1)!} \max_{t \in [0,1]} \|y^{(p+1)}(x_0 + th)\| \right. \\ &\quad \left. + \frac{1}{p!} \sum_{i=1}^s |b_i| \max_{t \in [0,1]} \|k_i^{(p)}(th)\| \right) \end{aligned} \quad (3.7)$$

and hence also

$$\|y(x_0 + h) - y_1\| \leq Ch^{p+1}. \quad (3.8)$$

□

Let us demonstrate this result on Runge's first method (1.4), which is of order $p = 2$, applied to a scalar differential equation. Differentiating (1.1) we obtain

$$y^{(3)}(x) = \left(f_{xx} + 2f_{xy}f + f_{yy}f^2 + f_y(f_x + f_yf) \right) (x, y(x)) \quad (3.9)$$

while the second derivative of $k_2(h) = f(x_0 + \frac{h}{2}, y_0 + \frac{h}{2}f_0)$ is given by

$$k_2^{(2)}(h) = \frac{1}{4} \left(f_{xx}(x_0 + \frac{h}{2}, y_0 + \frac{h}{2}f_0) + 2f_{xy}(\dots)f_0 + f_{yy}(\dots)f_0^2 \right) \quad (3.10)$$

(f_0 stands for $f(x_0, y_0)$). Under the assumptions of Theorem 3.1 we see that the expressions (3.9) and (3.10) are bounded by a constant independent of h , which gives (3.8).

The Principal Error Term

For higher order methods rigorous error bounds, like (3.7), become very unpractical. It is therefore much more realistic to consider the first non-zero term in the Taylor expansion of the error. For autonomous systems of equations (2.2), the error term is best obtained by subtracting the Taylor series and using (2.14) and (2.7;q).

Theorem 3.2. *If the Runge-Kutta method is of order p and if f is $(p+1)$ -times continuously differentiable, we have*

$$y^J(x_0 + h) - y_1^J = \frac{h^{p+1}}{(p+1)!} \sum_{t \in T_{p+1}} \alpha(t) e(t) F^J(t)(y_0) + \mathcal{O}(h^{p+2}) \quad (3.11)$$

where

$$e(t) = 1 - \gamma(t) \sum_{j=1}^s b_j \Phi_j(t). \quad (3.12)$$

□

$\gamma(t)$ and $\Phi_j(t)$ are given in Definitions 2.9 and 2.10; see also formulas (2.17) and (2.19). The expressions $e(t)$ are called the *error coefficients*.

Example 3.3. For the two-parameter family of 4th order RK methods (1.17) the error coefficients for the 9 trees of Table 2.2 are ($c_2 = u$, $c_3 = v$):

$$\begin{aligned} e(t_{51}) &= -\frac{1}{4} + \frac{5}{12}(u+v) - \frac{5}{6}uv, & e(t_{52}) &= \frac{5}{12}v - \frac{1}{4}, \\ e(t_{53}) &= \frac{5}{8}u - \frac{1}{4}, & e(t_{54}) &= -\frac{1}{4}, \\ e(t_{55}) &= 1 - \frac{5(b_4 + b_3(3-4v)^2)}{144b_3b_4(1-v)^2}, & & \\ e(t_{56}) &= -4e(t_{51}), & e(t_{57}) &= -4e(t_{52}), \\ e(t_{58}) &= -4e(t_{53}), & e(t_{59}) &= -4e(t_{54}). \end{aligned} \quad (3.13)$$

Proof. The last four formulas follow from (1.12). $e(t_{59})$ is trivial, $e(t_{58})$ and $e(t_{57})$ follow from (1.11h). Further

$$e(t_{51}) = 5 \int_0^1 t(t-1)(t-u)(t-v) dt$$

expresses the quadrature error. For $e(t_{55})$ one best introduces $c'_i = \sum_j a_{ij}c_j$ such that $e(t_{55}) = 1 - 20 \sum_i b_i c'_i c'_i$. Then from (1.1d,f) one obtains

$$c'_1 = c'_2 = 0, \quad b_3 c'_3 = \frac{1}{24(1-v)}, \quad b_4 c'_4 = \frac{3-4v}{24(1-v)}. \quad \square$$

For the classical 4th order method (Table 1.2a) these error coefficients are given by Kutta (1901), p. 448 (see also Lotkin 1951) as follows

$$\left(-\frac{1}{24}, -\frac{1}{24}, \frac{1}{16}, -\frac{1}{4}, -\frac{2}{3}, \frac{1}{6}, \frac{1}{6}, -\frac{1}{4}, 1\right)$$

Kutta remarked that for the second method (Table 1.2b) (“Als besser noch erweist sich . . .”) the error coefficients become

$$\left(-\frac{1}{54}, \frac{1}{36}, -\frac{1}{24}, -\frac{1}{4}, -\frac{1}{9}, \frac{2}{27}, -\frac{1}{9}, \frac{1}{6}, 1\right)$$

which, with the exception of the 4th and 9th term, are all smaller than for the above method. A tedious calculation was undertaken by Ralston (1962) (and by many others) to determine optimal coefficients of (1.17). For solutions which minimize the constants (3.13), see Exercise 3 below.

Estimation of the Global Error

Das war auch eine aufregende Zeit . . . (P. Henrici 1983)

The global error is the error of the computed solution after *several* steps. Suppose that we have a one-step method which, given an initial value (x_0, y_0) and a step size h , computes a numerical solution y_1 approximating $y(x_0 + h)$. We shall denote this process by Henrici’s notation

$$y_1 = y_0 + h\Phi(x_0, y_0, h) \quad (3.14)$$

and call Φ the *increment function* of the method.

The numerical solution for a point $X > x_0$ is then obtained by a step-by-step procedure

$$y_{i+1} = y_i + h_i \Phi(x_i, y_i, h_i), \quad h_i = x_{i+1} - x_i, \quad x_N = X \quad (3.15)$$

and our task is to estimate the *global error*

$$E = y(X) - y_N. \quad (3.16)$$

This estimate is found in a simple way, very similar to Cauchy’s convergence proof for Theorem 7.3 of Chapter I: *the local errors are transported to the final point x_N and then added up*. This “error transport” can be done in two different ways:

a) either along the exact solution curves (see Fig. 3.1); this method can yield sharp results when sharp estimates of error propagation for the exact solutions are known, e.g., from Theorem 10.6 of Chapter I based on the logarithmic norm $\mu(\partial f / \partial y)$.

b) or along $N - i$ steps of the numerical method (see Fig. 3.2); this is the method used in the proofs of Cauchy (1824) and Runge (1905), it generalizes easily to multistep methods (see Chapter III) and will be an important tool for the existence of asymptotic expansions (see II.8).

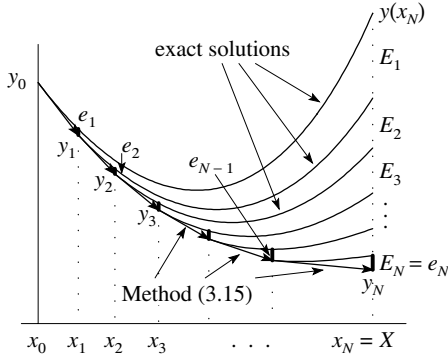


Fig. 3.1. Global error estimation, method (a)

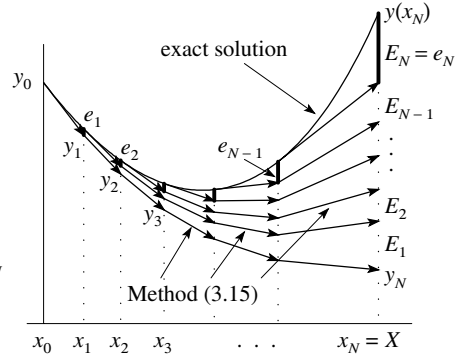


Fig. 3.2. Global error estimation, method (b)

In both cases we first estimate the local errors e_i with the help of Theorem 3.1 to obtain

$$\|e_i\| \leq C \cdot h_{i-1}^{p+1}. \quad (3.17)$$

Warning. The e_i of Fig. 3.1 and Fig. 3.2, for $i \neq 1$, are *not* the same, but they allow similar estimates.

We then estimate the transported errors E_i : for method (a) we use the known results from Chapter I, especially Theorem I.10.6, Theorem I.10.2, or formula (I.7.17). The result is

Theorem 3.4. Let U be a neighbourhood of $\{(x, y(x)) | x_0 \leq x \leq X\}$ where $y(x)$ is the exact solution of (1.1). Suppose that in U

$$\left\| \frac{\partial f}{\partial y} \right\| \leq L \quad \text{or} \quad \mu \left(\frac{\partial f}{\partial y} \right) \leq L, \quad (3.18)$$

and that the local error estimates (3.17) are valid in U . Then the global error (3.16) can be estimated by

$$\|E\| \leq h^p \frac{C'}{L} \left(\exp(L(X - x_0)) - 1 \right) \quad (3.19)$$

where $h = \max h_i$,

$$C' = \begin{cases} C & L \geq 0 \\ C \exp(-Lh) & L < 0, \end{cases}$$

and h is small enough for the numerical solution to remain in U .

Remark. For $L \rightarrow 0$ the estimate (3.19) tends to $h^p C (x_N - x_0)$.

Proof. From Theorem I.10.2 (with $\varepsilon = 0$) or Theorem I.10.6 (with $\delta = 0$) we obtain

$$\|E_i\| \leq \exp(L(x_N - x_i)) \|e_i\|. \quad (3.20)$$

We then insert this together with (3.17) into

$$\|E\| \leq \sum_{i=1}^N \|E_i\|.$$

Using $h_{i-1}^{p+1} \leq h^p \cdot h_{i-1}$ this leads to

$$\|E\| \leq h^p C \left(h_0 \exp(L(x_N - x_1)) + h_1 \exp(L(x_N - x_2)) + \dots \right).$$

The expression in large brackets can be bounded by

$$\int_{x_0}^{x_N} \exp(L(x_N - x)) dx \quad \text{for} \quad L \geq 0 \quad (3.21)$$

$$\int_{x_0}^{x_N} \exp(L(x_N - h - x)) dx \quad \text{for} \quad L < 0 \quad (3.22)$$

(see Fig. 3.3). This gives (3.19). \square

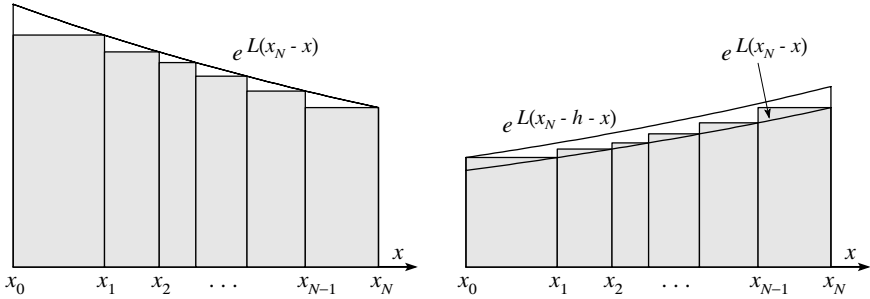


Fig. 3.3. Estimation of Riemann sums

For the second method (b) we need an estimate for $\|z_{i+1} - y_{i+1}\|$ in terms of $\|z_i - y_i\|$, where, besides (3.15),

$$z_{i+1} = z_i + h_i \Phi(x_i, z_i, h_i)$$

is a second pair of numerical solutions. For RK-methods z_{i+1} is defined by

$$\begin{aligned} \ell_1 &= f(x_i, z_i), \\ \ell_2 &= f(x_i + c_2 h_i, z_i + h_i a_{21} \ell_1), \quad \text{etc.} \end{aligned}$$

We now subtract formulas (1.8) from this and obtain

$$\begin{aligned}\|\ell_1 - k_1\| &\leq L\|z_i - y_i\|, \\ \|\ell_2 - k_2\| &\leq L(1 + |a_{21}|h_iL)\|z_i - y_i\|, \quad \text{etc.}\end{aligned}$$

This leads to the following

Lemma 3.5. *Let L be a Lipschitz constant for f and let $h_i \leq h$. Then the increment function Φ of method (1.8) satisfies*

$$\|\Phi(x_i, z_i, h_i) - \Phi(x_i, y_i, h_i)\| \leq \Lambda\|z_i - y_i\| \quad (3.23)$$

where

$$\Lambda = L\left(\sum_i |b_i| + hL \sum_{i,j} |b_i a_{ij}| + h^2 L^2 \sum_{i,j,k} |b_i a_{ij} a_{jk}| + \dots\right). \quad (3.24)$$

□

From (3.23) we obtain

$$\|z_{i+1} - y_{i+1}\| \leq (1 + h_i \Lambda)\|z_i - y_i\| \leq \exp(h_i \Lambda)\|z_i - y_i\| \quad (3.25)$$

and for the errors in Fig. 3.2,

$$\|E_i\| \leq \exp(\Lambda(x_N - x_i))\|e_i\| \quad (3.26)$$

instead of (3.20). The same proof as for Theorem 3.4 now gives us

Theorem 3.6. *Suppose that the local error satisfies, for initial values on the exact solution,*

$$\|y(x+h) - y(x) - h\Phi(x, y(x), h)\| \leq Ch^{p+1}, \quad (3.27)$$

and suppose that in a neighbourhood of the solution the increment function Φ satisfies

$$\|\Phi(x, z, h) - \Phi(x, y, h)\| \leq \Lambda\|z - y\|. \quad (3.28)$$

Then the global error (3.16) can be estimated by

$$\|E\| \leq h^p \frac{C}{\Lambda} \left(\exp(\Lambda(x_N - x_0)) - 1 \right) \quad (3.29)$$

where $h = \max h_i$.

□

Exercises

1. (Runge 1905). Show that for explicit Runge Kutta methods with $b_i \geq 0$, $a_{ij} \geq 0$ (all i, j) of order s the Lipschitz constant Λ for Φ satisfies

$$1 + h\Lambda < \exp(hL)$$

and that (3.29) is valid with Λ replaced by L .

2. Show that $e(t_{55})$ of (3.13) becomes

$$e(t_{55}) = 1 - 5 \frac{(4v^2 - 15v + 9) - u(6v^2 - 42v + 27) - u^2(26v - 18)}{12(1 - 2u)(6uv - 4(u + v) + 3)}$$

after inserting (1.17).

3. Determine u and v in (1.17) such that in (3.13)

$$\begin{array}{ll} \text{a) } \max_{i=5,6,7,8} |e(t_{5i})| = \min & \text{b) } \sum_{i=1}^9 |e(t_{5i})| = \min \\ \text{c) } \max_{i=5,6,7,8} \alpha(t_{5i}) |e(t_{5i})| = \min & \text{d) } \sum_{i=1}^9 \alpha(t_{5i}) |e(t_{5i})| = \min \end{array}$$

Results.

$$\begin{array}{lll} \text{a) } u = 0.3587, & v = 0.6346, & \min = 0.1033; \\ \text{b) } u = 0.3995, & v = 0.6, & \min = 1.55; \\ \text{c) } u = 0.3501, & v = 0.5839, & \min = 0.1248; \\ \text{d) } u = 0.3716, & v = 0.6, & \min = 2.53. \end{array}$$

Such optimal formulas were first studied by Ralston (1962), Hull & Johnston (1964), and Hull (1967).

4. Apply an explicit Runge-Kutta method to the problem $y' = f(x, y)$, $y(0) = 0$, where

$$f(x, y) = \begin{cases} \frac{\lambda}{x} y + g(x) & \text{if } x > 0 \\ (1 - \lambda)^{-1} g(0) & \text{if } x = 0, \end{cases}$$

$\lambda \leq 0$ and $g(x)$ is sufficiently differentiable (see Exercise 10 of Section I.5).

- a) Show that the error after the first step is given by

$$y(h) - y_1 = C_2 h^2 g'(0) + \mathcal{O}(h^3)$$

where C_2 is a constant depending on λ and on the coefficients of the method. Also for high order methods we have in general $C_2 \neq 0$.

- b) Compute C_2 for the classical 4th order method (Table 1.2).

II.4 Practical Error Estimation and Step Size Selection

Ich glaube indessen, dass ein practischer Rechner sich meistens mit der geringeren Sicherheit begnügen wird, die er aus der Uebereinstimmung seiner Resultate für grössere und kleinere Schritte gewinnt. (C. Runge 1895)

Even the simplified error estimates of Section II.3, which are content with the leading error term, are of little practical interest, because they require the computation and majorization of several partial derivatives of high orders. But the main advantage of Runge-Kutta methods, compared with Taylor series, is precisely that the computation of derivatives should be no longer necessary. However, since practical error estimates are necessary (on the one hand to ensure that the step sizes h_i are chosen sufficiently small to yield the required precision of the computed results, and on the other hand to ensure that the step sizes are sufficiently large to avoid unnecessary computational work), we shall now discuss alternative methods for error estimates.

The oldest device, used by Runge in his numerical examples, is to repeat the computations with *halved* step sizes and to compare the results: those digits which haven't changed are assumed to be correct (“... woraus ich schliessen zu dürfen glaube ...”).

Richardson Extrapolation

... its usefulness for practical computations can hardly be overestimated. (G. Birkhoff & G.C. Rota)

The idea of Richardson, announced in his classical paper Richardson (1910) which treats mainly partial differential equations, and explained in full detail in Richardson (1927), is to use more carefully the known behaviour of the error as a function of h .

Suppose that, with a given initial value (x_0, y_0) and step size h , we compute *two* steps, using a fixed Runge-Kutta method of order p , and obtain the numerical results y_1 and y_2 . We then compute, starting from (x_0, y_0) , *one big step* with step size $2h$ to obtain the solution w . The error of y_1 is known to be (Theorem 3.2)

$$e_1 = y(x_0 + h) - y_1 = C \cdot h^{p+1} + \mathcal{O}(h^{p+2}) \quad (4.1)$$

where C contains the error coefficients of the method and the elementary differentials $F^J(t)(y_0)$ of order $p+1$. The error of y_2 is composed of two parts: the

transported error of the first step, which is

$$\left(I + h \frac{\partial f}{\partial y} + \mathcal{O}(h^2)\right)e_1,$$

and the local error of the second step, which is the same as (4.1), but with the elementary differentials evaluated at $y_1 = y_0 + \mathcal{O}(h)$. Thus we obtain

$$\begin{aligned} e_2 = y(x_0 + 2h) - y_2 &= (I + \mathcal{O}(h))Ch^{p+1} + (C + \mathcal{O}(h))h^{p+1} + \mathcal{O}(h^{p+2}) \\ &= 2Ch^{p+1} + \mathcal{O}(h^{p+2}). \end{aligned} \quad (4.2)$$

Similarly to (4.1), we have for the big step

$$y(x_0 + 2h) - w = C(2h)^{p+1} + \mathcal{O}(h^{p+2}). \quad (4.3)$$

Neglecting the terms $\mathcal{O}(h^{p+2})$, formulas (4.2) and (4.3) allow us to eliminate the unknown constant C and to “extrapolate” a better value \hat{y}_2 for $y(x_0 + 2h)$, for which we obtain:

Theorem 4.1. *Suppose that y_2 is the numerical result of two steps with step size h of a Runge-Kutta method of order p , and w is the result of one big step with step size $2h$. Then the error of y_2 can be extrapolated as*

$$y(x_0 + 2h) - y_2 = \frac{y_2 - w}{2^p - 1} + \mathcal{O}(h^{p+2}) \quad (4.4)$$

and

$$\hat{y}_2 = y_2 + \frac{y_2 - w}{2^p - 1} \quad (4.5)$$

is an approximation of order $p + 1$ to $y(x_0 + 2h)$. □

Formula (4.4) is a very simple device to estimate the error of y_2 and formula (4.5) allows one to increase the precision by one additional order (“... The better theory of the following sections is complicated, and tends thereby to suggest that the practice may also be complicated; whereas it is really simple.” Richardson).

Embedded Runge-Kutta Formulas

Scruton is right in his criticism of Merson’s process, although Merson did not claim as much for his process as some people expect. (R. England 1969)

The idea is, rather than using Richardson extrapolation, to construct Runge-Kutta formulas which themselves contain, besides the numerical approximation y_1 , a second approximation \hat{y}_1 . The difference then yields an estimate of the local error for the less precise result and can be used for step size control (see below). Since

it is at our disposal at every step, this gives more flexibility to the code and makes step rejections less expensive.

We consider two Runge-Kutta methods (one for y_1 and one for \hat{y}_1) such that both use the *same* function values. We thus have to find a scheme of coefficients (see (1.8')),

$$\begin{array}{c|cccc}
 0 & & & & \\
 c_2 & a_{21} & & & \\
 c_3 & a_{32} & a_{32} & & \\
 \vdots & \vdots & \ddots & & \\
 c_s & a_{s1} & a_{s2} & \dots & a_{s,s-1} \\
 \hline
 & b_1 & b_2 & \dots & b_{s-1} & b_s \\
 & \hat{b}_1 & \hat{b}_2 & \dots & \hat{b}_{s-1} & \hat{b}_s
 \end{array} \tag{4.6}$$

such that

$$y_1 = y_0 + h(b_1 k_1 + \dots + b_s k_s) \tag{4.7}$$

is of order p , and

$$\hat{y}_1 = y_0 + h(\hat{b}_1 k_1 + \dots + \hat{b}_s k_s) \tag{4.7'}$$

is of order \hat{p} (usually $\hat{p} = p - 1$ or $\hat{p} = p + 1$). The approximation y_1 is used to continue the integration.

From Theorem 2.13, we have to satisfy the conditions

$$\sum_{j=1}^s b_j \Phi_j(t) = \frac{1}{\gamma(t)} \quad \text{for all trees of order } \leq p, \tag{4.8}$$

$$\sum_{j=1}^s \hat{b}_j \Phi_j(t) = \frac{1}{\gamma(t)} \quad \text{for all trees of order } \leq \hat{p}. \tag{4.8'}$$

The first methods of this type were proposed by Merson (1957), Ceschino (1962), and Zonneveld (1963). Those of Merson and Zonneveld are given in Tables 4.1 and 4.2. Here, “name $p(\hat{p})$ ” means that the order of y_1 is p and the order of the error estimator \hat{y}_1 is \hat{p} . Merson’s \hat{y}_1 is of order 5 only for *linear* equations with constant coefficients; for nonlinear problems it is of order 3. This method works quite well and has been used very often, especially by NAG users. Further embedded methods were then derived by Sarafyan (1966), England (1969), and Fehlberg (1964, 1968, 1969). Let us start with the construction of some low order embedded methods.

Methods of order 3(2). It is a simple task to construct embedded formulas of order 3(2) with $s = 3$ stages. Just take a 3-stage method of order 3 (Exercise II.1.4) and put $\hat{b}_3 = 0$, $\hat{b}_2 = 1/2c_2$, $\hat{b}_1 = 1 - 1/2c_2$.

Table 4.1. Merson 4(“5”)

0					
$\frac{1}{3}$	$\frac{1}{3}$				
$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{6}$			
$\frac{1}{2}$	$\frac{1}{8}$	0	$\frac{3}{8}$		
1	$\frac{1}{2}$	0	$-\frac{3}{2}$	2	
y_1	$\frac{1}{6}$	0	0	$\frac{2}{3}$	$\frac{1}{6}$
\hat{y}_1	$\frac{1}{10}$	0	$\frac{3}{10}$	$\frac{2}{5}$	$\frac{1}{5}$

Table 4.2. Zonneveld 4(3)

0					
$\frac{1}{2}$	$\frac{1}{2}$				
$\frac{1}{2}$	0	$\frac{1}{2}$			
1	0	0	1		
$\frac{3}{4}$	$\frac{5}{32}$	$\frac{7}{32}$	$\frac{13}{32}$	$-\frac{1}{32}$	
y_1	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$	
\hat{y}_1	$-\frac{1}{2}$	$\frac{7}{3}$	$\frac{7}{3}$	$\frac{13}{6}$	$-\frac{16}{3}$

Methods of order 4(3). With $s = 4$ it is impossible to find a pair of order 4(3) (see Exercise 2). The idea is to add y_1 as 5th stage of the process (i.e., $a_{5i} = b_i$ for $i = 1, \dots, 4$) and to search for a third order method which uses all five function values. Whenever the step is accepted this represents no extra work, because $f(x_0 + h, y_1)$ has to be computed anyway for the following step. This idea is called FSAL (First Same As Last). Then the order conditions (4.8') with $\hat{p} = 3$ represent 4 linear equations for the five unknowns $\hat{b}_1, \dots, \hat{b}_5$. One can arbitrarily fix $\hat{b}_5 \neq 0$ and solve the system for the remaining parameters. With \hat{b}_5 chosen such that $\hat{b}_4 = 0$ the result is

$$\begin{aligned} \hat{b}_1 &= 2b_1 - 1/6, & \hat{b}_2 &= 2(1 - c_2)b_2, \\ \hat{b}_3 &= 2(1 - c_3)b_3, & \hat{b}_4 &= 0, & \hat{b}_5 &= 1/6. \end{aligned} \quad (4.9)$$

Automatic Step Size Control

D'ordinaire, on se contente de multiplier ou de diviser par 2 la valeur du pas ... (Ceschino 1961)

We now want to write a code which automatically adjusts the step size in order to achieve a prescribed tolerance of the local error.

Whenever a starting step size h has been chosen, the program computes two approximations to the solution, y_1 and \hat{y}_1 . Then an estimate of the error for the less precise result is $y_1 - \hat{y}_1$. We want this error to satisfy componentwise

$$|y_{1i} - \hat{y}_{1i}| \leq sc_i, \quad sc_i = Atol_i + \max(|y_{0i}|, |y_{1i}|) \cdot Rtol_i \quad (4.10)$$

where $Atol_i$ and $Rtol_i$ are the desired tolerances prescribed by the user (relative errors are considered for $Atol_i = 0$, absolute errors for $Rtol_i = 0$; usually both

tolerances are different from zero; they may depend on the component of the solution). As a measure of the error we take

$$err = \sqrt{\frac{1}{n} \sum_{i=1}^n \left(\frac{y_{1i} - \hat{y}_{1i}}{sc_i} \right)^2}; \quad (4.11)$$

other norms, such as the max norm, are also of frequent use. Then err is compared to 1 in order to find an optimal step size. From the error behaviour $err \approx C \cdot h^{q+1}$ and from $1 \approx C \cdot h_{opt}^{q+1}$ (where $q = \min(p, \hat{p})$) the optimal step size is obtained as (“... le procédé connu”, Ceschino 1961)

$$h_{opt} = h \cdot (1/err)^{1/(q+1)}. \quad (4.12)$$

Some care is now necessary for a good code: we multiply (4.12) by a safety factor fac , usually $fac = 0.8, 0.9, (0.25)^{1/(q+1)}$, or $(0.38)^{1/(q+1)}$, so that the error will be acceptable the next time with high probability. Further, h is not allowed to increase nor to decrease too fast. For example, we may put

$$h_{new} = h \cdot \min(facmax, \max(facmin, fac \cdot (1/err)^{1/(q+1)})) \quad (4.13)$$

for the new step size. Then, if $err \leq 1$, the computed step is *accepted* and the solution is advanced with y_1 and a new step is tried with h_{new} as step size. Else, the step is *rejected* and the computations are repeated with the new step size h_{new} . The maximal step size increase $facmax$, usually chosen between 1.5 and 5, prevents the code from too large step increases and contributes to its safety. It is clear that, when chosen too small, it may also unnecessarily increase the computational work. It is also advisable to put $facmax = 1$ in the steps right after a step-rejection (Shampine & Watts 1979).

Whenever y_1 is of lower order than \hat{y}_1 , then the difference $y_1 - \hat{y}_1$ is (at least asymptotically) an estimate of the local error and the above algorithm keeps this estimate below the given tolerance. But isn't it more natural to continue the integration with the higher order approximation? Then the concept of “error estimation” is abandoned and the difference $y_1 - \hat{y}_1$ is only used for the purpose of step size selection. This is justified by the fact that, due to unknown stability and instability properties of the differential system, the local errors have in general very little in common with the global errors. The procedure of continuing the integration with the higher order result is called “local extrapolation”.

A modification of the above procedure (PI step size control), which is particularly interesting when applied to mildly stiff problems, is described in Section IV.2 (Volume II).

Starting Step Size

If anything has been made foolproof, a better fool will be developed.
(Heard from Dr. Pirkle, Baden)

For many years, the starting step size had to be supplied to a code. Users were assumed to have a rough idea of a good step size from mathematical knowledge or previous experience. Anyhow, a bad starting choice for h was quickly repaired by the step size control. Nevertheless, when this happens too often and when the choices are too bad, much computing time can be wasted. Therefore, several people (e.g., Watts 1983, Hindmarsh 1980) developed ideas to let the computer do this choice. We take up an idea of Gladwell, Shampine & Brankin (1987) which is based on the hypothesis that

$$\text{local error} \approx Ch^{p+1}y^{(p+1)}(x_0).$$

Since $y^{(p+1)}(x_0)$ is unknown we shall replace it by approximations of the first and second derivative of the solution. The resulting algorithm is the following one:

- a) Do one function evaluation $f(x_0, y_0)$ at the initial point. It is in any case needed for the first RK step. Then put $d_0 = \|y_0\|$ and $d_1 = \|f(x_0, y_0)\|$, where the norm is that of (4.11) with $sc_i = Atol_i + |y_{0i}| \cdot Rtol_i$.
- b) As a first guess for the step size let

$$h_0 = 0.01 \cdot (d_0/d_1)$$

so that the increment of an explicit Euler step is small compared to the size of the initial value. If either d_0 or d_1 is smaller than 10^{-5} we put $h_0 = 10^{-6}$.

- c) Perform one explicit Euler step, $y_1 = y_0 + h_0 f(x_0, y_0)$, and compute $f(x_0 + h_0, y_1)$.
- d) Compute $d_2 = \|f(x_0 + h_0, y_1) - f(x_0, y_0)\|/h_0$ as an estimate of the second derivative of the solution; the norm being the same as in (a).
- e) Compute a step size h_1 from the relation

$$h_1^{p+1} \cdot \max(d_1, d_2) = 0.01.$$

If $\max(d_1, d_2) \leq 10^{-15}$ we put $h_1 = \max(10^{-6}, h_0 \cdot 10^{-3})$.

- f) Finally we propose as starting step size

$$h = \min(100 \cdot h_0, h_1). \quad (4.14)$$

An algorithm like the one above, or a similar one, usually gives a good guess for the initial step size (or at least avoids a very bad choice). Sometimes, more information about h is known, e.g., from previous experience or computations of similar problems.

Numerical Experiments

As a representative of 4-stage 4th order methods we consider the “3/8 Rule” of Table 1.2. We equipped it with the embedded formula (4.9) of order 3.

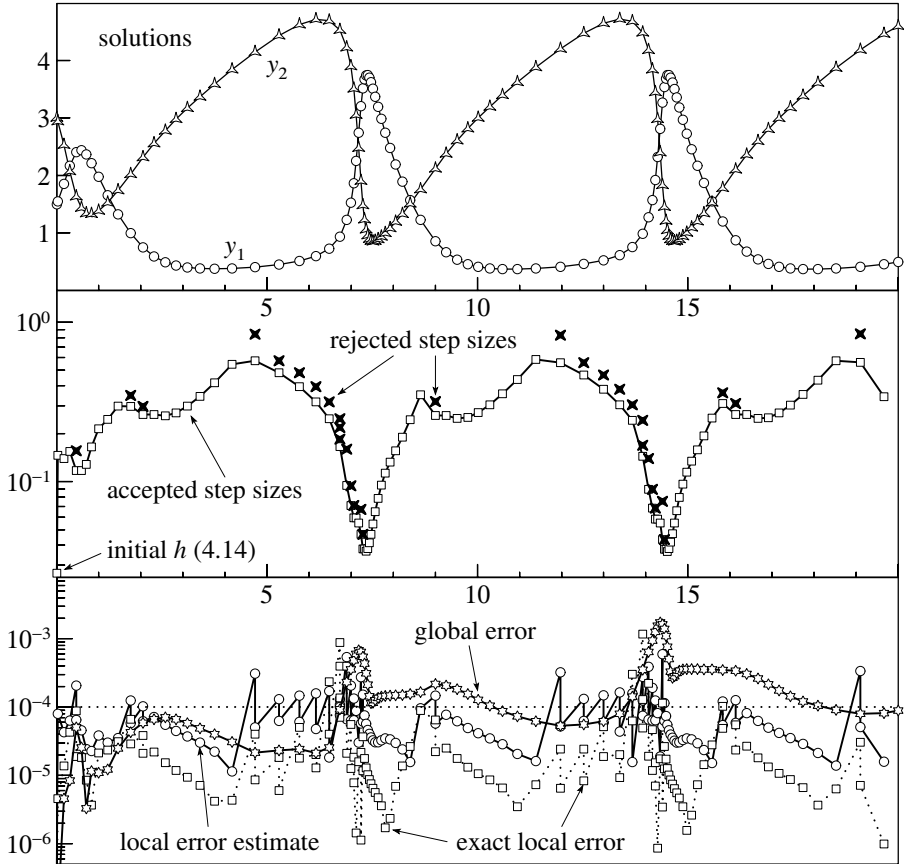


Fig. 4.1. Step size control, $Rtol = Atol = 10^{-4}$, 96 steps + 32 rejected

Step control mechanism. Fig. 4.1 presents the results of the step control mechanism (4.13) described above. As an example we choose the Brusselator (see Section I.16).

$$\begin{aligned} y_1' &= 1 + y_1^2 y_2 - 4y_1 \\ y_2' &= 3y_1 - y_1^2 y_2 \end{aligned} \quad (4.15)$$

with initial values $y_1(0) = 1.5$, $y_2(0) = 3$, integration interval $0 \leq x \leq 20$ and $Atol = Rtol = 10^{-4}$. The following results are plotted in this figure:

- i) At the top, the solutions $y_1(x)$ and $y_2(x)$ with all accepted integration steps;
- ii) then all step sizes used; the accepted ones are connected by a polygon; the rejected ones are indicated by \times ;
- iii) the third graph shows the local error estimate err , the exact local error and the global error; the desired tolerance is indicated by a broken horizontal line.

It can be seen that, due to the instabilities of the solutions with respect to the initial values, quite large global errors occur during the integration with small local tolerances everywhere. Further many step rejections can be observed in regions where the step size has to be decreased. This cannot easily be prevented, because right after an accepted step, the step size proposed by formula (4.13) is (apart from the safety factor) always increasing.

Numerical comparison. We are now curious to see the behaviour of the variable step size code, when compared to a fixed step size implementation. We applied both implementations to the Brusselator problem (4.15) with the initial values used there. The tolerances ($Atol = Rtol$) are chosen between 10^{-2} and 10^{-10} with ratio $\sqrt[3]{10}$. The results are then plotted in Fig. 4.2. There, the abscissa is the global error at the endpoint of integration (the “precision”), and the ordinate is the number of function evaluations (the “work”). We observe that for this problem the variable step size code is about twice as fast as the fixed step size code. There are, of course, problems (such as equation (0.1)) where variable step sizes are *much* more important than here.

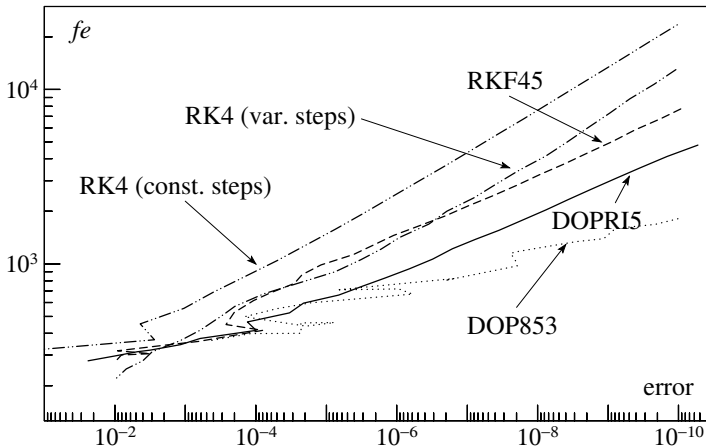


Fig. 4.2. Precision-Work diagram

In this comparison we have included some higher order methods, which will be discussed in Section II.5. The code RKF45 (written by H.A. Watts and L.F. Shampine) is based on an embedded method of order 5(4) due to Fehlberg. The codes DOPRI5 (order 5(4)) and DOP853 (order 8(5,3)) are based on methods of

Dormand & Prince. They will be discussed in the following section. It can clearly be seen that higher order methods are, especially for higher precision, more efficient than lower order methods. We shall also understand why the 5th order method of Dormand & Prince is clearly superior to RKF45.

Exercises

1. Show that Runge's method (1.4) can be interpreted as two Euler steps (with step size $h/2$), followed by a Richardson extrapolation.
2. Prove that no 4-stage Runge-Kutta method of order 4 admits an embedded formula of order 3.

Hint. Replace d_j by $\hat{b}_j - b_j$ in the proof of Lemma 1.4 and deduce that $\hat{b}_j = b_j$ for all j , which is a contradiction.

3. Show that the step size strategy (4.13) is invariant with respect to a rescaling of the independent variable. This means that it produces equivalent step size sequences when applied to the two problems

$$\begin{aligned} y' &= f(x, y), & y(0) &= y_0, & y(x_{\text{end}}) &=? \\ z' &= \sigma \cdot f(\sigma t, z), & z(0) &= y_0, & z(x_{\text{end}}/\sigma) &=? \end{aligned}$$

with initial step sizes h_0 and h_0/σ , respectively.

Remark. This is no longer the case if one replaces err in (4.13) by err/h and q by $q - 1$ ("error per unit step").

II.5 Explicit Runge-Kutta Methods of Higher Order

Gehen wir endlich zu Näherungen von der fünften Ordnung über,
so werden die Verhältnisse etwas andere. (W. Kutta 1901)

This section describes the construction of Runge-Kutta methods of higher orders, particularly of orders $p = 5$ and $p = 8$. As can be seen from Table 2.3, the complexity and number of the order conditions to be solved increases rapidly with p . An increasingly skilful use of simplifying assumptions will be the main tool for this task.

The Butcher Barriers

For methods of order 5 there are 17 order conditions to be satisfied (see Table 2.2). If we choose $s = 5$ we have 15 free parameters. Already Kutta raised the question whether there might nevertheless exist a solution (“Nun wäre es zwar möglich . . .”), but he had no hope for this and turned straight away to the case $s = 6$ (see II.2, Exercise 5). Kutta’s question remained open for more than 60 years and was answered around 1963 by three authors independently (Ceschino & Kuntzmann 1963, p. 89, Shanks 1966, Butcher 1964b, 1965b). Butcher’s work is the farthest reaching and we shall mainly follow his ideas in the following:

Theorem 5.1. *For $p \geq 5$ no explicit Runge-Kutta method exists of order p with $s = p$ stages.*

Proof. We first treat the case $s = p = 5$: define the matrices U and V by

$$U = \begin{pmatrix} \sum_i b_i a_{i2} & \sum_i b_i a_{i3} & \sum_i b_i a_{i4} \\ \sum_i b_i a_{i2} c_2 & \sum_i b_i a_{i3} c_3 & \sum_i b_i a_{i4} c_4 \\ g_2 & g_3 & g_4 \end{pmatrix}, \quad V = \begin{pmatrix} c_2 & c_2^2 & \sum_j a_{2j} c_j - c_2^2/2 \\ c_3 & c_3^2 & \sum_j a_{3j} c_j - c_3^2/2 \\ c_4 & c_4^2 & \sum_j a_{4j} c_j - c_4^2/2 \end{pmatrix} \quad (5.1)$$

where

$$g_k = \sum_{i,j} b_i a_{ij} a_{jk} - \frac{1}{2} \sum_i b_i a_{ik} (1 - c_k). \quad (5.2)$$

Then the order conditions for order 5 imply

$$UV = \begin{pmatrix} 1/6 & 1/12 & 0 \\ 1/12 & 1/20 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \quad (5.3)$$

Lemma 1.5 gives $g_4 = 0$ and consequently $c_4 = 1$ as in Lemma 1.4. Next we put in (5.1)

$$g_j = \left(\sum_i b_i a_{ij} - b_j(1 - c_j) \right) (c_j - c_5). \quad (5.4)$$

Again it can be verified by trivial computations that UV is the same as above. This time it follows that $c_4 = c_5$, hence $c_5 = 1$. Consequently, the expression

$$\sum_{i,j,k} b_i(1 - c_i)a_{ij}a_{jk}c_k \quad (5.5)$$

must be zero (because of $2 \leq k < j < i$). However, by multiplying out and using two fifth-order conditions, the expression in (5.5) should be $1/120$, a contradiction.

The case $p = s = 6$ is treated by considering all “one-leg trees”, i.e., the trees which consist of one leg above the root and the 5th order trees grafted on. The corresponding order conditions have the form

$$\sum_{i,j,\dots} b_i a_{ij} (a_{j,\dots} \dots \text{expressions for order 5}) = \frac{1}{\gamma(t)}.$$

If we let $b'_j = \sum_i b_i a_{ij}$ we are back in the 5th order 5-stage business and can follow the above ideas again. However, the $\gamma(t)$ values are not the same as before; as a consequence, the product UV in (5.3) now becomes

$$UV = \begin{pmatrix} \frac{1!}{(s-2)!} & \frac{2!}{(s-1)!} & 0 \\ \frac{2!}{(s-1)!} & \frac{3!}{s!} & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad (s=6). \quad (5.3')$$

Further, for $p = s = 7$ we use the “stork-trees” with order conditions

$$\sum_{i,j,\dots} b_i a_{ij} a_{jk} (a_{k,\dots} \dots \text{expressions for order 5}) = \frac{1}{\gamma(t)}$$

and let $b''_k = \sum_{i,j} b_i a_{ij} a_{jk}$ and so on. The general case $p = s \geq 5$ is now clear. \square

6-Stage, 5th Order Processes

We now demonstrate the construction of 5th order processes with 6 stages in full detail following the ideas which allowed Butcher (1964b) to construct 7-stage, 6th order formulas.

“In searching for such processes we are guided by the analysis of the previous section to make the following assumptions:”

$$\sum_{i=1}^6 b_i a_{ij} = b_j (1 - c_j) \quad j = 1, \dots, 6, \quad (5.6)$$

$$\sum_{j=1}^{i-1} a_{ij} c_j = \frac{c_i^2}{2} \quad i = 3, \dots, 6, \quad (5.7)$$

$$b_2 = 0. \quad (5.8)$$

The advantage of condition (5.6) is known to us already from Section II.1 (see Lemma 1.3): we can disregard all one-leg trees other than t_{21} .

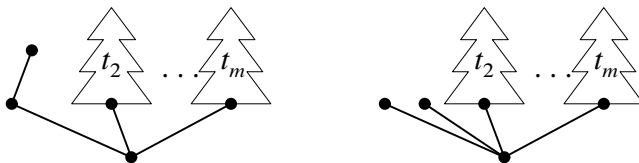


Fig. 5.1. Use of simplifying assumptions

Condition (5.7) together with (5.8) has a similar effect: for $[[\tau], t_2, \dots, t_m]$ and $[\tau, \tau, t_2, \dots, t_m]$ of Fig. 5.1 (with identical but arbitrary subtrees t_2, \dots, t_m) the order conditions read

$$\sum_{i,j} b_i a_{ij} c_j \Phi_i = \frac{1}{r \cdot 2} \quad \text{and} \quad \sum_i b_i c_i^2 \Phi_i = \frac{1}{r} \quad (5.9)$$

with known values for Φ_i and r . Since $b_2 = 0$ by (5.8) it follows from (5.7) that both conditions of (5.9) are equivalent (the condition $b_2 = 0$ is necessary for this reduction, because (5.7) cannot be satisfied for $i = 2$; otherwise we would have $c_2 = 0$ and the method would be equivalent to one of fewer stages).

The only trees left after the above reduction are the quadrature conditions

$$\sum_{i=1}^6 b_i c_i^{q-1} = \frac{1}{q} \quad q = 1, 2, 3, 4, 5 \quad (5.10)$$

and the two equations

$$\sum_{i,j,k} b_i c_i a_{ij} a_{jk} c_k = \frac{1}{5 \cdot 3 \cdot 2}, \quad (5.11)$$

$$\sum_{i,j} b_i c_i a_{ij} c_j^2 = \frac{1}{5 \cdot 3}. \quad (5.12)$$

We multiply (5.12) by $1/2$ and then subtract both equations to obtain

$$\sum_{i,j} b_i c_i a_{ij} \left(\sum_k a_{jk} c_k - c_j^2/2 \right) = 0.$$

From (5.7) the parenthesis is zero except when $j = 2$, and therefore

$$\sum_{i=3}^6 b_i c_i a_{i2} = 0 \quad (5.13)$$

replaces (5.11). Our last simplification is to subtract other order conditions from (5.12) to obtain

$$\sum_{i,j} b_i (1 - c_i) a_{ij} c_j (c_j - c_3) = \frac{1}{60} - \frac{c_3}{24}, \quad (5.14)$$

which has fewer terms than before, in particular because $c_6 = 1$ by (5.6) with $j = 6$. The resulting *reduced system* (5.6)-(5.8), (5.10), (5.13), (5.14) can easily be solved as follows:

Algorithm 5.2 (construction of 6-stage 5th order Runge-Kutta methods).

- a) $c_1 = 0$ and $c_6 = 1$ from (5.6) with $j = 6$; c_2, c_3, c_4, c_5 can be chosen as free parameters subject only to some trivial exceptions;
- b) $b_2 = 0$ from (5.8) and b_1, b_3, b_4, b_5, b_6 from the linear system (5.10);
- c) a_{32} from (5.7), $i = 3$; $a_{42} = \lambda$ arbitrary; a_{43} from (5.7), $i = 4$;
- d) a_{52} and a_{62} from the two linear equations (5.13) and (5.6), $j = 2$;
- e) a_{54} from (5.14) and a_{53} from (5.7), $i = 5$;
- f) a_{63}, a_{64}, a_{65} from (5.6), $j = 3, 4, 5$;
- g) finally a_{i1} ($i = 2, \dots, 6$) from (1.9).

Condition (5.6) for $j = 1$ and (5.7) for $i = 6$ are automatically satisfied. This follows as in the proof of Lemma 1.4.

Embedded Formulas of Order 5

Methods of Fehlberg. The methods obtained from Algorithm 5.2 do not all possess an embedded formula of order 4. Fehlberg, interested in the construction of Runge-Kutta pairs of order 4(5), looked mainly for simplifying assumptions which depend only on c_i and a_{ij} , but not on the weights b_i . In this case the simplifying assumptions are useful for the embedded method too. Therefore Fehlberg (1969) considered (5.7), (5.8) and replaced (5.6) by

$$\sum_{j=1}^{i-1} a_{ij} c_j^2 = \frac{c_i^3}{3}, \quad i = 3, \dots, 6. \quad (5.15)$$

As with (5.9) this allows us to disregard all trees of the form $[[\tau, \tau], t_2, \dots, t_m]$. In order that the reduction process of Fig. 5.1 also work on a higher level, we suppose, in addition to $b_2 = 0$, that

$$\sum_i b_i a_{i2} = 0, \quad \sum_i b_i c_i a_{i2} = 0, \quad \sum_{i,j} b_i a_{ij} a_{j2} = 0. \quad (5.16)$$

Then the last equations to be satisfied are

$$\sum_{i,j} b_i a_{ij} c_j^3 = \frac{1}{20} \quad (5.17)$$

and the quadrature conditions (5.10). We remark that the equations (5.7) and (5.15) for $i = 3$ imply

$$c_3 = \frac{3}{2} c_2. \quad (5.18)$$

We now want the method to possess an embedded formula of order 4. Analogously to (5.8) we set $\widehat{b}_2 = 0$. Then conditions (5.7) and (5.15) simplify the conditions of order 4 to 5 linear equations (the 4 quadrature conditions and $\sum_i \widehat{b}_i a_{i2} = 0$) for the 5 unknowns $\widehat{b}_1, \widehat{b}_3, \widehat{b}_4, \widehat{b}_5, \widehat{b}_6$. This system has a second solution (other than the b_i) only if it is singular, which is the case if (see Exercise 1 below)

$$c_4 = \frac{3c_2}{4 - 24c_2 + 45c_2^2}. \quad (5.19)$$

With c_2, c_5, c_6 as free parameters, the above system can be solved and yields an embedded formula of order 4(5). The coefficients of a very popular method, constructed by Fehlberg (1969), are given in Table 5.1.

Table 5.1. Fehlberg 4(5)

0						
$\frac{1}{4}$	$\frac{1}{4}$					
$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$				
$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$			
1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$-\frac{845}{4104}$		
$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$	
y_1	$\frac{25}{216}$	0	$\frac{1408}{2565}$	$\frac{2197}{4104}$	$-\frac{1}{5}$	0
\widehat{y}_1	$\frac{16}{135}$	0	$\frac{6656}{12825}$	$\frac{28561}{56430}$	$-\frac{9}{50}$	$\frac{2}{55}$

All of the methods of Fehlberg are of the type $p(\hat{p})$ with $p < \hat{p}$. Hence, the lower order approximation is intended to be used as initial value for the next step. In order to make his methods optimal, Fehlberg tried to minimize the error coefficients for the lower order result y_1 . This has the disadvantage that the local extrapolation mode (continue the integration with the higher order result) does not make sense and the estimated “error” can become substantially smaller than the true error.

It is possible to do a lot better than the pair of Fehlberg currently
regarded as “best.” (L.F. Shampine 1986)

Table 5.2. Dormand-Prince 5(4) (DOPRI5)

0							
$\frac{1}{5}$	$\frac{1}{5}$						
$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$					
$\frac{4}{5}$	$\frac{44}{45}$	$-\frac{56}{15}$	$\frac{32}{9}$				
$\frac{8}{9}$	$\frac{19372}{6561}$	$-\frac{25360}{2187}$	$\frac{64448}{6561}$	$-\frac{212}{729}$			
1	$\frac{9017}{3168}$	$-\frac{355}{33}$	$\frac{46732}{5247}$	$\frac{49}{176}$	$-\frac{5103}{18656}$		
1	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	
y_1	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	0
\hat{y}_1	$\frac{5179}{57600}$	0	$\frac{7571}{16695}$	$\frac{393}{640}$	$-\frac{92097}{339200}$	$\frac{187}{2100}$	$\frac{1}{40}$

Dormand & Prince pairs. The first efforts at minimizing the error coefficients of the higher order result, which is then used as numerical solution, were undertaken by Dormand & Prince (1980). Their methods of order 5 are constructed with the help of Algorithm 5.2 under the additional hypothesis (5.15). This condition is achieved by fixing the parameters c_3 and a_{42} in such a way that (5.15) holds for $i = 3$ and $i = 4$. The remaining two relations ($i = 5, 6$) are then automatically satisfied. To see this, multiply the difference $e_i = \sum_{j=1}^{i-1} a_{ij}c_j^2 - c_i^3/3$ by b_i and $b_i c_i$, respectively, sum up and deduce that all e_i must vanish.

In order to equip the method with an embedded formula, Dormand & Prince propose to use the FSAL idea (i.e., add y_1 as 7th stage). In this way the restriction (5.19) for c_4 is no longer necessary. We fix arbitrarily $\hat{b}_7 \neq 0$, put $\hat{b}_2 = 0$ (as in (5.8)) and compute the remaining \hat{b}_i , as above for the Fehlberg case from the 4 quadrature conditions and from $\sum_i \hat{b}_i a_{i2} = 0$.

We have thus obtained a family of 5th order Runge-Kutta methods with 4th

order embedded solution with c_2, c_4, c_5 as free parameters. Dormand & Prince (1980) have undertaken an extensive search to determine these parameters in order to minimize the error coefficients for y_1 and found that $c_2 = 1/5$, $c_4 = 4/5$ and $c_5 = 8/9$ was a close rational approximation to an optimal choice. Table 5.2 presents the coefficients of this method. The corresponding code of the Appendix is called DOPRI5.

Higher Order Processes

Order 6. By Theorem 5.1 at least 7 stages are necessary for order 6. A. Huřa (1956) constructed 6th order processes with 8 stages. Finally, methods with $s=7$, the optimal number, were derived by Butcher (1964b) along similar lines as above. He arrived at an algorithm where c_2, c_3, c_5, c_6 are free parameters.

Order 7. The existence of such a method with 8 stages is impossible by the following barrier:

Theorem 5.3 (Butcher 1965b). *For $p \geq 7$ no explicit Runge-Kutta method exists of order p with $s = p + 1$ stages.*

Since the proof of this theorem is much more complicated than that of Theorem 5.1, we do not reproduce it here.

This raises the question, whether 7th order methods with 9 stages exist. Such methods, announced by Butcher (1965b), do exist; see Verner (1971).

Order 8. As to methods of order 8, Curtis (1970) and Cooper & Verner (1972) have constructed such processes with $s = 11$. It was for a long time an open question whether there exist methods with 10 stages. John Butcher's dream of settling this difficult question before his 50th birthday did not become true. But he finally succeeded in proving the non-existence for Dahlquist's 60th birthday:

Theorem 5.4 (Butcher 1985b). *For $p \geq 8$ no explicit Runge-Kutta method exists of order p with $s = p + 2$ stages.*

For the proof, which is still more complicated, we again refer to Butcher's original paper.

Order 10. These are the highest order explicitly constructed explicit Runge-Kutta methods. Curtis (1975) constructed an 18-stage method of order 10. His construction was based solely on simplifying assumptions of the type (5.7), (5.8) and their extensions. Hairer (1978) then constructed a 17-stage method by using the complete arsenal of simplifying ideas. For more details, see the first edition, p. 189.

Embedded Formulas of High Order

It was mainly the formula manipulation genius Fehlberg who first derived high order embedded formulas. His greatest success was his 7th order formula with 8th order error estimate (Fehlberg 1968) which is of frequent use in all high precision computations, e.g., in astronomy. The coefficients are reproduced in Table 5.3.

Table 5.3. Fehlberg 7(8)													
0													
$\frac{2}{27}$	$\frac{2}{27}$												
$\frac{1}{9}$	$\frac{1}{36}$	$\frac{1}{12}$											
$\frac{1}{6}$	$\frac{1}{24}$	0	$\frac{1}{8}$										
$\frac{5}{12}$	$\frac{5}{12}$	0	$-\frac{25}{16}$	$\frac{25}{16}$									
$\frac{1}{2}$	$\frac{1}{20}$	0	0	$\frac{1}{4}$	$\frac{1}{5}$								
$\frac{5}{6}$	$-\frac{25}{108}$	0	0	$\frac{125}{108}$	$-\frac{65}{27}$	$\frac{125}{54}$							
$\frac{1}{6}$	$\frac{31}{300}$	0	0	0	$\frac{61}{225}$	$-\frac{2}{9}$	$\frac{13}{900}$						
$\frac{2}{3}$	2	0	0	$-\frac{53}{6}$	$\frac{704}{45}$	$-\frac{107}{9}$	$\frac{67}{90}$	3					
$\frac{1}{3}$	$-\frac{91}{108}$	0	0	$\frac{23}{108}$	$-\frac{976}{135}$	$\frac{311}{54}$	$-\frac{19}{60}$	$\frac{17}{6}$	$-\frac{1}{12}$				
1	$\frac{2383}{4100}$	0	0	$-\frac{341}{164}$	$\frac{4496}{1025}$	$-\frac{301}{82}$	$\frac{2133}{4100}$	$\frac{45}{82}$	$\frac{45}{164}$	$\frac{18}{41}$			
0	$\frac{3}{205}$	0	0	0	0	$-\frac{6}{41}$	$-\frac{3}{205}$	$-\frac{3}{41}$	$\frac{3}{41}$	$\frac{6}{41}$	0		
1	$-\frac{1777}{4100}$	0	0	$-\frac{341}{164}$	$\frac{4496}{1025}$	$-\frac{289}{82}$	$\frac{2193}{4100}$	$\frac{51}{82}$	$\frac{33}{164}$	$\frac{19}{41}$	0	1	
y_1	$\frac{41}{840}$	0	0	0	0	$\frac{34}{105}$	$\frac{9}{35}$	$\frac{9}{35}$	$\frac{9}{280}$	$\frac{9}{280}$	$\frac{41}{840}$	0	0
\hat{y}_1	0	0	0	0	0	$\frac{34}{105}$	$\frac{9}{35}$	$\frac{9}{35}$	$\frac{9}{280}$	$\frac{9}{280}$	0	$\frac{41}{840}$	$\frac{41}{840}$

Fehlberg's methods suffer from the fact that they give identically zero error estimates for quadrature problems $y' = f(x)$. The first high order embedded formulas which avoid this drawback were constructed by Verner (1978). One of Verner's methods (see Table 5.4) has been implemented by T.E. Hull, W.H. Enright and K.R. Jackson as DVERK and is widely used.

Table 5.4. Verner's method of order 6(5) (DVERK)

0								
$\frac{1}{6}$	$\frac{1}{6}$							
$\frac{4}{15}$	$\frac{4}{15}$	$\frac{16}{75}$						
$\frac{2}{3}$	$\frac{5}{6}$	$-\frac{8}{3}$	$\frac{5}{2}$					
$\frac{5}{6}$	$-\frac{165}{64}$	$\frac{55}{6}$	$-\frac{425}{64}$	$\frac{85}{96}$				
1	$\frac{12}{5}$	-8	$\frac{4015}{612}$	$-\frac{11}{36}$	$\frac{88}{255}$			
$\frac{1}{15}$	$-\frac{8263}{15000}$	$\frac{124}{75}$	$-\frac{643}{680}$	$-\frac{81}{250}$	$\frac{2484}{10625}$	0		
1	$\frac{3501}{1720}$	$-\frac{300}{43}$	$\frac{297275}{52632}$	$-\frac{319}{2322}$	$\frac{24068}{84065}$	0	$\frac{3850}{26703}$	
y_1	$\frac{3}{40}$	0	$\frac{875}{2244}$	$\frac{23}{72}$	$\frac{264}{1955}$	0	$\frac{125}{11592}$	$\frac{43}{616}$
\hat{y}_1	$\frac{13}{160}$	0	$\frac{2375}{5984}$	$\frac{5}{16}$	$\frac{12}{85}$	$\frac{3}{44}$	0	0

An 8th Order Embedded Method

The first high order methods with small error constants of the *higher* order solution were constructed by Prince & Dormand (1981, Code DOPRI8 of the first edition). In the following we describe the construction of a new Dormand & Prince pair of order 8(6) which will also allow a cheap and accurate dense output (see Section II.6). This method has been announced, but not published, in Dormand & Prince (1989, p. 983). We are grateful to P. Prince for mailing us the coefficients and for his help in recovering their construction.

The essential difficulty for the construction of a high order Runge-Kutta method is to set up a “good” reduced system which implies all order conditions of Theorem 2.13. At the same time it should be simple enough to be easily solved. In extending the ideas for the construction of a 5th order process (see above), Dormand & Prince proceed as follows:

Reduced system. Suppose $s = 12$ and consider for the coefficients c_i , b_i and a_{ij} the equations:

$$\sum_{i=1}^s b_i c_i^{q-1} = 1/q, \quad q = 1, \dots, 8 \quad (5.20a)$$

$$\sum_{j=1}^{i-1} a_{ij} = c_i, \quad i = 1, \dots, s \quad (5.20b)$$

$$\sum_{j=1}^{i-1} a_{ij} c_j = c_i^2/2, \quad i = 3, \dots, s \quad (5.20c)$$

$$\sum_{j=1}^{i-1} a_{ij} c_j^2 = c_i^3/3, \quad i = 3, \dots, s \quad (5.20d)$$

$$\sum_{j=1}^{i-1} a_{ij} c_j^3 = c_i^4/4, \quad i = 6, \dots, s \quad (5.20e)$$

$$\sum_{j=1}^{i-1} a_{ij} c_j^4 = c_i^5/5, \quad i = 6, \dots, s \quad (5.20f)$$

$$b_2 = b_3 = b_4 = b_5 = 0 \quad (5.20g)$$

$$a_{i2} = 0 \quad \text{for } i \geq 4, \quad a_{i3} = 0 \quad \text{for } i \geq 6 \quad (5.20h)$$

$$\sum_{i=j+1}^s b_i a_{ij} = b_j(1 - c_j), \quad j = 4, 5, 10, 11, 12 \quad (5.20i)$$

$$\sum_{i=j+1}^s b_i c_i a_{ij} = 0, \quad j = 4, 5 \quad (5.20j)$$

$$\sum_{i=j+1}^s b_i c_i^2 a_{ij} = 0, \quad j = 4, 5 \quad (5.20k)$$

$$\sum_{i=k+2}^s b_i c_i \sum_{j=k+1}^{i-1} a_{ij} a_{jk} = 0, \quad k = 4, 5 \quad (5.20l)$$

$$\sum_{i=1}^s b_i c_i \sum_{j=1}^{i-1} a_{ij} c_j^5 = 1/48. \quad (5.20m)$$

Verification of the order conditions. The equations (5.20a) are the order conditions for the bushy trees $[\tau, \dots, \tau]$ and (5.20m) is that for the tree $[\tau, [\tau, \tau, \tau, \tau, \tau]]$. For the verification of further order conditions we shall show that the reduced system implies

$$\sum_{i=j+1}^s b_i a_{ij} = b_j(1 - c_j) \quad \text{for all } j. \quad (5.21)$$

If we denote the difference by $d_j = \sum_{i=j+1}^s b_i a_{ij} - b_j(1 - c_j)$ then $d_2 = d_3 = 0$ by (5.20g,h) and $d_4 = d_5 = d_{10} = d_{11} = d_{12} = 0$ by (5.20i). The conditions (5.20a-g) imply

$$\sum_{j=1}^s d_j c_j^{q-1} = 0 \quad \text{for } q = 1, \dots, 5. \quad (5.22)$$

Hence, the remaining 5 values must also vanish if c_1, c_6, c_7, c_8, c_9 are distinct. The significance of condition (5.21) is already known from Lemma 1.3 and from formula (5.6). It implies that all one-leg trees $t = [t_1]$ can be disregarded.



Fig. 5.2. Use of simplifying assumptions

Conditions (5.20c-f) are an extension of (5.6) and (5.15). Their importance will be, once more, demonstrated on an example. Consider the two trees of Fig. 5.2 and suppose that their encircled parts are identical. Then the corresponding order

conditions are

$$\sum_{i,j=1}^s \Theta_i a_{ij} c_j^3 = \frac{1}{r \cdot 5 \cdot 4} \quad \text{and} \quad \sum_{i=1}^s \Theta_i c_i^4 = \frac{1}{r \cdot 5} \quad (5.23)$$

with known values for Θ_i and r . If (5.20e) is satisfied and if

$$\Theta_2 = \Theta_3 = \Theta_4 = \Theta_5 = 0 \quad (5.24)$$

then both conditions are equivalent so that the left-hand tree can be neglected. The conditions (5.20g,i-l) correspond to (5.24) for certain trees. Finally the assumption (5.20h) together with (5.20g,i-k) implies that for arbitrary Φ_i , Ψ_j and for $q \in \{1, 2, 3\}$,

$$\begin{aligned} \sum_i b_i \Phi_i a_{i2} &= 0 \\ \sum_{i,j} b_i \Phi_i a_{ij} \Psi_j a_{j2} &= 0 \\ \sum_{i,j,k} b_i c_i^{q-1} a_{ij} \Phi_j a_{jk} \Psi_k a_{k2} &= 0 \end{aligned} \quad \text{and} \quad \begin{aligned} \sum_i b_i \Phi_i a_{i3} &= 0 \\ \sum_{i,j} b_i c_i^{q-1} a_{ij} \Phi_j a_{j3} &= 0 \end{aligned}$$

which are again conditions of type (5.24). Using these relations the verification of the order conditions (order 8) is straightforward; all trees are reduced to those corresponding to (5.20a) and (5.20m).

Solving the reduced system. Compared to the original 200 order conditions of Theorem 2.13 for the 78 coefficients b_i, a_{ij} (the c_i are defined by (5.20b)), the 74 conditions of the reduced system present a considerable simplification. We can hope for a solution with 4 degrees of freedom.

We start by expressing the coefficients b_i, a_{ij} in terms of the c_i . Because of (5.20g), condition (5.20a) represents a linear system for b_1, b_6, \dots, b_{12} , which has a unique solution if c_1, c_6, \dots, c_{12} are distinct. For a fixed i ($1 \leq i \leq 8$) conditions (5.20b-f) represent a linear system for $a_{i1}, \dots, a_{i,i-1}$. Since there are sometimes less unknowns than equations (mainly due to (5.20h)) restrictions have to be imposed on the c_i . One verifies (similarly to (5.18)) that the relations

$$\begin{aligned} c_1 &= 0, & c_2 &= \frac{2}{3} c_3, & c_3 &= \frac{2}{3} c_4, \\ c_4 &= \frac{6 - \sqrt{6}}{10} c_6, & c_5 &= \frac{6 + \sqrt{6}}{10} c_6, & c_6 &= \frac{4}{3} c_7 \end{aligned} \quad (5.25a)$$

allow the computation of the a_{ij} with $i \leq 8$ (Step 1 in Fig. 5.3).

If $b_{12} \neq 0$ (which will be assumed in our construction), condition (5.20i) for $j = 12$ implies

$$c_{12} = 1, \quad (5.25b)$$

and for $j = 11$ it yields the value for $a_{12,11}$. We next compute the expressions

$$e_j = \sum_{i=j+1}^s b_i c_i a_{ij} - \frac{b_j}{2} (1 - c_j^2), \quad j = 1, \dots, s. \quad (5.26)$$

& Prince propose the following numerical values:

$$c_7 = 1/4, \quad c_8 = 4/13, \quad c_{10} = 3/5, \quad c_{11} = 6/7.$$

All remaining coefficients are then determined by the above procedure. Since c_4 and c_5 (see (5.25a)) are not rational, there is no easy way to present the coefficients in a tableau.

Embedded method. We look for a second method with the same c_i, a_{ij} but with different weights, say \hat{b}_i . If we require that

$$\sum_{i=1}^s \hat{b}_i c_i^{q-1} = 1/q, \quad q = 1, \dots, 6 \quad (5.29a)$$

$$\hat{b}_2 = \hat{b}_3 = \hat{b}_4 = \hat{b}_5 = 0 \quad (5.29b)$$

$$\sum_{i=j+1}^s \hat{b}_i a_{ij} = 0, \quad j = 4, 5 \quad (5.29c)$$

then one can verify (similarly as above for the 8th order method) that the corresponding Runge-Kutta method is of order 6. The system (5.29) consists of 12 linear equations for 12 unknowns. A comparison with (5.20) shows that b_1, \dots, b_{12} is a solution of (5.29). Furthermore, the corresponding homogeneous system has the nontrivial solution e_1, \dots, e_{12} (see (5.27) and (5.20)). Therefore

$$\hat{b}_i = b_i + \alpha e_i \quad (5.30)$$

is a solution of (5.29) for all values of α . Dormand & Prince suggest taking α in such a way that $\hat{b}_6 = 2$.

A program based on this method (with a different error estimator, see Section II.10) has been written and is called DOP853. It is documented in the Appendix. The performance of this code, compared to methods of lower order, is impressive. See for example the results for the Brusselator in Fig. 4.2.

Exercises

1. Consider a Runge-Kutta method with s stages that satisfies (5.7)-(5.8), (5.15), (5.17) and the first two relations of (5.16).
 - a) If the relation (5.19) holds, then the method possesses an embedded formula of order 4.
 - b) The condition (5.19) implies that the last relation of (5.16) is automatically satisfied.

Hint. The order conditions for the embedded method constitute a linear system for the \hat{b}_i which has to be singular. This implies that

$$a_{i2} = \alpha c_i + \beta c_i^2 + \gamma c_i^3 \quad \text{for} \quad i \neq 2. \quad (5.31)$$

Multiplying (5.31) with b_i and $b_i c_i$ and summing up, yields two relations for $\alpha, \beta, J\gamma$. These together with (5.31) for $i = 3, 4$ yield (5.19).

2. Construct a 6-stage 5th order formula with $c_3 = 1/3$, $c_4 = 1/2$, $c_5 = 2/3$ possessing an embedded formula of order 4.
3. (Butcher). Show that for any Runge-Kutta method of order 5,

$$\sum_i b_i \left(\sum_j a_{ij} c_j - \frac{c_i^2}{2} \right)^2 = 0.$$

Consequently, there exists no explicit Runge-Kutta method of order 5 with all $b_i > 0$.

Hint. Multiply out and use order conditions.

4. Write a code with a high order Runge-Kutta method (or take one) and solve numerically the Arenstorf orbit of the restricted three body problem (0.1) (see the introduction) with initial values

$$\begin{aligned} y_1(0) &= 0.994, & y_1'(0) &= 0, & y_2(0) &= 0, \\ y_2'(0) &= -2.0317326295573368357302057924, \end{aligned}$$

Compute the solutions for

$$x_{\text{end}} = 11.124340337266085134999734047.$$

The initial values are chosen such that the solution is periodic to this precision. The plotted solution curve has one loop less than that of the introduction.

5. (Shampine 1979). Show that the storage requirement of a Runge-Kutta method can be substantially decreased if s is large.

Hint. Suppose, for example, that $s = 15$.

After computing (see (1.8)) k_1, k_2, \dots, k_9 , compute the sums

$$\sum_{j=1}^9 a_{ij} k_j \quad \text{for } i = 10, 11, 12, 13, 14, 15, \quad \sum_{j=1}^9 b_j k_j, \quad \sum_{j=1}^9 \hat{b}_j k_j;$$

then the memories occupied by k_2, k_3, \dots, k_9 are not needed any longer. Another possibility for reducing the memory requirement is offered by the zero-pattern of the coefficients.

6. Show that the reduced system (5.20) implies (5.25c).

Hint. The equations (5.20b-f) imply that for $i \in \{1, 6, 7, 8, 9\}$

$$\alpha a_{i4} + \beta a_{i5} = \sigma_3 \frac{c_i^2}{2} - \sigma_2 \frac{c_i^3}{3} + \sigma_1 \frac{c_i^4}{4} - \frac{c_i^5}{5} \quad (5.32)$$

with σ_j given by (5.28). The constants α and β are not important. Further, for the same values of i one has

$$\begin{aligned} 0 &= c_i(c_i - c_6)(c_i - c_7)(c_i - c_8)(c_i - c_9) \\ &= \sigma_3 c_9 c_i - (\sigma_3 + c_9 \sigma_2) c_i^2 + (\sigma_2 + c_9 \sigma_1) c_i^3 - (\sigma_1 + c_9) c_i^4 + c_i^5. \end{aligned} \quad (5.33)$$

Multiplying (5.32) and (5.33) by $e_i, b_i, b_i c_i, b_i c_i^2$, summing up from $i = 1$ to s and using (5.20) gives the relation

$$\begin{pmatrix} \times & \times & \times \\ \times & \times & \times \\ 0 & 0 & b_{12}^{-1} \end{pmatrix} \begin{pmatrix} e_{10} & b_{10} & b_{10} c_{10} & b_{10} c_{10}^2 \\ e_{11} & b_{11} & b_{11} c_{11} & b_{11} c_{11}^2 \\ 0 & b_{12} & b_{12} & b_{12} \end{pmatrix} = \begin{pmatrix} 0 & \gamma_1 & \gamma_2 & \gamma_3 \\ 0 & \delta_1 & \delta_2 & \delta_3 \\ 0 & 1 & 1 & 1 \end{pmatrix} \quad (5.34)$$

where

$$\begin{aligned} \gamma_j &= \frac{\sigma_3}{2 \cdot (j+2)} - \frac{\sigma_2}{3 \cdot (j+3)} + \frac{\sigma_1}{4 \cdot (j+4)} - \frac{1}{5 \cdot (j+5)} \\ \delta_j &= \frac{\sigma_3 c_9}{j+1} - \frac{\sigma_3 + c_9 \sigma_2}{j+2} + \frac{\sigma_2 + c_9 \sigma_1}{j+3} - \frac{\sigma_1 + c_9}{j+4} + \frac{1}{j+5} \end{aligned}$$

and the “ \times ” indicate certain values. Deduce from (5.34) and $e_{11} \neq 0$ that the most left matrix of (5.34) is singular. This implies that the right-hand matrix of (5.34) is of rank 2 and yields equation (5.25c).

7. Prove that the 8th order method given by (5.20; $s = 12$) does not possess a 6th order embedding with $\hat{b}_{12} \neq b_{12}$, not even if one adds the numerical result y_1 as 13th stage (FSAL).

II.6 Dense Output, Discontinuities, Derivatives

... providing “interpolation” for Runge-Kutta methods. ... this capability and the features it makes possible will be the hallmark of the next generation of Runge-Kutta codes.

(L.F. Shampine 1986)

The present section is mainly devoted to the construction of dense output formulas for Runge-Kutta methods. This is important for many practical questions such as graphical output, event location or the treatment of discontinuities in differential equations. Further, the numerical computation of derivatives with respect to initial values and parameters is discussed, which is particularly useful for the integration of boundary value problems.

Dense Output

Classical Runge-Kutta methods are inefficient, if the number of output points becomes very large (Shampine, Watts & Davenport 1976). This motivated the construction of dense output formulas (Horn 1983). These are Runge-Kutta methods which provide, in addition to the numerical result y_1 , cheap numerical approximations to $y(x_0 + \theta h)$ for the *whole* integration interval $0 \leq \theta \leq 1$. “Cheap” means without or, at most, with only a few additional function evaluations.

We start from an s -stage Runge-Kutta method with given coefficients c_i, a_{ij} and b_j , eventually add $s^* - s$ new stages, and consider formulas of the form

$$u(\theta) = y_0 + h \sum_{i=1}^{s^*} b_i(\theta) k_i, \quad (6.1)$$

where

$$k_i = f\left(x_0 + c_i h, y_0 + h \sum_{j=1}^{i-1} a_{ij} k_j\right), \quad i = 1, \dots, s^* \quad (6.2)$$

and $b_i(\theta)$ are polynomials to be determined such that

$$u(\theta) - y(x_0 + \theta h) = \mathcal{O}(h^{p^*+1}). \quad (6.3)$$

Usually $s^* \geq s + 1$ since we include (at least) the first function evaluation of the subsequent step $k_{s+1} = hf(x_0 + h, y_1)$ in the formula with $a_{s+1,j} = b_j$ for all j . A Runge-Kutta method, provided with a formula (6.1), will be called a *continuous* Runge-Kutta method.

Theorem 6.1. *The error of the approximation (6.1) is of order p^* (i.e., the local error satisfies (6.3)), if and only if*

$$\sum_{j=1}^{s^*} b_j(\theta) \Phi_j(t) = \frac{\theta^{\varrho(t)}}{\gamma(t)} \quad \text{for} \quad \varrho(t) \leq p^* \quad (6.4)$$

with $\Phi_j(t)$, $\varrho(t)$, $\gamma(t)$ given in Section II.2.

Proof. The q th derivative (with respect to h) of the numerical approximation is given by (2.14) with b_j replaced by $b_j(\theta)$; that of the exact solution $y(x_0 + \theta h)$ is $\theta^q y^{(q)}(x_0)$. The statement thus follows as in Theorem 2.13. \square

Corollary 6.2. *Condition (6.4) implies that the derivatives of (6.1) approximate the derivatives of the exact solution as*

$$h^{-k} u^{(k)}(\theta) - y^{(k)}(x_0 + \theta h) = \mathcal{O}(h^{p^* - k + 1}). \quad (6.5)$$

Proof. Comparing the q th derivative (with respect to h) of $u'(\theta)$ with that of $hy'(x_0 + \theta h)$ we find that (6.5) (for $k = 1$) is equivalent to

$$\sum_{j=1}^{s^*} b'_j(\theta) \Phi_j(t) = \frac{\varrho(t) \theta^{\varrho(t)-1}}{\gamma(t)} \quad \text{for} \quad \varrho(t) \leq p^*.$$

This, however, follows from (6.4) by differentiation. The case $k > 1$ is obtained similarly. \square

We write the polynomials $b_j(\theta)$ as

$$b_j(\theta) = \sum_{q=1}^{p^*} b_{jq} \theta^q, \quad (6.6)$$

so that the equations (6.4) become a system of simultaneous linear equations of the form

$$\underbrace{\begin{pmatrix} 1 & 1 & \dots & 1 \\ \Phi_1(t_{21}) & \Phi_2(t_{21}) & \dots & \Phi_{s^*}(t_{21}) \\ \Phi_1(t_{31}) & \Phi_2(t_{31}) & \dots & \Phi_{s^*}(t_{31}) \\ \vdots & \vdots & & \vdots \end{pmatrix}}_{\Phi} \underbrace{\begin{pmatrix} b_{11} & b_{12} & b_{13} & \dots \\ b_{21} & b_{22} & b_{23} & \dots \\ \vdots & \vdots & \vdots & \dots \\ b_{s^*1} & b_{s^*2} & b_{s^*3} & \dots \end{pmatrix}}_B = \underbrace{\begin{pmatrix} 1 & 0 & 0 & \dots \\ 0 & \frac{1}{2} & 0 & \dots \\ 0 & 0 & \frac{1}{3} & \dots \\ 0 & 0 & \frac{1}{6} & \dots \\ \vdots & \vdots & \vdots & \dots \end{pmatrix}}_G \quad (6.4')$$

where the $\Phi_j(t)$ are known numbers depending on a_{ij} and c_i . Using standard linear algebra the solution of this system can easily be discussed. It may happen,

however, that the order p^* of the dense output is smaller than the order p of the underlying method.

Example. For “the” Runge-Kutta method of Table 1.2 (with $s^* = s = 4$) equations (6.4') with $p^* = 3$ produce a unique solution

$$b_1(\theta) = \theta - \frac{3\theta^2}{2} + \frac{2\theta^3}{3}, \quad b_2(\theta) = b_3(\theta) = \theta^2 - \frac{2\theta^3}{3}, \quad b_4(\theta) = -\frac{\theta^2}{2} + \frac{2\theta^3}{3}$$

which constitutes a dense output solution which is globally continuous but not C^1 .

Hermite interpolation. A much easier way (than solving (6.4')) and more efficient for low order dense output formulas is the use of Hermite interpolation (Shampine 1985). Whatever the method is, we have two function values y_0, y_1 and two derivatives $f_0 = f(x_0, y_0)$, $f_1 = f(x_0 + h, y_1)$ at our disposal and can thus do cubic polynomial interpolation. The resulting formula is

$$u(\theta) = (1-\theta)y_0 + \theta y_1 + \theta(\theta-1) \left((1-2\theta)(y_1 - y_0) + (\theta-1)hf_0 + \theta hf_1 \right). \quad (6.7)$$

Inserting the definition of y_1 into (6.7) shows that Hermite interpolation is a special case of (6.1). Whenever the underlying method is of order $p \geq 3$ we thus obtain a continuous Runge-Kutta method of order 3.

Since the function and derivative values on the right side of the first interval coincide with those on the left side of the second interval, Hermite interpolation leads to a globally C^1 approximation of the solution.

The 4-stage 4th order methods of Section II.1 do not possess a dense output of order 4 without any additional function evaluations (see Exercise 1). Therefore the question arises whether it is really important to have a dense output of the same order. Let us consider an interval far away from the initial value, say $[x_n, x_{n+1}]$, and denote by $z(x)$ the local solution, i.e., the solution of the differential equation which passes through (x_n, y_n) . Then the error of the dense output is composed of two terms:

$$u(\theta) - y(x_n + \theta h) = (u(\theta) - z(x_n + \theta h)) + (z(x_n + \theta h) - y(x_n + \theta h)).$$

The term to the far right reflects the global error of the method and is of size $\mathcal{O}(h^p)$. In order that both terms be of the same order of magnitude it is thus sufficient to require $p^* = p - 1$.

The situation changes, if we also need accurate values of the derivative $y'(x_n + \theta h)$ (see Section 5 of Enright, Jackson, Nørsett & Thomsen (1986) for a discussion of problems where this is important). We have

$$h^{-1}u'(\theta) - y'(x_n + \theta h) = (h^{-1}u'(\theta) - z'(x_n + \theta h)) + (z'(x_n + \theta h) - y'(x_n + \theta h))$$

and the term to the far right is of size $\mathcal{O}(h^p)$ if $f(x, y)$ satisfies a Lipschitz condition. A comparison with (6.5) shows that we need $p^* = p$ in order that both error terms be of comparable size.

Boot-strapping process (Enright, Jackson, Nørsett & Thomsen 1986). This is a general procedure for increasing iteratively the order of dense output formulas.

Suppose that we already have a 3rd order dense output at our disposal (e.g., from Hermite interpolation). We then fix arbitrarily an $\alpha \in (0, 1)$ and denote the 3rd order approximation at $x_0 + \alpha h$ by y_α . The idea is now that $hf(x_0 + \alpha h, y_\alpha)$ is a 4th order approximation to $hy'(x_0 + \alpha h)$. Consequently, the 4th degree polynomial $u(\theta)$ defined by

$$\begin{aligned} u(0) &= y_0, & u'(0) &= hf(x_0, y_0) \\ u(1) &= y_1, & u'(1) &= hf(x_0 + h, y_1) \\ u'(\alpha) &= hf(x_0 + \alpha h, y_\alpha) \end{aligned} \quad (6.8)$$

(which exists uniquely for $\alpha \neq 1/2$) yields the desired formula. The interpolation error is $\mathcal{O}(h^5)$ and each quantity of (6.8) approximates the corresponding exact solution value with an error of $\mathcal{O}(h^5)$.

The extension to arbitrary order is straightforward. Suppose that a dense output formula $u_0(\theta)$ of order $p^* < p$ is known. We then evaluate this polynomial at $p^* - 2$ distinct points $\alpha_i \in (0, 1)$ and compute the values $f(x_0 + \alpha_i h, u_0(\alpha_i))$. The interpolation polynomial $u_1(\theta)$ of degree $p^* + 1$, defined by

$$\begin{aligned} u_1(0) &= y_0, & u_1'(0) &= hf(x_0, y_0) \\ u_1(1) &= y_1, & u_1'(1) &= hf(x_0 + h, y_1) \\ u_1'(\alpha_i) &= hf(x_0 + \alpha_i h, u_0(\alpha_i)), & i &= 1, \dots, p^* - 2, \end{aligned} \quad (6.9)$$

yields an interpolation formula of order $p^* + 1$. Obviously, the α_i in (6.9) have to be chosen such that the corresponding interpolation problem admits a solution.

Continuous Dormand & Prince Pairs

The method of Dormand & Prince (Table 5.2) is of order 5(4) so that we are mainly interested in dense output formulas with $p^* = 4$ and $p = 5$.

Order 4. A continuous formula of order 4 can be obtained without any additional function evaluation. Since the coefficients satisfy (5.7), it follows from the difference of the order conditions for the trees t_{31} and t_{32} (notation of Table 2.2) that

$$b_2(\theta) = 0 \quad (6.10)$$

is necessary. This condition together with (5.7) and (5.15) then implies that the order conditions are equivalent for the following pairs of trees: t_{31} and t_{32} , t_{41} and t_{42} , t_{41} and t_{43} . Hence, for order 4, only 5 conditions have to be considered (the four quadrature conditions and $\sum_i b_i(\theta)a_{i2} = 0$). We can arbitrarily choose $b_7(\theta)$ and the coefficients $b_1(\theta), b_3(\theta), \dots, b_6(\theta)$ are then uniquely determined.

As for the choice of $b_7(\theta)$, Shampine (1986) proposed minimizing, for each θ , the error coefficients (Theorem 3.2)

$$e(t) = \theta^5 - \gamma(t) \sum_{j=1}^7 b_j(\theta) \Phi_j(t) \quad \text{for } t \in T_5, \quad (6.11)$$

weighted by $\alpha(t)$ of Definition 2.5, in the square norm. These expressions can be seen to depend linearly on $b_7(\theta)$,

$$\alpha(t)e(t) = \zeta(t, \theta) - b_7(\theta)\eta(t),$$

thus the minimal value is found for

$$b_7(\theta) = \sum_{t \in T_5} \zeta(t, \theta) \eta(t) / \sum_{t \in T_5} \eta^2(t).$$

The resulting formula, given by Dormand & Prince (1986), is

$$b_7(\theta) = \theta^2(\theta - 1) + \theta^2(\theta - 1)^2 10 \cdot (7414447 - 829305\theta) / 29380423. \quad (6.12)$$

The other coefficients, written in a fashion which makes the Hermite-part clearly visible, are then given by

$$\begin{aligned} b_1(\theta) &= \theta^2(3 - 2\theta) \cdot b_1 + \theta(\theta - 1)^2 \\ &\quad - \theta^2(\theta - 1)^2 5 \cdot (2558722523 - 31403016\theta) / 11282082432 \\ b_3(\theta) &= \theta^2(3 - 2\theta) \cdot b_3 + \theta^2(\theta - 1)^2 100 \cdot (882725551 - 15701508\theta) / 32700410799 \\ b_4(\theta) &= \theta^2(3 - 2\theta) \cdot b_4 - \theta^2(\theta - 1)^2 25 \cdot (443332067 - 31403016\theta) / 1880347072 \\ b_5(\theta) &= \theta^2(3 - 2\theta) \cdot b_5 + \theta^2(\theta - 1)^2 32805 \cdot (23143187 - 3489224\theta) / 199316789632 \\ b_6(\theta) &= \theta^2(3 - 2\theta) \cdot b_6 - \theta^2(\theta - 1)^2 55 \cdot (29972135 - 7076736\theta) / 822651844. \end{aligned} \quad (6.13)$$

It can be directly verified that the interpolation polynomial $u(\theta)$ defined by (6.10), (6.12) and (6.13) satisfies

$$\begin{aligned} u(0) &= y_0, & u'(0) &= hf(x_0, y_0), \\ u(1) &= y_1, & u'(1) &= hf(x_0 + h, y_1), \end{aligned} \quad (6.14)$$

so that it produces globally a C^1 approximation of the solution.

Instead of using the above 5th degree polynomial $u(\theta)$, Shampine (1986) suggests evaluating it only at the midpoint, $y_{1/2} = u(1/2)$, and then doing quartic polynomial interpolation with the five values y_0 , $hf(x_0, y_0)$, y_1 , $hf(x_0 + h, y_1)$, $y_{1/2}$. This dense output is also C^1 , is easier to implement and the difference to the above formula "... is not significant" (Dormand & Prince 1986).

We have implemented Shampine's dense output in the code DOPRI5 (see Appendix). The advantages of such a dense output for graphical representations of the solution can already be seen from Fig. 0.1 of the introduction to Chapter II. For a more thorough study we have applied DOPRI5 to the Brusselator (4.15) with initial

values $y_1(0) = 1.5$, $y_2(0) = 3$, integration interval $0 \leq x \leq 10$ and error tolerance $Atol = Rtol = 10^{-4}$. The global error of the above 4th order continuous solution is displayed in Fig. 6.1 for both components. The error shows the same quality throughout; the grid points, which are represented by the symbols \square and \circ , are by no means outstanding.

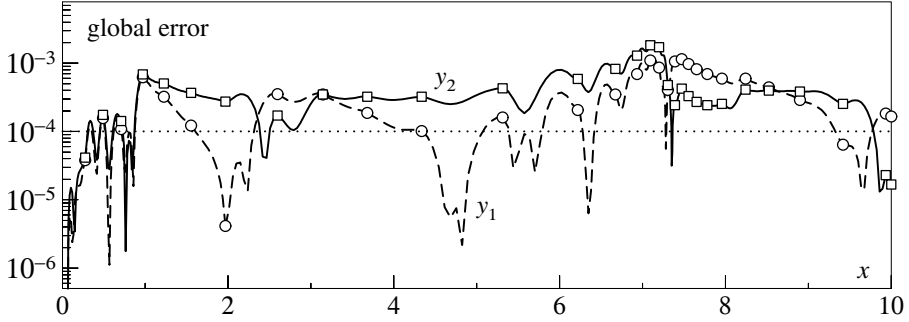


Fig. 6.1. Error of dense output of DOPRI5

Order 5. For a dense output of order $p^* = 5$ for the Dormand & Prince method the linear system (6.4') has no solution since

$$\text{rank}(\Phi|G) = 9 \quad \text{and} \quad \text{rank}(\Phi) = 7 \quad (6.15)$$

as can be verified by Gaussian elimination. Such a linear system has a solution if and only if the two ranks in (6.15) are *equal*. So we must append additional stages to the method. Each new stage adds a new column to the matrix Φ , thus may increase the rank of Φ by one without changing $\text{rank}(\Phi|G)$. Therefore we obtain

Lemma 6.3 (Owren & Zennaro 1991). *Consider a Runge-Kutta method of order p . For the construction of a continuous extension of order $p^* = p$ one has to add at least*

$$\delta := \text{rank}(\Phi|G) - \text{rank}(\Phi) \quad (6.16)$$

stages.

□

For the Dormand & Prince method we thus need at least two additional stages. There are several possibilities for constructing such dense output formulas:

- a) Shampine (1986) shows that one new function evaluation allows one to compute a 5th order approximation at the midpoint $x_0 + h/2$. If one evaluates anew the function at this point to get an approximation of $y'(x_0 + h/2)$, one can do quintic Hermite interpolation to get a dense output of order 5.

- b) Use the 4th order formula constructed above at two different output points and do boot-strapping. This has been done by Calvé & Vaillancourt (1990).
- c) Add two arbitrary new stages and solve the order conditions. This leads to methods with 10 free parameters (Calvo, Montijano & Rández 1992) which can then be used to minimize the error terms. This seems to give the best output formulas.

New methods. If anyhow the Dormand & Prince pair needs two additional function evaluations for a 5th order dense output, the suggestion lies at hand to search for completely new methods which use *all* stages for the solution y_1 and \hat{y}_1 as well. Owren & Zennaro (1992) constructed an 8-stage continuous Runge-Kutta method of order 5(4). It uses the FSAL idea so that the effective cost is 7 function evaluations (*fe*) per step. Bogacki & Shampine (1989) present a 7-stage method of order 5(4) with very small error coefficients, so that it nearly behaves like a 6th order method. The effective cost of its dense output is 10 *fe*. A method of order 6(5) with a dense output of order $p^* = 5$ is given by Calvo, Montijano & Rández (1990).

Dense Output for DOP853

We are interested in a continuous extension of the 8th order method of Section II.5 (formula (5.20)). A dense output of order 6 can be obtained for free (add y_1 as 13th stage and solve the linear system (6.19a-c) below with $s^* = s + 1 = 13$). Following Dormand & Prince we shall construct a dense output of order $p^* = 7$. We add three further stages (by Lemma 6.3 this is the minimal number of additional stages). The values for c_{14}, c_{15}, c_{16} are chosen arbitrarily as

$$c_{14} = 0.1, \quad c_{15} = 0.2, \quad c_{16} = 7/9 \quad (6.17)$$

and the coefficients a_{ij} are assumed to satisfy, for $i \in \{14, 15, 16\}$,

$$\sum_{j=1}^{i-1} a_{ij} c_j^{q-1} = c_i^q / q, \quad q = 1, \dots, 6 \quad (6.18a)$$

$$a_{i2} = a_{i3} = a_{i4} = a_{i5} = 0 \quad (6.18b)$$

$$\sum_{j=k+1}^{i-1} a_{ij} a_{jk} = 0, \quad k = 4, 5. \quad (6.18c)$$

This system can easily be solved (step 5 of Fig. 5.3). We are still free to set some coefficients equal to 0 (see Fig. 5.3).

We next search for polynomials $b_i(\theta)$ such that the conditions (6.4) are satisfied for all trees of order ≤ 7 . We find the following necessary conditions ($s^* = 16$)

$$\sum_{i=1}^{s^*} b_i(\theta) c_i^{q-1} = \theta^q / q, \quad q = 1, \dots, 7 \quad (6.19a)$$

$$b_2(\theta) = b_3(\theta) = b_4(\theta) = b_5(\theta) = 0 \quad (6.19b)$$

$$\sum_{i=j+1}^{s^*} b_i(\theta) a_{ij} = 0, \quad j = 4, 5 \quad (6.19c)$$

$$\sum_{i=j+1}^{s^*} b_i(\theta) c_i a_{ij} = 0, \quad j = 4, 5 \quad (6.19d)$$

$$\sum_{i,j=1}^{s^*} b_i(\theta) a_{ij} c_j^5 = \theta^7/42. \quad (6.19e)$$

Here (6.19a,e) are order conditions for $[\tau, \dots, \tau]$ and $[[\tau, \tau, \tau, \tau, \tau]]$. The property $b_2(\theta) = 0$ follows from $0 = \sum_i b_i(\theta) (\sum_j a_{ij} c_j - c_i^2/2) = -b_2(\theta) c_2^2/2$ and the other three conditions of (6.19b) are a consequence of the relations $0 = \sum_i b_i(\theta) c_i^{q-1} (\sum_j a_{ij} c_j^3 - c_i^4/4) = 0$ for $q = 1, 2, 3$. The necessity of the conditions (6.19c,d) is seen similarly.

On the other hand, the conditions (6.19) are also sufficient for the dense output to be of order 7. We first remark that (6.19), (6.18) and (5.20) imply

$$\sum_{i,j=k+1}^{s^*} b_i(\theta) a_{ij} a_{jk} = 0, \quad k = 4, 5 \quad (6.20)$$

(see Exercise 3). The verification of the order conditions (6.4) is then possible without difficulty.

System (6.19) consists of 16 linear equations for 16 unknowns which possess a unique solution. An interesting property of the continuous solution (6.1) obtained in this manner is that it yields a global C^1 -approximation to the solution, i.e.,

$$u(0) = y_0, \quad u(1) = y_1, \quad u'(0) = hf(y_0), \quad u'(1) = hf(y_1). \quad (6.21)$$

For the verification of this property we define a polynomial $q(\theta)$ of degree 7 by the relations (6.21) and by $q(\theta_i) = u(\theta_i)$ for 4 distinct values θ_i which are different from 0 and 1. Obviously, $q(\theta)$ is of the form (6.1) and defines a dense output of order 7. Due to the uniqueness of the $b_i(\theta)$ we must have $q(\theta) \equiv u(\theta)$ so that (6.21) is verified.

Event Location

Often the output value x_{end} for which the solutions are wanted is not known in advance, but depends implicitly on the computed solutions. An example of such a situation is the search for periodic solutions and limit cycles discussed in Section I.16, where we wanted to know when the solution reaches the Poincaré-section for the first time.

Such problems are very easily treated when a dense output $u(x)$ is available. Suppose we want to determine x such that

$$g(x, y(x)) = 0. \quad (6.22)$$

Algorithm 6.4. Compute the solution step-by-step until a sign change appears between $g(x_i, y_i)$ and $g(x_{i+1}, y_{i+1})$ (this is, however, not completely safe because g may change sign twice in an integration interval; use the dense output at intermediate values if more safety is needed). Then replace $y(x)$ in (6.22) by the

approximation $u(x)$ and solve the resulting equation numerically, e.g. by bisection or Newton iterations.

This algorithm can be conveniently done in the subroutine SOLOUT, which is called after every accepted step (see Appendix). If the value of x , satisfying (6.22), has been found, the integration is stopped by setting ITRN = -1.

Whenever the function g of (6.22) also depends on $y'(x)$, it is advisable to use a dense output of order $p^* = p$.

Discontinuous Equations

If you write some software which is half-way useful, sooner or later someone will use it on discontinuities. You have to scope about ... (A.R. Curtis 1986)

In many applications the function defining a differential equation is not analytic or continuous everywhere. A common example is a problem which (at least locally) can be written in the form

$$y' = \begin{cases} f_I(y) & \text{if } g(y) > 0 \\ f_{II}(y) & \text{if } g(y) < 0 \end{cases} \quad (6.23)$$

with sufficiently differentiable functions g , f_I and f_{II} . The derivative of the solution is thus in general discontinuous on the surface

$$S = \{y; g(y) = 0\}.$$

The function $g(y)$ is called a *switching function*.

In order to understand the situations which can occur when the solution of (6.23) meets the surface S in a point y_0 (i.e., $g(y_0) = 0$), we consider the scalar products

$$\begin{aligned} a_I &= \langle \text{grad } g(y_0), f_I(y_0) \rangle \\ a_{II} &= -\langle \text{grad } g(y_0), f_{II}(y_0) \rangle \end{aligned} \quad (6.24)$$

which can be approximated numerically by $a_I \approx g(y_0 + \delta f_I(y_0)) / \delta$ with small enough δ . Since the vector $\text{grad } g(y_0)$ points towards the domain of f_I , the inequality $a_I < 0$ tells us that the flow for f_I is “pushing” against S , while for $a_I > 0$ the flow is “pulling”. The same argument holds for a_{II} and the flow for f_{II} . Therefore, apart from degenerate cases where either a_I or a_{II} vanishes, we can distinguish the following four cases (see Fig. 6.2):

- 1) $a_I > 0, a_{II} < 0$: the flow traverses S from $g < 0$ to $g > 0$.
- 2) $a_I < 0, a_{II} > 0$: the flow traverses S from $g > 0$ to $g < 0$.
- 3) $a_I > 0, a_{II} > 0$: the flow “pulls” on both sides; the solution is not unique; except in the case of an unhappily chosen initial value, this situation would normally not occur.

- 4) $a_I < 0, a_{II} < 0$: here *both* flows push against S ; the solution is trapped in S and the problem no longer has a classical solution.

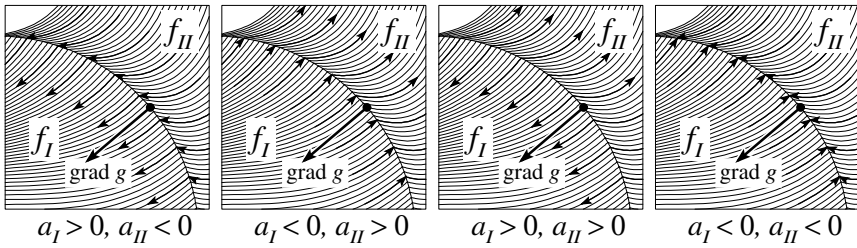


Fig. 6.2. Solutions near the surface of discontinuity

Crossing a discontinuity. The *numerical* computation of a solution crossing a discontinuity (cases 1 and 2) can be performed as follows:

- a) *Ignoring the discontinuity*: apply a variable step size code with local error control (such as DOPRI5) and hope that the step size mechanism would handle the discontinuity appropriately. Consider the example (which represents the flow of the second picture of Fig. 6.2)

$$y' = \begin{cases} x^2 + 2y^2 & \text{if } (x + 0.05)^2 + (y + 0.15)^2 \leq 1 \\ 2x^2 + 3y^2 - 2 & \text{if } (x + 0.05)^2 + (y + 0.15)^2 > 1 \end{cases} \quad (6.25)$$

with initial value $y(0) = 0.3$. The discontinuity for this problem occurs at $x \approx 0.6234$ and the code, applied with $Atol = Rtol = 10^{-5}$, detects the discontinuity fairly well by means of numerous rejected steps (see Fig. 6.3; this figure, however, is much less dramatic than an analogous drawing (see Gear & Østerby 1984) for multistep methods). The numerical solution for $x = 1$ then has an error of $5.9 \cdot 10^{-4}$.

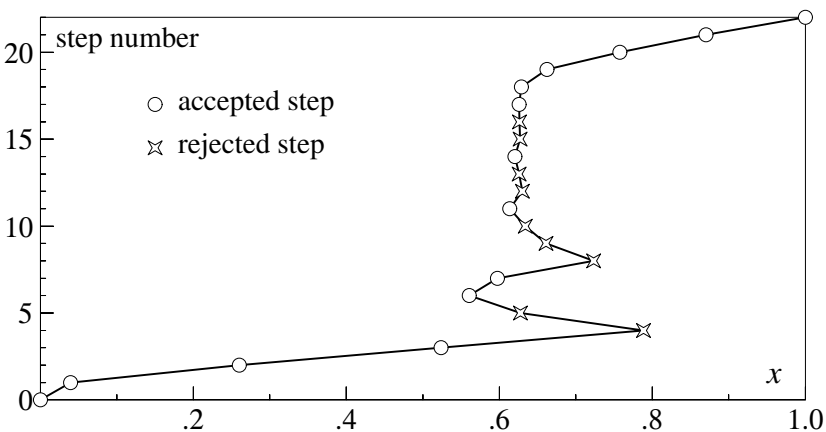


Fig. 6.3. Ignoring the discontinuity at problem (6.23)

- b) *Singularity detecting codes.* Concepts have been developed (Gear & Østerby (1984) for multistep methods, Enright, Jackson, Nørsett & Thomsen (1988) for Runge-Kutta methods) to modify existing codes in such a way that singularities are detected more precisely and handled more appropriately. These concepts are mainly based on the behaviour of the local error estimate compared to the step size.
- c) *Use the switching function:* stop the computation at the surface of discontinuity using Algorithm 6.4 and restart the integration with the new right-hand side. One has to take care that during one integration step only function values of either f_I or f_{II} are used. This algorithm, applied to Example (6.25), uses less than half of the function evaluations as the “ignoring algorithm” and gives an error of $6.6 \cdot 10^{-6}$ at the point $x = 1$. It is thus not only faster, but also much more reliable.

Example 6.5. Coulomb’s law of friction (Coulomb 1785), which states that the force of friction is *independent* of the speed, gives rise to many situations with discontinuous differential equations. Consider the example (see Den Hartog 1930, Reissig 1954, Taubert 1976)

$$y'' + 2Dy' + \mu \operatorname{sign} y' + y = A \cos(\omega x). \quad (6.26)$$

where the Coulomb-force $\mu \operatorname{sign} y'$ is accompanied by a viscosity term Dy' . We fix the parameters as $D = 0.1$, $\mu = 4$, $A = 2$ and $\omega = \pi$, and choose the initial values

$$y(0) = 3, \quad y'(0) = 4. \quad (6.27)$$

Equation (6.26), written in the form (6.23), is

$$\begin{aligned} y' &= v \\ v' &= -0.2v - y + 2 \cos(\pi x) - \begin{cases} 4 & \text{if } v > 0 \\ -4 & \text{if } v < 0. \end{cases} \end{aligned} \quad (6.28)$$

Its solution is plotted in Fig. 6.4.

The initial value (6.27) is in the region $v > 0$ and we follow the solution until it hits the manifold $v = 0$ for the first time. This happens for $x_1 \approx 0.5628$. An investigation of the values

$$a_I = -y(x_1) + 2 \cos(\pi x_1) - 4, \quad a_{II} = y(x_1) - 2 \cos(\pi x_1) - 4 \quad (6.29)$$

shows that $a_I < 0$, $a_{II} > 0$, so that we have to continue the integration into the region $v < 0$. The next intersection of the solution with the manifold of discontinuity is at $x_2 \approx 2.0352$. Here $a_I < 0$, $a_{II} < 0$, so that a classical solution does not exist beyond this point and the solution remains “trapped” in the manifold ($v = 0$, $y = \text{Const} = y(x_2)$) until one of the values a_I or a_{II} changes sign. This happens for a_{II} at the point $x_3 \approx 2.6281$ and we can continue the integration of (6.28) in the region $v < 0$ (see Fig. 6.4). The same situation then repeats periodically.

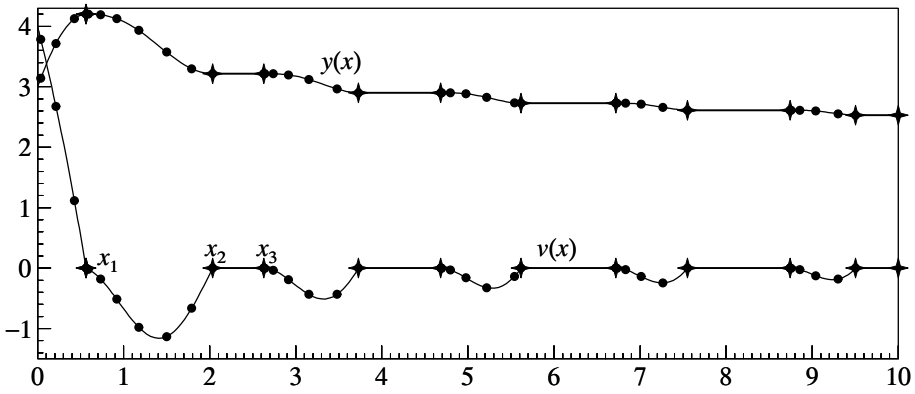


Fig. 6.4. Solutions of (6.28)

Solutions in the manifold. In the case $a_I < 0$, $a_{II} < 0$ the solution of (6.23) can neither be continued along the flow of $y' = f_I(y)$ nor along that of $y' = f_{II}(y)$. However, the physical process, described by the differential equation (6.23), possesses a solution (see Example 6.5). Early papers on this subject studied the convergence of Euler polygons, pushed across the border again and again by the conflicting vector fields (see, e.g., Taubert 1976). Later it became clear that it is much more advantageous to pursue the solution *in* the manifold S , i.e., solve a so-called differential algebraic problem. This approach is advocated by Eich (1992), who attributes the ideas to the thesis of G. Bock, by Eich, Kastner-Maresch & Reich (unpublished manuscript, 1991), and by Stewart (1990). We must decide, however, *which* vector field in S should determine the solution. Several motivations (see Exercises 8 and 9 below) suggest to search this field in the convex hull

$$f(y, \lambda) = (1 - \lambda)f_I(y) + \lambda f_{II}(y), \quad (6.30)$$

of f_I and f_{II} . This coincides, for the special problem (6.23), with Filippov's "generalized solution" (Filippov 1960); but other homotopies may be of interest as well. The value of λ must be chosen in such a way that the solution remains in S . This means that we have to solve the problem

$$y' = f(y, \lambda) \quad (6.31a)$$

$$0 = g(y). \quad (6.31b)$$

Differentiating (6.31b) with respect to time yields

$$0 = \text{grad } g(y)y' = \text{grad } g(y)f(y, \lambda). \quad (6.32)$$

If this relation allows λ to be expressed as a function of y , say as $\lambda = G(y)$, then (6.31a) becomes the ordinary differential equation

$$y' = f(y, G(y)) \quad (6.33)$$

which can be solved by standard integration methods. Obviously, the solution of

(6.33) together with $\lambda = G(y)$ satisfy (6.32) and after integration also (6.31b) (because the initial value satisfies $g(y_0) = 0$).

For the homotopy (6.30) the relation (6.32) becomes

$$(1 - \lambda)a_I(y) - \lambda a_{II}(y) = 0, \quad \text{i.e.,} \quad \lambda = \frac{a_I(y)}{a_I(y) + a_{II}(y)}, \quad (6.34)$$

where $a_I(y)$ and $a_{II}(y)$ are given in (6.24).

Remark . Problem (6.31) is a “differential-algebraic system of index 2” and direct numerical methods are discussed in Chapter VI of Volume II. The instances where a_I or a_{II} change sign can again be computed by using a dense output and Algorithm 6.4.

Numerical Computation of Derivatives with Respect to Initial Values and Parameters

For the efficient computation of boundary value problems by a shooting technique as explained in Section I.15, we need to compute the derivatives of the solutions with respect to (the missing) initial values. Also, if we want to adjust unknown parameters from given data, say by a nonlinear least squares procedure, we have to compute the derivatives of the solutions with respect to parameters in the differential equation.

We shall restrict our discussion to the problem

$$y' = f(x, y, B), \quad y(x_0) = y_0(B) \quad (6.35)$$

where the right-hand side function and the initial values depend on a real parameter B . The generalization to more than one parameter is straightforward. There are several possibilities for computing the derivative $\partial y / \partial B$.

External differentiation. Denote the numerical solution, obtained by a variable step size code with a fixed tolerance, by $y_{Tol}(x_{\text{end}}, x_0, B)$. Then the most simple device is to approximate the derivative by a finite difference

$$\frac{1}{\Delta B} \left(y_{Tol}(x_{\text{end}}, x_0, B + \Delta B) - y_{Tol}(x_{\text{end}}, x_0, B) \right). \quad (6.36)$$

However, due to the error control mechanism with its IF's and THEN's and step rejections, the function $y_{Tol}(x_{\text{end}}, x_0, B)$ is by no means a smooth function of the parameter B . Therefore, the errors of the two numerical results in (6.36) are not correlated, so that the error of (6.36) as an approximation to $\partial y / \partial B(x_{\text{end}}, x_0, B)$ is of size $\mathcal{O}(\text{Tot}/\Delta B) + \mathcal{O}(\Delta B)$, the second term coming from the discretization (6.36). This suggests taking for ΔB something like $\sqrt{\text{Tot}}$, and the error of (6.36) becomes of size $\mathcal{O}(\sqrt{\text{Tot}})$.

Internal differentiation. We know from Section I.14 that $\Psi = \partial y / \partial B$ is the solution of the variational equation

$$\Psi' = \frac{\partial f}{\partial y}(x, y, B)\Psi + \frac{\partial f}{\partial B}(x, y, B), \quad \Psi(x_0) = \frac{\partial y_0}{\partial B}(B). \quad (6.37)$$

Here y is the solution of (6.35). Hence, (6.35) and (6.37) together constitute a differential system for y and Ψ , which can be solved simultaneously by any code. If the partial derivatives $\partial f / \partial y$ and $\partial f / \partial B$ are available analytically, then the error of $\partial y / \partial B$, obtained by this procedure, is obviously of size Tol . This algorithm is equivalent to “internal differentiation” as introduced by Bock (1981).

If $\partial f / \partial y$ and $\partial f / \partial B$ are not available one can approximate them by finite differences so that (6.37) becomes

$$\Psi' = \frac{1}{\Delta B} \left(f(x, y + \Delta B \cdot \Psi, B + \Delta B) - f(x, y, B) \right). \quad (6.38)$$

The solution of (6.38), when inserted into (6.37), gives raise to a defect of size $\mathcal{O}(\Delta B) + \mathcal{O}(eps / \Delta B)$, where eps is the precision of the computer (independent of Tol). By Theorem I.10.2, the difference of the solutions of (6.38) and (6.37) is of the same size. Choosing $\Delta B \approx \sqrt{eps}$ the error of the approximation to $\partial y / \partial B$, obtained by solving (6.35), (6.38), will be of order $Tol + \sqrt{eps}$, so that for $Tol \geq \sqrt{eps}$ the result is as precise as that obtained by integration of (6.37). Observe that external differentiation and the numerical solution of (6.35), (6.38) need about the same number of function evaluations.

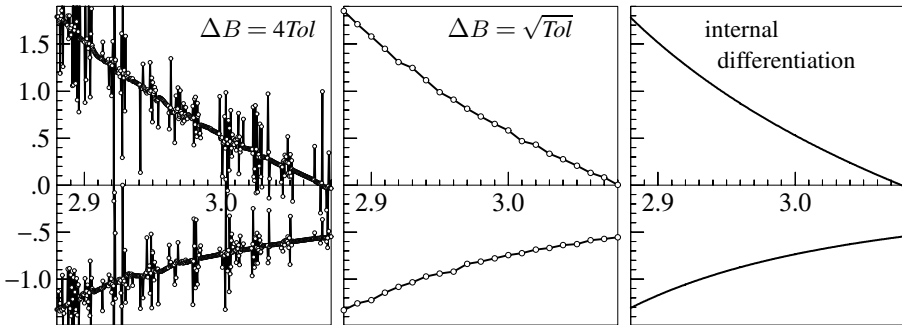


Fig. 6.5. Derivatives of the solution of (6.39) with respect to B

As an example we consider the Brusselator

$$\begin{aligned} y_1' &= 1 + y_1^2 y_2 - (B + 1)y_1 & y_1(0) &= 1.3 \\ y_2' &= B y_1 - y_1^2 y_2 & y_2(0) &= B \end{aligned} \quad (6.39)$$

and compute $\partial y / \partial B$ at $x = 20$ for various B ranging from $B = 2.88$ to $B = 3.08$. We applied the code DOPRI5 with $Atol = Rtol = Tol = 10^{-4}$. The numerical

result is displayed in Fig. 6.5. External differentiation has been applied, once with $\Delta B = \sqrt{\text{Tol}}$ and a second time with $\Delta B = 4\text{Tol}$. This numerical example clearly demonstrates that internal differentiation is to be preferred.

Exercises

1. (Owren & Zennaro 1991, Carnicer 1991). The 4-stage 4th order methods of Section II.1 do not possess a dense output of order 4 (also if the numerical solution y_1 is included as 5th stage). Prove this statement.
2. Consider a Runge-Kutta method of order p and use Richardson extrapolation for step size control. Besides the numerical solution y_0, y_1, y_2 we consider the extrapolated values (see Section II.4)

$$\hat{y}_1 = y_1 + \frac{y_2 - w}{(2^p - 1)2}, \quad \hat{y}_2 = y_2 + \frac{y_2 - w}{2^p - 1}$$

and do quintic polynomial interpolation based on $y_0, f(x_0, y_0), \hat{y}_1, f(x_0 + h, y_1), \hat{y}_2, f(x_0 + 2h, \hat{y}_2)$. Prove that the resulting dense output formula is of order $p^* = \min(5, p + 1)$.

Remark. It is not necessary to evaluate f at \hat{y}_1 .

3. Prove that the conditions (6.19), (6.18) and (5.20) imply (6.20).

Hint. The system (6.19) together with one relation of (6.20) is overdetermined. However, it possesses the solution b_i for $\theta = 1$. Further, the values $b_i c_i$ also solve this system if the right-hand side of (6.19a) is adapted. These properties imply that for $k \in \{4, 5\}$ and for $i \in \{1, 6, \dots, 16\}$

$$\sum_{j=k+1}^{i-1} a_{ij} a_{jk} = \alpha a_{i4} + \beta a_{i5} + \gamma c_i a_{i4} + \delta c_i a_{i5} + \varepsilon \left(\sum_{j=1}^{i-1} a_{ij} c_j^5 - \frac{c_i^6}{6} \right),$$

where the parameters $\alpha, \beta, \gamma, \delta, \varepsilon$ may depend on k .

4. (Butcher). Try your favorite code on the example

$$\begin{aligned} y_1' &= f_1(y_1, y_2), & y_1(0) &= 1 \\ y_2' &= f_2(y_1, y_2), & y_2(0) &= 0 \end{aligned}$$

where f is defined as follows.

If $(|y_1| > |y_2|)$ then

$$f_1 = 0, \quad f_2 = \text{sign}(y_1)$$

Else

$$f_2 = 0, \quad f_1 = -\text{sign}(y_2)$$

End If .

Compute $y_1(8), y_2(8)$. Show that the exact solution is periodic.

5. Do numerical computations for the problem $y' = f(y)$, $y(0) = 1$, $y(3) = ?$ where

$$f(y) = \begin{cases} y^2 & \text{if } 0 \leq y \leq 2 \\ \begin{matrix} \text{a) } 1 \\ \text{b) } 4 \\ \text{c) } -4 + 4y \end{matrix} & \text{if } 2 < y \end{cases}$$

Remark. The correct answer would be (a) 4.5, (b) 12, (c) $\exp(10) + 1$.

6. Consider an s -stage Runge-Kutta method and denote by \tilde{s} the number of distinct c_i . Prove that the order of any continuous extension is $\leq \tilde{s}$.

Hint. Let $q(x)$ be a polynomial of degree \tilde{s} satisfying $q(c_i) = 0$ (for $i = 1, \dots, s$) and investigate the expression $\sum_i b_i(\theta)q(c_i)$.

7. (Step size freeze). Consider the following algorithm for the computation of $\partial y / \partial B$: first compute numerically the solution of (6.35) and denote it by $y_h(x_{\text{end}}, B)$. At the same time memorize all the selected step sizes. This step size sequence is then used to solve (6.35) with B replaced by $B + \Delta B$. The result is denoted by $y_h(x_{\text{end}}, B + \Delta B)$. Then approximate the derivative $\partial y / \partial B$ by

$$\frac{1}{\Delta B} (y_h(x_{\text{end}}, B + \Delta B) - y_h(x_{\text{end}}, B)).$$

Prove that this algorithm is equivalent to the solution of the system (6.35), (6.38), if only the components of y are considered for error control and step size selection.

Remark. For large systems this algorithm needs less storage requirements than internal differentiation, in particular if the derivative with respect to several parameters is computed.

8. (Taubert 1976). Show that for the discontinuous problem (6.23) the Euler polygons converge to Filippov's solution (6.30), (6.31).

Hint. The difference quotient of a piece of the Euler polygon lies in the convex hull of points $f_I(y)$ and $f_{II}(y)$.

Remark. This result can either be interpreted as pleading for myriads of Euler steps, or as a motivation for the homotopy (6.30).

9. Another motivation for formula (6.30): suppose that a small particle of radius ε is transported in a possibly discontinuous flow. Then its movement might be described by the mean of f

$$f_\varepsilon(y) = \int_{B_\varepsilon(y)} f(z) dz / \int_{B_\varepsilon(y)} dz$$

which is continuous in y . Show that the solution of $y'_\varepsilon = f_\varepsilon(y)$ becomes, for $\varepsilon \rightarrow 0$, that of (6.33) and (6.34).

II.7 Implicit Runge-Kutta Methods

It has been traditional to consider only explicit processes
(J.C. Butcher 1964a)

The high speed computing machines make it possible to enjoy
the advantage of intricate methods
(P.C. Hammer & J.W. Hollingsworth 1955)

The first *implicit* RK methods were used by Cauchy (1824) for the sake of — you have guessed correctly — error estimation (Méthodes diverses qui peuvent être employées au Calcul numérique ...; see Exercise 5). Cauchy inserted the mean value theorem into the integral studied in Sections I.8 and II.1,

$$y(x_1) = y(x_0) + \int_{x_0}^{x_1} f(x, y(x)) dx, \quad (7.1)$$

to obtain

$$y_1 = y_0 + hf(x_0 + \theta h, y_0 + \Theta(y_1 - y_0)) \quad (7.2)$$

with $0 \leq \theta, \Theta \leq 1$ (the “ θ -method”). The extreme cases are $\theta = \Theta = 0$ (the explicit Euler method) and $\theta = \Theta = 1$

$$y_1 = y_0 + hf(x_1, y_1), \quad (7.3)$$

which we call the *implicit* or *backward Euler method*.

For the sake of more efficient numerical processes, we apply, as we did in Section II.1, the midpoint rule ($\theta = \Theta = 1/2$) and obtain from (7.2) by setting $k_1 = (y_1 - y_0)/h$:

$$\begin{aligned} k_1 &= f\left(x_0 + \frac{h}{2}, y_0 + \frac{h}{2} k_1\right), \\ y_1 &= y_0 + hk_1. \end{aligned} \quad (7.4)$$

This method is called the *implicit midpoint rule*.

Still another possibility is to approximate (7.1) by the *trapezoidal rule* and to obtain

$$y_1 = y_0 + \frac{h}{2} \left(f(x_0, y_0) + f(x_1, y_1) \right). \quad (7.5)$$

Let us also look at the Radau scheme

$$\begin{aligned} y(x_1) - y(x_0) &= \int_{x_0}^{x_0+h} f(x, y(x)) dx \\ &\approx \frac{h}{4} \left(f(x_0, y_0) + 3f\left(x_0 + \frac{2}{3}h, y\left(x_0 + \frac{2}{3}h\right)\right) \right). \end{aligned}$$

Here we need to approximate $y(x_0 + 2h/3)$. One idea would be the use of quadratic interpolation based on y_0 , y'_0 and $y(x_1)$,

$$y\left(x_0 + \frac{2}{3}h\right) \approx \frac{5}{9}y_0 + \frac{4}{9}y(x_1) + \frac{2}{9}hf(x_0, y_0).$$

The resulting method, given by Hammer & Hollingsworth (1955), is

$$\begin{aligned} k_1 &= f(x_0, y_0) \\ k_2 &= f\left(x_0 + \frac{2}{3}h, y_0 + \frac{h}{3}(k_1 + k_2)\right) \\ y_1 &= y_0 + \frac{h}{4}(k_1 + 3k_2). \end{aligned} \quad (7.6)$$

All these schemes are of the form (1.8) if the summations are extended up to “ s ”.

Definition 7.1. Let b_i , a_{ij} ($i, j = 1, \dots, s$) be real numbers and let c_i be defined by (1.9). The method

$$\begin{aligned} k_i &= f\left(x_0 + c_i h, y_0 + h \sum_{j=1}^s a_{ij} k_j\right) \quad i = 1, \dots, s \\ y_1 &= y_0 + h \sum_{i=1}^s b_i k_i \end{aligned} \quad (7.7)$$

is called an *s-stage Runge-Kutta method*. When $a_{ij} = 0$ for $i \leq j$ we have an explicit (ERK) method. If $a_{ij} = 0$ for $i < j$ and at least one $a_{ii} \neq 0$, we have a *diagonal implicit Runge-Kutta method* (DIRK). If in addition all diagonal elements are identical ($a_{ii} = \gamma$ for $i = 1, \dots, s$), we speak of a *singly diagonal implicit* (SDIRK) method. In all other cases we speak of an *implicit Runge-Kutta method* (IRK).

The tableau of coefficients used above for ERK-methods is obviously extended to include all the other non-zero a_{ij} ’s above the diagonal. For methods (7.3), (7.4) and (7.6) it is given in Table 7.1.

Renewed interest in implicit Runge-Kutta methods arose in connection with *stiff* differential equations (see Volume II).

Table 7.1. Implicit Runge-Kutta methods

			0	0	0
			2/3	1/3	1/3
				1/4	3/4
1	1	1/2	1/2		
	1		1		
Implicit Euler	Implicit midpoint rule	Hammer & Hollingsworth			

Existence of a Numerical Solution

For implicit methods, the k_i 's can no longer be evaluated successively, since (7.7) constitutes a system of implicit equations for the determination of k_i . For DIRK-methods we have a sequence of implicit equations of dimension n for k_1 , then for k_2 , etc. For fully implicit methods $s \cdot n$ unknowns (k_i , $i = 1, \dots, s$; each of dimension n) have to be determined simultaneously, which still increases the difficulty. A natural question is therefore (the reason for which the original version of Butcher (1964a) was returned by the editors): do equations (7.7) possess a solution at all?

Theorem 7.2. *Let $f: \mathbb{R} \times \mathbb{R}^n \rightarrow \mathbb{R}^n$ be continuous and satisfy a Lipschitz condition with constant L (with respect to y). If*

$$h < \frac{1}{L \max_i \sum_j |a_{ij}|} \quad (7.8)$$

there exists a unique solution of (7.7), which can be obtained by iteration. If $f(x, y)$ is p times continuously differentiable, the functions k_i (as functions of h) are also in C^p .

Proof. We prove the existence by iteration (“... on la résoudra facilement par des approximations successives ...”, Cauchy 1824)

$$k_i^{(m+1)} = f\left(x_0 + c_i h, y_0 + h \sum_{j=1}^s a_{ij} k_j^{(m)}\right).$$

We define $K \in \mathbb{R}^{sn}$ as $K = (k_1, \dots, k_s)^T$ and use the norm $\|K\| = \max_i (\|k_i\|)$. Then (7.7) can be written as $K = F(K)$ where

$$F_i(K) = f\left(x_0 + c_i h, y_0 + h \sum_{j=1}^s a_{ij} k_j\right), \quad i = 1, \dots, s.$$

The Lipschitz condition and a repeated use of the triangle inequality then show that

$$\|F(K_1) - F(K_2)\| \leq hL \max_{i=1, \dots, s} \sum_{j=1}^s |a_{ij}| \cdot \|K_1 - K_2\|$$

which from (7.8) is a contraction. The contraction mapping principle then ensures the existence and uniqueness of the solution and the convergence of the fixed-point iteration.

The differentiability result is ensured by the Implicit Function Theorem of classical analysis: (7.7) is written as $\Phi(h, K) = K - F(K) = 0$. The matrix of partial derivatives $\partial\Phi/\partial K$ for $h = 0$ is the identity matrix and therefore the solution of $\Phi(h, K) = 0$, which for $h = 0$ is $k_i = f(x_0, y_0)$, is continuously differentiable in a neighbourhood of $h = 0$. \square

If the assumptions on f in Theorem 7.2 are only satisfied in a neighbourhood of the initial value, then further restrictions on h are needed in order that the argument of f remains in this neighbourhood. Uniqueness is then only of local nature.

The step size restriction (7.8) becomes useless for stiff problems (L large). We return to this question in Vol. II, Sections IV.8 and IV.14.

The definition of *order* is the same as for explicit methods and the order conditions are derived in precisely the same way as in Section II.2.

Example 7.3. Let us study implicit two-stage methods of order 3: the order conditions become (see Theorem 2.1)

$$\begin{aligned} b_1 + b_2 &= 1, & b_1 c_1 + b_2 c_2 &= \frac{1}{2}, & b_1 c_1^2 + b_2 c_2^2 &= \frac{1}{3} \\ b_1(a_{11}c_1 + a_{12}c_2) + b_2(a_{21}c_1 + a_{22}c_2) &= \frac{1}{6}. \end{aligned} \quad (7.9)$$

The first three equations imply the following orthogonality relation (from the theory of Gaussian integration):

$$\int_0^1 (x - c_1)(x - c_2) dx = 0, \quad \text{i.e., } c_2 = \frac{2 - 3c_1}{3 - 6c_1} \quad (c_1 \neq 1/2) \quad (7.10)$$

and

$$b_1 = \frac{c_2 - 1/2}{c_2 - c_1}, \quad b_2 = \frac{c_1 - 1/2}{c_1 - c_2}.$$

In the fourth equation we insert $a_{21} = c_2 - a_{22}$, $a_{11} = c_1 - a_{12}$ and consider a_{12} and c_1 as free parameters. This gives

$$a_{22} = \frac{1/6 - b_1 a_{12}(c_2 - c_1) - c_1/2}{b_2(c_2 - c_1)}. \quad (7.11)$$

For $a_{12} = 0$ we obtain a one-parameter family of DIRK-methods of order 3. An SDIRK-method is obtained if we still require $a_{11} = a_{22}$ (Nørsett 1974b, Crouzeix 1975, see Table 7.2). For order 4 we have 4 additional conditions, with only two free parameters left. Nevertheless there exists a unique solution (see Table 7.3).

Table 7.2. SDIRK method, order 3

γ	γ	0	$\gamma = \frac{3 \pm \sqrt{3}}{6}$
$1 - \gamma$	$1 - 2\gamma$	γ	
	$1/2$	$1/2$	

Table 7.3. Hammer & Hollingsworth, order 4

$\frac{1}{2} - \frac{\sqrt{3}}{6}$	$\frac{1}{4}$	$\frac{1}{4} - \frac{\sqrt{3}}{6}$
$\frac{1}{2} + \frac{\sqrt{3}}{6}$	$\frac{1}{4} + \frac{\sqrt{3}}{6}$	$\frac{1}{4}$
	$1/2$	$1/2$

The Methods of Kuntzmann and Butcher of Order 2s

It is clear that formula (7.4) and the method of Table 7.3 extend the one-point and two-point Gaussian quadrature formulas, respectively. Kuntzmann (1961) (see Ceschino & Kuntzmann 1963, p. 106) and Butcher (1964a) then discovered that for all s there exist IRK-methods of order $2s$. The main tools of proof are the following *simplifying assumptions*

$$\begin{aligned}
 B(p) : \quad & \sum_{i=1}^s b_i c_i^{q-1} = \frac{1}{q} \quad q = 1, \dots, p, \\
 C(\eta) : \quad & \sum_{j=1}^s a_{ij} c_j^{q-1} = \frac{c_i^q}{q} \quad i = 1, \dots, s, \quad q = 1, \dots, \eta, \\
 D(\zeta) : \quad & \sum_{i=1}^s b_i c_i^{q-1} a_{ij} = \frac{b_j}{q} (1 - c_j^q) \quad j = 1, \dots, s, \quad q = 1, \dots, \zeta.
 \end{aligned}$$

Condition $B(p)$ simply means that the quadrature formula (b_i, c_i) is of order p or, equivalently, that the order conditions (2.21) are satisfied for the bushy trees $[\tau, \dots, \tau]$ up to order p .

The assumption $C(\eta)$ implies that the pairs of trees in Fig. 7.1 give identical order conditions for $q \leq \eta$. In contrast to explicit Runge-Kutta methods (see (5.7) and (5.15)) there is no need to require conditions such as $b_2 = 0$ (see (5.8)), because $\sum_j a_{ij} c_j^{q-1} = c_i^q / q$ is valid for all i .

The assumption $D(\zeta)$ is an extension of (1.12). It means that the order condition of the left-hand tree of Fig. 7.2 is implied by those of the two right-hand trees if $q \leq \zeta$.

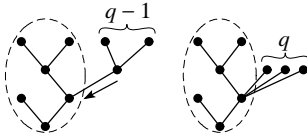


Fig. 7.1. Reduction with $C(q)$

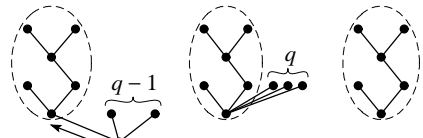


Fig. 7.2. Reduction with $D(q)$

Theorem 7.4 (Butcher 1964a). *If $B(p)$, $C(\eta)$ and $D(\zeta)$ are satisfied with $p \leq 2\eta + 2$ and $p \leq \zeta + \eta + 1$, then the method is of order p .*

Proof. The above reduction by $C(\eta)$ implies that it is sufficient to consider trees $t = [t_1, \dots, t_m]$ of order $\leq p$, where the subtrees t_1, \dots, t_m are either equal to τ or of order $\geq \eta + 1$. Since $p \leq 2\eta + 2$ either all subtrees are equal to τ or there is exactly one subtree different from τ . In the second case the number of τ 's is $\leq \zeta - 1$ by $p \leq \eta + \zeta + 1$ and the reduction by $D(\zeta)$ can be applied. Therefore, after all these reductions, only the bushy trees are left and they are satisfied by $B(p)$. \square

To obtain the formulas of order $2s$, Butcher assumed $B(2s)$ (i.e., the c_i and b_i are the coefficients of the Gaussian quadrature formula) and $C(s)$. This implies $D(s)$ (see Exercise 7) so that Theorem 7.4 can be applied with $p = 2s$, $\eta = s$ and $\zeta = s$. Hence the method, obtained in this way, is of order $2s$. For $s = 3$ and 4 the coefficients are given in Tables 7.4 and 7.5. They can still be expressed by radicals for $s = 5$ and are given in Butcher (1964a), p. 57.

Impressive numerical results from celestial mechanics for these methods were first reported in the thesis of D. Sommer (see Sommer 1965).

Table 7.4. Kuntzmann & Butcher method, order 6

$\frac{1}{2} - \frac{\sqrt{15}}{10}$	$\frac{5}{36}$	$\frac{2}{9} - \frac{\sqrt{15}}{15}$	$\frac{5}{36} - \frac{\sqrt{15}}{30}$
$\frac{1}{2}$	$\frac{5}{36} + \frac{\sqrt{15}}{24}$	$\frac{2}{9}$	$\frac{5}{36} - \frac{\sqrt{15}}{24}$
$\frac{1}{2} + \frac{\sqrt{15}}{10}$	$\frac{5}{36} + \frac{\sqrt{15}}{30}$	$\frac{2}{9} + \frac{\sqrt{15}}{15}$	$\frac{5}{36}$
	$\frac{5}{18}$	$\frac{4}{9}$	$\frac{5}{18}$

Table 7.5. Kuntzmann & Butcher method, order 8

$\frac{1}{2} - \omega_2$	ω_1	$\omega'_1 - \omega_3 + \omega'_4$	$\omega'_1 - \omega_3 - \omega'_4$	$\omega_1 - \omega_5$
$\frac{1}{2} - \omega'_2$	$\omega_1 - \omega'_3 + \omega_4$	ω'_1	$\omega'_1 - \omega'_5$	$\omega_1 - \omega'_3 - \omega_4$
$\frac{1}{2} + \omega'_2$	$\omega_1 + \omega'_3 + \omega_4$	$\omega'_1 + \omega'_5$	ω'_1	$\omega_1 + \omega'_3 - \omega_4$
$\frac{1}{2} + \omega_2$	$\omega_1 + \omega_5$	$\omega'_1 + \omega_3 + \omega'_4$	$\omega'_1 + \omega_3 - \omega'_4$	ω_1
	$2\omega_1$	$2\omega'_1$	$2\omega'_1$	$2\omega_1$
	$\omega_1 = \frac{1}{8} - \frac{\sqrt{30}}{144},$	$\omega'_1 = \frac{1}{8} + \frac{\sqrt{30}}{144},$		
	$\omega_2 = \frac{1}{2} \sqrt{\frac{15 + 2\sqrt{30}}{35}},$	$\omega'_2 = \frac{1}{2} \sqrt{\frac{15 - 2\sqrt{30}}{35}},$		
	$\omega_3 = \omega_2 \left(\frac{1}{6} + \frac{\sqrt{30}}{24} \right),$	$\omega'_3 = \omega'_2 \left(\frac{1}{6} - \frac{\sqrt{30}}{24} \right),$		
	$\omega_4 = \omega_2 \left(\frac{1}{21} + \frac{5\sqrt{30}}{168} \right),$	$\omega'_4 = \omega'_2 \left(\frac{1}{21} - \frac{5\sqrt{30}}{168} \right),$		
	$\omega_5 = \omega_2 - 2\omega_3,$	$\omega'_5 = \omega'_2 - 2\omega'_3.$		

An important interpretation of the assumption $C(\eta)$ is the following:

Lemma 7.5. *The assumption $C(\eta)$ implies that the internal stages*

$$g_i = y_0 + h \sum_{j=1}^s a_{ij} k_j, \quad k_j = f(x_0 + c_j h, g_j) \quad (7.12)$$

satisfy for $i = 1, \dots, s$

$$g_i - y(x_0 + c_i h) = \mathcal{O}(h^{\eta+1}). \quad (7.13)$$

Proof. Because of $C(\eta)$ the exact solution satisfies (Taylor expansion)

$$y(x_0 + c_i h) = y_0 + h \sum_{j=1}^s a_{ij} y'(x_0 + c_j h) + \mathcal{O}(h^{\eta+1}). \quad (7.14)$$

Subtracting (7.14) from (7.12) yields

$$\begin{aligned} g_i - y(x_0 + c_i h) &= h \sum_{j=1}^s a_{ij} \left(f(x_0 + c_j h, g_j) - f(x_0 + c_j h, y(x_0 + c_j h)) \right) \\ &\quad + \mathcal{O}(h^{\eta+1}) \end{aligned}$$

and Lipschitz continuity of f proves (7.13). \square

IRK Methods Based on Lobatto Quadrature

Lobatto quadrature rules (Lobatto 1852, Radau 1880, p. 307) modify the idea of Gaussian quadrature by requiring that the first and the last node coincide with the interval ends, i.e., $c_1 = 0$, $c_s = 1$. These points are easier to handle and, in a step-by-step procedure, can be used twice. The remaining c 's are then adjusted optimally, i.e., as the zeros of the Jacobi orthogonal polynomial $P_{s-2}^{(1,1)}(x)$ or of $P'_{s-1}(x)$ (see e.g., Abramowitz & Stegun 1964, 25.4.32 for the interval $[-1, 1]$) and lead to formulas of order $2s - 2$.

J.C. Butcher (1964a, p. 51, 1964c) then found that Lobatto quadrature rules can be extended to IRK-methods whose coefficient matrix is zero in the first line and the last column. The first and the last stage then become *explicit* and the number of implicit stages reduces to $s - 2$. The methods are characterized by $B(2s - 2)$ and $C(s - 1)$. As in Exercise 7 this implies $D(s - 1)$ so that by Theorem 7.4 the method is of order $2s - 2$. For $s = 3$ and 4, the coefficients are given in Table 7.6.

We shall see in Volume II (Section IV.3, Table 3.1) that these methods, although preferable as concerns the relation between order and implicit stages, are not sufficiently stable for stiff differential equations.

Table 7.6. Butcher's Lobatto formulas of orders 4 and 6

				0	0	0	0	0
0	0	0	0	$\frac{5-\sqrt{5}}{10}$	$\frac{5+\sqrt{5}}{60}$	$\frac{1}{6}$	$\frac{15-7\sqrt{5}}{60}$	0
$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{5+\sqrt{5}}{10}$	$\frac{5-\sqrt{5}}{60}$	$\frac{15+7\sqrt{5}}{60}$	$\frac{1}{6}$	0
1	0	1	0	1	$\frac{1}{6}$	$\frac{5-\sqrt{5}}{12}$	$\frac{5+\sqrt{5}}{12}$	0
	$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$		$\frac{1}{12}$	$\frac{5}{12}$	$\frac{5}{12}$	$\frac{1}{12}$

Collocation Methods

Es ist erstaunlich dass die Methode trotz ihrer Primitivität und der geringen Rechenarbeit in vielen Fällen ... sogar gute Ergebnisse liefert.
(L. Collatz 1951)

Nous allons montrer l'équivalence de notre définition avec la définition traditionnelle de certaines formules de Runge Kutta implicites.
(Guillou & Soulé 1969)

The concept of collocation is old and universal in numerical analysis (see e.g., pp. 28,29,32,181,411,453,483,495 of Collatz 1960, Frazer, Jones & Skan 1937). For ordinary differential equations it consists in searching for a polynomial of degree s whose derivative coincides ("co-locates") at s given points with the vector field of the differential equation (Guillou & Soulé 1969, Wright 1970). Still another approach is to combine Galerkin's method with numerical quadrature (see Hulme 1972).

Definition 7.6. For s a positive integer and c_1, \dots, c_s distinct real numbers (typically between 0 and 1), the corresponding *collocation polynomial* $u(x)$ of degree s is defined by

$$u(x_0) = y_0 \quad (\text{initial value}) \quad (7.15a)$$

$$u'(x_0 + c_i h) = f(x_0 + c_i h, u(x_0 + c_i h)), \quad i = 1, \dots, s. \quad (7.15b)$$

The numerical solution is then given by

$$y_1 = u(x_0 + h). \quad (7.15c)$$

If some of the c_i coincide, the collocation condition (7.15b) will contain higher derivatives and lead to multi-derivative methods (see Section II.13). Accordingly, for the moment, we suppose them all distinct.

Theorem 7.7 (Guillou & Soulé 1969, Wright 1970). *The collocation method (7.15) is equivalent to the s -stage IRK-method (7.7) with coefficients*

$$a_{ij} = \int_0^{c_i} \ell_j(t) dt, \quad b_j = \int_0^1 \ell_j(t) dt \quad i, j = 1, \dots, s, \quad (7.16)$$

where the $\ell_j(t)$ are the Lagrange polynomials

$$\ell_j(t) = \prod_{k \neq j} \frac{(t - c_k)}{(c_j - c_k)}. \quad (7.17)$$

Proof. Put $u'(x_0 + c_i h) = k_i$, so that

$$u'(x_0 + th) = \sum_{j=1}^s k_j \cdot \ell_j(t) \quad (\text{Lagrange}).$$

Then integrate

$$u(x_0 + c_i h) = y_0 + h \int_0^{c_i} u'(x_0 + th) dt \quad (7.18)$$

and insert into (7.15b) together with (7.16). The IRK-method (7.7) then comes out. \square

As a consequence of this result, the existence and uniqueness of the collocation polynomial (for sufficiently small h) follows from Theorem 7.2.

Theorem 7.8. *An implicit Runge-Kutta method with all c_i different and of order at least s is a collocation method iff $C(s)$ is true.*

Proof. $C(s)$ determines the a_{ij} uniquely. We write it as

$$\sum_{j=1}^s a_{ij} p(c_j) = \int_0^{c_i} p(t) dt \quad (7.19)$$

for all polynomials p of degree $\leq s - 1$. The a_{ij} given by (7.16) satisfy this relation, because (7.16) inserted into (7.19) is just the Lagrange interpolation formula. \square

Theorem 7.9. *Let $M(t) = \prod_{i=1}^s (t - c_i)$ and suppose that M is orthogonal to polynomials of degree $r - 1$,*

$$\int_0^1 M(t) t^{q-1} dt = 0, \quad q = 1, \dots, r, \quad (7.20)$$

then method (7.15) has order $p = s + r$.

Proof. The following proof uses the Gröbner & Alekseev Formula, which gives nice insight in the background of the result. An alternative proof is indicated in Exercise 7 below. One can also linearize the equation, apply the *linear* variation-of-constants formula and estimate the error (Guillou & Soulé 1969).

The orthogonality condition (7.20) means that the quadrature formula

$$\int_{x_0}^{x_0+h} g(t) dt = h \sum_{j=1}^s b_j g(x_0 + c_j h) + \text{err}(g) \quad (7.21)$$

is of order $s + r = p$, and its error is bounded by

$$|\text{err}(g)| \leq Ch^{p+1} \cdot \max |g^{(p)}(x)|. \quad (7.22)$$

The principal idea of the proof is now the following: we consider

$$u'(x) = f(x, u(x)) + (u'(x) - f(x, u(x)))$$

as a perturbation of

$$y'(x) = f(x, y(x))$$

and integrate the Gröbner & Alekseev Formula (I.14.18) with the quadrature formula (7.21). Due to (7.15b), the result is identically zero, since at the collocation points the defect is zero. Thus from (7.21) and (7.22)

$$\|y(x_0 + h) - u(x_0 + h)\| = \|\text{err}(g)\| \leq C \cdot h^{p+1} \cdot \max_{x_0 \leq t \leq x_0+h} \|g^{(p)}(t)\|, \quad (7.23)$$

where

$$g(t) = \frac{\partial y}{\partial y_0}(x, t, u(t)) \cdot (u'(t) - f(t, u(t))),$$

and we see that the local error behaves like $\mathcal{O}(h^{p+1})$.

There remains, however, a small technical detail: to show that the derivatives of $g(t)$ remain bounded for $h \rightarrow 0$. These derivatives contain partial derivatives of $f(t, y)$ and derivatives of $u(t)$. We shall see in the next theorem that these derivatives remain bounded for $h \rightarrow 0$. \square

Theorem 7.10. *The collocation polynomial $u(x)$ gives rise to a continuous IRK method of order s , i.e., for all $x_0 \leq x \leq x_0 + h$ we have*

$$\|y(x) - u(x)\| \leq C \cdot h^{s+1}. \quad (7.24)$$

Moreover, for the derivatives of $u(x)$ we have

$$\|y^{(k)}(x) - u^{(k)}(x)\| \leq C \cdot h^{s+1-k} \quad k = 0, \dots, s. \quad (7.25)$$

Proof. The exact solution $y(x)$ satisfies the collocation condition everywhere, hence *also* at the points $x_0 + c_i h$. So, in exactly the same way as in the proof

of Theorem 7.7, we apply the Lagrange interpolation formula to $y'(x)$:

$$y'(x_0 + th) = \sum_{j=1}^s f(x_0 + c_j h, y(x_0 + c_j h)) \ell_j(t) + h^s R(t, h)$$

where $R(t, h)$ is a smooth function of both variables. Integration and subtraction from (7.18) gives

$$y(x_0 + th) - u(x_0 + th) = h \sum_{j=1}^s \Delta f_j \cdot \int_0^t \ell_j(\tau) d\tau + h^{s+1} \int_0^t R(\tau, h) d\tau, \quad (7.26)$$

where

$$\Delta f_j = f(x_0 + c_j h, y(x_0 + c_j h)) - f(x_0 + c_j h, u(x_0 + c_j h)).$$

The k th derivative of (7.26) with respect to t is

$$h^k \left(y^{(k)}(x_0 + th) - u^{(k)}(x_0 + th) \right) = h \sum_{j=1}^s \Delta f_j \cdot \ell_j^{(k-1)}(t) + h^{s+1} \frac{\partial^{k-1} R}{\partial t^{k-1}}(t, h),$$

so that the result follows from the boundedness of the derivatives of $R(t, h)$ and from $\Delta f_j = \mathcal{O}(h^{s+1})$ which is a consequence of Lemma 7.5. \square

Remark. Only *some* IRK methods are collocation methods. An extension of the collocation idea (“Perturbed Collocation”, see Nørsett & Wanner 1981) applies to *all* IRK methods.

Exercises

1. Compute the one-point collocation method ($s = 1$) with $c_i = \theta$ and compare with (7.2). Determine its order in dependence of θ .
2. Compute all collocation methods with $s = 2$ of order 2 in dependence of c_1 and c_2 .
3. Specify in the method of Exercise 2 $c_1 = 1/3$, $c_2 = 1$ as well as $c_1 = 0$, $c_2 = 2/3$. Determine the orders of the obtained methods and explain.
4. Interpret the implicit midpoint rule (7.4) and the explicit Euler method as collocation methods. Is method (7.5) a collocation method? Method (7.6)?
5. (Cauchy 1824). Find from equation (7.2) conditions for the function $f(x, y)$ such that for scalar differential equations

$$y_1(\text{explicit Euler}) \geq y(x_1) \geq y_1(\text{implicit Euler}).$$

Compute five steps with $h = 0.2$ with both methods to obtain upper and lower bounds for $y(1)$, the solution of

$$y' = \cos \frac{x+y}{5}, \quad y(0) = 0.$$

Cauchy's result: $0.9659 \leq y(1) \leq 0.9810$. For one single step with $h = 1$ he obtained $0.926 \leq y(1) \leq 1$.

Compute the exact solution by elementary integration.

6. Determine the orders of the methods of Table 7.7. Generalize to arbitrary s (Ehle 1968).

Hint. Use Theorems 7.8 and 7.9.

Table 7.7. Methods of Ehle

Radau IIA, order 5				Lobatto IIIA, order 4			
$\frac{4-\sqrt{6}}{10}$	$\frac{88-7\sqrt{6}}{360}$	$\frac{296-169\sqrt{6}}{1800}$	$\frac{-2+3\sqrt{6}}{225}$	0	0	0	0
$\frac{4+\sqrt{6}}{10}$	$\frac{296+169\sqrt{6}}{1800}$	$\frac{88+7\sqrt{6}}{360}$	$\frac{-2-3\sqrt{6}}{225}$	$\frac{1}{2}$	$\frac{5}{24}$	$\frac{1}{3}$	$-\frac{1}{24}$
1	$\frac{16-\sqrt{6}}{36}$	$\frac{16+\sqrt{6}}{36}$	$\frac{1}{9}$	1	$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$
	$\frac{16-\sqrt{6}}{36}$	$\frac{16+\sqrt{6}}{36}$	$\frac{1}{9}$		$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$

7. (Butcher 1964a). Give an algebraic proof of Theorem 7.9.

Hint. From Theorem 7.8 we have $C(s)$.

Next the condition $B(p)$ with $p = s + r$ (theory of Gaussian quadrature formulas) implies $D(r)$. To see this, multiply the two vectors $u_j = \sum_i b_i c_i^{q-1} a_{ij}$ and $v_j = b_j(1 - c_j^q)/q$ ($j = 1, \dots, s$) by the Vandermonde matrix

$$V = \begin{pmatrix} 1 & 1 & \dots & 1 \\ c_1 & c_2 & \dots & c_s \\ \vdots & \vdots & \dots & \vdots \\ c_1^{s-1} & c_2^{s-1} & \dots & c_s^{s-1} \end{pmatrix}.$$

Finally apply Theorem 7.4.

II.8 Asymptotic Expansion of the Global Error

Mein Verzicht auf das Restglied war leichtsinnig . . .
(W. Romberg 1979)

Our next goal will be to perfect Richardson's extrapolation method (see Section II.4) by doing *repeated* extrapolation and eliminating more and more terms Ch^{p+k} of the error. A sound theoretical basis for this procedure is given by the study of the asymptotic behaviour of the global error. For problems of the type $y' = f(x)$, which lead to integration, the answer is given by the Euler-Maclaurin formula and has been exploited by Romberg (1955) and his successors. The first rigorous treatments for differential equations are due to Henrici (1962) and Gragg (1964) (see also Stetter 1973). We shall follow here the successive elimination of the error terms given by Hairer & Lubich (1984), which also generalizes to multistep methods.

Suppose we have a one-step method which we write, in Henrici's notation, as

$$y_{n+1} = y_n + h\Phi(x_n, y_n, h). \quad (8.1)$$

If the method is of order p , it possesses at each point of the solution $y(x)$ a *local error* of the form

$$\begin{aligned} y(x+h) - y(x) - h\Phi(x, y(x), h) = \\ d_{p+1}(x)h^{p+1} + \dots + d_{N+1}(x)h^{N+1} + \mathcal{O}(h^{N+2}) \end{aligned} \quad (8.2)$$

whenever the differential equation is sufficiently differentiable. For Runge-Kutta methods these error terms were computed in Section II.2 (see also Theorem 3.2).

The Global Error

Let us now set $y_n =: y_h(x)$ for the numerical solution at $x = x_0 + nh$. We then know from Theorem 3.6 that the global error behaves like h^p . We shall search for a function $e_p(x)$ such that

$$y(x) - y_h(x) = e_p(x)h^p + o(h^p). \quad (8.3)$$

The idea is to consider

$$y_h(x) + e_p(x)h^p =: \hat{y}_h(x) \quad (8.4a)$$

as the numerical solution of a new method

$$\widehat{y}_{n+1} = \widehat{y}_n + h\widehat{\Phi}(x_n, \widehat{y}_n, h). \quad (8.4b)$$

By comparison with (8.1), we see that the increment function for the new method is

$$\widehat{\Phi}(x, \widehat{y}, h) = \Phi(x, \widehat{y} - e_p(x)h^p, h) + (e_p(x+h) - e_p(x))h^{p-1}. \quad (8.5)$$

Our task is to find a function $e_p(x)$, with $e_p(x_0) = 0$, such that the method with increment function $\widehat{\Phi}$ is of order $p+1$.

Expanding the local error of the one-step method $\widehat{\Phi}$ into powers of h we obtain

$$\begin{aligned} y(x+h) - y(x) - h\widehat{\Phi}(x, y(x), h) \\ = \left(d_{p+1}(x) + \frac{\partial f}{\partial y}(x, y(x))e_p(x) - e'_p(x) \right) h^{p+1} + \mathcal{O}(h^{p+2}) \end{aligned} \quad (8.6)$$

where we have used

$$\frac{\partial \Phi}{\partial y}(x, y, 0) = \frac{\partial f}{\partial y}(x, y). \quad (8.7)$$

The term in h^{p+1} vanishes if $e_p(x)$ is defined as the solution of

$$e'_p(x) = \frac{\partial f}{\partial y}(x, y(x))e_p(x) + d_{p+1}(x), \quad e_p(x_0) = 0. \quad (8.8)$$

By Theorem 3.6, applied to the method $\widehat{\Phi}$, we now have

$$y(x) - y_h(x) = e_p(x)h^p + \mathcal{O}(h^{p+1}) \quad (8.9)$$

and the first term of the desired asymptotic expansion has been determined.

We now repeat the procedure with the method with increment function $\widehat{\Phi}$. It is of order $p+1$ and again satisfies condition (8.7). The final result of this procedure is the following

Theorem 8.1 (Gragg 1964). *Suppose that a given method with sufficiently smooth increment function Φ satisfies the consistency condition $\Phi(x, y, 0) = f(x, y)$ and possesses an expansion (8.2) for the local error. Then the global error has an asymptotic expansion of the form*

$$y(x) - y_h(x) = e_p(x)h^p + \dots + e_N(x)h^N + E_h(x)h^{N+1} \quad (8.10)$$

where the $e_j(x)$ are solutions of inhomogeneous differential equations of the form (8.8) with $e_j(x_0) = 0$ and $E_h(x)$ is bounded for $x_0 \leq x \leq x_{\text{end}}$ and $0 \leq h \leq h_0$. \square

The differentiability properties of the $e_j(x)$ depend on those of f and Φ (see (8.8) and (8.2)). The expansion (8.10) will be the theoretical basis for all discussions of extrapolation methods.

Examples. 1. For the equation $y' = y$ and Euler's method we have with $h = 1/n$ and $x = 1$, using the binomial theorem,

$$y_h(1) = \left(1 + \frac{1}{n}\right)^n = 1 + 1 + \left(1 - \frac{1}{n}\right) \frac{1}{2!} + \left(1 - \frac{1}{n}\right) \left(1 - \frac{2}{n}\right) \frac{1}{3!} + \dots$$

By multiplying out, this gives

$$y(1) - y_h(1) = - \sum_{i=1}^{\infty} h^i \sum_{j=1}^{\infty} \frac{S_{i+j}^{(j)}}{(i+j)!} = 1.359h - 1.246h^2 \pm \dots$$

where the $S_i^{(j)}$ are the Stirling numbers of the first kind (1730, see Abramowitz & Stegun 1964, Section 24.1.3). This is, of course, the Taylor series for the function

$$e - (1+h)^{1/h} = e - \exp\left(1 - \frac{h}{2} + \frac{h^2}{3} \pm \dots\right) = e\left(\frac{1}{2}h - \frac{11}{24}h^2 + \frac{7}{16}h^3 \pm \dots\right)$$

with *convergence radius* $r = 1$.

2. For the differential equation $y' = f(x)$ and the trapezoidal rule (7.5), the expansion (8.10) becomes

$$\int_0^1 f(x) dx - y_h(1) = - \sum_{k=1}^N \frac{h^{2k}}{(2k)!} B_{2k} \left(f^{(2k-1)}(1) - f^{(2k-1)}(0) \right) + \mathcal{O}(h^{2N+1}),$$

the well known Euler-Maclaurin formula (1736). For $N \rightarrow \infty$, the series will usually diverge, due to the fast growth of the Bernoulli numbers for large k . It may, however, be useful for small values of N and we call it an *asymptotic expansion* (Poincaré 1893).

Variable h

Theorem 8.1 is not only valid for equal step sizes. A reasonable assumption for the case of variable step sizes is the existence of a function $\tau(x) > 0$ such that the step sizes depend as

$$x_{n+1} - x_n = \tau(x_n) h \quad (8.11)$$

on a parameter h . Then the local error expansion (8.2) becomes

$$y(x + \tau(x)h) - y(x) - h\tau(x)\Phi(x, y(x), \tau(x)h) = d_{p+1}(x)\tau^{p+1}(x)h^{p+1} + \dots$$

and instead of (8.5) we have

$$\widehat{\Phi}(x, \widehat{y}, \tau(x)h) = \Phi(x, \widehat{y} - e_p(x)h^p, \tau(x)h) + \frac{h^p}{h\tau(x)} \left(e_p(x + \tau(x)h) - e_p(x) \right).$$

With this the local error expansion for the new method becomes, instead of (8.6),

$$y(x + \tau(x)h) - y(x) - h\tau(x)\widehat{\Phi}(x, y(x), \tau(x)h)$$

$$= \tau(x) \left(d_{p+1}(x) \tau^p(x) + \frac{\partial f}{\partial y}(x, y(x)) e_p(x) - e'_p(x) \right) h^{p+1} + \mathcal{O}(h^{p+2})$$

and the proof of Theorem 8.1 generalizes with slight modifications.

Negative h

The most important extrapolation algorithms will use asymptotic expansions with *even* powers of h . In order to provide a theoretical basis for these methods, we need to explain the meaning of $y_h(x)$ for h *negative*.

Motivation. We write (8.1) as

$$y_h(x+h) = y_h(x) + h\Phi(x, y_h(x), h) \quad (8.1')$$

and replace h by $-h$ to obtain

$$y_{-h}(x-h) = y_{-h}(x) - h\Phi(x, y_{-h}(x), -h).$$

Next we replace x by $x+h$ which gives

$$y_{-h}(x) = y_{-h}(x+h) - h\Phi(x+h, y_{-h}(x+h), -h). \quad (8.12)$$

This is an implicit equation for $y_{-h}(x+h)$, which possesses a unique solution for sufficiently small h (by the implicit function theorem). We write this solution in the form

$$y_{-h}(x+h) = y_{-h}(x) + h\Phi^*(x, y_{-h}(x), h). \quad (8.13)$$

The comparison of (8.12) and (8.13) (with $A = y_{-h}(x+h)$, $B = y_{-h}(x)$) leads us to the following definition.

Definition 8.2. Let $\Phi(x, y, h)$ be the increment function of a method. Then we define the increment function $\Phi^*(x, y, h)$ of the *adjoint method* by the pair of formulas

$$\begin{aligned} B &= A - h\Phi(x+h, A, -h) \\ A &= B + h\Phi^*(x, B, h). \end{aligned} \quad (8.14)$$

Example. The adjoint method of explicit Euler is implicit Euler.

Theorem 8.3. Let Φ be the Runge-Kutta method (7.7) with coefficients a_{ij} , b_j , c_i ($i, j = 1, \dots, s$). Then the adjoint method Φ^* is equivalent to a Runge-Kutta method with s stages and with coefficients

$$\begin{aligned} c_i^* &= 1 - c_{s+1-i} \\ a_{ij}^* &= b_{s+1-j} - a_{s+1-i, s+1-j} \\ b_j^* &= b_{s+1-j}. \end{aligned}$$

Proof. The formulas (8.14) indicate that for the definition of the adjoint method we have, starting from (7.7), to exchange $y_0 \leftrightarrow y_1$, $h \leftrightarrow -h$ and replace $x_0 \rightarrow x_0 + h$. This then leads to

$$k_i = f\left(x_0 + (1 - c_i)h, y_0 + h \sum_{j=1}^s (b_j - a_{ij})k_j\right)$$

$$y_1 = y_0 + h \sum_{j=1}^s b_j k_j.$$

In order to preserve the usual natural ordering of c_1, \dots, c_s , we also permute the k_i -values and replace all indices i by $s + 1 - i$. \square

Properties of the Adjoint Method

Theorem 8.4. $\Phi^{**} = \Phi$.

Proof. This property, which is the reason for the name “adjoint”, is seen by replacing $h \rightarrow -h$ and then $x \rightarrow x + h$, $B \rightarrow A$, $A \rightarrow B$ in (8.14). \square

Theorem 8.5. *The adjoint method has the same order as the original method. Its principal error term is the error term of the first method multiplied by $(-1)^p$.*

Proof. We replace h by $-h$ in (8.2), then $x \rightarrow x + h$ and rearrange the terms. This gives (using $d_{p+1}(x + h) = d_{p+1}(x) + \mathcal{O}(h)$)

$$y(x) + d_{p+1}(x)h^{p+1}(-1)^p + \mathcal{O}(h^{p+2})$$

$$= y(x + h) - h\Phi(x + h, y(x + h), -h).$$

Here we let B be the left-hand side of this identity, $A = y(x + h)$, and use (8.14). This leads to

$$y(x + h) = y(x) + d_{p+1}(x)h^{p+1}(-1)^p + h\Phi^*(x, y(x), h) + \mathcal{O}(h^{p+2}),$$

which expresses the statement of the theorem. \square

Theorem 8.6. *The adjoint method has exactly the same asymptotic expansion (8.10) as the original method, with h replaced by $-h$.*

Proof. We repeat the procedure which led to the proof of Theorem 8.1, with h negative. The first separated term corresponding to (8.9) will be

$$y(x) - y_{-h}(x) = e_p(x)(-h)^p + \mathcal{O}(h^{p+1}). \quad (8.9')$$

This is true because the solution of (8.8) with initial value $e_p(x_0) = 0$ has the same sign change as the inhomogeneity $d_{p+1}(x)$. This settles the first term. To continue, we prove that the transformation (8.4b) commutes with the adjunction operation, i.e., that

$$(\widehat{\Phi})^* = (\Phi^*)^\widehat{}. \quad (8.15)$$

In order to prove (8.15), we obtain from (8.4a) and the definition of $\widehat{\Phi}$

$$y_h(x+h) + e_p(x+h)h^p = y_h(x) + e_p(x)h^p + h\widehat{\Phi}(x, y_h(x) + e_p(x)h^p, h).$$

Here again, we substitute $h \rightarrow -h$ followed by $x \rightarrow x+h$. Finally, we apply (8.14) with $B = y_{-h}(x) + e_p(x)(-h)^p$ and $A = y_{-h}(x+h) + e_p(x+h)(-h)^p$ to obtain

$$\begin{aligned} y_{-h}(x+h) + e_p(x+h)(-h)^p \\ = y_{-h}(x) + e_p(x)(-h)^p + h(\widehat{\Phi})^*(x, y_{-h}(x) + e_p(x)(-h)^p, h). \end{aligned} \quad (8.16)$$

On the other hand, if we perform the transformation (see Theorem 8.5)

$$\widehat{y}_{-h}(x) = y_{-h}(x) + e_p(x)(-h)^p \quad (8.4')$$

and insert this into (8.13), we obtain (8.16) again, but this time with $(\Phi^*)^\widehat{}$ instead of $(\widehat{\Phi})^*$. This proves (8.15). \square

Symmetric Methods

Definition 8.7. A method is *symmetric* if $\Phi = \Phi^*$.

Example. The trapezoidal rule (7.5) and the implicit mid-point rule (7.4) are symmetric: the exchanges $y_1 \leftrightarrow y_0$, $h \leftrightarrow -h$ and $x_0 \leftrightarrow x_0 + h$ leave these methods invariant. The following two theorems (Wanner 1973) characterize symmetric IRK methods.

Theorem 8.8. *If*

$$a_{s+1-i, s+1-j} + a_{ij} = b_{s+1-j} = b_j, \quad i, j = 1, \dots, s, \quad (8.17)$$

then the corresponding Runge-Kutta method is symmetric. Moreover, if the b_i are nonzero and the c_i distinct and ordered as $c_1 < c_2 < \dots < c_s$, then condition (8.17) is also necessary for symmetry.

Proof. The sufficiency of (8.17) follows from Theorem 8.3. The condition $c_i = 1 - c_{s+1-i}$ can be verified by adding up (8.17) for $j = 1, \dots, s$.

Symmetry implies that the original method (with coefficients c_i, a_{ij}, b_j) and the adjoint method (c_i^*, a_{ij}^*, b_j^*) give identical numerical results. If we apply both methods to $y' = f(x)$ we obtain

$$\sum_{i=1}^s b_i f(c_i) = \sum_{i=1}^s b_i^* f(c_i^*)$$

for all $f(x)$. Our assumption on b_i and c_i thus yields

$$b_i^* = b_i, \quad c_i^* = c_i \quad \text{for all } i.$$

We next apply both methods to $y_1' = f(x)$, $y_2' = x^q y_1$ and obtain

$$\sum_{i,j=1}^s b_i c_i^q a_{ij} f(c_j) = \sum_{i,j=1}^s b_i^* c_i^{*q} a_{ij}^* f(c_j^*).$$

This implies $\sum_i b_i c_i^q a_{ij} = \sum_i b_i c_i^q a_{ij}^*$ for $q = 0, 1, \dots$ and hence also $a_{ij}^* = a_{ij}$ for all i, j . \square

Theorem 8.9. *A collocation method based on symmetrically distributed collocation points is symmetric.*

Proof. If $c_i = 1 - c_{s+1-i}$, the Lagrange polynomials satisfy $\ell_i(t) = \ell_{s+1-i}(1-t)$. Condition (8.17) is then an easy consequence of (7.19). \square

The following important property of symmetric methods, known intuitively for many years, now follows from the above results.

Theorem 8.10. *If in addition to the assumptions of Theorem 8.1 the underlying method is symmetric, then the asymptotic expansion (8.10) contains only even powers of h :*

$$y(x) - y_h(x) = e_{2q}(x)h^{2q} + e_{2q+2}(x)h^{2q+2} + \dots \quad (8.18)$$

with $e_{2j}(x_0) = 0$.

Proof. If $\Phi^* = \Phi$, we have $y_{-h}(x) = y_h(x)$ from (8.13) and the result follows from Theorem 8.6. \square

Exercises

1. Assume the one-step method (8.1) to be of order $p \geq 2$ and in addition to $\Phi(x, y, 0) = f(x, y)$ assume

$$\frac{\partial \Phi}{\partial h}(x, y, 0) = \frac{1}{2} \left(\frac{\partial f}{\partial x}(x, y) + \frac{\partial f}{\partial y}(x, y) \cdot f(x, y) \right). \quad (8.19)$$

Show that the principal local error term of the method $\hat{\Phi}$ defined in (8.5) is then given by

$$\hat{d}_{p+2}(x) = d_{p+2}(x) - \frac{1}{2} \frac{\partial f}{\partial y}(x, y(x)) d_{p+1}(x) - \frac{1}{2} d'_{p+1}(x).$$

Verify that (8.19) is satisfied for all RK-methods of order ≥ 2 .

2. Consider the second order method

0		
1		1
	1/2	1/2

applied to the problem $y' = y$, $y(0) = 1$. Show that

$$d_3(x) = \frac{1}{6} e^x, \quad d_4(x) = \frac{1}{24} e^x, \quad e_2(x) = \frac{1}{6} x e^x, \quad \hat{d}_4(x) = -\frac{1}{8} e^x.$$

3. Consider the second order method

0			
1/2		1/2	
1	0	1	
	1/4	1/2	1/4

Show that for this method

$$d_3(x) = \frac{1}{24} \left(F(t_{32})(y(x)) - \frac{1}{2} F(t_{31})(y(x)) \right)$$

$$d_4(x) = \frac{1}{24} \left(F(t_{44})(y(x)) + \frac{1}{4} F(t_{43})(y(x)) - \frac{1}{4} F(t_{41})(y(x)) \right)$$

in the notation of Table 2.2. Show that this implies

$$\hat{d}_4(x) = 0 \quad \text{and} \quad e_3(x) = 0,$$

so that one step of Richardson extrapolation increases the order of the method by two. Find a connection between this method and the GBS-algorithm of Section II.9.

4. Discuss the symmetry of the IRK methods of Section II.7.

II.9 Extrapolation Methods

The following method of approximation may or may not be new, but as I believe it to be of practical importance . . .

(S.A. Corey 1906)

The h^2 -extrapolation was discovered by a hint from theory followed by arithmetical experiments, which gave pleasing results.

(L.F. Richardson 1927)

Extrapolation constitutes a powerful means . . .

(R. Bulirsch & J. Stoer 1966)

Extrapolation does not appear to be a particularly effective way . . . , our tests raise the question as to whether there is any point to pursuing it as a separate method.

(L.F. Shampine & L.S. Baca 1986)

Definition of the Method

Let $y' = f(x, y)$, $y(x_0) = y_0$ be a given differential system and $H > 0$ a basic step size. We choose a sequence of positive integers

$$n_1 < n_2 < n_3 < \dots \quad (9.1)$$

and define the corresponding step sizes $h_1 > h_2 > h_3 > \dots$ by $h_i = H/n_i$. We then choose a numerical method of order p and compute the numerical results of our initial value problem by performing n_i steps with step size h_i to obtain

$$y_{h_i}(x_0 + H) =: T_{i,1} \quad (9.2)$$

(the letter “ T ” stands historically for “trapezoidal rule”). We then eliminate as many terms as possible from the asymptotic expansion (8.10) by computing the interpolation polynomial

$$p(h) = \hat{y} - e_p h^p - e_{p+1} h^{p+1} - \dots - e_{p+k-2} h^{p+k-2} \quad (9.3)$$

such that

$$p(h_i) = T_{i,1} \quad i = j, j-1, \dots, j-k+1. \quad (9.4)$$

Finally we “*extrapolate to the limit*” $h \rightarrow 0$ and use

$$p(0) = \hat{y} =: T_{j,k}$$

as numerical result. Conditions (9.4) consist of k linear equations for the k unknowns $\hat{y}, e_p, \dots, e_{p+k-2}$.

Example. For $k = 2$, $n_1 = 1$, $n_2 = 2$ the above definition is identical to Richardson’s extrapolation discussed in Section II.4.

Theorem 9.1. *The value $T_{j,k}$ represents a numerical method of order $p+k-1$.*

Proof. We compare (9.4) and (9.3) with the asymptotic expansion (8.10) which we write in the form (with $N = p+k-1$)

$$T_{i,1} = y(x_0+H) - e_p(x_0+H)h_i^p - \dots - e_{p+k-2}(x_0+H)h_i^{p+k-2} - \Delta_i, \quad (9.4')$$

where

$$\Delta_i = e_{p+k-1}(x_0+H)h_i^{p+k-1} + E_{h_i}(x_0+H)h_i^{p+k} = \mathcal{O}(H^{p+k})$$

because $e_{p+k-1}(x_0) = 0$ and $h_i \leq H$. This is a linear system for the unknowns $y(x_0+H)$, $H^p e_p(x_0+H)$, \dots , $H^{p+k-2} e_{p+k-2}(x_0+H)$ with the Vandermonde-like matrix

$$A = \begin{pmatrix} 1 & \frac{1}{n_j^p} & \dots & \frac{1}{n_j^{p+k-2}} \\ \vdots & \vdots & & \vdots \\ 1 & \frac{1}{n_{j-k+1}^p} & \dots & \frac{1}{n_{j-k+1}^{p+k-2}} \end{pmatrix}.$$

It is the same as (9.4), just with the right-hand side perturbed by the $\mathcal{O}(H^{p+k})$ -terms Δ_i . The matrix A is invertible (see Exercise 6). Therefore by subtraction we obtain

$$|y(x_0+H) - \hat{y}| \leq \|A^{-1}\|_\infty \cdot \max |\Delta_i| = \mathcal{O}(H^{p+k}). \quad \square$$

Remark. The case $p=1$ (as well as $p=2$ with expansions in h^2) can also be treated by interpreting the difference $y(x_0+H) - \hat{y}$ as an interpolation error (see (9.21)).

A great advantage of the method is that it provides a complete table of numerical results

$$\begin{array}{ccccccc} T_{11} & & & & & & \\ T_{21} & T_{22} & & & & & \\ T_{31} & T_{32} & T_{33} & & & & \\ T_{41} & T_{42} & T_{43} & T_{44} & & & \\ \dots & \dots & \dots & \dots & \dots & & \end{array} \quad (9.5)$$

which form a sequence of embedded methods and allow easy estimates of the local error and strategies for variable order. Several step-number sequences are in use for (9.1):

The “Romberg sequence” (Romberg 1955):

$$1, 2, 4, 8, 16, 32, 64, 128, 256, 512, \dots \quad (9.6)$$

The “*Bulirsch sequence*” (see also Romberg 1955):

$$1, 2, 3, 4, 6, 8, 12, 16, 24, 32, \dots \quad (9.7)$$

alternating powers of 2 with 1.5 times 2^k . This sequence needs fewer function evaluations for higher orders than the previous one and became prominent through the success of the “*Gragg-Bulirsch-Stoer algorithm*” (Bulirsch & Stoer 1966).

The above sequences have the property that for integration problems $y' = f(x)$ many function values can be saved and re-used for smaller h_i . Further, $\liminf(n_{i+1}/n_i)$ remains bounded away from 1 (“*Toeplitz condition*”) which allows convergence proofs for $j = k \rightarrow \infty$ (Bauer, Rutishauser & Stiefel 1963). However, if we work with differential equations and with fixed or bounded order, the most economic sequence is the “*harmonic sequence*” (Deuffhard 1983)

$$1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \dots \quad (9.8)$$

The Aitken - Neville Algorithm

For the case $p = 1$, (9.3) and (9.4) become a classical interpolation problem and we can compute the values of $T_{j,k}$ economically by the use of classical methods. Since we need only the values of the interpolation polynomials at the point $h = 0$, the most economical algorithm is that of “*Aitken - Neville*” (Aitken 1932, Neville 1934, based on ideas of Jordan 1928) which leads to

$$T_{j,k+1} = T_{j,k} + \frac{T_{j,k} - T_{j-1,k}}{(n_j/n_{j-k}) - 1}. \quad (9.9)$$

If the basic method used is *symmetric*, we know that the underlying asymptotic expansion is in powers of h^2 (Theorem 8.9), and each extrapolation eliminates *two* powers of h . We may thus simply replace in (9.3) h by h^2 and for $p = 2$ (i.e., $q = 1$ in (8.18)) also use the Aitken - Neville algorithm with this modification. This leads to

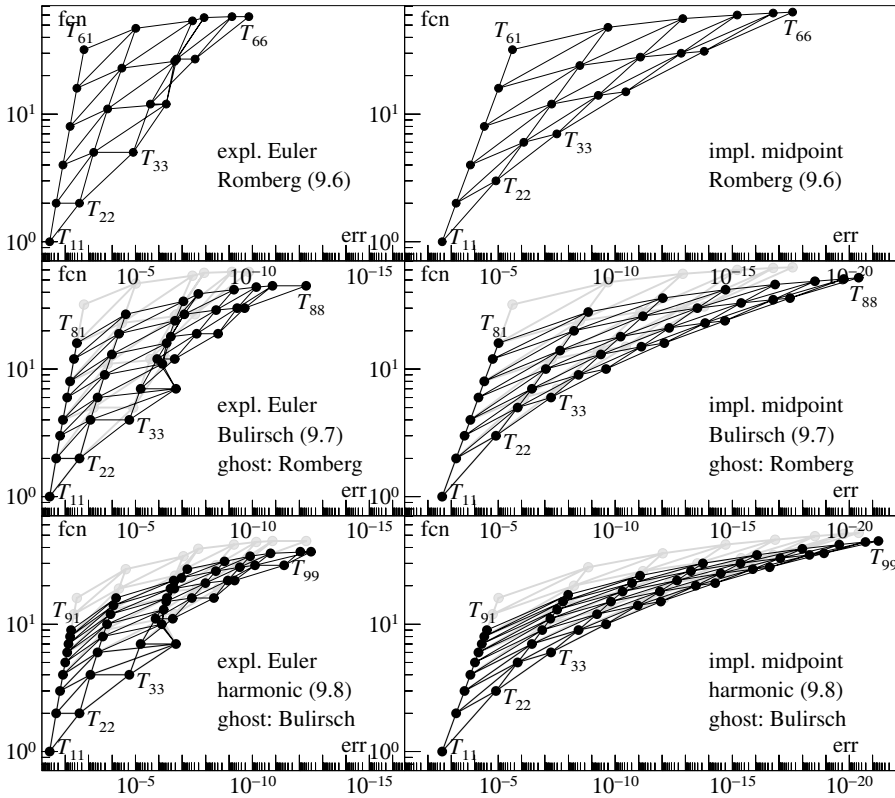
$$T_{j,k+1} = T_{j,k} + \frac{T_{j,k} - T_{j-1,k}}{(n_j/n_{j-k})^2 - 1} \quad (9.10)$$

instead of (9.9).

Numerical example. We solve the problem

$$y' = (-y \sin x + 2 \tan x)y, \quad y(\pi/6) = 2/\sqrt{3} \quad (9.11)$$

with true solution $y(x) = 1/\cos x$ and basic step size $H = 0.2$ by Euler’s method. Fig. 9.1 represents, for each of the entries $T_{j,k}$ of the extrapolation tableau, the *numerical work* $(1 + n_j - 1 + n_{j-1} - 1 + \dots + n_{j-k+1} - 1)$ compared to the *precision* $(|T_{j,k} - y(x_0 + H)|)$ in double logarithmic scale. The first picture is for the Romberg sequence (9.6), the second for the Bulirsch sequence (9.7), and the last

Fig. 9.1. h -extrap. expl. EulerFig. 9.2. h^2 -extrap. impl. midpoint

for the harmonic sequence (9.8). In pictures 2 and 3 the results of the foregoing graphics are repeated as a shaded “ghost” (. . . of Canterbury) in order to demonstrate how the results are better than those for the predecessor. Nobody is perfect, however. The “best” method in these comparisons, the harmonic sequence, suffers for high orders from a strong influence of rounding errors (see Exercise 5 below; the computations of Fig. 9.1, 9.2 and 9.4 have been made in quadruple precision).

The analogous results for the symmetric implicit mid-point rule (7.4) are presented in Fig. 9.2. Although implicit, this method is easy to implement for this particular example. We again use the same basic step size $H = 0.2$ as above and the same step-number sequences (9.6), (9.7), (9.8). Here, the “numerical work” ($n_j + n_{j-1} + \dots + n_{j-k+1}$) represents *implicit* stages and therefore can not be compared to the values of the explicit method. The precisions, however, show a drastic improvement.

Rational Extrapolation. Many authors in the sixties claimed that it is better to use rational functions instead of polynomials in (9.3). In this case the formula (9.9)

must be replaced by (Bulirsch & Stoer 1964)

$$T_{j,k+1} = T_{j,k} + \frac{T_{j,k} - T_{j-1,k}}{\left(\frac{n_j}{n_{j-k}}\right) \left(1 - \frac{T_{j,k} - T_{j-1,k}}{T_{j,k} - T_{j-1,k-1}}\right) - 1} \quad (9.12)$$

where

$$T_{j,0} = 0.$$

For systems of differential equations the division of vectors is to be understood componentwise.

Later numerical experiments (Deuffhard 1983) showed that rational extrapolation is nearly never more advantageous than polynomial extrapolation.

The Gragg or GBS Method

Since it is fully explicit GRAGG's algorithm is so ideally suited as a basis for RICHARDSON extrapolation that no other symmetric two-step algorithm can compete with it. (H.J. Stetter 1970)

Here we can not do better than quote from Stetter (1970): "Expansions in powers of h^2 are extremely important for an efficient application of Richardson extrapolation. Therefore it was a great achievement when Gragg proved in 1963 that the quantity $S_h(x)$ produced by the algorithm ($x = x_0 + 2nh$, $x_i = x_0 + ih$)

$$y_1 = y_0 + hf(x_0, y_0) \quad (9.13a)$$

$$y_{i+1} = y_{i-1} + 2hf(x_i, y_i) \quad i = 1, 2, \dots, 2n \quad (9.13b)$$

$$S_h(x) = \frac{1}{4} (y_{2n-1} + 2y_{2n} + y_{2n+1}) \quad (9.13c)$$

possesses an asymptotic expansion in even powers of h and has satisfactory stability properties. This led to the construction of the very powerful G(ragg)-B(ulirsch)-S(toer)-extrapolation algorithm . . .".

Gragg's *proof* of this property was very long and complicated and it was again "a great achievement" that Stetter had the elegant idea of interpreting (9.13b) as a *one-step* algorithm by rewriting (9.13) in terms of odd and even indices: for this purpose we define

$$\begin{aligned} h^* &= 2h, & x_k^* &= x_0 + kh^*, & u_0 &= v_0 = y_0, \\ u_k &= y_{2k}, & v_k &= y_{2k+1} - hf(x_{2k}, y_{2k}) = \frac{1}{2} (y_{2k+1} + y_{2k-1}). \end{aligned} \quad (9.14)$$

Then the method (9.13) can be rewritten as (see Fig. 9.3)

$$\begin{pmatrix} u_{k+1} \\ v_{k+1} \end{pmatrix} = \begin{pmatrix} u_k \\ v_k \end{pmatrix} + h^* \begin{pmatrix} f\left(x_k^* + \frac{h^*}{2}, v_k + \frac{h^*}{2} f(x_k^*, u_k)\right) \\ \frac{1}{2} \left(f(x_k^* + h^*, u_{k+1}) + f(x_k^*, u_k)\right) \end{pmatrix}. \quad (9.15)$$

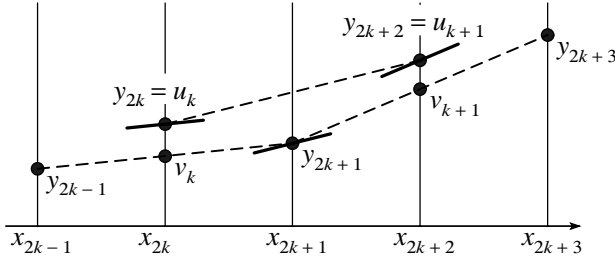


Fig. 9.3. Symmetry of the Gragg method

This method, which maps the pair (u_k, v_k) to (u_{k+1}, v_{k+1}) , can be seen from Fig. 9.3 to be *symmetric*. The symmetry can also be checked analytically (see Definition 8.7) by exchanging $u_{k+1} \leftrightarrow u_k$, $v_{k+1} \leftrightarrow v_k$, $h^* \leftrightarrow -h^*$, $x_k^* \leftrightarrow x_k^* + h^*$. A trivial calculation then shows that this leaves formula (9.15) invariant. Method (9.15) is consistent with the differential equation (let $h^* \rightarrow 0$ in the increment function)

$$\begin{aligned} u' &= f(x, v) & u(x_0) &= y_0 \\ v' &= f(x, u) & v(x_0) &= y_0, \end{aligned} \quad (9.16)$$

whose exact solution is simply $u(x) = v(x) = y(x)$. Therefore, we have from Theorem 8.10 that

$$y(x) - u_{h^*}(x) = \sum_{j=1}^{\ell} a_{2j}(x)(h^*)^{2j} + (h^*)^{2\ell+2} A(x, h^*) \quad (9.17a)$$

$$y(x) - v_{h^*}(x) = \sum_{j=1}^{\ell} b_{2j}(x)(h^*)^{2j} + (h^*)^{2\ell+2} B(x, h^*) \quad (9.17b)$$

and $a_{2j}(x_0) = b_{2j}(x_0) = 0$. We see from (9.14) and (9.17a) that $y_h(x)$ possesses an expansion in even powers of h , provided that the number of steps is even; i.e., for $x = x_0 + 2nh$,

$$y(x) - y_h(x) = \sum_{j=1}^{\ell} \hat{a}_{2j}(x) h^{2j} + h^{2\ell+2} \hat{A}(x, h) \quad (9.18)$$

where $\hat{a}_{2j}(x) = 2^{2j} a_{2j}(x)$ and $\hat{A}(x, h) = 2^{2\ell+2} A(x, 2h)$.

The so-called *smoothing step*, i.e., formula

$$S_h(x_0 + 2nh) = \frac{1}{4} (y_{2n-1} + 2y_{2n} + y_{2n+1}) = \frac{1}{2} (u_n + v_n)$$

(see (9.13c) and (9.14)) had its historical origin in the “weak stability” of the explicit midpoint rule (9.13b) (see also Fig. III.9.2). However, since the method is anyway followed by extrapolation, this step is not of great importance (Shampine & Baca 1983). It is a little more costly and increases the “stability domain” by

approximately the same amount (see Fig. IV.2.3 of Vol. II). Further, it has the advantage of evaluating the function f at the end of the basic step.

Theorem 9.2. *Let $f(x, y) \in \mathcal{C}^{2\ell+2}$, then the numerical solution defined in (9.13) possesses for $x = x_0 + 2nh$ an asymptotic expansion of the form*

$$y(x) - S_h(x) = \sum_{j=1}^{\ell} e_{2j}(x)h^{2j} + h^{2\ell+2}C(x, h) \quad (9.19)$$

with $e_{2j}(x_0) = 0$ and $C(x, h)$ bounded for $x_0 \leq x \leq \bar{x}$ and $0 \leq h \leq h_0$.

Proof. By adding (9.17a) and (9.17b) and using $h^* = 2h$ we obtain (9.19) with $e_{2j}(x) = (a_{2j}(x) + b_{2j}(x))2^{2j-1}$. \square

This method can thus be used for Richardson extrapolation in the same way as symmetric methods above: we choose a step-number sequence, with the condition that the n_j are even, i.e.,

$$2, 4, 8, 16, 32, 64, 128, 256, \dots \quad (9.6')$$

$$2, 4, 6, 8, 12, 16, 24, 32, 48, \dots \quad (9.7')$$

$$2, 4, 6, 8, 10, 12, 14, 16, 18, \dots \quad (9.8')$$

set

$$T_{i,1} := S_{h_i}(x_0 + H)$$

and compute the extrapolated expressions $T_{i,j}$, based on the h^2 -expansion, by the Aitken-Neville formula (9.10).

Numerical example. Fig. 9.4 represents the numerical results of this algorithm applied to Example (9.11) with step size $H = 0.2$. The step size sequences are Romberg (9.6') (above), Bulirsch (9.7') (middle), and harmonic (9.8') (below). The algorithm *with* smoothing step (numerical work $= 1 + n_j + n_{j-1} + \dots + n_{j-k+1}$) is represented *left*, the results *without* smoothing step (numerical work $= 1 + n_j - 1 + n_{j-1} - 1 + \dots + n_{j-k+1} - 1$) are on the *right*.

The results are nearly identical to those for the implicit midpoint rule (Fig. 9.2), but much more valuable, since here the method is explicit. In the pictures on the left the values for extrapolated Euler (from Fig. 9.1) are repeated as a “ghost” and demonstrate clearly the importance of the h^2 -expansion, especially in the diagonal T_{kk} for large values of k . The ghost in the pictures on the right are the values *with* smoothing step from the left; the differences are seen to be tiny.

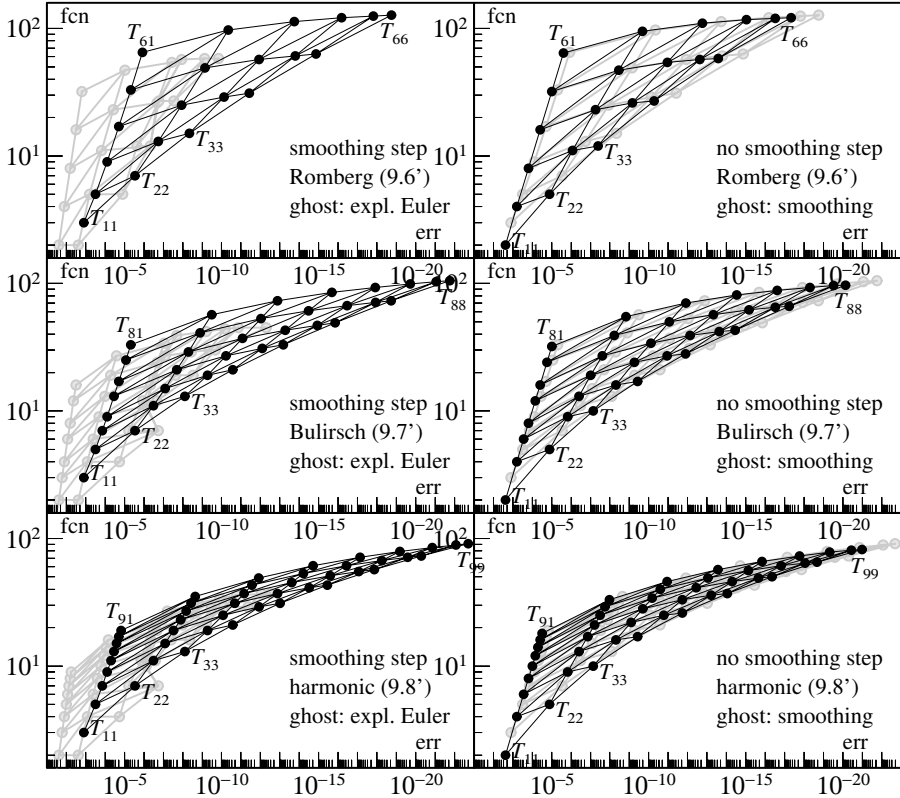


Fig. 9.4. Precision of h^2 -extrapolated Gragg method for Example (9.11)

Asymptotic Expansion for Odd Indices

For completeness, we still want to derive the existence of an h^2 expansion for y_{2k+1} from (9.17b), although this is of no practical importance for the numerical algorithm described above.

Theorem 9.3 (Gragg 1964). *For $x = x_0 + (2k+1)h$ we have*

$$y(x) - y_h(x) = \sum_{j=1}^{\ell} \hat{b}_{2j}(x) h^{2j} + h^{2\ell+2} \hat{B}(x, h) \quad (9.20)$$

where the coefficients $\hat{b}_{2j}(x)$ are in general different from those for even indices and $\hat{b}_{2j}(x_0) \neq 0$.

Proof. y_{2k+1} can be computed (see Fig. 9.3) either from v_k by a forward step or from v_{k+1} by a backward step. For the sake of symmetry, we take the mean of

both expressions and write

$$y_{2k+1} = \frac{1}{2}(v_k + v_{k+1}) + \frac{h}{2}(f(x_k^*, u_k) - f(x_{k+1}^*, u_{k+1})).$$

We now subtract the exact solution and obtain

$$\begin{aligned} 2(y_h(x) - y(x)) &= v_{2h}(x - h) - y(x - h) \\ &\quad + v_{2h}(x + h) - y(x + h) + y(x - h) - 2y(x) + y(x + h) \\ &\quad + h\left(f(x - h, u_{2h}(x - h)) - f(x + h, u_{2h}(x + h))\right). \end{aligned}$$

Due to the symmetry of $u_{2h}(x)$ ($u_{2h}(\xi) = u_{-2h}(\xi)$) and of $v_{2h}(x)$ the whole expression becomes symmetric in h . Thus the asymptotic expansion for y_{2k+1} contains no odd powers of h . \square

Both expressions, for even and for odd indices, can still be combined into a single formula (see Exercise 2).

Existence of Explicit RK Methods of Arbitrary Order

Each of the expressions $T_{j,k}$ clearly represents an explicit RK-method (see Exercise 1). If we apply the well-known error formula for polynomial interpolation (see e.g., Abramowitz & Stegun 1964, formula 25.2.27) to (9.19), we obtain

$$y(x_0 + H) - T_{j,k} = \frac{(-1)^k}{n_j^2 \cdot \dots \cdot n_{j-k+1}^2} e_{2k}(x_0 + H) H^{2k} + \mathcal{O}(H^{2k+2}). \quad (9.21)$$

Since $e_k(x_0) = 0$, we have

$$y(x_0 + H) - T_{j,k} = \frac{(-1)^k}{n_j^2 \cdot \dots \cdot n_{j-k+1}^2} e'_{2k}(x_0) H^{2k+1} + \mathcal{O}(H^{2k+2}). \quad (9.22)$$

This shows that $T_{j,k}$ represents an explicit Runge-Kutta method of order $2k$. As an application of this result we have:

Theorem 9.4 (Gragg 1964). *For p even, there exists an explicit RK-method of order p with $s = p^2/4 + 1$ stages.*

Proof. This result is obtained by counting the number of necessary function evaluations of the GBS-algorithm using the harmonic sequence and without the final smoothing step. \square

Remark. The extrapolated Euler method leads to explicit Runge-Kutta methods with $s = p(p-1)/2 + 1$ stages. This shows once again the importance of the h^2 expansion.

Order and Step Size Control

Extrapolation methods have the advantage that in addition to the step size also the order (i.e., number of columns) can be changed at each step. Because of this double freedom, the *practical implementation* in an optimal way is more complicated than for fixed-order RK-methods. The first codes were developed by Bulirsch & Stoer (1966) and their students. Very successful extrapolation codes due to P. Deuffhard and his collaborators are described in Deuffhard (code DIFEX1, 1983).

The choice of the *step size* can be performed in exactly the same way as for fixed-order embedded methods (see Section II.4). If the first k lines of the extrapolation tableau are computed, we have T_{kk} as the highest-order approximation (of order $2k$ by (9.22)) and in addition $T_{k,k-1}$ of order $2k-2$. It is therefore natural to use the expression

$$err_k = \|T_{k,k-1} - T_{k,k}\| \quad (9.23)$$

for step size control. The norm is the same as in (4.11). As in (4.12) we get for the optimal step size the formula

$$H_k = H \cdot 0.94 \cdot (0.65/err_k)^{1/(2k-1)} \quad (9.24)$$

where this time we have chosen a safety factor depending partly on the order.

For the choice of an *optimal order* we need a measure of work, which allows us to compare different methods. The work for computing T_{kk} can be measured by the number A_k of function evaluations. For the GBS-algorithm it is given recursively by

$$\begin{aligned} A_1 &= n_1 + 1 \\ A_k &= A_{k-1} + n_k. \end{aligned} \quad (9.25)$$

However, a large number of function evaluations can be compensated by a large step size H_k , given by (9.24). We therefore consider

$$W_k = \frac{A_k}{H_k}, \quad (9.26)$$

the *work per unit step*, as a measure of work. The idea is now to choose the order (i.e., the index k) in such a way that W_k is minimized.

Let us describe the *combined order and step size control* in some more detail. We assume that at some point of integration the step size H and the index k ($k > 2$) are proposed. The step is then realized in the following way: we first compute $k-1$ lines of the extrapolation tableau and also the values H_{k-2} , W_{k-2} , err_{k-1} , H_{k-1} , W_{k-1} .

a) *Convergence in line $k-1$.* If $err_{k-1} \leq 1$, we accept $T_{k-1,k-1}$ as numerical solution and continue the integration with the new proposed quantities

$$\begin{aligned} k_{\text{new}} &= \begin{cases} k & \text{if } W_{k-1} < 0.9 \cdot W_{k-2} \\ k-1 & \text{else} \end{cases} \\ H_{\text{new}} &= \begin{cases} H_{k_{\text{new}}} & \text{if } k_{\text{new}} \leq k-1 \\ H_{k-1}(A_k/A_{k-1}) & \text{if } k_{\text{new}} = k. \end{cases} \end{aligned} \quad (9.27)$$

In (9.27), the only non-trivial formula is the choice of the step size H_{new} in the case of an order-increase $k_{\text{new}} = k$. In this case we want to avoid the computation of err_k , so that H_k and W_k are unknown. However, since our k is assumed to be close to the optimal value, we have $W_k \approx W_{k-1}$ which leads to the proposed step size increase.

b) *Convergence monitor.* If $err_{k-1} > 1$, we first decide whether we may expect convergence at least in line $k+1$. It follows from (9.22) that, asymptotically,

$$\|T_{k,k-2} - T_{k,k-1}\| \approx \left(\frac{n_2}{n_k}\right)^2 err_{k-1} \quad (9.28)$$

with err_{k-1} given by (9.23). Unfortunately, err_k cannot be compared with (9.28), since different factors (depending on the differential equation to be solved) are involved in the asymptotic formula (cf. (9.22)). If we nevertheless assume that err_k is $(n_2/n_1)^2$ times smaller than (9.28) we obtain $err_k \approx (n_1/n_k)^2 err_{k-1}$. We therefore already reject the step at this point, if

$$err_{k-1} > \left(\frac{n_{k+1}n_k}{n_1n_1}\right)^2 \quad (9.29)$$

and restart with $k_{\text{new}} \leq k-1$ and H_{new} according to (9.27). If the contrary of (9.29) holds, we compute the next line of the extrapolation tableau, i.e., $T_{k,k}$, err_k , H_k and W_k .

c) *Convergence in line k .* If $err_k \leq 1$, we accept T_{kk} as numerical solution and continue the integration with the new proposed values

$$\begin{aligned} k_{\text{new}} &= \begin{cases} k-1 & \text{if } W_{k-1} < 0.9 \cdot W_k \\ k+1 & \text{if } W_k < 0.9 \cdot W_{k-1} \\ k & \text{in all other cases} \end{cases} \\ H_{\text{new}} &= \begin{cases} H_{k_{\text{new}}} & \text{if } k_{\text{new}} \leq k \\ H_k(A_{k+1}/A_k) & \text{if } k_{\text{new}} = k+1. \end{cases} \end{aligned} \quad (9.30)$$

d) *Second convergence monitor.* If $err_k > 1$, we check, as in (b), the relation

$$err_k > \left(\frac{n_{k+1}}{n_1}\right)^2. \quad (9.31)$$

If (9.31) is satisfied, the step is rejected and we restart with $k_{\text{new}} \leq k$ and H_{new} of (9.30). Otherwise we continue.

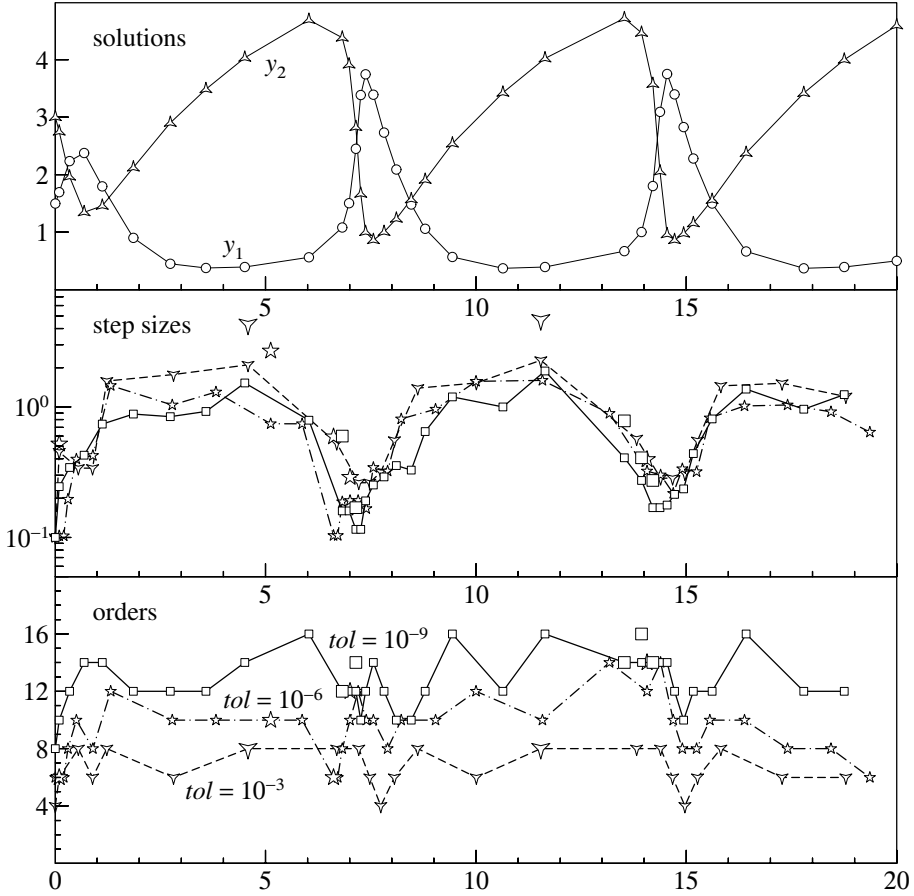


Fig. 9.5. Solution, step size and order variation obtained by ODEX

e) *Hope for convergence in line $k+1$.* We compute err_{k+1} , H_{k+1} and W_{k+1} . If $err_{k+1} \leq 1$, we accept $T_{k+1,k+1}$ as numerical solution and continue the integration with the new proposed order

$$\begin{aligned}
 k_{\text{new}} &:= k \\
 \text{if } (W_{k-1} < 0.9 \cdot W_k) & \quad k_{\text{new}} := k-1 \\
 \text{if } (W_{k+1} < 0.9 \cdot W_{k_{\text{new}}}) & \quad k_{\text{new}} := k+1.
 \end{aligned} \tag{9.32}$$

If $err_{k+1} > 1$ the step is rejected and we restart with $k_{\text{new}} \leq k$ and H_{new} of (9.24).

The following slight modifications of the above algorithm are recommended:

i) Storage considerations lead to a limitation of the number of columns of the extrapolation tableau, say by k_{max} (e.g., $k_{\text{max}} = 9$). For the proposed index k_{new} we require $2 \leq k_{\text{new}} \leq k_{\text{max}} - 1$. This allows us to activate (e) at each step.

ii) After a step-rejection the step size and the order may not be increased.

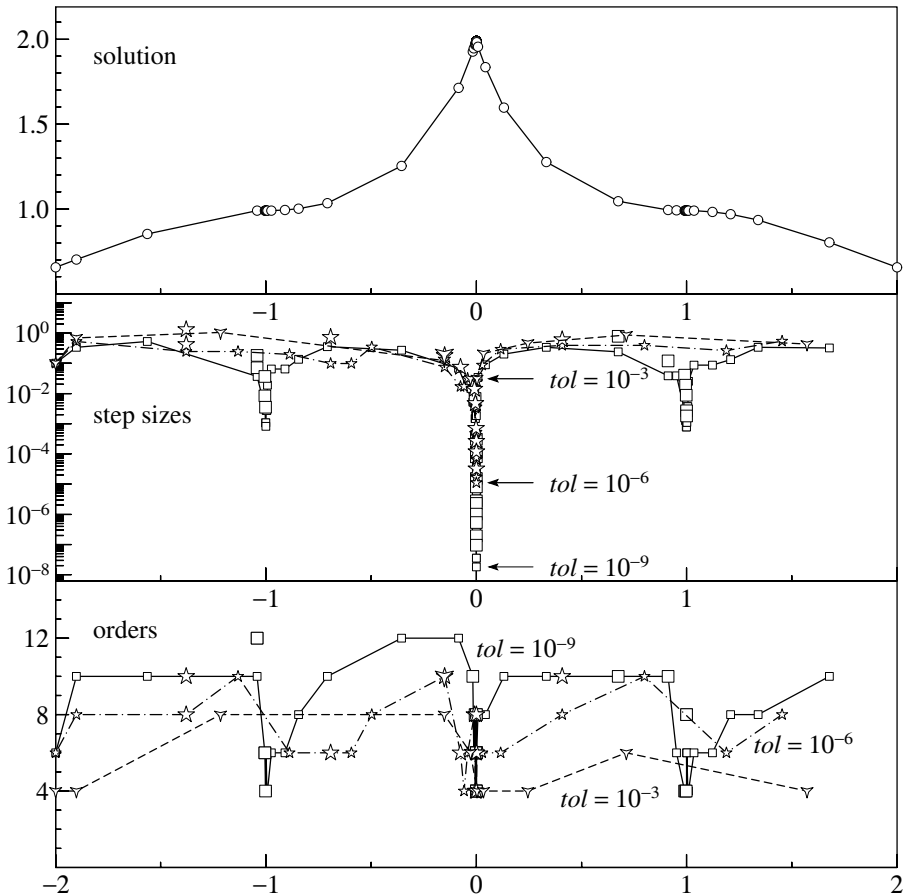


Fig. 9.6. Solution, step size and order variation obtained by ODEX at the discontinuous example (9.33)

Numerical study of the combined step size and order control. We show in the following examples how the step size and the order vary for the above algorithm. For this purpose we have written the FORTRAN-subroutine ODEX (see Appendix).

As a first example we again take the *Brusselator* (cf. Section II.4). As in Fig. 4.1, the first picture of Fig. 9.5 shows the two components of the solution (obtained with $Atol = Rtol = 10^{-9}$). In the remaining two pictures we have plotted the step sizes and orders for the three tolerances 10^{-3} (broken line), 10^{-6} (dashes and dots) and 10^{-9} (solid line). One can easily observe that the extrapolation code automatically chooses a suitable order (depending essentially on Tol). Step-rejections are indicated by larger symbols.

We next study the behaviour of the order control near discontinuities. In the example

$$y' = -\text{sign}(x) |1 - |x|| y^2, \quad y(-2) = 2/3, \quad -2 \leq x \leq 2 \quad (9.33)$$

we have a discontinuity in the first derivative of $y(x)$ at $x = 0$ and two discontinuities in the second derivative (at $x = \pm 1$). The numerical results are shown in Fig. 9.6 for three tolerances. In all cases the error at the endpoint is about $10 \cdot \text{tol}$. The discontinuities at $x = \pm 1$ are not recognized in the computations with $\text{tol} = 10^{-3}$ and $\text{tol} = 10^{-6}$. Whenever a discontinuity is detected, the order drops to 4 (lowest possible) in its neighbourhood, so that these points are passed rather efficiently.

Dense Output for the GBS Method

Extrapolation methods are methods best suited for high precision which typically take very large (basic) step sizes during integration. The reasons for the need of a dense output formula (discussed in Section II.6) are therefore particularly important here. First attempts to provide extrapolation methods with a dense output are due to Lindberg (1972) for the implicit trapezoidal rule, and to Shampine, Baca & Bauer (1983) who constructed a 3rd order dense output for the GBS method. We present here the approach of Hairer & Ostermann (1990) (see also Simonsen 1990).

It turned out that the existence of high order dense output is only possible if the step number sequence satisfies some restrictions such as

$$n_{j+1} - n_j = 0 \pmod{4} \quad \text{for } j = 1, 2, 3, \dots \quad (9.34)$$

which, for example, is fulfilled by the sequence

$$\{2, 6, 10, 14, 18, 22, 26, 30, 34, \dots\}. \quad (9.35)$$

The idea is, once again, to do Hermite interpolation. To begin with, high order approximations are as usual at our disposal for the values y_0, y'_0, y_1, y'_1 by using $y_0, f(x_0, y_0), T_{kk}, f(x_0 + H, T_{kk})$, where T_{kk} is supposed to be the highest order approximation computed and used for continuation of the solution.

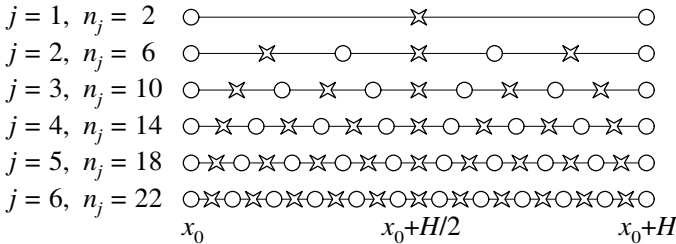


Fig. 9.7. Evaluation points for a GBS step

For more inspiration, we represent in Fig. 9.7 the steps taken by Gragg's midpoint rule for the step number sequence (9.35). The symbols \circ and \times indicate that the even steps and the odd steps possess a *different* asymptotic expansion (see Theorem 9.3) and must not be blended. We see that, owing to condition (9.34), the midpoint values $y_{n_j/2}^{(j)}$, obtained during the computation of T_{j1} , all have the same parity and can therefore also be extrapolated to yield an approximation for $y(x_0 + H/2)$ of order $2k - 1$ (remember that in Theorem 9.3, $\widehat{b}_{2j}(x_0) \neq 0$).

We next insert (9.20) for $x = x_0 + H/2$ into $f(x, y)$

$$f_{n_j/2}^{(j)} := f(x, y_{n_j/2}^{(j)}) = f\left(x, y(x) - h_j^2 \widehat{b}_2(x) - h_j^4 \widehat{b}_4(x) \dots\right)$$

and develop in powers of h_j to obtain

$$y'(x) - f_{n_j/2}^{(j)} = h_j^2 a_{2,1}(x) + h_j^4 a_{4,1}(x) + \dots \quad (9.36)$$

This shows that the f -values at the midpoint $x_0 + H/2$ (for $j = 1, 2, \dots, k$) possess an asymptotic expansion and can be extrapolated $k - 1$ times to yield an approximation to $y'(x_0 + H/2)$ of order $2k - 1$.

But this is not enough. We now consider, similar to an idea which goes back to the papers of Deuflhard & Nowak (1987) and Lubich (1989), the central differences $\delta f_i = f_{i+1} - f_{i-1}$ at the midpoint which, by Fig. 9.7, are available for $j = 1, 2, \dots, k$ and are based on even parity. By using (9.18) and by developing into powers of h_j we obtain

$$\begin{aligned} \frac{\delta f_{n_j/2}^{(j)}}{2h_j} &= \frac{f(x + h_j, y_{n_j/2+1}^{(j)}) - f(x - h_j, y_{n_j/2-1}^{(j)})}{2h_j} \\ &= \left(f(x + h_j, y(x + h_j) - h_j^2 \widehat{a}_2(x + h_j) - h_j^4 \widehat{a}_4(x + h_j) - \dots) - \right. \\ &\quad \left. f(x - h_j, y(x - h_j) - h_j^2 \widehat{a}_2(x - h_j) - h_j^4 \widehat{a}_4(x - h_j) - \dots) \right) / 2h_j \\ &= \frac{y'(x + h_j) - y'(x - h_j)}{2h_j} - h_j^2 c_2(x) - h_j^4 c_4(x) - \dots \end{aligned}$$

Finally we insert the Taylor series for $y'(x + h)$ and $y'(x - h)$ to obtain an expansion

$$y''(x) - \frac{\delta f_{n_j/2}^{(j)}}{2h_j} = h_j^2 a_{2,2}(x) + h_j^4 a_{4,2}(x) + \dots \quad (9.38)$$

Therefore, $k - 1$ extrapolations of the expressions (9.37) yield an approximation to $y''(x_0 + H/2)$ of order $2k - 1$.

In order to get approximations to the third and fourth derivatives of the solution at $x_0 + H/2$, we use the second and third central differences of $f_i^{(j)}$ which exist for $j \geq 2$ (Fig. 9.7). These can be extrapolated $k - 2$ times to give approximations of order $2k - 3$.

The continuation of this process yields the following algorithm:

Step 1. For each $j \in \{1, \dots, k\}$, compute approximations to the derivatives of $y(x)$ at $x_0 + H/2$ by:

$$d_j^{(0)} = y_{n_j/2}^{(j)}, \quad d_j^{(\kappa)} = \frac{\delta^{\kappa-1} f_{n_j/2}^{(j)}}{(2h_j)^{\kappa-1}} \quad \text{for } \kappa = 1, \dots, 2j. \quad (9.39)$$

Step 2. Extrapolate $d_j^{(0)}$ $(k-1)$ times and $d_j^{(2\ell-1)}$, $d_j^{(2\ell)}$ $(k-\ell)$ times to obtain improved approximations $d^{(\kappa)}$ to $y^{(\kappa)}(x_0 + H/2)$.

Step 3. For given μ ($-1 \leq \mu \leq 2k$) define the polynomial $P_\mu(\theta)$ of degree $\mu + 4$ by

$$\begin{aligned} P_\mu(0) &= y_0, & P'_\mu(0) &= Hf(x_0, y_0), \\ P_\mu(1) &= T_{kk}, & P'_\mu(1) &= Hf(x_0 + H, T_{kk}) \\ P_\mu^{(\kappa)}(1/2) &= H^\kappa d^{(\kappa)} & \text{for } \kappa = 0, \dots, \mu. \end{aligned} \quad (9.40)$$

This computation of $P_\mu(\theta)$ does not need any further function evaluation since $f(x_0 + H, T_{kk})$ has to be computed anyway for the next step. Further, $P_\mu(\theta)$ gives a global \mathcal{C}^1 approximation to the solution.

Theorem 9.5 (Hairer & Ostermann 1990). *If the step number sequence satisfies (9.34), then the error of the dense output polynomial $P_\mu(\theta)$ satisfies*

$$y(x_0 + \theta H) - P_\mu(\theta) = \begin{cases} \mathcal{O}(H^{2k+1}) & \text{if } n_1 = 4 \text{ and } \mu \geq 2k-4 \\ \mathcal{O}(H^{2k}) & \text{if } n_1 = 2 \text{ and } \mu \geq 2k-5. \end{cases} \quad (9.40)$$

Proof. Since $P_\mu(\theta)$ is a polynomial of degree $\mu + 4$ the error due to interpolation is of size $\mathcal{O}(H^{\mu+5})$. This explains the restriction on μ in (9.40). As explained above, the function value and derivative data used for Hermite interpolation have the required precision

$$H^\kappa y^{(\kappa)}(x_0 + H/2) - H^\kappa d^{(\kappa)} = \begin{cases} \mathcal{O}(H^{2k}) & \text{if } \kappa = 0, \\ \mathcal{O}(H^{2k+1}) & \text{if } \kappa \text{ is odd,} \\ \mathcal{O}(H^{2k+2}) & \text{if } \kappa \geq 2 \text{ is even.} \end{cases}$$

In the case $n_1 = 4$ the parity of the central point $x_0 + H/2$ is *even* (in contrary to Fig. 9.7), we therefore apply (9.18) and gain one order because then the functions $a_{i,0}(x)$ vanish at x_0 . \square

Control of the Interpolation Error

At one time . . . every young mathematician was familiar with $\operatorname{sn} u$, $\operatorname{cn} u$, and $\operatorname{dn} u$, and algebraic identities between these functions figured in every examination.

(E.H. Neville, *Jacobian Elliptic Functions*, 1944)

Numerical example. We apply the above dense output formula with $\mu = 2k - 3$ (as is standard in ODEX) to the differential equations of the Jacobian elliptic functions sn , cn , dn (see Abramowitz & Stegun 1964, 16.16):

$$\begin{aligned} y_1' &= y_2 y_3 & y_1(0) &= 0 \\ y_2' &= -y_1 y_3 & y_2(0) &= 1 \\ y_3' &= -0.51 \cdot y_1 y_2 & y_3(0) &= 1 \end{aligned} \quad (9.41)$$

with integration interval $0 \leq x \leq 10$ and error tolerance $Atol = Rtol = 10^{-9}$. The error for the three components of the obtained continuous solution is displayed in Fig. 9.8 (upper picture; the ghosts are the solution curves) and gives a quite disappointing impression when compared with the precision at the grid points. We shall now see that these horrible bumps are nothing else than interpolation errors.

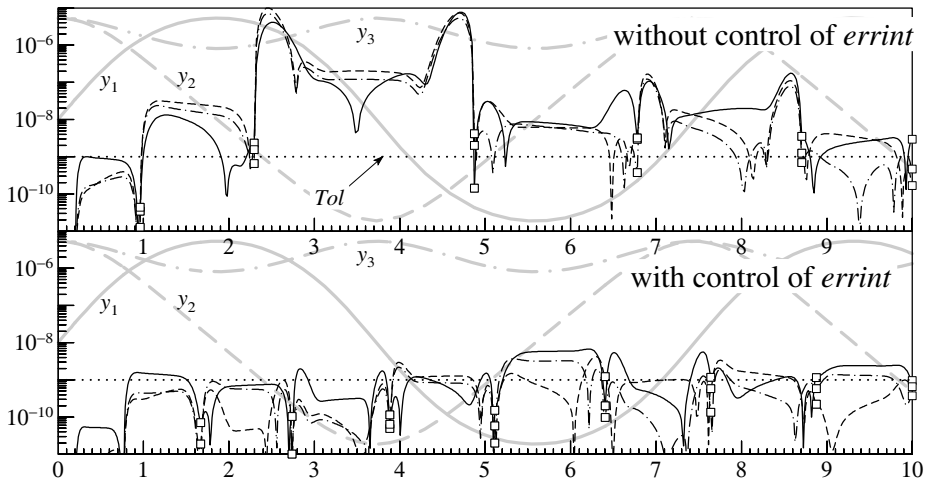


Fig. 9.8. Error of dense output without/with interpolation control

Assume that in the definition of $P_\mu(\theta)$ the basic function and derivative values are replaced by the exact values $y(x_0 + H)$, $y'(x_0 + H)$, and $y^{(\kappa)}(x_0 + H/2)$. Then the error of $P_\mu(\theta)$ is given by

$$\theta^2(1-\theta)^2 \left(\theta - \frac{1}{2} \right)^{\mu+1} \frac{y^{(\mu+5)}(\xi)}{(\mu+5)!} H^{\mu+5} \quad (9.42)$$

2. Combine (9.18) and (9.19) into the formula ($x = x_0 + kh$)

$$y(x) - y_k = \sum_{j=1}^{\ell} \left(\alpha_{2j}(x) + (-1)^k \beta_{2j}(x) \right) h^{2j} + h^{2\ell+2} E(x, h)$$

for the asymptotic expansion of the Gragg method defined by (9.13a,b).

3. (Stetter 1970). Prove that for every real b (generally between 0 and 1) the method

$$y_1 = y_0 + h \left(b f(x_0, y_0) + (1-b) f(x_1, y_1) \right)$$

$$y_{i+1} = y_{i-1} + h \left((1-b) f(x_{i-1}, y_{i-1}) + 2b f(x_i, y_i) + (1-b) f(x_{i+1}, y_{i+1}) \right)$$

possesses an expansion in powers of h^2 . Prove the same property for the smoothing step

$$S_h(x) = \frac{1}{2} \left(y_{2n} + y_{2n-1} + h(1-b) f(x_{2n-1}, y_{2n-1}) + h b f(x_{2n}, y_{2n}) \right).$$

4. (Stetter 1970). Is the Euler step (9.13a) essential for an h^2 -expansion? Prove that a first order starting procedure

$$y_1 = y_0 + h \Phi(x_0, y_0, h)$$

for (9.13a) produces an h^2 -expansion if the quantities

$y_{-1} = y_0 - h \Phi(x_0, y_0, -h)$, y_0 , and y_1 satisfy (9.13b) for $i = 0$.

5. Study the numerical instability of the extrapolation scheme for the harmonic sequence, i.e., suppose that the entries T_{11} , T_{21} , $T_{31} \dots$ are disturbed with rounding errors ε , $-\varepsilon$, ε, \dots and compute the propagation of these errors into the extrapolation tableau (9.5).

Result. Due to the linearity of the extrapolation scheme, we suppose the T_{ik} equal zero and $\varepsilon = 1$. Then the results for sequence (9.8') are

1.								
-1.	-1.67							
1.	2.60	3.13						
-1.	-3.57	-5.63	-6.21					
1.	4.56	9.13	11.94	12.69				
-1.	-5.55	-13.63	-21.21	-25.35	-26.44			
1.	6.54	19.13	35.01	47.65	54.14	55.82		
-1.	-7.53	-25.63	-54.31	-84.09	-105.64	-116.30	-119.03	
1.	8.53	33.13	80.13	140.14	195.34	232.96	251.10	255.73

hence, for order 18, we lose approximately two digits due to roundoff errors.

6. (Laguerre 1883^{*}). If a_1, a_2, \dots, a_n are distinct positive real numbers and r_1, r_2, \dots, r_n are distinct reals, then

$$A = \begin{pmatrix} a_1^{r_1} & a_1^{r_2} & \dots & a_1^{r_n} \\ a_2^{r_1} & a_2^{r_2} & \dots & a_2^{r_n} \\ \vdots & \vdots & & \vdots \\ a_n^{r_1} & a_n^{r_2} & \dots & a_n^{r_n} \end{pmatrix}$$

is invertible.

Hint (Pólya & Szegő 1925, Vol. II, Abschn. V, Problems 76-77^{*}). Show by induction on n that, if the function $g(t) = \sum_{i=1}^n \alpha_i t^{r_i}$ has n distinct positive zeros, then $g(t) \equiv 0$. By Rolle's theorem the function

$$\frac{d}{dt}(t^{-r_1} g(t)) = \sum_{i=2}^n \alpha_i (r_i - r_1) t^{r_i - r_1 - 1}$$

has $n - 1$ positive distinct zeros and the induction hypothesis can be applied.

^{*} We are grateful to our colleague J. Steinig for these references.

II.10 Numerical Comparisons

The Pleiades seem to be among the first stars mentioned in astronomical literature, appearing in Chinese annals of 2357 B.C. . . .

(R.H. Allen, *Star names, their love and meaning*, 1899, Dover 1963)

If you enjoy fooling around making pictures, instead of typesetting ordinary text, \TeX will be a source of endless frustration/amusement for you, . . .

(D. Knuth, *The \TeX book*, p. 389)

Problems

EULR — Euler’s equation of rotation of a rigid body (“Diese merkwürdig symmetrischen und eleganten Formeln . . .”, A. Sommerfeld 1942, vol. I, § 26.1, Euler 1758)

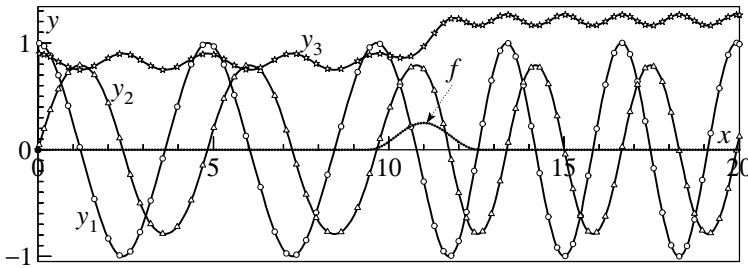


Fig. 10.1. Solutions of Euler’s equations (10.1)

$$\begin{aligned} I_1 y_1' &= (I_2 - I_3) y_2 y_3 \\ I_2 y_2' &= (I_3 - I_1) y_3 y_1 \\ I_3 y_3' &= (I_1 - I_2) y_1 y_2 + f(x) \end{aligned} \quad (10.1)$$

where y_1, y_2, y_3 are the coordinates of $\vec{\omega}$, the rotation vector, and I_1, I_2, I_3 are the principal moments of inertia. The third coordinate has an additional exterior force

$$f(x) = \begin{cases} 0.25 \cdot \sin^2 x & \text{if } 3\pi \leq x \leq 4\pi \\ 0 & \text{otherwise} \end{cases} \quad (10.1')$$

which is discontinuous in its second derivative. We choose the constants and initial values as

$$I_1 = 0.5, \quad I_2 = 2, \quad I_3 = 3, \quad y_1(0) = 1, \quad y_2(0) = 0, \quad y_3(0) = 0.9$$

(see Fig. 10.1) and check the numerical precision at the output points

$$x_{\text{end}} = 10 \quad \text{and} \quad x_{\text{end}} = 20.$$

AREN — the Arenstorf orbit (0.1) for the restricted three body problem with initial values (0.2) integrated over one period $0 \leq x \leq x_{\text{end}}$ (see Fig. 0.1). The precision is checked at the endpoint, here the solution is most sensitive to errors of the initial phase.

LRNZ — the solution of the Saltzman-Lorenz equations (I.16.17) displayed in Fig. I.16.8, i.e., with constants and initial values

$$\sigma = 10, \quad r = 28, \quad b = \frac{8}{3}, \quad y_1(0) = -8, \quad y_2(0) = 8, \quad y_3(0) = 27. \quad (10.2)$$

The solution is, for large values of x , *extremely* sensitive to the errors of the first integration steps (see Fig. I.16.10 and its discussion). For example, at $x = 50$ the numerical solution becomes totally wrong, even if the computations are performed in quadruple precision with $\text{Tot} = 10^{-20}$. Hence the numerical results of *all* methods would be equally useless and no comparison makes any sense. Therefore we choose

$$x_{\text{end}} = 16$$

and check the numerical solution at this point. Even here, all computations with $\text{Tot} \geq 10^{-7}$, say, fall into a chaotic cloud of meaningless results (see Fig. 10.5).

PLEI — a celestial mechanics problem (which we call “the Pleiades”): seven stars in the plane with coordinates x_i, y_i and masses $m_i = i$ ($i = 1, \dots, 7$):

$$\begin{aligned} x_i'' &= \sum_{j \neq i} m_j (x_j - x_i) / r_{ij} \\ y_i'' &= \sum_{j \neq i} m_j (y_j - y_i) / r_{ij} \end{aligned} \quad (10.3)$$

where

$$r_{ij} = ((x_i - x_j)^2 + (y_i - y_j)^2)^{3/2}, \quad i, j = 1, \dots, 7.$$

The initial values are

$$\begin{aligned} x_1(0) &= 3, & x_2(0) &= 3, & x_3(0) &= -1, & x_4(0) &= -3, \\ x_5(0) &= 2, & x_6(0) &= -2, & x_7(0) &= 2, \\ y_1(0) &= 3, & y_2(0) &= -3, & y_3(0) &= 2, & y_4(0) &= 0, \\ y_5(0) &= 0, & y_6(0) &= -4, & y_7(0) &= 4, \\ x_i'(0) &= y_i'(0) = 0, & \text{for all } i & \text{with the exception of} \\ x_6'(0) &= 1.75, & x_7'(0) &= -1.5, & y_4'(0) &= -1.25, & y_5'(0) &= 1, \end{aligned} \quad (10.4)$$

and we integrate for $0 \leq t \leq t_{\text{end}} = 3$. Fig. 10.2a represents the movement of these 7 bodies in phase coordinates. The initial value is marked by an “i”, the final value at $t = t_{\text{end}}$ is marked by an “f”. Between these points, 19 time-equidistant output points are plotted and connected by a dense output formula. There occur several quasi-collisions which are displayed in Table 10.1.

Table 10.1. Quasi-collisions in the PLEI problem

Body $_1$	1	1	3	1	2	5
Body $_2$	7	3	5	7	6	7
r_{ij}^2	0.0129	0.0193	0.0031	0.0011	0.1005	0.0700
time	1.23	1.46	1.63	1.68	1.94	2.14

The resulting violent shapes of the derivatives $x'_i(t), y'_i(t)$ are displayed in Fig. 10.2b and show that automatic step size control is essential for this example.

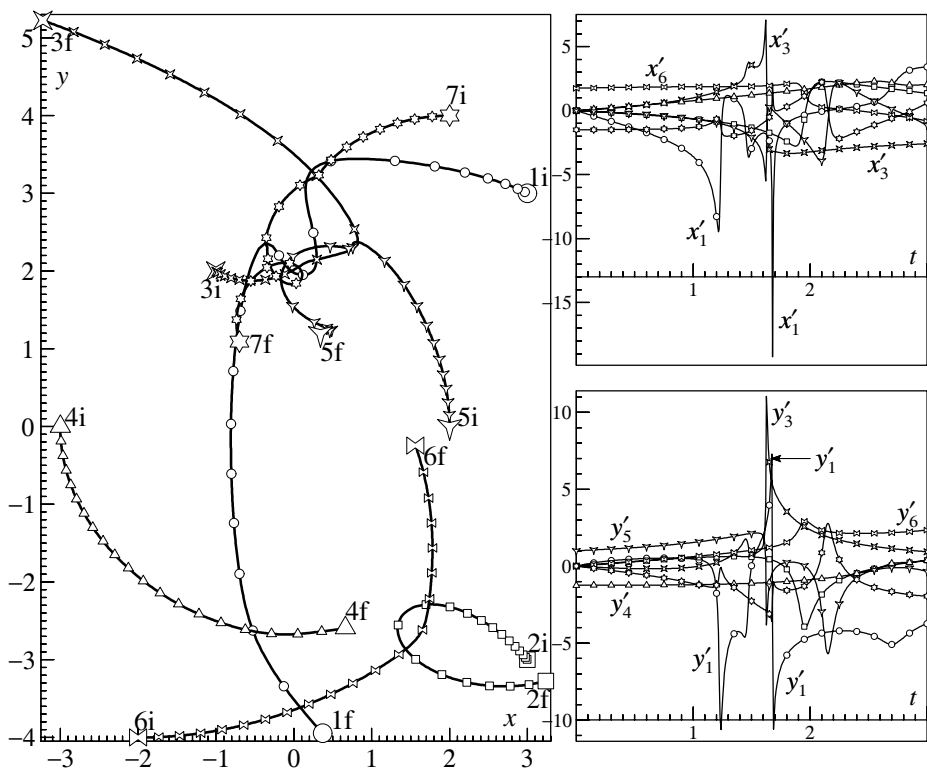


Fig. 10.2a. Solutions of (10.3)

Fig. 10.2b. Speeds

ROPE — the movement of a hanging rope (see Fig. 10.3a) of length 1 under gravitation and under the influence of a horizontal force

$$F_y(t) = \left(\frac{1}{\cosh(4t - 2.5)} \right)^4 \quad (10.5a)$$

acting at the point $s = 0.75$ as well as a vertical force

$$F_x(t) = 0.4 \quad (10.5b)$$

acting at the endpoint $s = 1$.

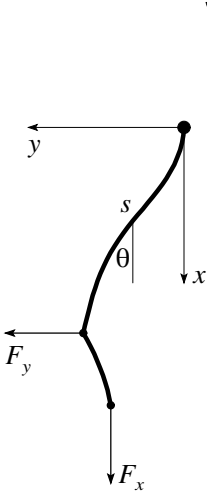


Fig. 10.3a. Hanging rope

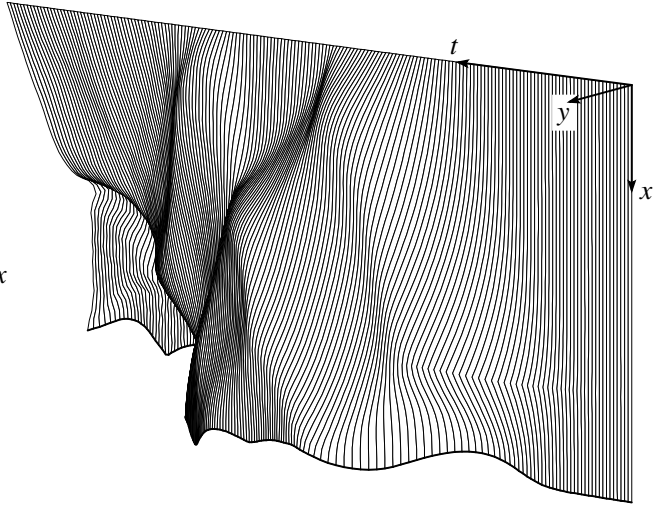


Fig. 10.3b. Solution for $0 \leq t \leq 3.723$.

If this problem is discretized, then Lagrange theory (see (I.6.18); see also Exercises IV.1.2 and IV.1.4 of Volume II) leads to the following equations for the unknown angles θ_k :

$$\sum_{k=1}^n a_{lk} \ddot{\theta}_k = - \sum_{k=1}^n b_{lk} \dot{\theta}_k^2 - n \left(n + \frac{1}{2} - l \right) \sin \theta_l - n^2 \sin \theta_l \cdot F_x(t) + \begin{cases} n^2 \cos \theta_l \cdot F_y(t) & \text{if } l \leq 3n/4 \\ 0 & \text{if } l > 3n/4, \end{cases} \quad l = 1, \dots, n \quad (10.6)$$

where

$$a_{lk} = g_{lk} \cos(\theta_l - \theta_k), \quad b_{lk} = g_{lk} \sin(\theta_l - \theta_k), \quad g_{lk} = n + \frac{1}{2} - \max(l, k). \quad (10.7)$$

We choose

$$n = 40, \quad \theta_l(0) = \dot{\theta}_l(0) = 0, \quad 0 \leq t \leq 3.723. \quad (10.8)$$

The resulting system is of dimension 80. The special structure of G^{-1} (see (IV.1.16–18) of Volume II) allows one to evaluate $\dot{\theta}_l$ with the following algorithm:

- a) Let $v_l = -n(n + \frac{1}{2} - l) \sin \theta_l - n^2 \sin \theta_l \cdot F_x + \begin{cases} n^2 \cos \theta_l \cdot F_y \\ 0 \end{cases}$
- b) Compute $w = Dv + \dot{\theta}^2$,
- c) Solve the tridiagonal system $Cu = w$,
- d) Compute $\ddot{\theta} = Cv + Du$,

where

$$C = \begin{pmatrix} 1 & -\cos(\theta_1 - \theta_2) & & & \\ -\cos(\theta_2 - \theta_1) & 2 & -\cos(\theta_2 - \theta_3) & & \\ & -\cos(\theta_3 - \theta_2) & \ddots & \ddots & \\ & & \ddots & 2 & -\cos(\theta_{n-1} - \theta_n) \\ & & & -\cos(\theta_n - \theta_{n-1}) & 3 \end{pmatrix} \quad (10.9)$$

$$D = \begin{pmatrix} 0 & -\sin(\theta_1 - \theta_2) & & & \\ -\sin(\theta_2 - \theta_1) & 0 & -\sin(\theta_2 - \theta_3) & & \\ & -\sin(\theta_3 - \theta_2) & \ddots & \ddots & \\ & & \ddots & 0 & -\sin(\theta_{n-1} - \theta_n) \\ & & & -\sin(\theta_n - \theta_{n-1}) & 0 \end{pmatrix}.$$

BRUS — the reaction-diffusion equation (Brusselator with diffusion)

$$\begin{aligned} \frac{\partial u}{\partial t} &= 1 + u^2 v - 4.4u + \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \\ \frac{\partial v}{\partial t} &= 3.4u - u^2 v + \alpha \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) \end{aligned} \quad (10.10)$$

for $0 \leq x \leq 1$, $0 \leq y \leq 1$, $t \geq 0$, $\alpha = 2 \cdot 10^{-3}$ together with the Neumann boundary conditions

$$\frac{\partial u}{\partial \mathbf{n}} = 0, \quad \frac{\partial v}{\partial \mathbf{n}} = 0, \quad (10.11)$$

and the initial conditions

$$u(x, y, 0) = 0.5 + y, \quad v(x, y, 0) = 1 + 5x. \quad (10.12)$$

By the method of lines (cf. Section I.6) this problem becomes a system of ordinary differential equations. We put

$$x_i = \frac{i-1}{N-1}, \quad y_j = \frac{j-1}{N-1}, \quad i, j = 1, \dots, N$$

and define

$$U_{ij}(t) = u(x_i, y_j, t), \quad V_{ij}(t) = v(x_i, y_j, t). \quad (10.13)$$

Discretizing the derivatives in (10.10) with respect to the space variables we obtain for $i, j = 1, \dots, N$

$$\begin{aligned} U'_{ij} &= 1 + U_{ij}^2 V_{ij} - 4.4 U_{ij} + \alpha(N-1)^2 (U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1} - 4U_{ij}) \\ V'_{ij} &= 3.4 U_{ij} - U_{ij}^2 V_{ij} + \alpha(N-1)^2 (V_{i+1,j} + V_{i-1,j} + V_{i,j+1} + V_{i,j-1} - 4V_{ij}), \end{aligned} \quad (10.14)$$

an ODE of dimension $2N^2$. Because of the boundary condition (10.11) we have

$$U_{0,j} = U_{2,j}, \quad U_{N+1,j} = U_{N-1,j}, \quad U_{i,0} = U_{i,2}, \quad U_{i,N+1} = U_{i,N-1}$$

and similarly for the V_{ij} -quantities. We choose $N = 21$ so that the system is of dimension 882 and check the numerical solutions at the output point $t_{\text{end}} = 7.5$. The solution of (10.14) (in the (x, y) -space) is represented in Fig. 10.4a and Fig. 10.4b for u and v respectively.

Performance of the Codes

Several codes were applied to each of the test problems with $Tol = 10^{-3}$, $Tol = 10^{-3-1/8}$, $Tol = 10^{-3-2/8}$, $Tol = 10^{-3-3/8}$, \dots (for the large problems with $Tol = 10^{-3}$, $Tol = 10^{-3-1/4}$, $Tol = 10^{-3-2/4}$, \dots) up to, in general, $Tol = 10^{-14}$, then the numerical result at the output points were compared with an “exact solution” (computed very precisely in quadruple precision). Each of these results then corresponds to one point of Fig. 10.5, where this precision is compared (in double logarithmic scale) to the number of function evaluations. The “integer” tolerances 10^{-3} , 10^{-4} , 10^{-5} , \dots are distinguishable as enlarged symbols. All codes were applied with complete “standard” parameter settings and were not at all “tuned” to these particular problems.

A comparison of the *computing time* (instead of the number of function evaluations) gave no significant difference. Therefore, only one representative of the small problems (LRNZ) and one large problem (BRUS) are displayed in Fig. 10.6. All computations have been performed in REAL*8 ($U_{\text{round}} = 1.11 \cdot 10^{-16}$) on a Sun Workstation (SunBlade 100).

The codes used are the following:

RKF45 — symbol \star — a product of Shampine and Watts’ programming art based on Fehlberg’s pair of orders 4 and 5 (Table 5.1). The method is used in the “local extrapolation mode”, i.e., the numerical solution is advanced with the 5th order result. The code is usually, except for low precision, the slowest of all, which is explained by its low order. The results of the “time”-picture Fig. 10.6 for this

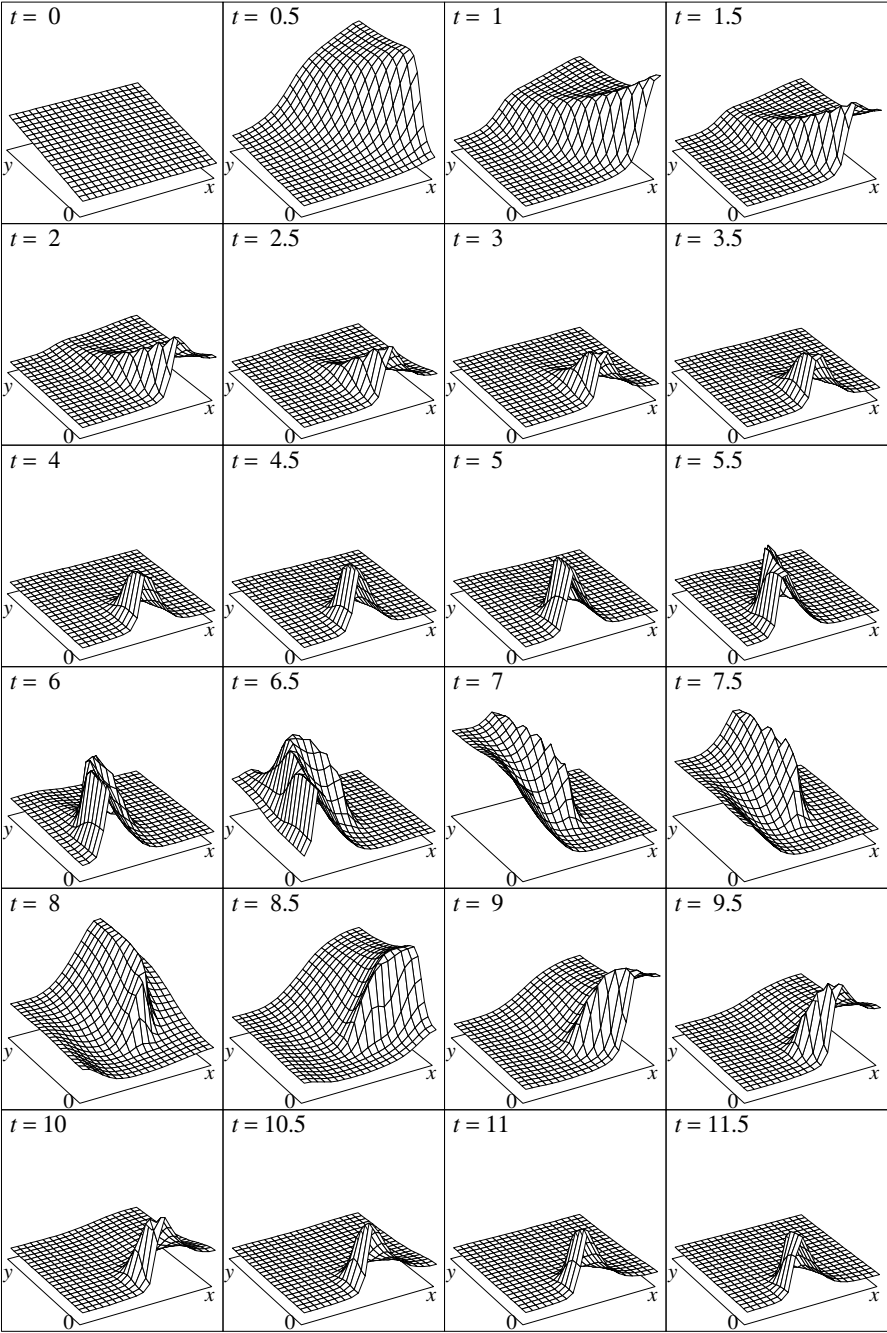


Fig. 10.4a. Solution $u(x, y, t)$ for the BRUS problem

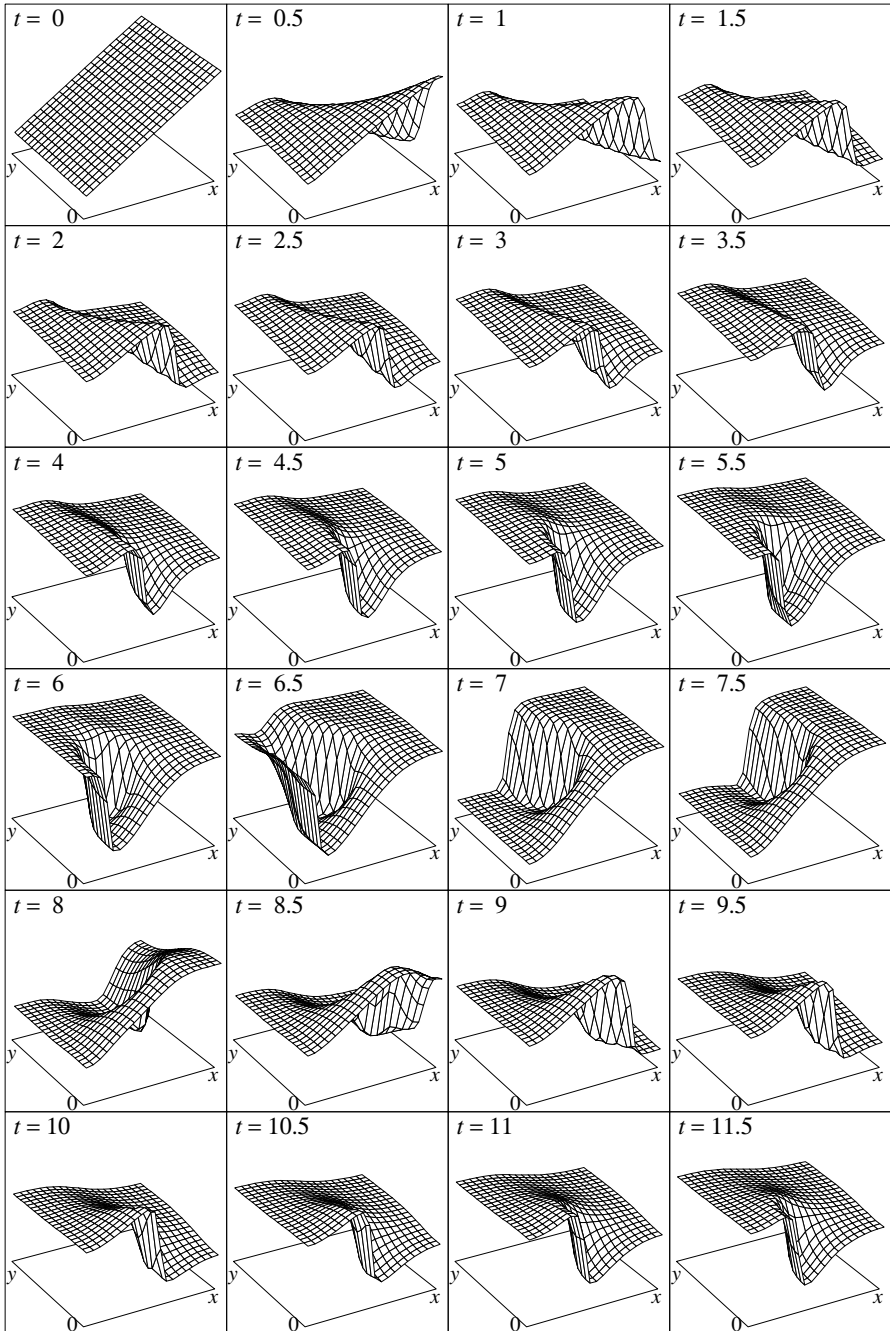


Fig. 10.4b. Solution $v(x, y, t)$ for the BRUS problem

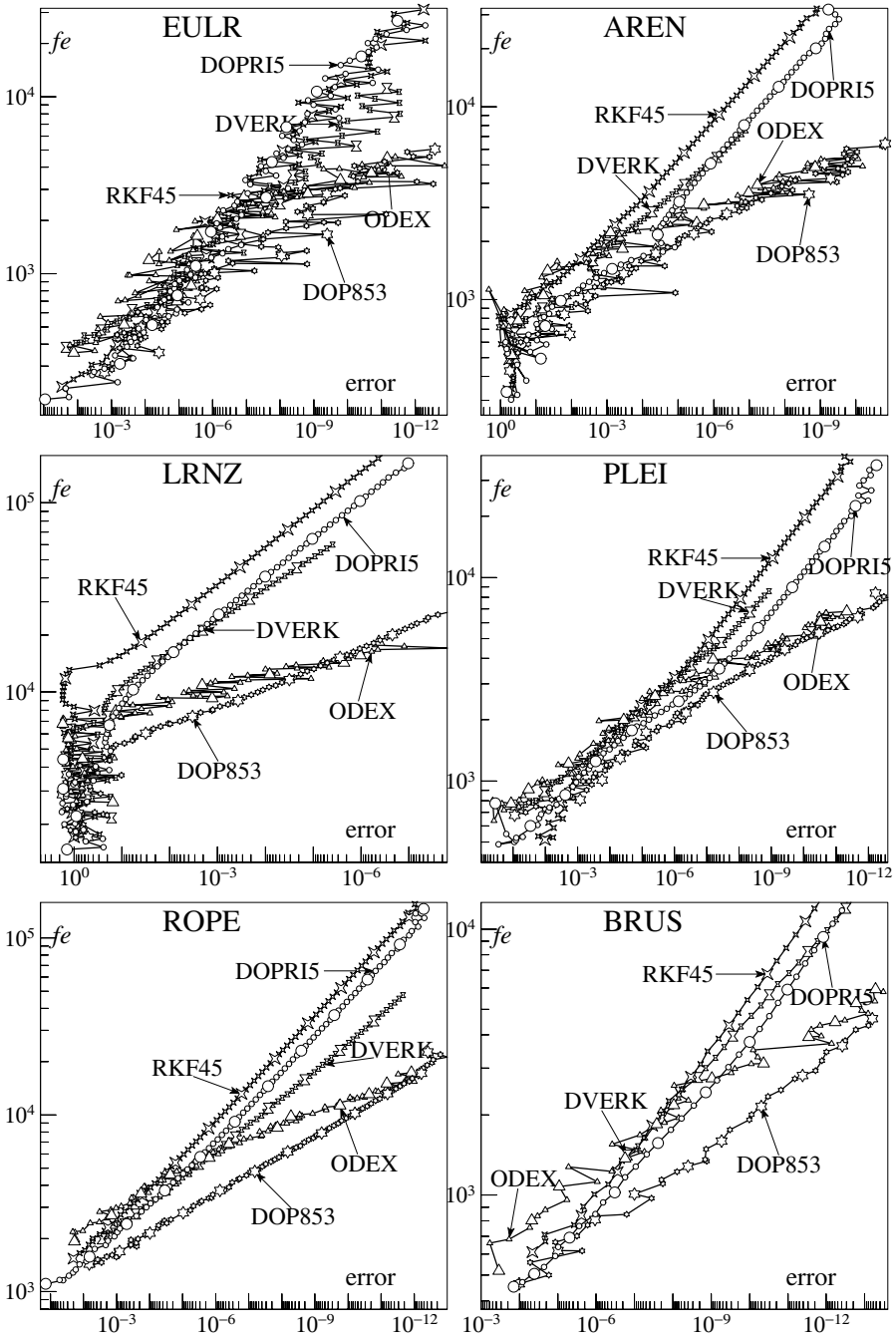


Fig. 10.5. Precision versus function calls

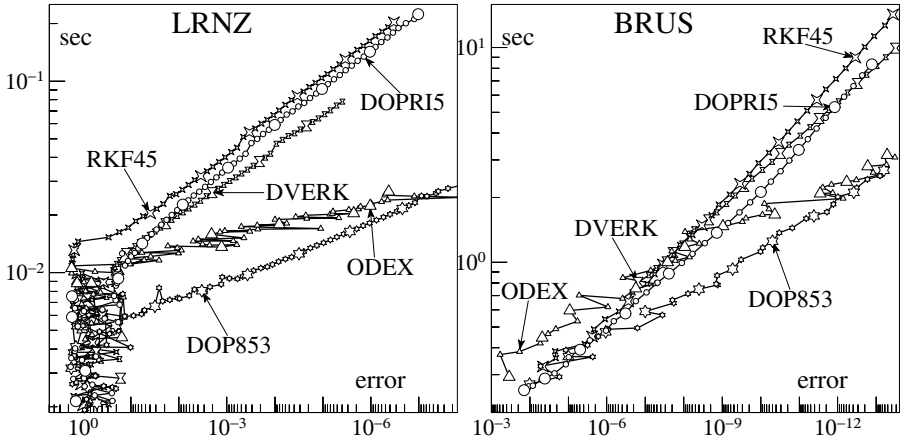


Fig. 10.6. Precision versus computing time

code are relatively better than those on the “function calls” front (Fig. 10.5). This indicates that the code has particularly small overhead.

DOPRI5 — symbol \bigcirc — the method of Dormand & Prince of order 5 with embedded error estimator of order 4 (see Table 5.2). The code is explained in the Appendix. The method has precisely the same order as that used in RKF45, but the error constants are much more optimized. Therefore the “error curves” in Fig. 10.5 are nicely parallel to those of RKF45, but appear translated to the side of higher precision. One usually gains between a half and one digit of numerical precision for comparable numerical work. The code performs specially well between $Tol = 10^{-3}$ and $Tol = 10^{-8}$ in the AREN problem. This is simply due to an accidental sign change of the error for the most sensitive solution component.

DVERK — symbol Σ — this widely known code implements Verner’s 6th order method of Table 5.4 and was written by Hull, Enright & Jackson. It has been included in the IMSL library for many years and the source code is available through na-net. The corresponding error curves in Fig. 10.5 appear to be less steep than those of DOPRI5, which illustrates the higher order of the method. However, the error constants seem to be less optimal so that this code surpasses the performance of DOPRI5 only for very stringent tolerances. It is significantly better than DOPRI5 solely in problems EULR and ROPE. The code, as it was, failed at the BRUS problem for $Tol = 10^{-3}$ and $Tol = 10^{-4}$. Therefore these computations were started with $Tol = 10^{-5}$.

DOP853 — symbol \star — is the method of Dormand & Prince of order 8 explained in Section II.5 (formulas (5.20) – (5.30), see Appendix). The 6th order error estimator (5.29), (5.30) has been replaced by a 5th order estimator with 3rd order correction (see below). This was necessary to make the code robust for the

EULR problem. The code works perfectly for all problems and nearly all tolerances. Whenever more than 3 or 4 digits are desired, this method seems to be highly recommendable. The most astonishing fact is that its use was never disastrous, even not for $Tol = 10^{-3}$.

ODEX — symbol \triangle — is an extrapolation code based on the Gragg-Bulirsch-Stoer algorithm with harmonic step number sequence (see Appendix). This method, which allows arbitrary high orders (in the standard version of the code limited to $p \leq 18$) is of course predestined for computations with high precision. The more stringent Tol is, the higher the used order becomes, the less steep the error curve is. This can best be observed in the picture for the ROPE problem. Finally, for $Tol \approx 10^{-12}$, the code surpasses the values of DOP853. As can be seen in Fig. 10.6, the code loses slightly on the “time”-front. This is due to the increased overhead of the extrapolation scheme.

The numerical results of ODEX behave very similarly to those of DIFEX1 (Deuffhard 1983).

A “Stretched” Error Estimator for DOP853

In preliminary stages of our numerical tests we had written a code “DOPR86” based on the method of order 8 of Dormand & Prince with the 6th order error estimator described in Section II.5. For most problems the results were excellent. However, there are some situations in which the error control of DOPR86 did not work safely:

When applied to the BRUS problem with $Tol = 10^{-3}$ or $Tol = 10^{-4}$ the code stopped with an overflow message. The reason was the following: when the step size is too large, the internal stages are too far away from the solution and their modulus increases at each stage (e.g., by a factor 10^5 between stage 11 and stage 12). Due to the fact that $\hat{b}_{12} = b_{12}$ (see (5.30) (5.26) and (5.25b)) the difference $\hat{y}_1 - y_1$ is not influenced by the last stage and is smaller (by a factor of 10^5) than the modulus of y_1 . Hence, the error estimator scaled by (4.10) is $\leq 10^{-5}$ and a completely wrong step will be accepted.

The code DOPR86 also had severe difficulties when applied to problems with discontinuities such as EULR. The worst results were obtained for the problem

$$\begin{aligned} y_1' &= y_2 y_3 & y_1(0) &= 0 \\ y_2' &= -y_3 y_1 & y_2(0) &= 1 \\ y_3' &= -0.51 \cdot y_1 y_2 + f(x) & y_3(0) &= 1 \end{aligned} \quad (10.15)$$

where $f(x)$, given in (10.1'), has a discontinuous second derivative. The results for this problem and the code DOPR86 for very many different Tol values ($Tol = 10^{-3}, 10^{-3-1/24}, 10^{-3-2/24}, \dots, 10^{-14}$) are displayed in Fig. 10.7. There,

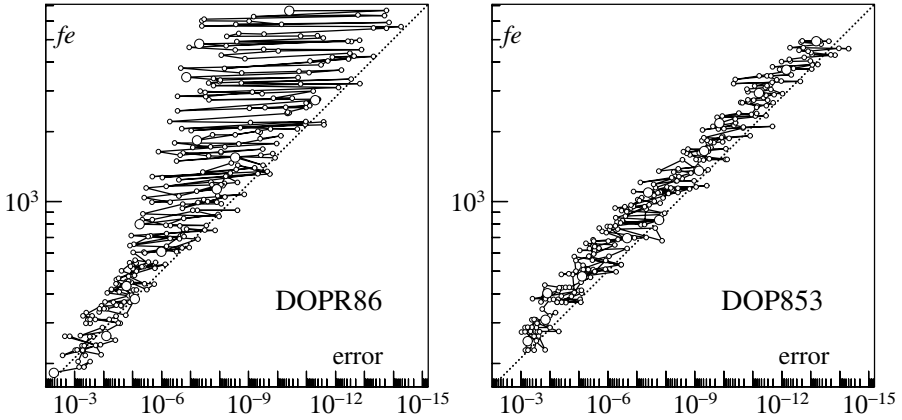


Fig. 10.7. Performances of DOPR86 and DOP853 at (10.15)

the (dotted) diagonal is of exact slope $1/8$ and represents the theoretical convergence speed of the method of order 8. It can be observed that this convergence is well attained by *some* results, but others lose precision of up to 8 digits from the desired tolerance. We explain this disappointing behaviour by the fact that $\hat{b}_{12} = b_{12}$ and that the 12th stage is the only one where the function is evaluated at the end-point of the step. Whenever the discontinuity of f'' is by accident slightly to the left of a grid point, the error estimator ignores it and the code reports a wrong value.

Unfortunately, the basic 8th order method does not possess a 6th order embedding with $\hat{b}_{12} \neq b_{12}$ (unless additional function evaluations are used). Therefore, we decided to construct a 5th order approximation \hat{y}_1 . It can be obtained by taking $\hat{b}_6, \hat{b}_7, \hat{b}_{12}$ as free parameters, e.g.,

$$\hat{b}_6 = b_6/2 + 1, \quad \hat{b}_7 = b_7/2 + 0.45, \quad \hat{b}_{12} = b_{12}/2,$$

by putting $\hat{b}_2 = \hat{b}_3 = \hat{b}_4 = \hat{b}_5 = 0$ and by determining the remaining coefficients such that this quadrature formula has order 5. Due to the simplifying assumptions (5.20) all conditions for order 5 are then satisfied. In order to prevent a serious *over-estimation* of the error, we consider a second embedded method \tilde{y}_1 of order 3 based on the nodes $c_1 = 0$, c_9 and $c_{12} = 1$ so that two error estimators

$$err_5 = \|\hat{y}_1 - y_1\| = \mathcal{O}(h^6), \quad err_3 = \|\tilde{y}_1 - y_1\| = \mathcal{O}(h^4) \quad (10.16)$$

are available. Similarly to a procedure which is common for quadrature formulas (R. Piessens, E. de Doncker-Kapenga, C.W. Überhuber & D.K. Kahaner 1983, Berntsen & Espelid 1991) we consider

$$err = err_5 \cdot \frac{err_5}{\sqrt{err_5^2 + 0.01 \cdot err_3^2}} = \mathcal{O}(h^8) \quad (10.17)$$

as error estimator. It behaves asymptotically like the global error of the method. The corresponding code DOP853 gives satisfactory results for all the above problems (see right picture in Fig. 10.7).

Effect of Step-Number Sequence in ODEX

We also study the influence of the different step-number sequences to the performance of the extrapolation code ODEX. Fig. 10.8 presents two examples of this study, a small problem (AREN) and a large problem (ROPE). The used sequences are

HARMONIC — symbol \bigcirc — the harmonic sequence (9.8') which is the standard choice in ODEX;

MOD4 — symbol \triangle — the sequence $\{2, 6, 10, 14, 18, \dots\}$ (see (9.35)) which allowed the construction of high-order dense output;

BULIRSCH — symbol \square — the Bulirsch sequence (9.7');

ROMBERG — symbol \diamond — the Romberg sequence (9.6');

DNSECTRL — symbol \star — the error control for the MOD4 sequence taking into account the interpolation error of the dense output solution (9.42). This is included only in the small problem, since (complete) dense output on large problems would need too much memory.

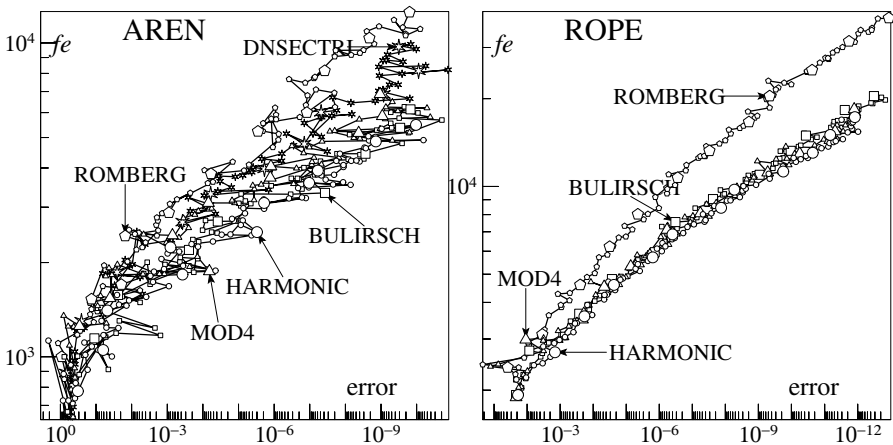


Fig. 10.8. Effect of step-number sequences in ODEX

Discussion. With the exception of the clear inferiority of the Romberg sequence, especially for high precision, and a certain price to be paid for the dense output error control, there is not much difference between the first three sequences. Although the harmonic sequence appears to be slightly superior, the difference is statistically not very significant.

II.11 Parallel Methods

We suppose that we have a computer with a number of arithmetic processors capable of simultaneous operation and seek to devise parallel integration algorithms for execution on such a computer.

(W.L. Miranker & W. Liniger 1967)

“PARALYSING ODES” (K. Burrage, talk in Helsinki 1990)

Parallel machines are computers with more than one processor and this facility might help us to speed up the computations in ordinary differential equations. This is particularly interesting for very large problems, for very costly function evaluation, or for fast real-time simulations. A second motivation is the desire to make a code, with the help of parallel computations, not necessarily faster but more robust and reliable.

Early attempts for finding parallel methods are Nievergelt (1964) and Miranker & Liniger (1967). See also the survey papers Miranker (1971) and Jackson (1991).

We distinguish today essentially between two types of parallel architectures:

SIMD (single instruction multiple data): all processors execute the same instructions with possibly different input data.

MIMD (multiple instruction multiple data): the different processors can act independently.

The exploitation of parallelism for an ordinary differential equation

$$y' = f(x, y), \quad y(x_0) = y_0 \quad (11.1)$$

can be classified into two main categories (Gear 1987, 1988):

Parallelism across the system. Often the problem itself offers more or less trivial applications for parallelism, e.g.,

- > if several solutions are required for various initial or parameter values;
- > if the right-hand side of (11.1) is very costly, but structured in such a way that the computation of *one* function evaluation can be split efficiently across the various processors;
- > space discretizations of partial differential equations (such as the Brusselator problem (10.14)) whose function evaluation can be done simultaneously for all components on an SIMD machine with thousands of processors;
- > the solution of boundary value problems with the multiple shooting method (see Section I.15) where all computations on the various sub-intervals can be done in parallel;

- > doing all the high-dimensional linear algebra in the Runge-Kutta method (11.2) in parallel;
- > parallelism in the linear algebra for Newton's method for *implicit* Runge-Kutta methods (see Section IV.8).

These types of parallelism, of course, depend strongly on the problem and on the type of the computer.

Parallelism across the method. This is problem-independent and means that, due to a special structure of the method, several function values can be evaluated in parallel within one integration step. This will be discussed in this section in more detail.

Parallel Runge-Kutta Methods

... it seems that *explicit* Runge-Kutta methods are not facilitated much by parallelism at the method level.

(Iserles & Nørsett 1990)

Consider an explicit Runge-Kutta method

$$k_i = f\left(x_0 + c_i h, y_0 + h \sum_{j=1}^{i-1} a_{ij} k_j\right), \quad i = 1, \dots, s$$

$$y_1 = y_0 + h \sum_{i=1}^s b_i k_i.$$
(11.2)

Suppose, for example, that the coefficients have the zero-pattern indicated in Fig. 11.1.

0				
×	×			
×	×	0		
×	×	×	×	
	×	×	×	×

Fig. 11.1. Parallel method

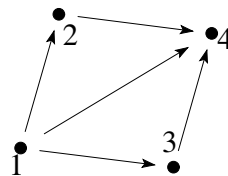


Fig. 11.2. Production graph

Each arrow in the corresponding “production graph” G (Fig. 11.2), pointing from vertex “ i ” to vertex “ j ”, stands for a non-zero a_{ji} . Here the vertices 2 and 3 are independent and can be evaluated in parallel. We call the number of vertices in the longest chain of successive arrows (here 3) the *number of sequential function evaluations* σ .

In general, if the Runge-Kutta matrix A can be partitioned (possibly after a permutation of the stages) as

$$A = \begin{pmatrix} 0 & & & & \\ A_{21} & 0 & & & \\ A_{31} & A_{32} & 0 & & \\ \vdots & \vdots & & \ddots & \\ A_{\sigma 1} & A_{\sigma 2} & \dots & A_{\sigma, \sigma-1} & 0 \end{pmatrix}, \quad (11.3)$$

where A_{ij} is a matrix of size $\mu_i \times \mu_j$, then the derivatives k_1, \dots, k_{μ_1} as well as $k_{\mu_1+1}, \dots, k_{\mu_1+\mu_2}$, and so on, can be computed in parallel and one step of the method is executed in σ sequential function evaluations (if $\mu = \max_i \mu_i$ processors are at disposal). The following theorem is a severe restriction on parallel methods. It appeared in hand-written notes by K. Jackson & S. Nørsett around 1986. For a publication see Jackson & Nørsett (1992) and Iserles & Nørsett (1990).

Theorem 11.1. *For an explicit Runge-Kutta method with σ sequential stages the order p satisfies*

$$p \leq \sigma, \quad (11.4)$$

for any number μ of available processors.

Proof. Each non-zero term of the expressions $\Phi_i(t)$ for the “tall” trees $t_{21}, t_{32}, t_{44}, t_{59}, \dots$ (see Table 2.2 and Definition 2.9) $\sum a_{ij} a_{jk} a_{k\ell} a_{\ell m} \dots$ corresponds to a connected chain of arrows in the production graph. Since their length is limited by σ , these terms are all zero for $\varrho(t) > \sigma$. \square

Methods with $p = \sigma$ will be called *P-optimal methods*. The Runge-Kutta methods of Section II.1 for $p \leq 4$ are all P-optimal. Only for $p > 4$ does the subsequent construction of P-optimal methods allow one to increase the order with the help of parallelism.

Remark. The fact that the “stability function” (see Section IV.2) of an explicit parallel Runge-Kutta method is a polynomial of degree $\leq \sigma$ allows a second proof of Theorem 11.1. Further, P-optimal methods all have the same stability function $1 + z + z^2/2! + \dots + z^\sigma/\sigma!$.

Parallel Iterated Runge-Kutta Methods

One possibility of constructing P-optimal methods is by fixed point iteration. Consider an arbitrary (explicit or implicit) Runge-Kutta method with coefficients

$$c = (c_1, \dots, c_s)^T, \quad A = (a_{ij})_{i,j=1}^s, \quad b^T = (b_1, \dots, b_s)$$

and define \hat{y}_1 by

$$\begin{aligned} k_i^{(0)} &= 0 \\ k_i^{(\ell)} &= f\left(x_0 + c_i h, y_0 + h \sum_{j=1}^s a_{ij} k_j^{(\ell-1)}\right), \quad \ell = 1, \dots, \sigma \\ \hat{y}_1 &= y_0 + h \sum_{i=1}^s b_i k_i^{(\sigma)}. \end{aligned} \quad (11.5)$$

This algorithm can be interpreted as an explicit Runge-Kutta method with scheme

$$\begin{array}{c|cccccc} 0 & 0 & & & & \\ c & A & 0 & & & \\ c & 0 & A & 0 & & \\ \vdots & \vdots & & \ddots & \ddots & \\ c & 0 & \dots & 0 & A & 0 \\ \hline & 0 & \dots & 0 & 0 & b^T \end{array} \quad (11.6)$$

It has σ sequential stages if s processors are available. To compute its order we use a Lipschitz condition for $f(x, y)$ and obtain

$$\max_i \|k_i^{(\ell)} - k_i\| \leq Ch \cdot \max_i \|k_i^{(\ell-1)} - k_i\|$$

where k_i are the stage-vectors of the basic method. Since $k_i^{(0)} - k_i = \mathcal{O}(1)$ this implies $k_i^{(\sigma)} - k_i = \mathcal{O}(h^\sigma)$ and consequently the difference to the solution of the basic method satisfies $\hat{y}_1 - y_1 = \mathcal{O}(h^{\sigma+1})$.

Theorem 11.2. *The parallel iterated Runge-Kutta method (11.5) is of order*

$$p = \min(p_0, \sigma), \quad (11.7)$$

if p_0 denotes the order of the basic method.

Proof. The statement follows from

$$\hat{y}_1 - y(x_0 + h) = \hat{y}_1 - y_1 + y_1 - y(x_0 + h) = \mathcal{O}(h^{\sigma+1}) + \mathcal{O}(h^{p_0+1}). \quad \square$$

This theorem shows that the choice $\sigma = p_0$ in (11.5) yields P-optimal explicit Runge-Kutta methods (i.e., $\sigma = p$). If we take as basic method the s -stage collocation method based on the Gaussian quadrature ($p_0 = 2s$) then we obtain a method of order $p = 2s$ which is P-optimal on s processors. P.J. van der Houwen

& B.P. Sommeijer (1990) have done extensive numerical experiments with this method.

Extrapolation Methods

It turns out that the GBS-algorithm (Section II.9) without smoothing step is also P-optimal. Indeed, all the values T_{j1} can be computed independently of each other. If we choose the step number sequence $\{2, 4, 6, 8, 10, 12, \dots\}$ then the computation of T_{k1} requires $2k$ sequential function evaluations. Hence, if k processors are available (one for each T_{j1}), the numerical approximation T_{kk} , which is of order $p=2k$, can be computed with $\sigma=2k$ sequential stages. When the processors are of type MIMD we can compute T_{11} and $T_{k-1,1}$ on one processor ($2+2(k-1)=2k$ function evaluations). Similarly, T_{21} and $T_{k-2,1}$ occupy another processor, etc. In this way, the number of necessary processors is reduced by a factor close to 2 without increasing the number of sequential stages.

The order and step size strategy, discussed in Section II.9, should, of course, be adapted for an implementation on parallel computers. The “hope for convergence in line $k+1$ ” no longer makes sense because this part of the algorithm is now as costly as the whole step. Similarly, there is no reason to accept already $T_{k-1,k-1}$ as numerical approximation, because T_{kk} is computed on the same time level as $T_{k-1,k-1}$. Moreover, the numbers A_k of (9.25) should be replaced by $A_k = n_k$ which will in general increase the order used by the code.

Increasing Reliability

... using parallelism to improve *reliability* and *functionality*
rather than efficiency. (W.H. Enright & D.J. Higham 1991)

For a given Runge-Kutta method parallel computation can be used to give a reliable error estimate or an accurate dense output. This has been advocated by Enright & Higham (1991) and will be the subject of this subsection.

Consider a Runge-Kutta method of order p , choose distinct numbers $0 = \sigma_0 < \sigma_1 < \dots < \sigma_k = 1$ and apply the Runge-Kutta method in parallel with step sizes $\sigma_1 h, \dots, \sigma_{k-1} h, \sigma_k h = h$. This gives approximations

$$y_{\sigma_i} \approx y(x_0 + \sigma_i h) . \quad (11.8)$$

Then compute $f(x_0 + \sigma_i h, y_{\sigma_i})$ and do Hermite interpolation with the values

$$y_{\sigma_i}, h f(x_0 + \sigma_i h, y_{\sigma_i}), \quad i = 0, 1, \dots, k, \quad (11.9)$$

i.e., compute

$$u(\theta) = \sum_{i=0}^k v_i(\theta) y_{\sigma_i} + h \sum_{i=0}^k w_i(\theta) f(x_0 + \sigma_i h, y_{\sigma_i}) \quad (11.10)$$

where $v_i(\theta)$ and $w_i(\theta)$ are the scalar polynomials

$$\left. \begin{aligned} v_i(\theta) &= \ell_i^2(\theta) \cdot (1 - 2\ell_i'(\sigma_i)(\theta - \sigma_i)) \\ w_i(\theta) &= \ell_i^2(\theta) \cdot (\theta - \sigma_i) \end{aligned} \right\} \text{ with } \ell_i(\theta) = \prod_{\substack{j=0 \\ j \neq i}}^k \frac{(\theta - \sigma_j)}{(\sigma_i - \sigma_j)}. \quad (11.11)$$

The interpolation error, which is $\mathcal{O}(h^{2k+2})$, may be neglected if $2k+2 > p+1$.

As to the choice of σ_i we denote the local error of the method by $le = y_1 - y(x_0 + h)$. It follows from Taylor expansion (see Theorem 3.2) that

$$y_{\sigma_i} - y(x_0 + \sigma_i h) = \sigma_i^{p+1} \cdot le + \mathcal{O}(h^{p+2})$$

and consequently the error of (11.10) satisfies (for $2k+2 > p+1$)

$$u(\theta) - y(x_0 + \theta h) = \left(\sum_{i=1}^k \sigma_i^{p+1} v_i(\theta) \right) \cdot le + \mathcal{O}(h^{p+2}). \quad (11.12)$$

The coefficient of le is equal to 1 for $\theta = 1$ and it is natural to search for suitable σ_i such that

$$\left| \sum_{i=1}^k \sigma_i^{p+1} v_i(\theta) \right| \leq 1 \quad \text{for all } \theta \in [0, 1]. \quad (11.13)$$

Indeed, under the assumption $2k-1 \leq p < 2k+1$, it can be shown that numbers $0 = \sigma_0 < \sigma_1 < \dots < \sigma_{k-1} < \sigma_k = 1$ exist satisfying (11.13) (see Exercise 1). Selected values of σ_i proposed by Enright & Higham (1991), which satisfy this condition are given in Table 11.1. For such a choice of σ_i the error (11.12) of the dense output is bounded (at least asymptotically) by the local error le at the endpoint of integration. This implementation of a dense output provides a simple way to estimate le . Since $u(\theta)$ is an $\mathcal{O}(h^{p+1})$ -approximation of $y(x_0 + \theta h)$, the defect of $u(\theta)$ satisfies

$$u'(\theta) - hf(x_0 + \theta h, u(\theta)) = \left(\sum_{i=1}^k \sigma_i^{p+1} v_i'(\theta) \right) \cdot le + \mathcal{O}(h^{p+2}). \quad (11.14)$$

If we take a σ^* different from σ_i such that $\sum_{i=1}^k \sigma_i^{p+1} v_i'(\sigma^*) \neq 0$ (see Table 11.1) then only one function evaluation, namely $f(x_0 + \sigma^* h, u(\sigma^*))$, allows the computation of an asymptotically correct approximation of le from (11.14). This error estimate can be used for step size selection and for improving the numerical result (local extrapolation). In the local extrapolation mode one then loses the \mathcal{C}^1 continuity of the dense output.

With the use of an additional processor the quantities y_{σ^*} and $f(x_0 + \sigma^* h, y_{\sigma^*})$ can be computed simultaneously with y_{σ_i} and $f(x_0 + \sigma_i h, y_{\sigma_i})$. If the polynomial $u(\theta)$ is required to satisfy $u(\sigma^*) = y_{\sigma^*}$, but not $u'(\sigma^*) = hf(x_0 + \sigma^* h, y_{\sigma^*})$, then the estimate (11.14) of the local error le does not need any further evaluation of f .

Table 11.1. Good values for σ_i

p	k	$\sigma_1, \dots, \sigma_{k-1}$	σ^*
5	3	0.2, 0.4	0.88
6	3	0.2, 0.4	0.88
7	4	0.2, 0.4, 0.7	0.94
8	4	0.2, 0.4, 0.6	0.93

Exercises

1. Let the positive integers k and p satisfy $2k - 1 \leq p < 2k + 1$. Then show that there exist numbers $0 = \sigma_0 < \sigma_1 < \dots < \sigma_{k-1} < \sigma_k = 1$ such that (11.13) is true for all $\theta \in [0, 1]$.

Hint. Put $\sigma_j = j\varepsilon$ for $j = 1, \dots, k - 1$ and show that (11.13) is verified for sufficiently small $\varepsilon > 0$. Of course, in a computer program, one should use σ_j which satisfy (11.13) and are well separated in order to avoid roundoff errors.

II.12 Composition of B-Series

At the Dundee Conference in 1969, a paper by J. Butcher was read which contained a surprising result. (H.J. Stetter 1971)

We shall now derive a theorem on the composition of what we call B-series (in honour of J. Butcher). This will have many applications and will lead to a better understanding of order conditions for all general classes of methods (composition of methods, multiderivative methods of Section II.13, general linear methods of Section III.8, Rosenbrock methods in Exercise 2 of Section IV.7).

Composition of Runge-Kutta Methods

There is no five-stage explicit Runge-Kutta method of order 5 (Section II.5). This led Butcher (1969) to the idea of searching for different five-stage methods such that a certain *composition* of these methods produces a fifth-order result (“effective order”). Although not of much practical interest (mainly due to the problem of changing step size), this was the starting point of a fascinating algebraic theory of numerical methods.

Suppose we have two methods, say of three stages,

$$\begin{array}{c|ccc}
 0 & & & \\
 \hat{c}_2 & \hat{a}_{21} & & \\
 \hat{c}_3 & \hat{a}_{31} & \hat{a}_{32} & \\
 \hline
 & \hat{b}_1 & \hat{b}_2 & \hat{b}_3
 \end{array}
 \qquad
 \begin{array}{c|ccc}
 0 & & & \\
 \tilde{c}_2 & \tilde{a}_{21} & & \\
 \tilde{c}_3 & \tilde{a}_{31} & \tilde{a}_{32} & \\
 \hline
 & \tilde{b}_1 & \tilde{b}_2 & \tilde{b}_3
 \end{array}
 \quad (12.1)$$

which are applied one after the other to a starting value y_0 with the same step size:

$$g_i = y_0 + h \sum_j \hat{a}_{ij} f(g_j), \quad y_1 = y_0 + h \sum_j \hat{b}_j f(g_j) \quad (12.2)$$

$$\ell_i = y_1 + h \sum_j \tilde{a}_{ij} f(\ell_j), \quad y_2 = y_1 + h \sum_j \tilde{b}_j f(\ell_j). \quad (12.3)$$

If we insert y_1 from (12.2) into (12.3) and group all g_i, ℓ_i together, we see that the

composition can be interpreted as a large Runge-Kutta method with coefficients

$$\begin{array}{c|cccc}
 0 & & & & \\
 \widehat{c}_2 & \widehat{a}_{21} & & & \\
 \widehat{c}_3 & \widehat{a}_{31} & \widehat{a}_{32} & & \\
 \sum \widehat{b}_i & \widehat{b}_1 & \widehat{b}_2 & \widehat{b}_3 & \\
 \sum \widehat{b}_i + \widetilde{c}_2 & \widehat{b}_1 & \widehat{b}_2 & \widehat{b}_3 & \widetilde{a}_{21} \\
 \sum \widehat{b}_i + \widetilde{c}_3 & \widehat{b}_1 & \widehat{b}_2 & \widehat{b}_3 & \widetilde{a}_{31} \quad \widetilde{a}_{32} \\
 \hline
 & \widehat{b}_1 & \widehat{b}_2 & \widehat{b}_3 & \widetilde{b}_1 \quad \widetilde{b}_2 \quad \widetilde{b}_3
 \end{array}
 \equiv
 \begin{array}{c|cccccc}
 0 & & & & & \\
 c_2 & a_{21} & & & & \\
 c_3 & a_{31} & a_{32} & & & \\
 c_4 & a_{41} & a_{42} & a_{43} & & \\
 c_5 & a_{51} & a_{52} & a_{53} & a_{54} & \\
 c_6 & a_{61} & a_{62} & a_{63} & a_{64} & a_{65} \\
 \hline
 & b_1 & b_2 & b_3 & b_4 & b_5 \quad b_6
 \end{array}
 \quad (12.4)$$

It is now of interest to study the *order conditions* of the new method. For this, we have to compute the expressions (see Table 2.2)

$$\sum b_i, \quad 2 \sum b_i c_i, \quad 3 \sum b_i c_i^2, \quad 6 \sum b_i a_{ij} c_j, \quad \text{etc.}$$

If we insert the values from the left tableau of (12.4), a computation, which for low orders is still not too difficult, shows that these expressions can be written in terms of the corresponding expressions for the two methods (12.1). We shall denote these expressions for the *first* method by $\mathbf{a}(t)$, for the *second* method by $\mathbf{b}(t)$, and for the *composite* method by $\mathbf{ab}(t)$:

$$\mathbf{a}(\cdot) = \sum \widehat{b}_i, \quad \mathbf{a}(\cdot) = 2 \cdot \sum \widehat{b}_i \widehat{c}_i, \quad \mathbf{a}(\cdot) = 3 \cdot \sum \widehat{b}_i \widehat{c}_i^2, \quad \dots \quad (12.5a)$$

$$\mathbf{b}(\cdot) = \sum \widetilde{b}_i, \quad \mathbf{b}(\cdot) = 2 \cdot \sum \widetilde{b}_i \widetilde{c}_i, \quad \mathbf{b}(\cdot) = 3 \cdot \sum \widetilde{b}_i \widetilde{c}_i^2, \quad \dots \quad (12.5b)$$

$$\mathbf{ab}(\cdot) = \sum b_i, \quad \mathbf{ab}(\cdot) = 2 \cdot \sum b_i c_i, \quad \mathbf{ab}(\cdot) = 3 \cdot \sum b_i c_i^2, \quad \dots \quad (12.5c)$$

The above mentioned formulas are then

$$\begin{aligned}
 \mathbf{ab}(\cdot) &= \mathbf{a}(\cdot) + \mathbf{b}(\cdot) \\
 \mathbf{ab}(\cdot) &= \mathbf{a}(\cdot) + 2\mathbf{b}(\cdot)\mathbf{a}(\cdot) + \mathbf{b}(\cdot) \\
 \mathbf{ab}(\cdot) &= \mathbf{a}(\cdot) + 3\mathbf{b}(\cdot)\mathbf{a}(\cdot)^2 + 3\mathbf{b}(\cdot)\mathbf{a}(\cdot) + \mathbf{b}(\cdot) \\
 \mathbf{ab}(\cdot) &= \mathbf{a}(\cdot) + 3\mathbf{b}(\cdot)\mathbf{a}(\cdot) + 3\mathbf{b}(\cdot)\mathbf{a}(\cdot) + \mathbf{b}(\cdot)
 \end{aligned}
 \quad (12.6)$$

etc.

It is now, of course, of interest to have a general understanding of these formulas for arbitrary trees. This, however, is not easy in the above framework (“... a tedious calculation shows that ...”). Further, there are problems of identifying different methods with identical numerical results (see Exercise 1 below). Also, we want the theory to include more general processes than Runge-Kutta methods, for example the exact solution or multi-derivative methods.

B-Series

All these difficulties can be avoided if we consider directly the composition of the series appearing in Section II.2. We define by

$$T = \{\emptyset\} \cup T_1 \cup T_2 \cup \dots, \quad LT = \{\emptyset\} \cup LT_1 \cup LT_2 \cup \dots$$

the sets of all trees and labelled trees, respectively.

Definition 12.1 (Hairer & Wanner 1974). Let $\mathbf{a}(\emptyset)$, $\mathbf{a}(\cdot)$, $\mathbf{a}(\curvearrowright)$, $\mathbf{a}(\blacktriangledown)$, \dots be a sequence of real coefficients defined for all trees $\mathbf{a} : T \rightarrow \mathbb{R}$. Then we call the series (see Theorem 2.11, Definitions 2.2, 2.3)

$$\begin{aligned} B(\mathbf{a}, y) &= \mathbf{a}(\emptyset)y + h\mathbf{a}(\cdot)f(y) + \frac{h^2}{2!}\mathbf{a}(\curvearrowright)F(\cdot)(y) + \dots \\ &= \sum_{t \in LT} \frac{h^{\varrho(t)}}{\varrho(t)!} \mathbf{a}(t)F(t)(y) = \sum_{t \in LT} \frac{h^{\varrho(t)}}{\varrho(t)!} \alpha(t) \mathbf{a}(t)F(t)(y) \end{aligned} \quad (12.7)$$

a *B-series*.

We have seen in Theorems 2.11 and 2.6 that the numerical solution of a Runge-Kutta method as well as the exact solution are B-series. The coefficients of the latter are all equal to 1.

Usually we are only interested in a finite number of terms of these series (only as high as the orders of the methods under consideration, or as far as f is differentiable) and all subsequent results are valid modulo error terms $\mathcal{O}(h^{k+1})$.

Definition 12.2. Let $t \in LT$ be a labelled tree of order $q = \varrho(t)$ and $0 \leq i \leq q$ be a fixed integer. Then we denote by $s_i(t) = s$ the *subtree* formed by the first i indices and by $d_i(t)$ (the *difference set*) the set of subtrees formed by the remaining indices. In the graphical representation we distinguish the subtree s by fat nodes and doubled lines.

Example 12.3. For the labelled tree $t = \begin{array}{c} m \\ \swarrow \downarrow \searrow \\ p \quad l \quad k \\ \quad \quad \quad j \end{array}$ we have:

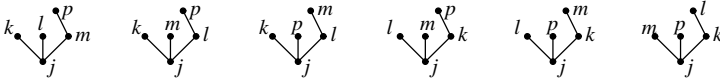
$i = 0$:		$s_0(t) = \emptyset,$	$d_0(t) = \{\blacktriangledown\}$
$i = 1$:		$s_1(t) = \cdot,$	$d_1(t) = \{\cdot, \cdot, \cdot, \curvearrowright\}$
$i = 2$:		$s_2(t) = \curvearrowright,$	$d_2(t) = \{\cdot, \cdot, \cdot, \cdot\}$
$i = 3$:		$s_3(t) = \blacktriangledown,$	$d_3(t) = \{\cdot, \cdot, \cdot\}$
$i = 4$:		$s_4(t) = \curvearrowright,$	$d_4(t) = \{\cdot\}$
$i = 5$:		$s_5(t) = t = \blacktriangledown,$	$d_5(t) = \emptyset$

Definition 12.4. Let $\mathbf{a} : T \rightarrow \mathbb{R}$ and $\mathbf{b} : T \rightarrow \mathbb{R}$ be two sequences of coefficients such that $\mathbf{a}(\emptyset) = 1$. Then for a tree t of order $q = \varrho(t)$ we define the *composition*

$$\mathbf{ab}(t) = \frac{1}{\alpha(t)} \sum_{i=0}^q \left(\sum_i^q \binom{q}{i} \mathbf{b}(s_i(t)) \prod_{z \in d_i(t)} \mathbf{a}(z) \right) \quad (12.8)$$

where the first summation is over all $\alpha(t)$ different labellings of t (see Definition 2.5).

Example 12.5. It is easily seen that the formulas of (12.6) are special cases of (12.8). The tree t of Example 12.3 possesses 6 different labellings



These lead to

$$\begin{aligned} \mathbf{ab}(\heartsuit) &= \mathbf{b}(\emptyset)\mathbf{a}(\heartsuit) + 5\mathbf{b}(\cdot)\mathbf{a}(\cdot)^2\mathbf{a}(\cdot) \\ &\quad + 10\left(\frac{1}{2}\mathbf{b}(\cdot)\mathbf{a}(\cdot)\mathbf{a}(\cdot) + \frac{1}{2}\mathbf{b}(\cdot)\mathbf{a}(\cdot)^3\right) \\ &\quad + 10\left(\frac{1}{6}\mathbf{b}(\heartsuit)\mathbf{a}(\cdot) + \frac{4}{6}\mathbf{b}(\heartsuit)\mathbf{a}(\cdot)^2 + \frac{1}{6}\mathbf{b}(\heartsuit)\mathbf{a}(\cdot)^2\right) \\ &\quad + 5\left(\frac{1}{2}\mathbf{b}(\heartsuit)\mathbf{a}(\cdot) + \frac{1}{2}\mathbf{b}(\heartsuit)\mathbf{a}(\cdot)\right) + \mathbf{b}(\heartsuit). \end{aligned} \quad (12.9)$$

Here is the main theorem of this section:

Theorem 12.6 (Hairer & Wanner 1974). *As above, let $\mathbf{a} : T \rightarrow \mathbb{R}$ and $\mathbf{b} : T \rightarrow \mathbb{R}$ be two sequences of coefficients such that $\mathbf{a}(\emptyset) = 1$. Then the composition of the two corresponding B-series is again a B-series*

$$B(\mathbf{b}, B(\mathbf{a}, y)) = B(\mathbf{ab}, y) \quad (12.10)$$

where the “product” $\mathbf{ab} : T \rightarrow \mathbb{R}$ is that of Definition 12.4.

Proof. We denote the inner series by

$$B(\mathbf{a}, y) = g(h). \quad (12.11)$$

Then the proof is similar to the development of Section II.2 (see Fig. 2.2), with the difference that, instead of $f(g)$, we now start from

$$B(\mathbf{b}, g) = \sum_{s \in LT} \frac{h^{\varrho(s)}}{\varrho(s)!} \mathbf{b}(s) F(s)(g) \quad (12.12)$$

and have to compute the derivatives of this function: let us select the term $s = \heartsuit$

of this series,

$$\frac{h^3}{3!} \mathbf{b}(\mathfrak{z}) \sum_{L,M} f_L^K(g) f_M^L(g) f^M(g). \quad (12.13)$$

The q th derivative of this expression, for $h = 0$, is by Leibniz' formula

$$\binom{q}{3} \mathbf{b}(\mathfrak{z}) \sum_{L,M} (f_L^K(g) f_M^L(g) f^M(g))^{(q-3)}|_{h=0}. \quad (12.14)$$

We now compute, as we did in Lemma 2.8, the derivatives of

$$f_L^K(g) f_M^L(g) f^M(g) \quad (12.15)$$

using the classical rules of differential calculus; this gives for the first derivative

$$\sum_N f_{LN}^K \cdot (g^N)' f_M^L f^M + \sum_N f_L^K f_{MN}^L \cdot (g^N)' f^M + \sum_N f_L^K f_M^L f_N^M \cdot (g^N)'$$

and so on. We again represent this in graphical form in Fig. 12.1.

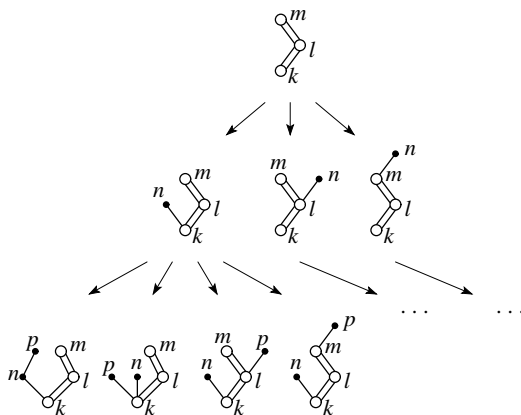


Fig. 12.1. Derivatives of (12.15)

We see that we arrive at trees u of order q such that $s_3(u) = s$ (where $3 = \varrho(s)$) and the elements of $d_3(u)$ have no ramifications. The corresponding expressions are similar to (2.6;q-1) in Lemma 2.8. We finally have to insert the derivatives of g (see (12.11)) and rearrange the terms. Then, as in Fig. 2.4, the tall branches of $d_3(u)$ are replaced by trees z of order δ , multiplied by $\mathbf{a}(z)$. Thus the coefficient which we obtain for a given tree t is just given by (12.8).

The factor $1/\alpha(t)$ is due to the fact that in $B(\mathbf{ab}, y)$ the term with $\mathbf{ab}(t)F(t)$ appears $\alpha(t)$ times. \square

Since $hf(y) = B(\mathbf{b}, y)$ is a special B-series with $\mathbf{b}(\cdot) = 1$ and all other $\mathbf{b}(t) = 0$, we have the following

Corollary 12.7. *If $\mathbf{a} : T \rightarrow \mathbb{R}$ with $\mathbf{a}(\emptyset) = 1$, then*

$$hf(B(\mathbf{a}, y)) = B(\mathbf{a}', y)$$

with

$$\begin{aligned} \mathbf{a}'(\emptyset) &= 0, \quad \mathbf{a}'(\cdot) = 1 \\ \mathbf{a}'([t_1, \dots, t_m]) &= \varrho(t) \mathbf{a}(t_1) \cdots \mathbf{a}(t_m) \end{aligned} \quad (12.16)$$

where $t = [t_1, \dots, t_m]$ means that $d_1(t) = \{t_1, t_2, \dots, t_m\}$ (Definition 2.12).

Proof. We obtain (12.16) from (12.8) with $i = 1$, $q = \varrho(t)$ and the fact that the expression in brackets is independent of the labelling of t . \square

Order Conditions for Runge-Kutta Methods

As an application of Corollary 12.7, we demonstrate the derivation of order conditions for Runge-Kutta methods: we write method (2.3) as

$$g_i = y_0 + \sum_{j=1}^s a_{ij} k_j, \quad k_i = hf(g_i), \quad y_1 = y_0 + \sum_{j=1}^s b_j k_j. \quad (12.17)$$

If we assume g_i , k_i and y_1 to be B-series, whose coefficients we denote by $\mathbf{g}_i, \mathbf{k}_i, \mathbf{y}_1$

$$g_i = B(\mathbf{g}_i, y_0), \quad k_i = B(\mathbf{k}_i, y_0), \quad y_1 = B(\mathbf{y}_1, y_0),$$

then Corollary 12.7 immediately allows us to transcribe formulas (12.17) as

$$\begin{aligned} \mathbf{g}_i(\emptyset) &= 1, & \mathbf{k}_i(\cdot) &= 1, & \mathbf{y}_1(\emptyset) &= 1, \\ \mathbf{g}_i(t) &= \sum_{j=1}^s a_{ij} \mathbf{k}_j(t), & \mathbf{k}_i(t) &= \varrho(t) \mathbf{g}_i(t_1) \cdots \mathbf{g}_i(t_m), & \mathbf{y}_1(t) &= \sum_{j=1}^s b_j \mathbf{k}_j(t) \end{aligned}$$

which leads easily to formulas (2.17), (2.19) and Theorem 2.11.

Also, if we put $y(h) = B(\mathbf{y}, y_0)$ for the *true solution*, and compare the derivative $hy'(h)$ of the series (12.7) with $hf(y(h))$ from Corollary 12.7, we immediately obtain $\mathbf{y}(t) = 1$ for all t , so that Theorem 2.6 drops out. The order conditions are then obtained as in Theorem 2.13 by comparing the coefficients of the B-series $B(\mathbf{y}, y_0)$ and $B(\mathbf{y}_1, y_0)$.

Butcher's "Effective Order"

We search for a 5-stage Runge-Kutta method **a** and for a method **d**, such that **dad**⁻¹ represents a fifth order method **u**. This means that we have to satisfy

$$\mathbf{da}(t) = \mathbf{yd}(t) \quad \text{for} \quad \varrho(t) \leq 5, \quad (12.18)$$

where **y**(*t*) = 1 represents the B-series of the exact solution. Then

$$(\mathbf{dad}^{-1})^k = \mathbf{da}^k \mathbf{d}^{-1} = (\mathbf{da}) \mathbf{a}^{k-2} (\mathbf{ad}^{-1}). \quad (12.19)$$

If now two Runge-Kutta methods **b** and **c** are constructed such that **b** = **da** and **c** = **ad**⁻¹ up to order 5, then applying one step of **b** followed by *k* - 2 steps of **a** and a final step of **c** is equivalent (up to order 5) to *k* steps of the 5th order method **dad**⁻¹ (see Fig. 12.2). A possible set of coefficients, computed by Butcher (1969), is given in Table 12.1 (method **a** has classical order 4).

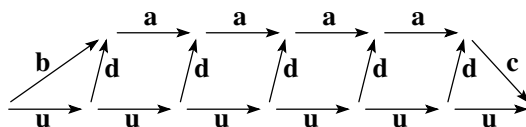


Fig. 12.2. Effective increase of order

Stetter's approach. Soon after the appearance of Butcher's purely algebraic proof, Stetter (1971) gave an elegant analytic explanation. Consider the principal global error term $e_p(x)$ which satisfies the variational equation (8.8). The question is, under which conditions on the local error $d_{p+1}(x)$ (see (8.8)) this equation can be solved, for special initial values, without effort. We write equation (8.8) as

$$e'(x) - \frac{\partial f}{\partial y}(y(x)) \cdot e(x) = d(x) \quad (12.20)$$

and want $e(x)$ to possess an expansion of the form

$$e(x) = \sum_{t \in T_p} \alpha(t) \mathbf{e}(t) F(t)(y(x)) \quad (12.21)$$

with constant coefficients $\mathbf{e}(t)$. Simply inserting (12.21) into (12.20) yields

$$d(x) = \sum_{t \in T_p} \alpha(t) \mathbf{e}(t) \left(\frac{d}{dx} (F(t)(y(x))) - f'(y(x)) \cdot F(t)(y(x)) \right). \quad (12.22)$$

Thus, (12.21) is the exact solution of the variational equation, if the local error $d(x)$ has the symmetric form (12.22). Then, if we replace the initial value y_0 by the "starting procedure"

$$\hat{y}_0 := y_0 - h^p e(x_0) = y_0 - h^p \sum_{t \in T_p} \alpha(t) \mathbf{e}(t) F(t)(y_0) \quad (12.23)$$

Table 12.1. Butcher's method of effective order 5

0	Method a				
$\frac{1}{5}$	$\frac{1}{5}$				
$\frac{2}{5}$	0	$\frac{2}{5}$			
$\frac{1}{2}$	$\frac{3}{16}$	0	$\frac{5}{16}$		
1	$\frac{1}{4}$	0	$-\frac{5}{4}$	2	
	$\frac{1}{6}$	0	0	$\frac{2}{3}$	$\frac{1}{6}$

0	Method b				
$\frac{1}{5}$	$\frac{1}{5}$				
$\frac{2}{5}$	0	$\frac{2}{5}$			
$\frac{3}{4}$	$\frac{75}{64}$	$-\frac{9}{4}$	$\frac{117}{64}$		
1	$-\frac{37}{36}$	$\frac{7}{3}$	$-\frac{3}{4}$	$\frac{4}{9}$	
	$\frac{19}{144}$	0	$\frac{25}{48}$	$\frac{2}{9}$	$\frac{1}{8}$

0	Method c				
$\frac{1}{5}$	$\frac{1}{5}$				
$\frac{2}{5}$	0	$\frac{2}{5}$			
$\frac{3}{4}$	$\frac{161}{192}$	$-\frac{19}{12}$	$\frac{287}{192}$		
1	$-\frac{27}{28}$	$\frac{19}{7}$	$-\frac{291}{196}$	$\frac{36}{49}$	
	$\frac{7}{48}$	0	$\frac{475}{1008}$	$\frac{2}{7}$	$\frac{7}{72}$

(or by a Runge-Kutta method equivalent to this up to order $p+1$; this would represent “method **d**” in Fig. 12.2), its error satisfies $y(x_0) - \hat{y}_0 = h^p e(x_0) + \mathcal{O}(h^{p+1})$. By Theorem 8.1 the numerical solution \hat{y}_n of the Runge-Kutta method applied to \hat{y}_0 satisfies $y(x_n) - \hat{y}_n = h^p e(x_n) + \mathcal{O}(h^{p+1})$. Therefore the “finishing procedure”

$$y_n := \hat{y}_n + h^p e(x_n) = \hat{y}_n + h^p \sum_{t \in T_p} \alpha(t) \mathbf{e}(t) F(t)(\hat{y}_n) + \mathcal{O}(h^{p+1}) \quad (12.24)$$

(or some equivalent Runge-Kutta method) gives a $(p+1)$ th order approximation to the solution.

Example. Butcher's method **a** of Table 12.1 has the local error

$$d_6(x) = \frac{1}{6!} \left(-\frac{1}{24} F(\Psi) - \frac{1}{4} F(\Psi) - \frac{1}{8} F(\Psi) + \frac{1}{6} F(\Psi) + \frac{1}{2} F(\Psi) \right). \quad (12.25)$$

The right-hand side of (12.22) would be (the derivation $\frac{d}{dx} F$ attaches a new twig

to each of the nodes, the product $f'(y) \cdot F$ lifts the tree on a stilt)

$$\begin{aligned}
 & e(\Psi) \left(F(\Psi) + 3F(\Phi) - F(\Upsilon) \right) \\
 & + 3e(\Phi) \left(F(\Phi) + F(\Theta) + F(\Psi) + F(\dot{\Phi}) - F(\dot{\Psi}) \right) \\
 & + e(\Upsilon) \left(F(\Psi) + F(\Upsilon) + 2F(\dot{\Psi}) - F(\dot{\Upsilon}) \right) \\
 & + e(\dot{\Psi}) \left(F(\dot{\Psi}) + F(\dot{\Upsilon}) + F(\dot{\Upsilon}) + F(\dot{\dot{\Psi}}) - F(\dot{\dot{\Upsilon}}) \right).
 \end{aligned} \tag{12.26}$$

Comparison of (12.25) and (12.26) shows that this method does indeed have the desired symmetry if

$$e(\Psi) = e(\Phi) = -\frac{1}{6!} \cdot \frac{1}{24}, \quad e(\Upsilon) = e(\dot{\Psi}) = \frac{1}{6!} \cdot \frac{1}{8}.$$

This allows one to construct a Runge-Kutta method as starting procedure corresponding to (12.23) up to the desired order.

Exercises

1. Show that the pairs of methods given in Tables 12.2 - 12.4 produce, at least for h sufficiently small, identical numerical results.

Result. a) is seen by permutation of the stages, b) by neglecting superfluous stages (Dahlquist & Jeltsch 1979), c) by identifying equal stages (Stetter 1973, Hundsdorfer & Spijker 1981). See also the survey on “The Runge-Kutta space” by Butcher (1984).

2. Extend formulas (12.6) by computing the composition $\mathbf{ab}(t)$ for all trees of order 4 and 5.
3. Verify that the methods given in Table 12.1 satisfy the stated order properties.
4. Prove, using Theorem 12.6, that the set

$$G = \{\mathbf{a} : T \rightarrow \mathbb{R} \mid \mathbf{a}(\emptyset) = 1\}$$

together with the composition law of Definition 12.4 is a (non-commutative) group.

5. (Equivalence of Butcher’s and Stetter’s approach). Let $\mathbf{a} : T \rightarrow \mathbb{R}$ represent a Runge-Kutta method of classical order p and effective order $p+1$, i.e., $\mathbf{a}(t) = 1$ for $\varrho(t) \leq p$ and

$$\mathbf{da}(t) = \mathbf{yd}(t) \quad \text{for} \quad \varrho(t) \leq p+1 \tag{12.27}$$

for some $\mathbf{d} : T \rightarrow \mathbb{R}$ and with $\mathbf{y}(t)$ as in (12.18). Prove that then the local error $h^{p+1}d(x) + \mathcal{O}(h^{p+2})$ of the method \mathbf{a} has the symmetric form (12.22). This

Table 12.2. Equivalent methods a)

0				1	0	1
1	1	0		0	0	0
			1/4	3/4		

Table 12.3. Equivalent methods b)

1	2	0	0	-1	1	2	-1
3	0	1	2	0	2	1	1
7	0	3	4	0			
2	1	0	0	1			
					1/2	0	1/2

Table 12.4. Equivalent methods c)

1	1	1	1	-2	1	3	-2
1	2	2	-1	-2	-1	2	-3
1	-1	-1	5	-2			
-1	-1	2	1	-3			
					3/4	1/4	

means that, in this situation, Butcher's effective order is equivalent to Stetter's approach.

Hint. Start by expanding condition (12.27) (using (12.8)) for the first trees. Possible simplifications are then best seen if the second sum $\sum_{i=0}^q$ (for $\mathbf{y}\mathbf{d}$) is arranged *downwards* ($i = q, q-1, \dots, 0$). One then arrives recursively at the result

$$\mathbf{d}(t) = \mathbf{d}(\cdot)^{\varrho(t)} \quad \text{for } \varrho(t) \leq p-1.$$

Then express the error coefficients $\mathbf{a}(t) - 1$ for $\varrho(t) = p+1$ in terms of $\mathbf{d}(s) - \mathbf{d}(\cdot)^{\varrho(s)}$ where $\varrho(s) = p$. Formula (12.22) then becomes visible.

6. Prove that for $t = [t_1, \dots, t_m]$ the coefficient $\alpha(t)$ of Definition 2.5 satisfies the recurrence relation

$$\alpha(t) = \binom{\varrho(t)-1}{\varrho(t_1), \dots, \varrho(t_m)} \alpha(t_1) \cdot \dots \cdot \alpha(t_m) \cdot \frac{1}{\mu_1! \mu_2! \dots}. \quad (12.28)$$

The integers μ_1, μ_2, \dots count the equal trees among t_1, \dots, t_m .

Hint. The multinomial coefficient in (12.28) counts the possible partitionings of the labels $2, \dots, \varrho(t)$ to the m subtrees t_1, \dots, t_m . Equal subtrees lead to equal labellings. Hence the division by $\mu_1! \mu_2! \dots$.

II.13 Higher Derivative Methods

In Section I.8 we studied the computation of higher derivatives of solutions of

$$(y^J)' = f^J(x, y^1, \dots, y^n), \quad J = 1, \dots, n. \quad (13.1)$$

The chain rule

$$(y^J)'' = \frac{\partial f^J}{\partial x}(x, y) + \frac{\partial f^J}{\partial y^1}(x, y) \cdot f^1(x, y) + \dots + \frac{\partial f^J}{\partial y^n}(x, y) \cdot f^n(x, y) \quad (13.2)$$

leads to the differential operator D which, when applied to a function $\Psi(x, y)$, is given by

$$(D\Psi)(x, y) = \frac{\partial \Psi}{\partial x}(x, y) + \frac{\partial \Psi}{\partial y^1}(x, y) \cdot f^1(x, y) + \dots + \frac{\partial \Psi}{\partial y^n}(x, y) \cdot f^n(x, y). \quad (13.2')$$

Since $Dy^J = f^J$, we see by extending (13.2) that

$$(y^J)^{(\ell)} = (D^\ell y^J)(x, y), \quad \ell = 0, 1, 2, \dots \quad (13.3)$$

This notation allows us to define a new class of methods which combine features of Runge-Kutta methods as well as Taylor series methods:

Definition 13.1. Let $a_{ij}^{(r)}$, $b_j^{(r)}$, $(i, j = 1, \dots, s, r = 1, \dots, q)$ be real coefficients. Then the method

$$\begin{aligned} k_i^{(\ell)} &= \frac{h^\ell}{\ell!} (D^\ell y) \left(x_0 + c_i h, y_0 + \sum_{r=1}^q \sum_{j=1}^s a_{ij}^{(r)} k_j^{(r)} \right) \\ y_1 &= y_0 + \sum_{r=1}^q \sum_{j=1}^s b_j^{(r)} k_j^{(r)} \end{aligned} \quad (13.4)$$

is called an s -stage q -derivative Runge-Kutta method. If $a_{ij}^{(r)} = 0$ for $i \leq j$, the method is *explicit*, otherwise *implicit*.

A natural extension of (1.9) is here, because of $Dx = 1$, $D^\ell x = 0$ ($\ell \geq 2$),

$$c_i = \sum_{j=1}^s a_{ij}^{(1)}. \quad (13.5)$$

Definition 13.1 is from Kastlunger & Wanner (1972), but special methods of this type have been considered earlier in the literature. In particular, the very successful methods of Fehlberg (1958, 1964) have this structure.

Collocation Methods

A natural way of obtaining s -stage q -derivative methods is to use the collocation idea with *multiple nodes*, i.e., to replace (7.15b) by

$$u^{(\ell)}(x_0 + c_i h) = (D^\ell y)(x_0 + c_i h, u(x_0 + c_i h)) \quad i = 1, \dots, s, \quad \ell = 1, \dots, q_i \quad (13.6)$$

where $u(x)$ is a polynomial of degree $q_1 + q_2 + \dots + q_s$ and q_1, \dots, q_s , the “multiplicities” of the nodes c_1, \dots, c_s , are given integers. For example $q_1 = m$, $q_2 = \dots = q_s = 1$ leads to Fehlberg-type methods.

In order to generalize the results and ideas of Section II.7, we have to replace the Lagrange interpolation of Theorem 7.7 by *Hermite* interpolation (Hermite 1878: “Je me suis proposé de trouver un polynôme . . .”). The reason is that (13.6) can be interpreted as an ordinary collocation condition with clusters of q_i nodes “infinitely” close together (Rolle’s theorem). We write Hermite’s formula as

$$p(t) = \sum_{j=1}^s \sum_{r=1}^{q_j} \frac{1}{r!} \ell_{jr}(t) p^{(r-1)}(c_j) \quad (13.7)$$

for polynomials $p(t)$ of degree $\sum q_j - 1$. Here the “basis” polynomials $\ell_{jr}(t)$ of degree $\sum q_j - 1$ must satisfy

$$l_{jr}^{(k)}(c_i) = \begin{cases} r! & \text{if } i = j \text{ and } k = r - 1 \\ 0 & \text{else} \end{cases} \quad (13.8)$$

and are best obtained from Newton’s interpolation formula (with multiple nodes). We now use this formula, as we did in Section II.7, for $p(t) = hu'(x_0 + th)$:

$$hu'(x_0 + th) = \sum_{j=1}^s \sum_{r=1}^{q_j} \ell_{jr}(t) k_j^{(r)}, \quad (13.9)$$

with

$$k_j^{(r)} = \frac{h^r}{r!} u^{(r)}(x_0 + c_j h). \quad (13.10)$$

If we insert

$$u(x_0 + c_i h) = y_0 + \int_0^{c_i} hu'(x_0 + th) dt$$

together with (13.9) into (13.6), we get:

Theorem 13.2. *The collocation method (13.6) is equivalent to an s -stage q -derivative implicit Runge-Kutta method (13.4) with*

$$a_{ij}^{(r)} = \int_0^{c_i} \ell_{jr}(t) dt, \quad b_j^{(r)} = \int_0^1 \ell_{jr}(t) dt. \quad (13.11)$$

□

Theorems 7.8, 7.9, and 7.10 now generalize immediately to the case of “confluent” quadrature formulas; i.e., the q -derivative Runge-Kutta method possesses the *same order* as the underlying quadrature formula

$$\int_0^1 p(t) dt \approx \sum_{j=1}^s \sum_{r=1}^{q_j} b_j^{(r)} p^{(r-1)}(c_j).$$

The “algebraic” proof of this result (extending Exercise 7 of Section II.7) is more complicated and is given, for the case $q_j = q$, in Kastlunger & Wanner (1972b).

The formulas corresponding to condition $C(\eta)$ are given by

$$\sum_{j=1}^s \sum_{r=1}^{q_j} a_{ij}^{(r)} \binom{\varrho}{r} c_j^{\varrho-r} = c_i^{\varrho}, \quad \varrho = 1, 2, \dots, \sum_{j=1}^s q_j. \quad (13.12)$$

These equations uniquely determine the $a_{ij}^{(r)}$, once the c_i have been chosen, by a linear system with a “confluent” Vandermonde matrix (see e.g., Gautschi 1962). Formula (13.12) is obtained by setting $p(t) = t^{\varrho-1}$ in (13.7) and then integrating from 0 to c_i .

Examples of methods. “Gaussian” quadrature formulas with multiple nodes exist for *odd* q (Stroud & Stancu 1965) and extend to q -derivative implicit Runge-Kutta methods (Kastlunger & Wanner 1972b): for $s = 1$ we have, of course, $c_1 = 1/2$ which yields

$$b_1^{(2k)} = 0, \quad b_1^{(2k+1)} = 2^{-2k}, \quad a_{11}^{(k)} = (-1)^{k+1} 2^{-k}.$$

We give also the coefficients for the case $s = 2$ and $q_1 = q_2 = 3$. The nodes c_i and the weights $b_i^{(k)}$ are those of Stroud & Stancu. The method has order 8:

$$\begin{array}{ll} c_1 = 0.185394435825045 & c_2 = 1 - c_1 \\ b_1^{(1)} = 0.5 & b_2^{(1)} = b_1^{(1)} \\ b_1^{(2)}/2! = 0.0240729420844974 & b_2^{(2)} = -b_1^{(2)} \\ b_1^{(3)}/3! = 0.00366264960671727 & b_2^{(3)} = b_1^{(3)} \end{array}$$

$$\begin{aligned}
a_{ij}^{(1)} &= \begin{pmatrix} 0.201854115831005 & -0.0164596800059598 \\ 0.516459680005959 & 0.298145884168994 \end{pmatrix} \\
a_{ij}^{(2)} &= \begin{pmatrix} -0.0223466569080541 & 0.00868878773082417 \\ 0.0568346718998190 & -0.0704925410770490 \end{pmatrix} \\
a_{ij}^{(3)} &= \begin{pmatrix} 0.0116739668400997 & -0.00215351251065784 \\ 0.0241294101509615 & 0.0103019308002039 \end{pmatrix}
\end{aligned}$$

Hermite-Obreschkoff Methods

We now consider the special case of collocation methods with $s=2$, $c_1=0$, $c_2=1$. These methods can be obtained in closed form by repeated partial integration as follows (Darboux 1876, Hermite 1878):

Lemma 13.3. *Let m be a given positive integer and $P(t)$ a polynomial of exact degree m . Then*

$$\sum_{j=0}^m h^j (D^j y)(x_1, y_1) P^{(m-j)}(0) = \sum_{j=0}^m h^j (D^j y)(x_0, y_0) P^{(m-j)}(1) \quad (13.13)$$

defines a multiderivative method (13.4) of order m .

Proof. We let $y(x)$ be the exact solution and start from

$$h^{m+1} \int_0^1 y^{(m+1)}(x_0 + ht) P(1-t) dt = \mathcal{O}(h^{m+1}).$$

This integral is now transformed by repeated partial integration until all derivatives of the polynomial $P(1-t)$ are used up. This leads to

$$\sum_{j=0}^m h^j y^{(j)}(x_1) P^{(m-j)}(0) = \sum_{j=0}^m h^j y^{(j)}(x_0) P^{(m-j)}(1) + \mathcal{O}(h^{m+1}).$$

If this is subtracted from (13.13) we find the difference of the left-hand sides to be $\mathcal{O}(h^{m+1})$, which shows by the implicit function theorem that (13.13) determines y_1 to this order if $P^{(m)}$, which is a constant, is $\neq 0$. \square

The argument $1-t$ in P (instead of the more natural t) avoids the sign changes in the partial integrations.

A good choice for $P(t)$ is, of course, a polynomial for which most derivatives disappear at $t=0$ and $t=1$. Then the method (13.13) is, by keeping the same order m , most economical. We write

$$P(t) = \frac{t^k(t-1)^\ell}{(k+\ell)!}$$

and obtain

$$\begin{aligned}
 y_1 - \frac{\ell}{(k+\ell)} \frac{h}{1!} (Dy)(x_1, y_1) + \frac{\ell(\ell-1)}{(k+\ell)(k+\ell-1)} \frac{h^2}{2!} (D^2y)(x_1, y_1) \pm \dots \\
 = y_0 + \frac{k}{(k+\ell)} \frac{h}{1!} (Dy)(x_0, y_0) + \frac{k(k-1)}{(k+\ell)(k+\ell-1)} \frac{h^2}{2!} (D^2y)(x_0, y_0) + \dots
 \end{aligned} \tag{13.14}$$

which is a method of order $m = k + \ell$. After the ℓ th term in the first line and the k th term in the second line, the coefficients automatically become zero. Special cases of this method are:

$$\begin{aligned}
 k = 1, \quad \ell = 0 &: \text{explicit Euler} \\
 k \geq 1, \quad \ell = 0 &: \text{Taylor series} \\
 k = 0, \quad \ell = 1 &: \text{implicit Euler} \\
 k = 1, \quad \ell = 1 &: \text{trapezoidal rule.}
 \end{aligned}$$

Darboux and Hermite advocated the use of this formula for the approximations of functions, Obreschkoff (1940) for the computation of integrals, Loscalzo & Schoenberg (1967), Loscalzo (1969) as well as Nørsett (1974a) for the solution of differential equations.

Fehlberg Methods

Another class of multiderivative methods is due to Fehlberg (1958, 1964): the idea is to subtract from the solution of $y' = f(x, y)$, $y(x_0) = y_0$ m terms of the Taylor series (see Section I.8)

$$\hat{y}(x) := y(x) - \sum_{i=0}^m Y_i(x - x_0)^i, \tag{13.15}$$

and to solve the resulting differential equation $\hat{y}'(x) = \hat{f}(x, \hat{y}(x))$, where

$$\hat{f}(x, \hat{y}(x)) = f\left(x, \hat{y} + \sum_{i=0}^m Y_i(x - x_0)^i\right) - \sum_{i=1}^m Y_i i (x - x_0)^{i-1}, \tag{13.16}$$

by a Runge-Kutta method. Thus, knowing that the solution of (13.16) and its first m derivatives are zero at the initial value, we can achieve much higher orders.

In order to understand this, we develop the Taylor series of the solution for the non-autonomous case, as we did at the beginning of Section II.1. We thereby omit the hats and suppose the transformation (13.15) already carried out. We then have from (1.6) (see also Exercise 3 of Section II.2)

$$\begin{aligned}
 f &= 0, \\
 f_x + f_y f &= 0, \\
 f_{xx} + 2f_{xy}f + f_{yy}f^2 + f_y(f_x + f_y f) &= 0, \text{ etc.}
 \end{aligned}$$

These formulas recursively imply that $f = 0$, $f_x = 0$, \dots , $\partial^{m-1} f / \partial x^{m-1} = 0$. All elementary differentials of order $\leq m$ and most of those of higher orders then become zero and the corresponding order conditions can be omitted. The first non-zero terms are

$$\begin{aligned} & \frac{\partial^m f}{\partial x^m} && \text{for order } m+1, \\ & \frac{\partial^{m+1} f}{\partial x^{m+1}} \quad \text{and} \quad \frac{\partial f}{\partial y} \cdot \frac{\partial^m f}{\partial x^m} && \text{for order } m+2, \end{aligned}$$

and so on. The corresponding order conditions are then

$$\sum_{i=1}^s b_i c_i^m = \frac{1}{m+1}$$

for order $m+1$,

$$\sum_{i=1}^s b_i c_i^{m+1} = \frac{1}{m+2} \quad \text{and} \quad \sum_{i,j} b_i a_{ij} c_j^m = \frac{1}{(m+1)(m+2)}$$

for order $m+2$, and so on.

The condition $\sum a_{ij} = c_i$, which usually allows several terms of (1.6) to be grouped together, is not necessary, because all these other terms are zero.

A complete insight is obtained by considering the method as being *partitioned* applied to the *partitioned system* $y' = f(x, y)$, $x' = 1$. This will be explained in Section II.15 (see Fig. 15.3).

Example 13.4. A solution with $s = 3$ stages of the (seven) conditions for order $m+3$ is given by Fehlberg (1964). The choice $c_1 = c_3 = 1$ minimizes the numerical work for the evaluation of (13.16) and the other coefficients are then uniquely determined (see Table 13.1).

Fehlberg (1964) also derived an embedded method with two additional stages of orders $m+3$ ($m+4$). These methods were widely used in the sixties for scientific computations.

Table 13.1. Fehlberg, order $m+3$

1			$\theta = \frac{m+1}{m+3}$
θ	$\frac{\theta^m}{m+3}$		
1	$-\frac{1}{m+1}$	$\frac{2}{(m+1)\theta^m}$	
	0	$\frac{m+3}{2(m+1)(m+2)\theta^m}$	
			$\frac{1}{2(m+2)}$

General Theory of Order Conditions

For the same reason as in Section II.2 we assume that (13.1) is autonomous. The general form of the order conditions for method (13.4) was derived in the thesis of Kastlunger (see Kastlunger & Wanner 1972). It later became a simple application of the composition theorem for B-series (Hairer & Wanner 1974). The point is that from Theorem 2.6,

$$\frac{h^i}{i!} (D^i y)(y_0) = \sum_{t \in LT, \varrho(t)=i} \frac{h^i}{i!} F(t)(y_0) = B(\mathbf{y}^{(i)}, y_0) \quad (13.17)$$

is a B-series with coefficients

$$\mathbf{y}^{(i)}(t) = \begin{cases} 1 & \text{if } \varrho(t) = i \\ 0 & \text{otherwise.} \end{cases} \quad (13.18)$$

Thus, in extension of Corollary 12.7, we have

$$\frac{h^i}{i!} (D^i y)(B(\mathbf{a}, y_0)) = B(\mathbf{a}^{(i)}, y_0) \quad (13.19)$$

where, from formula (12.8) with $q = \varrho(t)$,

$$\mathbf{a}^{(i)}(t) = (\mathbf{a}\mathbf{y}^{(i)})(t) = \frac{1}{\alpha(t)} \binom{q}{i} \sum \prod_{z \in d_i(t)} \mathbf{a}(z), \quad (13.20)$$

and the sum is over all $\alpha(t)$ different labellings of t . This allows us to compute recursively the coefficients of the B-series which appear in (13.4).

Example 13.5. The tree $t = \spadesuit$ sketched in Fig. 13.1 possesses three different labellings, two of which produce the same difference set $d_2(t)$, so that formula (13.20) becomes

$$\mathbf{a}''(\spadesuit) = 2(2(\mathbf{a}(\cdot))^2 + \mathbf{a}(\spadesuit)). \quad (13.21)$$

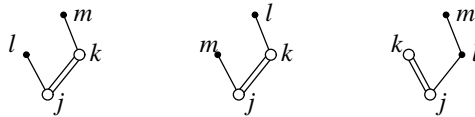


Fig. 13.1. Different labellings of \spadesuit

For all other trees of order ≤ 4 we have $\alpha(t) = 1$ and (13.20) leads to the following table of second derivatives

$$\begin{array}{ll} \mathbf{a}''(\cdot) = 0 & \mathbf{a}''(\spadesuit) = 1 \\ \mathbf{a}''(\heartsuit) = 3\mathbf{a}(\cdot) & \mathbf{a}''(\clubsuit) = 3\mathbf{a}(\cdot) \\ \mathbf{a}''(\padesuit) = 6(\mathbf{a}(\cdot))^2 & \mathbf{a}''(\spadesuit) = 4(\mathbf{a}(\cdot))^2 + 2\mathbf{a}(\spadesuit) \\ \mathbf{a}''(\heartsuit) = 6(\mathbf{a}(\cdot))^2 & \mathbf{a}''(\spadesuit) = 6\mathbf{a}(\spadesuit). \end{array} \quad (13.22)$$

Once these expressions have been established, we write formulas (13.4) in the form

$$k_i^{(\ell)} = \frac{h^\ell}{\ell!} (D^\ell y)(g_i)$$

$$g_i = y_0 + \sum_{r=1}^q \sum_{j=1}^s a_{ij}^{(r)} k_j^{(r)}, \quad y_1 = y_0 + \sum_{r=1}^q \sum_{j=1}^s b_j^{(r)} k_j^{(r)} \quad (13.23)$$

and suppose the expressions $k_i^{(\ell)}$, g_i , y_1 to be B-series

$$k_i^{(\ell)} = B(\mathbf{k}_i^{(\ell)}, y_0), \quad g_i = B(\mathbf{g}_i, y_0), \quad y_1 = B(\mathbf{y}_1, y_0).$$

Then equations (13.23) can be translated into

$$\mathbf{k}_i^{(1)}(t) = \varrho(t) \mathbf{g}_i(t_1) \cdot \dots \cdot \mathbf{g}_i(t_m), \quad \mathbf{k}_i^{(1)}(\tau) = 1 \quad (\text{see (12.16)})$$

$$\mathbf{k}_i^{(2)}(t) = \mathbf{g}_i''(t) \quad \text{from (13.22)}$$

$$\mathbf{k}_i^{(3)}(t) = \mathbf{g}_i'''(t) \quad \text{from Exercise 1 or Exercise 2, etc.}$$

$$\mathbf{g}_i(t) = \sum_{r=1}^q \sum_{j=1}^s a_{ij}^{(r)} \mathbf{k}_j^{(r)}(t), \quad \mathbf{y}_1(t) = \sum_{r=1}^q \sum_{j=1}^s b_j^{(r)} \mathbf{k}_j^{(r)}(t).$$

These formulas recursively determine all the coefficients. Method (13.4) (together with (13.5)) is then of order p if, as usual,

$$\mathbf{y}_1(t) = 1 \quad \text{for all } t \text{ with } \varrho(t) \leq p. \quad (13.24)$$

More details and special methods are given in Kastlunger & Wanner (1972); see also Exercise 3.

Exercises

1. Extend Example 13.5 and obtain formulas for $\mathbf{a}^{(3)}(t)$ for all trees of order ≤ 4 .
2. (Kastlunger). Prove the following variant form of formula (13.20) which extends (12.16) more directly and can also be used to obtain the formulas of Example 13.5. If $t = [t_1, \dots, t_m]$ then

$$\mathbf{a}^{(i)}(t) = \frac{\varrho(t)}{i} \sum_{\substack{\lambda_1 + \dots + \lambda_m = i-1 \\ \lambda_1, \dots, \lambda_m \geq 0}} \mathbf{a}^{(\lambda_1)}(t_1) \dots \mathbf{a}^{(\lambda_m)}(t_m)$$

Hint. See Kastlunger & Wanner (1972); Hairer & Wanner (1973), Section 5.

3. Show that the conditions for order 3 of method (13.4) are given by

$$\begin{aligned}\sum_i b_i^{(1)} &= 1 \\ 2 \sum_i b_i^{(1)} c_i + \sum_i b_i^{(2)} &= 1 \\ 3 \sum_i b_i^{(1)} c_i^2 + 3 \sum_i b_i^{(2)} c_i + \sum_i b_i^{(3)} &= 1 \\ 6 \sum_{i,j} b_i^{(1)} a_{ij}^{(1)} c_j + 3 \sum_i b_i^{(1)} e_i + 3 \sum_i b_i^{(2)} c_i + \sum_i b_i^{(3)} &= 1,\end{aligned}$$

where $c_i = \sum_j a_{ij}^{(1)}$, $e_i = \sum_j a_{ij}^{(2)}$.

4. (Zurmühl 1952, Albrecht 1955). Differentiate a given first order system of differential equations $y' = f(x, y)$ to obtain

$$y'' = (D^2 y)(x, y), \quad y(x_0) = y_0, \quad y'(x_0) = f_0.$$

Apply to this equation a special method for higher order systems (see the following Section II.14) to obtain higher-derivative methods. Show that the following method is of order six

$$\begin{aligned}k_1 &= h^2 g(x_0, y_0) \\ k_2 &= h^2 g\left(x_0 + \frac{h}{4}, y_0 + \frac{h}{4} f_0 + \frac{1}{32} k_1\right) \\ k_3 &= h^2 g\left(x_0 + \frac{h}{2}, y_0 + \frac{h}{2} f_0 + \frac{1}{24} (-k_1 + 4k_2)\right) \\ k_4 &= h^2 g\left(x_0 + \frac{3h}{4}, y_0 + \frac{3h}{4} f_0 + \frac{1}{32} (3k_1 + 4k_2 + 2k_3)\right) \\ y_1 &= y_0 + h f_0 + \frac{1}{90} (7k_1 + 24k_2 + 6k_3 + 8k_4)\end{aligned}$$

where $g(x, y) = (D^2 y)(x, y) = Df(x, y) = f_x(x, y) + f_y(x, y) \cdot f(x, y)$.

II.14 Numerical Methods for Second Order Differential Equations

Mutationem motus proportionalem esse vi motrici impressae
(Newton's Lex II, 1687)

Many differential equations which appear in practice are systems of the *second order*

$$y'' = f(x, y, y'). \quad (14.1)$$

This is mainly due to the fact that the forces are proportional to acceleration, i.e., to second derivatives. As mentioned in Section I.1, such a system can be transformed into a first order differential equation of doubled dimension by considering the vector (y, y') as the new variable:

$$\begin{pmatrix} y \\ y' \end{pmatrix}' = \begin{pmatrix} y' \\ f(x, y, y') \end{pmatrix} \quad \begin{matrix} y(x_0) = y_0 \\ y'(x_0) = y'_0 \end{matrix} \quad (14.2)$$

In order to solve (14.1) numerically, one can for instance apply a Runge-Kutta method (explicit or implicit) to (14.2). This yields

$$\begin{aligned} k_i &= y'_0 + h \sum_{j=1}^s a_{ij} k'_j \\ k'_i &= f(x_0 + c_i h, y_0 + h \sum_{j=1}^s a_{ij} k_j, y'_0 + h \sum_{j=1}^s a_{ij} k'_j) \\ y_1 &= y_0 + h \sum_{i=1}^s b_i k_i, \quad y'_1 = y'_0 + h \sum_{i=1}^s b_i k'_i. \end{aligned} \quad (14.3)$$

If we insert the first formula of (14.3) into the others we obtain (assuming (1.9) and an order ≥ 1)

$$\begin{aligned} k'_i &= f(x_0 + c_i h, y_0 + c_i h y'_0 + h^2 \sum_{j=1}^s \bar{a}_{ij} k'_j, y'_0 + h \sum_{j=1}^s a_{ij} k'_j) \\ y_1 &= y_0 + h y'_0 + h^2 \sum_{i=1}^s \bar{b}_i k'_i, \quad y'_1 = y'_0 + h \sum_{i=1}^s b_i k'_i \end{aligned} \quad (14.4)$$

where

$$\bar{a}_{ij} = \sum_{k=1}^s a_{ik} a_{kj}, \quad \bar{b}_i = \sum_{j=1}^s b_j a_{ji}. \quad (14.5)$$

For an implementation the representation (14.4) is preferable to (14.3), since about half of the storage can be saved. This may be important, in particular if the dimension of equation (14.1) is large.

Nyström Methods

R.H. Merson: "... I have not seen the paper by Nyström. Was it in English?"

J.M. Bennett: "In German actually, not Finnish."

(From the discussion following a talk of Merson 1957)

E.J. Nyström (1925) was the first to consider methods of the form (14.4) in which the coefficients do not necessarily satisfy (14.5) ("Da bis jetzt die *direkte* Anwendung der Rungeschen Methode auf den wichtigen Fall von Differentialgleichungen zweiter Ordnung nicht behandelt war ...". Nyström, 1925). Such direct methods are called *Nyström methods*.

Definition 14.1. A Nyström method (14.4) has *order* p if for sufficiently smooth problems (14.1)

$$y(x_0 + h) - y_1 = \mathcal{O}(h^{p+1}), \quad y'(x_0 + h) - y'_1 = \mathcal{O}(h^{p+1}). \quad (14.6)$$

An example of an explicit Nyström method where condition (14.5) is violated is given in Table 14.1. Nyström claimed that this method would be simpler to apply than "Runge-Kutta's" and reduce the work by about 25%. This is, of course, not true if the Runge-Kutta method is applied as in (14.4) (see also Exercise 2).

Table 14.1. Nyström, order 4

c_i	0								
	$\frac{1}{2}$	$\frac{1}{8}$	\bar{a}_{ij}			$\frac{1}{2}$	a_{ij}		
	$\frac{1}{2}$	$\frac{1}{8}$	0			0	$\frac{1}{2}$		
	1	0	0	$\frac{1}{2}$		0	0	1	
$\bar{b}_i \rightarrow$		$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	0	$\frac{1}{6}$	$\frac{2}{6}$	$\frac{2}{6}$	$\frac{1}{6} \leftarrow b_i$

A *real* improvement can be achieved in the case where the right-hand side of (14.1) does not depend on y' , i.e.,

$$y'' = f(x, y). \quad (14.7)$$

Here the Nyström method becomes

$$\begin{aligned}
 k'_i &= f(x_0 + c_i h, y_0 + c_i h y'_0 + h^2 \sum_{j=1}^s \bar{a}_{ij} k'_j) \\
 y_1 &= y_0 + h y'_0 + h^2 \sum_{i=1}^s \bar{b}_i k'_i, \quad y'_1 = y'_0 + h \sum_{i=1}^s b_i k'_i,
 \end{aligned} \tag{14.8}$$

and the coefficients a_{ij} are no longer needed. Some examples are given in Table 14.2. The fifth-order method of Table 14.2 needs only four evaluations of f . This is a considerable improvement compared to Runge-Kutta methods where at least six evaluations are necessary (cf. Theorem 5.1).

Table 14.2. Methods for $y'' = f(x, y)$

Nyström, order 4				Nyström, order 5					
	0	\bar{a}_{ij}			0				
	$\frac{1}{2}$	$\frac{1}{8}$			$\frac{1}{50}$	\bar{a}_{ij}			
c_i	$\frac{1}{2}$	$\frac{1}{8}$			$\frac{2}{3}$	$\frac{-1}{27}$	$\frac{7}{27}$		
	1	0	$\frac{1}{2}$		1	$\frac{3}{10}$	$\frac{-2}{35}$	$\frac{9}{35}$	
	\bar{b}_i	$\frac{1}{6}$	$\frac{1}{3}$	0	\bar{b}_i	$\frac{14}{336}$	$\frac{100}{336}$	$\frac{54}{336}$	0
	b_i	$\frac{1}{6}$	$\frac{4}{6}$	$\frac{1}{6}$	b_i	$\frac{14}{336}$	$\frac{125}{336}$	$\frac{162}{336}$	$\frac{35}{336}$

Global convergence. Introducing the variable $z_n = (y_n, y'_n)^T$, a Nyström method (14.4) can be written in the form

$$z_1 = z_0 + h\Phi(x_0, z_0, h) \tag{14.9}$$

where

$$\Phi(x_0, z_0, h) = \begin{pmatrix} y'_0 + h \sum_i \bar{b}_i k'_i \\ \sum_i b_i k'_i \end{pmatrix}.$$

(14.9) is just a special one-step method for the differential equation (14.2). For a p th order Nyström method the local error $(y(x_0 + h) - y_1, y'(x_0 + h) - y'_1)^T$ can be bounded by Ch^{p+1} (Definition 14.1), which is in agreement with formula (3.27). The convergence theorems of Section II.3 and the results on asymptotic expansions of the global error (Section II.8) are also valid here.

Our next aim is to derive the order conditions for Nyström methods. For this purpose we extend the theory of Section II.2 to second order differential equations (Hairer & Wanner 1976).

The Derivatives of the Exact Solution

As for first order equations we may restrict ourselves to systems of autonomous differential equations

$$(y^J)'' = f^J(y^1, \dots, y^n, y'^1, \dots, y'^n) \quad (14.10)$$

(if necessary, add $x'' = 0$). The superscript index J denotes the J th component of the corresponding vector. We now calculate the derivatives of the exact solution of (14.10). The second derivative is given by (14.10):

$$(y^J)^{(2)} = f^J(y, y'). \quad (14.11;2)$$

A repeated differentiation of this equation, using (14.10), leads to

$$(y^J)^{(3)} = \sum_K \frac{\partial f^J}{\partial y^K} (y, y') \cdot y'^K + \sum_K \frac{\partial f^J}{\partial y'^K} (y, y') f^K(y, y') \quad (14.11;3)$$

$$\begin{aligned} (y^J)^{(4)} = & \sum_{K,L} \frac{\partial^2 f^J}{\partial y^K \partial y^L} (y, y') \cdot y'^K \cdot y'^L & (14.11;4) \\ & + \sum_{K,L} \frac{\partial^2 f^J}{\partial y^K \partial y'^L} (y, y') \cdot y'^K \cdot f^L(y, y') + \sum_K \frac{\partial f^J}{\partial y^K} (y, y') f^K(y, y') \\ & + \sum_{K,L} \frac{\partial^2 f^J}{\partial y'^K \partial y^L} (y, y') f^K(y, y') \cdot y'^L \\ & + \sum_{K,L} \frac{\partial^2 f^J}{\partial y'^K \partial y'^L} (y, y') f^K(y, y') f^L(y, y') \\ & + \sum_{K,L} \frac{\partial f^J}{\partial y'^K} (y, y') \frac{\partial f^K}{\partial y^L} (y, y') y'^L \\ & + \sum_{K,L} \frac{\partial f^J}{\partial y'^K} (y, y') \frac{\partial f^K}{\partial y'^L} (y, y') f^L(y, y') \end{aligned}$$

The continuation of this process becomes even more complex than for first order differential equations. A graphical representation of the above formulas will therefore be very helpful. In order to distinguish the derivatives with respect to y and y' we need two kinds of vertices: “meagre” and “fat”. Fig. 14.1 shows the graphs that correspond to the above formulas.

Definition 14.2. A labelled N -tree of order q is a labelled tree (see Definition 2.2)

$$t : A_q \setminus \{j\} \rightarrow A_q$$

together with a mapping

$$t' : A_q \rightarrow \{\text{“meagre”}, \text{“fat”}\}$$

$$\circ_j \quad (14.11;2)$$

$$\begin{array}{c} k \\ \bullet \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \circ \\ \circ_j \end{array} \quad (14.11;3)$$

$$\begin{array}{c} k \\ \bullet \\ \circ_j \end{array} \begin{array}{c} l \\ \bullet \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \bullet \\ \circ_j \end{array} \begin{array}{c} l \\ \circ \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \bullet \\ \circ_j \end{array} \begin{array}{c} l \\ \circ \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \circ \\ \circ_j \end{array} \begin{array}{c} l \\ \bullet \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \circ \\ \circ_j \end{array} \begin{array}{c} l \\ \circ \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \circ \\ \circ_j \end{array} \begin{array}{c} l \\ \bullet \\ \circ_j \end{array} \begin{array}{c} l \\ \circ \\ \circ_j \end{array} \quad (14.11;4)$$

$$\begin{array}{c} k \\ \bullet \\ \circ_j \end{array} \begin{array}{c} l \\ \bullet \\ \circ_j \end{array} \begin{array}{c} m \\ \bullet \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \bullet \\ \circ_j \end{array} \begin{array}{c} l \\ \bullet \\ \circ_j \end{array} \begin{array}{c} m \\ \circ \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \bullet \\ \circ_j \end{array} \begin{array}{c} l \\ \circ \\ \circ_j \end{array} \begin{array}{c} m \\ \bullet \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \circ \\ \circ_j \end{array} \begin{array}{c} l \\ \bullet \\ \circ_j \end{array} \begin{array}{c} m \\ \bullet \\ \circ_j \end{array} \quad \begin{array}{c} k \\ \circ \\ \circ_j \end{array} \begin{array}{c} l \\ \bullet \\ \circ_j \end{array} \begin{array}{c} m \\ \circ \\ \circ_j \end{array} \quad \dots \quad (14.11;5)$$

Fig. 14.1. The derivatives of the exact solution

which satisfies:

a) the root of t is always fat; i.e., $t'(j) = \text{“fat”}$;

b) a meagre vertex has at most one son and this son has to be fat.

We denote by LNT_q the set of all labelled N-trees of order q .

The reason for condition (b) is that all derivatives of $g(y, y') = y'$ vanish identically with the exception of the first derivative with respect to y' .

In the sequel we use the notation *end-vertex* for a vertex which has no son. If no confusion is possible, we write t instead of (t, t') for a labelled N-tree.

Definition 14.3. For a labelled N-tree t we denote by

$$F^J(t)(y, y')$$

the expression which is a *sum* over the indices of all fat vertices of t (without “ j ”, the index of the root) and over the indices of all meagre end-vertices. The *general term* of this sum is a product of expressions

$$\frac{\partial^r f^K}{\partial y^L \dots \partial y'^M \dots} (y, y') \quad \text{and} \quad y'^K. \quad (14.12)$$

A factor of the first type appears if the fat vertex k is connected via a meagre son with l, \dots and directly with a fat son m, \dots ; a factor y'^K appears if “ k ” is the index of a meagre end-vertex. The vector $F(t)(y, y')$ is again called an *elementary differential*.

For some examples see Table 14.3 below. Observe that the indices of the meagre vertices, which are not end-vertices, play no role in the above definition. In analogy to Definition 2.4 we have

Definition 14.4. Two labelled N-trees (t, t') and (u, u') are *equivalent*, if they differ only by a permutation of their indices; i.e., if they have the same order, say

q , and if there exists a bijection $\sigma : A_q \rightarrow A_q$ with $\sigma(j) = j$, such that $t\sigma = \sigma u$ on $A_q \setminus \{j\}$ and $t'\sigma = u'$.

For example, the second and fourth labelled N-trees of formula (14.11;4) in Fig. 14.1 are equivalent; and also the second and fifth of formula (14.11;5).

Definition 14.5. An equivalence class of q th order labelled N-trees is called an *N-tree of order q* . The set of all N-trees of order q is denoted by NT_q . We further denote by $\alpha(t)$ the number of elements in the equivalence class t , i.e., the number of possible different monotonic labellings of t .

Representatives of N-trees up to order 5 are shown in Table 14.3. We are now able to give a closed formula for the derivatives of the exact solution of (14.10).

Theorem 14.6. *The exact solution of (14.10) satisfies*

$$y^{(q)} = \sum_{t \in LNT_{q-1}} F(t)(y, y') = \sum_{t \in NT_{q-1}} \alpha(t) F(t)(y, y'). \quad (14.11;q)$$

Proof. The general formula is obtained by continuing the computation for (14.11;2-4) as in Section II.2. \square

The Derivatives of the Numerical Solution

We first rewrite (14.4) as

$$\begin{aligned} g_i &= y_0 + c_i h y'_0 + \sum_{j=1}^s \bar{a}_{ij} h^2 f(g_j, g'_j), & g'_i &= y'_0 + \sum_{j=1}^s a_{ij} h f(g_j, g'_j) \\ y_1 &= y_0 + h y'_0 + \sum_{i=1}^s \bar{b}_i h^2 f(g_i, g'_i), & y'_1 &= y'_0 + \sum_{i=1}^s b_i h f(g_i, g'_i) \end{aligned} \quad (14.13)$$

so that the intermediate values g_i, g'_i are treated in the same way as y_1, y'_1 . In (14.13) there appear expressions of the form $h^2 \varphi(h)$ and $h \varphi(h)$. Therefore we have to use in addition to (2.4) the formula

$$(h^2 \varphi(h))^{(q)} \big|_{h=0} = q \cdot (q-1) \cdot (\varphi(h))^{(q-2)} \big|_{h=0}. \quad (14.14)$$

We now compute successively the derivatives of g_i^J and $g_i'^J$ at $h=0$:

$$(g_i^J)^{(1)} \big|_{h=0} = c_i y_0'^J \quad (14.15;1)$$

$$(g_i'^J)^{(1)} \big|_{h=0} = \sum_j a_{ij} f^J \big|_{y_0, y_0'} \quad (14.16;1)$$

$$(g_i^J)^{(2)}|_{h=0} = 2 \sum_j \bar{a}_{ij} f^J|_{y_0, y'_0}. \quad (14.15;2)$$

For a further differentiation we need

$$(f^J(g_j, g'_j))^{(1)} = \sum_K \frac{\partial f^J}{\partial y^K} (g_j, g'_j) (g_j^K)^{(1)} + \sum_K \frac{\partial f^J}{\partial y'^K} (g_j, g'_j) (g_j'^K)^{(1)}. \quad (14.17)$$

With this formula we then obtain

$$\begin{aligned} (g_i'^J)^{(2)}|_{h=0} &= 2 \sum_j a_{ij} c_j \sum_K \frac{\partial f^J}{\partial y^K} \cdot y'^K|_{y_0, y'_0} \\ &\quad + 2 \sum_{j,k} a_{ij} a_{jk} \sum_K \frac{\partial f^J}{\partial y'^K} \cdot f^K|_{y_0, y'_0} \end{aligned} \quad (14.16;2)$$

$$\begin{aligned} (g_i^J)^{(3)}|_{h=0} &= 3 \cdot 2 \sum_j \bar{a}_{ij} c_j \sum_K \frac{\partial f^J}{\partial y^K} \cdot y'^K|_{y_0, y'_0} \\ &\quad + 3 \cdot 2 \sum_{j,k} \bar{a}_{ij} a_{jk} \sum_K \frac{\partial f^J}{\partial y'^K} \cdot f^K|_{y_0, y'_0}. \end{aligned} \quad (14.15;3)$$

To write down a general formula we need

Definition 14.7. For a labelled N-tree we denote by $\Phi_j(t)$ the expression which is a sum over the indices of all fat vertices of t (without “ j ”, the index of the root). The general term of the sum is a product of

- a_{kl} if the fat vertex “ k ” has a fat son “ l ”;
- \bar{a}_{kl} if the fat vertex “ k ” is connected via a meagre son with “ l ”; and
- c_k^m if the fat vertex “ k ” is connected with m meagre end-vertices.

Theorem 14.8. The g_i, g'_i of (14.13) satisfy

$$(g_i)^{(q+1)}|_{h=0} = (q+1) \sum_{t \in LNT_q} \gamma(t) \sum_{j=1}^s \bar{a}_{ij} \Phi_j(t) F(t)(y_0, y'_0) \quad (14.15;q+1)$$

$$(g'_i)^{(q)}|_{h=0} = \sum_{t \in LNT_q} \gamma(t) \sum_{j=1}^s a_{ij} \Phi_j(t) F(t)(y_0, y'_0) \quad (14.16;q)$$

where $\gamma(t)$ is given in Definition 2.10.

Proof. For small values of q these formulas were obtained above; for general values of q they are proved like Theorem 2.11. System (14.2) is a special case of what will later be treated as a *partitioned system* (see Section II.15). Theorem 14.8 will then appear again in a new light. \square

Because of the similarity of the formulas for g_i and y_1 , g'_i and y'_1 we have

Theorem 14.9. *The numerical solution y_1, y'_1 of (14.13) satisfies*

$$(y_1)^{(q)}|_{h=0} = q \sum_{t \in LNT_{q-1}} \gamma(t) \sum_{i=1}^s \bar{b}_i \Phi_i(t) F(t)(y_0, y'_0) \quad (14.18; q)$$

$$(y'_1)^{(q-1)}|_{h=0} = \sum_{t \in LNT_{q-1}} \gamma(t) \sum_{i=1}^s b_i \Phi_i(t) F(t)(y_0, y'_0). \quad (14.19; q-1)$$

□

The Order Conditions

For the study of the order of a Nyström method (Definition 14.1) one has to compare the Taylor series of y_1, y'_1 with that of the true solution $y(x_0 + h), y'(x_0 + h)$.

Theorem 14.10. *A Nyström method (14.4) is of order p iff*

$$\sum_{i=1}^s \bar{b}_i \Phi_i(t) = \frac{1}{(\varrho(t) + 1) \cdot \gamma(t)} \quad \text{for } N\text{-trees } t \text{ with } \varrho(t) \leq p-1, \quad (14.20)$$

$$\sum_{i=1}^s b_i \Phi_i(t) = \frac{1}{\gamma(t)} \quad \text{for } N\text{-trees } t \text{ with } \varrho(t) \leq p. \quad (14.21)$$

Here $\varrho(t)$ denotes the order of the N -tree t , $\Phi_i(t)$ and $\gamma(t)$ are given by Definition 14.7 and formula (2.17).

Proof. The “if” part is an immediate consequence of Theorems 14.6 and 14.9. The “only if” part can be shown in the same way as for first order equations (cf. Exercise 4 of Section II.2). □

Let us briefly discuss whether the extra freedom in the choice of the parameters of (14.4) (by discarding the assumption (14.5)) can lead to a considerable improvement. Since the order conditions for Runge-Kutta methods (Theorem 2.13) are a subset of (14.21) (see Exercise 3 below), it is impossible to gain order with this extra freedom. Only some (never all) error coefficients can be made smaller. Therefore we shall turn to Nyström methods (14.8) for special second order differential equations (14.7).

For the study of the order conditions for (14.8) we write (14.7) in autonomous form

$$y'' = f(y). \quad (14.22)$$

This special form implies that those elementary differentials which contain derivatives with respect to y' vanish identically. Consequently, only the following subset of N-trees has to be considered:

Definition 14.11. An N-tree t is called a *special N-tree* or *SN-tree*, if the fat vertices have only meagre sons.

Theorem 14.12. A Nyström method (14.8) for the special differential equation (14.7) is of order p , iff

$$\sum_{i=1}^s \bar{b}_i \Phi_i(t) = \frac{1}{(\varrho(t) + 1) \cdot \gamma(t)} \quad \text{for SN-trees } t \text{ with } \varrho(t) \leq p - 1, \quad (14.23)$$

$$\sum_{i=1}^s b_i \Phi_i(t) = \frac{1}{\gamma(t)} \quad \text{for SN-trees } t \text{ with } \varrho(t) \leq p. \quad (14.24) \quad \square$$

All SN-trees up to order 5, together with the elementary differentials and the expressions Φ_j , ϱ , α , and γ , which are needed for the order conditions, are given in Table 14.3.

Higher order systems. The extension of the ideas of this section to *higher order* systems

$$y^{(n)} = f(x, y, y', \dots, y^{(n-1)}) \quad (14.25)$$

is now more or less straightforward. Again, a real improvement is only possible in the case when the right-hand side of (14.25) depends only on x and y . A famous paper on this subject is the work of Zurmühl (1948). Tables of order conditions and methods are given in Hebsacker (1982).

On the Construction of Nyström Methods

The following simplifying assumptions are useful for the construction of Nyström methods.

Lemma 14.13. Under the assumption

$$\bar{b}_i = b_i(1 - c_i) \quad i = 1, \dots, s \quad (14.26)$$

the condition (14.24) implies (14.23).

Proof. Let t be an SN-tree of order $\leq p - 1$ and denote by u the SN-tree of order $\varrho(t) + 1$ obtained from t by attaching a new branch with a meagre vertex to the root of t . By Definition 14.7 we have $\Phi_i(u) = c_i \Phi_i(t)$ and from formula (2.17) it

Table 14.3. SN-trees, elementary differentials and order conditions

t	graph	$\varrho(t)$	$\alpha(t)$	$\gamma(t)$	$F^J(t)(y, y')$	$\Phi_j(t)$
t_1		1	1	1	f^J	1
t_2		2	1	2	$\sum_K f_K^J y'^K$	c_j
t_3		3	1	3	$\sum_{K,L} f_{KL}^J y'^K y'^L$	c_j^2
t_4		3	1	6	$\sum_L f_L^J f^L$	$\sum_l \bar{a}_{jl}$
t_5		4	1	4	$\sum_{K,L,M} f_{KLM}^J y'^K y'^L y'^M$	c_j^3
t_6		4	3	8	$\sum_{L,M} f_{LM}^J y'^L f^M$	$\sum_m c_j \bar{a}_{jm}$
t_7		4	1	24	$\sum_{L,M} f_L^J f_M^L y'^M$	$\sum_l \bar{a}_{jl} c_l$
t_8		5	1	5	$\sum_{K,L,M,P} f_{KLMP}^J y'^K y'^L y'^M y'^P$	c_j^4
t_9		5	6	10	$\sum_{L,M,P} f_{LMP}^J y'^L y'^M f^P$	$\sum_p c_j^2 \bar{a}_{jp}$
t_{10}		5	3	20	$\sum_{M,P} f_{MP}^J f^M f^P$	$\sum_{m,p} \bar{a}_{jm} \bar{a}_{jp}$
t_{11}		5	4	30	$\sum_{L,M,P} f_{LP}^J f_M^L y'^M y'^P$	$\sum_l c_j \bar{a}_{jl} c_l$
t_{12}		5	1	60	$\sum_{L,M,P} f_L^J f_{MP}^L y'^M y'^P$	$\sum_l \bar{a}_{jl} c_l^2$
t_{13}		5	1	120	$\sum_{L,P} f_L^J f_P^L f^P$	$\sum_{l,p} \bar{a}_{jl} \bar{a}_{lp}$

follows that $\gamma(u) = (\varrho(t) + 1)\gamma(t)/\varrho(t)$. The conclusion now follows since

$$\sum_{i=1}^s \bar{b}_i \Phi_i(t) = \sum_{i=1}^s b_i \Phi_i(t) - \sum_{i=1}^s b_i \Phi_i(u) = \frac{1}{\gamma(t)} - \frac{1}{\gamma(u)} = \frac{1}{(\varrho(t) + 1)\gamma(t)}. \quad \square$$

Lemma 14.14. *Let t and u be two SN-trees as sketched in Fig. 14.2, where the encircled parts are assumed to be identical. Then under the assumption*

$$\sum_{j=1}^s \bar{a}_{ij} = \frac{c_i^2}{2} \quad i = 1, \dots, s \quad (14.27)$$

the order conditions for t and u are the same.

Proof. It follows from Definition 14.7 and (14.27) that $\Phi_i(t) = \Phi_i(u)/2$ and from formula (2.17) that $\gamma(t) = 2\gamma(u)$. Both order conditions are thus identical. \square

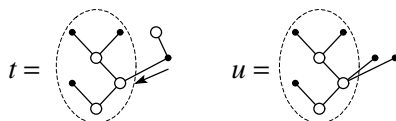


Fig. 14.2. Trees of Lemma 14.14

Condition (14.26) allows us to neglect the equations (14.23), while condition (14.27) plays a similar role to that of (1.9) for Runge-Kutta methods. It expresses the fact that the g_i of (14.13) approximate $y(x_0 + c_i h)$ up to $\mathcal{O}(h^3)$. As a consequence of Lemma 14.14, SN-trees which have at least one fat end-vertex can be left out (i.e., $t_4, t_6, t_9, t_{10}, t_{13}$ of Table 14.3).

With the help of (14.26) and (14.27) *explicit Nyström methods* (14.8) of order 5 with $s = 4$ can now easily be constructed: the order conditions for the trees t_1, t_2, t_3, t_5 and t_8 just indicate that the quadrature formula with nodes $c_1 = 0, c_2, c_3, c_4$ and weights b_1, b_2, b_3, b_4 is of order 5. Thus the nodes c_i have to satisfy the orthogonality relation

$$\int_0^1 x(x - c_2)(x - c_3)(x - c_4) dx = 0$$

and we see that two degrees of freedom are still left in the choice of the quadrature formula. The \bar{a}_{ij} are now uniquely determined and can be computed as follows: \bar{a}_{21} is given by (14.27) for $i = 2$. The order conditions for t_7 and t_{11} constitute two linear equations for the unknowns

$$\sum_{j=1}^2 \bar{a}_{3j} c_j \quad \text{and} \quad \sum_{j=1}^3 \bar{a}_{4j} c_j.$$

Together with (14.27, $i = 3$) one now obtains \bar{a}_{31} and \bar{a}_{32} . Finally, the order condition for t_{12} leads to $\sum_j \bar{a}_{4j} c_j^2$ and the remaining coefficients $\bar{a}_{41}, \bar{a}_{42}, \bar{a}_{43}$ can be computed from a Vandermonde-type linear system. The method of Table 14.2 is obtained in this way.

For still higher order methods it is helpful to use further simplifying assumptions; for example

$$\sum_{j=1}^s \bar{a}_{ij} c_j^q = \frac{c_i^{q+2}}{(q+2)(q+1)} \quad (14.28)$$

which, for $q = 0$, reduces to (14.27), and

$$\sum_{i=1}^s b_i c_i^q \bar{a}_{ij} = b_j \left(\frac{c_j^{q+2}}{(q+2)(q+1)} - \frac{c_j}{q+1} + \frac{1}{q+2} \right) \quad (14.29)$$

which can be considered a generalization of condition $D(\zeta)$ of Section II.7. For more details we refer to Hairer & Wanner (1976) and also to Albrecht (1955), Battin (1976), Beentjes & Gerritsen (1976), Hairer (1977, 1982), where Nyström methods of higher order are presented.

Embedded Nyström methods. For an efficient implementation we need a step size control mechanism. This can be performed in the same manner as for Runge-Kutta methods (see Section II.4). One can either apply Richardson extrapolation in order to estimate the local error, or construct embedded Nyström methods.

A series of embedded Nyström methods has been constructed by Fehlberg (1972). These methods use a $(p+1)$ -st order approximation to $y(x_0+h)$ for step size control. A $(p+1)$ -st order approximation to $y'(x_0+h)$ is not needed, since the lower order approximations are used for step continuation.

As for first order differential equations, local extrapolation — to use the higher order approximations for step continuation — turns out to be superior. Bettis (1973) was apparently the first to use this technique. His proposed method is of order 5(4). A method of order 7(6) has been constructed by Dormand & Prince (1978), methods of order 8(7), 9(8), 10(9) and 11(10) are given by Filippi & Gräf (1986) and further methods of order 8(6) and 12(10) are presented by Dormand, El-Mikkawy & Prince (1987).

In certain situations (see Section II.6) it is important that a Nyström method be equipped with a dense output formula. Such procedures are given by Dormand & Prince (1987) and, for general initial value problems $y'' = f(x, y, y')$, by Fine (1987).

An Extrapolation Method for $y'' = f(x, y)$

Les calculs originaux, comprenant environ 3.000 pages in-folio avec 358 grandes planches, et encore 3.800 pages de développements mathématiques correspondants, appartiennent maintenant à la collection de manuscrits de la Bibliothèque de l'Université, Christiania. (Störmer 1921)

If we rewrite the differential equation (14.7) as a first order system

$$\begin{pmatrix} y \\ y' \end{pmatrix}' = \begin{pmatrix} y' \\ f(x, y) \end{pmatrix}, \quad \begin{pmatrix} y \\ y' \end{pmatrix}(x_0) = \begin{pmatrix} y_0 \\ y'_0 \end{pmatrix} \quad (14.30)$$

we can apply the GBS-algorithm (9.13) directly to (14.30); this yields

$$y_1 = y_0 + hy'_0 \quad (14.31a)$$

$$y'_1 = y'_0 + hf(x_0, y_0)$$

$$y_{i+1} = y_{i-1} + 2hy'_i \quad (14.31b)$$

$$y'_{i+1} = y'_{i-1} + 2hf(x_i, y_i) \quad i = 1, 2, \dots, 2n$$

$$S_h(x) = (y_{2n-1} + 2y_{2n} + y_{2n+1})/4 \quad (14.31c)$$

$$S'_h(x) = (y'_{2n-1} + 2y'_{2n} + y'_{2n+1})/4.$$

Here, $S_h(x)$ and $S'_h(x)$ are the numerical approximations to $y(x)$ and $y'(x)$ at $x = x_0 + H$, where $H = 2nh$ and $x_i = x_0 + ih$. We now make the following important observation: for the computation of $y_0, y_2, y_4, \dots, y_{2n}$ (even indices) and

of $y'_1, y'_3, \dots, y'_{2n+1}$ (odd indices) only the function values $f(x_0, y_0)$, $f(x_2, y_2)$, \dots , $f(x_{2n}, y_{2n})$ have to be calculated. Furthermore, we know from (9.17) that y_{2n} and $(y'_{2n-1} + y'_{2n+1})/2$ each possess an asymptotic expansion in even powers of h . It is therefore obvious that (14.31c) should be replaced by (Gragg 1965)

$$\begin{aligned} S_h(x) &= y_{2n} \\ S'_h(x) &= (y'_{2n-1} + y'_{2n+1})/2. \end{aligned} \quad (14.31c')$$

Using this final step, the number of function evaluations is reduced by a factor of two. These numerical approximations can now be used for extrapolation. We take the harmonic sequence (9.8'), put

$$T_{i1} = S_h(x_0 + H), \quad T'_{i1} = S'_h(x_0 + H)$$

and compute the extrapolated expressions $T_{i,j}$ and $T'_{i,j}$ by the Aitken & Neville formula (9.10).

Remark. Eliminating the y'_j -values in (14.31b) we obtain the equivalent formula

$$y_{i+2} - 2y_i + y_{i-2} = (2h)^2 f(x_i, y_i), \quad (14.32)$$

which is often called *Störmer's rule*. For the implementation the formulation (14.31b) is to be preferred, since it is more stable with respect to round-off errors (see Section III.10).

Dense output. As for the derivation of Section II.9 for the GBS algorithm we shall do Hermite interpolation based on derivatives of the solution at x_0 , $x_0 + H$ and $x_0 + H/2$. At the endpoints of the considered interval we have y_0 , y'_0 , $y''_0 = f(x_0, y_0)$ and y_1 , y'_1 , y''_1 at our disposal. The derivatives at the midpoint can be obtained by extrapolation of suitable differences of function values. However, one has to take care of the fact that y_i and $f(x_i, y_i)$ are available only for even indices, whereas y'_i is available for odd indices only. For the same reason as for the GBS method, the step number sequence has to satisfy (9.34). For notational convenience, the following description is restricted to the sequence (9.35).

We suppose that T_{kk} and T'_{kk} are accepted approximations to the solution. Then the construction of a dense output formula can be summarized as follows:

Step 1. For each $j \in \{1, \dots, k\}$ compute the approximations to the derivatives of $y(x)$ at $x_0 + H/2$ by (δ is the central difference operator):

$$\begin{aligned} d_j^{(0)} &= \frac{1}{2} (y_{n_j/2-1} + y_{n_j/2+1}), & d_j^{(1)} &= y'_{n_j/2}, \\ d_j^{(\kappa)} &= \frac{1}{2} \cdot \frac{1}{(2h_j)^{\kappa-2}} \left(\delta^{\kappa-2} f_{n_j/2-1}^{(j)} + \delta^{\kappa-2} f_{n_j/2+1}^{(j)} \right), & \kappa &= 2, 4, \dots, 2j, \\ d_j^{(\kappa)} &= \frac{\delta^{\kappa-2} f_{n_j/2}^{(j)}}{(2h_j)^{\kappa-2}}, & \kappa &= 3, 5, \dots, 2j+1. \end{aligned} \quad (14.33)$$

Step 2. Extrapolate $d_j^{(0)}, d_j^{(1)}$ ($k-1$) times and $d_j^{(2\ell)}, d_j^{(2\ell+1)}$ ($k-\ell$) times to obtain improved approximations $d^{(\kappa)}$ to $y^{(\kappa)}(x_0 + H/2)$.

Step 3. For given μ ($-1 \leq \mu \leq 2k+1$) define the polynomial $P_\mu(\theta)$ of degree $\mu+6$ by

$$\begin{aligned} P_\mu(0) &= y_0, & P'_\mu(0) &= y'_0, & P''_\mu(0) &= f(x_0, y_0) \\ P_\mu(1) &= T_{kk}, & P'_\mu(1) &= T'_{kk}, & P''_\mu(1) &= f(x_0 + H, T_{kk}) \\ P_\mu^{(\kappa)}(1/2) &= H^\kappa d^{(\kappa)} & & \text{for } \kappa = 0, 1, \dots, \mu. \end{aligned} \quad (14.34)$$

Since T_{kk}, T'_{kk} are the initial values for the next step, the dense output obtained by the above algorithm is a global \mathcal{C}^2 approximation to the solution. It satisfies

$$y(x_0 + \theta H) - P_\mu(\theta) = \mathcal{O}(H^{2k}) \quad \text{if } \mu \geq 2k-7 \quad (14.35)$$

(compare Theorem 9.5). In the code ODEX2 of the Appendix the value $\mu = 2k-5$ is suggested as standard choice.

Problems for Numerical Comparisons

PLEI — the celestial mechanics problem (10.3) which is the only problem of Section II.10 already in the special form (14.7).

ARES — the AREnstorf orbit in Second order form (14.7). This is the restricted three body problem (0.1) with initial values (0.2) integrated over one period $0 \leq x \leq x_{\text{end}}$ (see Fig. 0.1) in a *fixed* coordinate system. Then the equations of motion become

$$\begin{aligned} y_1'' &= \mu' \frac{a_1(x) - y_1}{D_1} + \mu \frac{b_1(x) - y_1}{D_2} \\ y_2'' &= \mu' \frac{a_2(x) - y_2}{D_1} + \mu \frac{b_2(x) - y_2}{D_2} \end{aligned} \quad (14.36)$$

where

$$D_1 = ((y_1 - a_1(x))^2 + (y_2 - a_2(x))^2)^{3/2}, \quad D_2 = ((y_1 - b_1(x))^2 + (y_2 - b_2(x))^2)^{3/2}$$

and the movement of sun and moon are described by

$$a_1(x) = -\mu \cos x \quad a_2(x) = -\mu \sin x \quad b_1(x) = \mu' \cos x \quad b_2(x) = \mu' \sin x.$$

The initial values

$$\begin{aligned} y_1(0) &= 0.994, & y_1'(0) &= 0, & y_2(0) &= 0, \\ y_2'(0) &= -2.00158510637908252240537862224 + 0.994, \\ x_{\text{end}} &= 17.0652165601579625588917206249, \end{aligned}$$

are those of (0.2) enlarged by the speed of the rotation. The exact solution values are the initial values transformed by the rotation of the coordinate system.

CPEN — the nonlinear Coupled PENDulum (see Fig. 14.3).

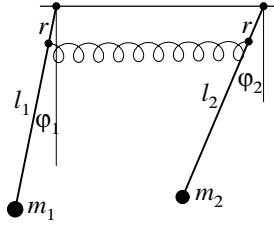


Fig. 14.3. Coupled pendulum

The kinetic as well as potential energies

$$\begin{aligned} T &= \frac{m_1 l_1^2 \dot{\varphi}_1^2}{2} + \frac{m_2 l_2^2 \dot{\varphi}_2^2}{2} \\ V &= -m_1 l_1 \cos \varphi_1 - m_2 l_2 \cos \varphi_2 + \frac{c_0 r^2 (\sin \varphi_1 - \sin \varphi_2)^2}{2} \end{aligned}$$

lead by Lagrange theory (equations (I.6.21)) to

$$\begin{aligned} \ddot{\varphi}_1 &= -\frac{\sin \varphi_1}{l_1} - \frac{c_0 r^2}{m_1 l_1^2} (\sin \varphi_1 - \sin \varphi_2) \cos \varphi_1 + f(t) \\ \ddot{\varphi}_2 &= -\frac{\sin \varphi_2}{l_2} - \frac{c_0 r^2}{m_2 l_1^2} (\sin \varphi_2 - \sin \varphi_1) \cos \varphi_2. \end{aligned} \tag{14.37}$$

We choose the parameters

$$l_1 = l_2 = 1, \quad m_1 = 1, \quad m_2 = 0.99, \quad r = 0.1, \quad c_0 = 0.01, \quad t_{\text{end}} = 496$$

and all initial values and speeds for $t = 0$ equal to zero. The first pendulum is then pushed into movement by a (somewhat idealized) hammer as

$$f(t) = \begin{cases} \sqrt{1 - (1 - t)^2} & \text{if } |t - 1| \leq 1; \\ 0 & \text{otherwise.} \end{cases}$$

The resulting solutions are displayed in Fig. 14.4. The nonlinearities in this problem produce quite different sausages (cf. “Mon Oncle” de Jacques Tati 1958) from those people are accustomed to from linear problems (cf. Sommerfeld 1942, §20).

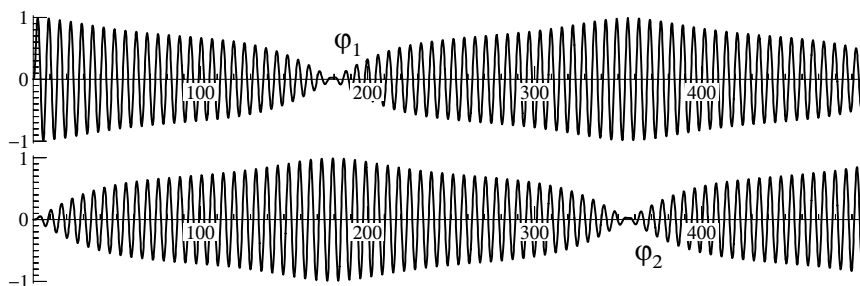


Fig. 14.4. Movement of the coupled pendulum (14.37)

WPLT — the Weak PLaTe, i.e., the PLATE problem of Section IV.10 (see Volume II) with weakened stiffness. We use precisely the same equations as (IV.10.6) and reduce the stiffness parameter σ from $\sigma = 100$ to $\sigma = 1/16$. We also remove the friction ($\omega = 0$ instead of $\omega = 1000$) so that the problem becomes purely of second order. It is linear, nonautonomous, and of dimension 40.

Performance of the Codes

Several codes were applied to each of the above four problems with 89 different tolerances between $Tol = 10^{-3}$ and $Tol = 10^{-14}$ (exactly as in Section II.10). The number of function evaluations (Fig. 14.5) and the computer time (Fig. 14.6) on a Sun Workstation (SunBlade 100) are plotted as a function of the global error at the endpoint of the integration interval. The codes used are the following:

RKN6 — symbol \star — is the low order option of the Runge-Kutta-Nyström code presented in Brankin, Gladwell, Dormand, Prince & Seward (1989). It is based on a fixed-order embedded Nyström method of order 6(4), whose coefficients are given in Dormand & Prince (1987). This code is provided with a dense output.

RKN12 — symbol \boxtimes — is the high order option of the Runge-Kutta-Nyström code presented in Brankin & al. (1989). It is based on the method of order 12(10), whose coefficients are given in Dormand, El-Mikkawy & Prince (1987). This code is not equipped with a dense output.

ODEX2 — symbol \bigcirc — is the extrapolation code based on formula (14.31a,b,c') and uses the harmonic step number sequence (see Appendix). It is implemented in the same way as ODEX (the extrapolation code for first order differential equations). In particular, the order and step size strategy is that of Section II.9. A dense output is available. Similar results are obtained by the code DIFEX2 of Deuffhard & Bauer (see Deuffhard 1985).

In order to demonstrate the superiority of the special methods for $y'' = f(x, y)$, we have included the results obtained by DOP853 (symbol \star) and ODEX (symbol

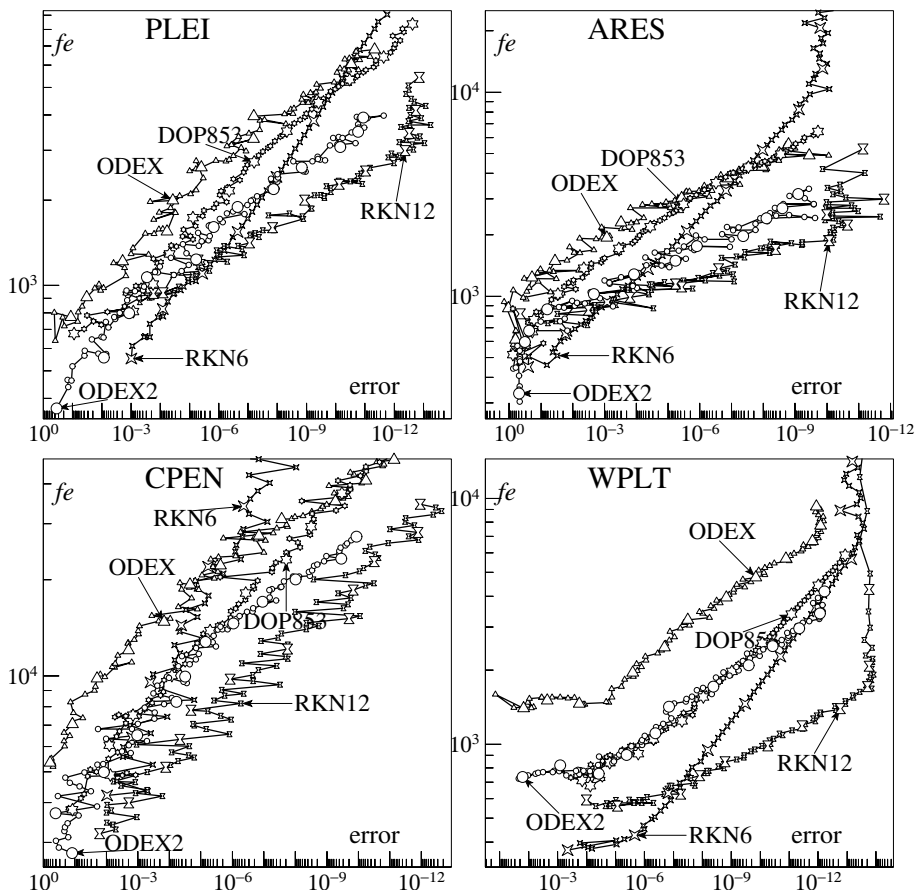


Fig. 14.5. Precision versus function evaluations

\triangle) which were already described in Section II.10. For their application we had to rewrite the four problems as a first order system by introducing the first derivatives as new variables. The code ODEX2 is nearly twice as efficient as ODEX which is in agreement with the theoretical considerations. Similarly the Runge-Kutta-Nyström codes RKN6 and RKN12 are a real improvement over DOP853.

A comparison of Fig. 14.5 and 14.6 shows a significant difference. The extrapolation codes ODEX and ODEX2 are relatively better on the “time”-pictures than for the function evaluation counts. With the exception of problem WPLT the performance of the code ODEX2 then becomes comparable to that of RKN12. As can be observed especially at the WPLT problem, the code RKN12 overshoots, for stringent tolerances, significantly the desired precision. It becomes less efficient if Tol is chosen too close to $Uround$.

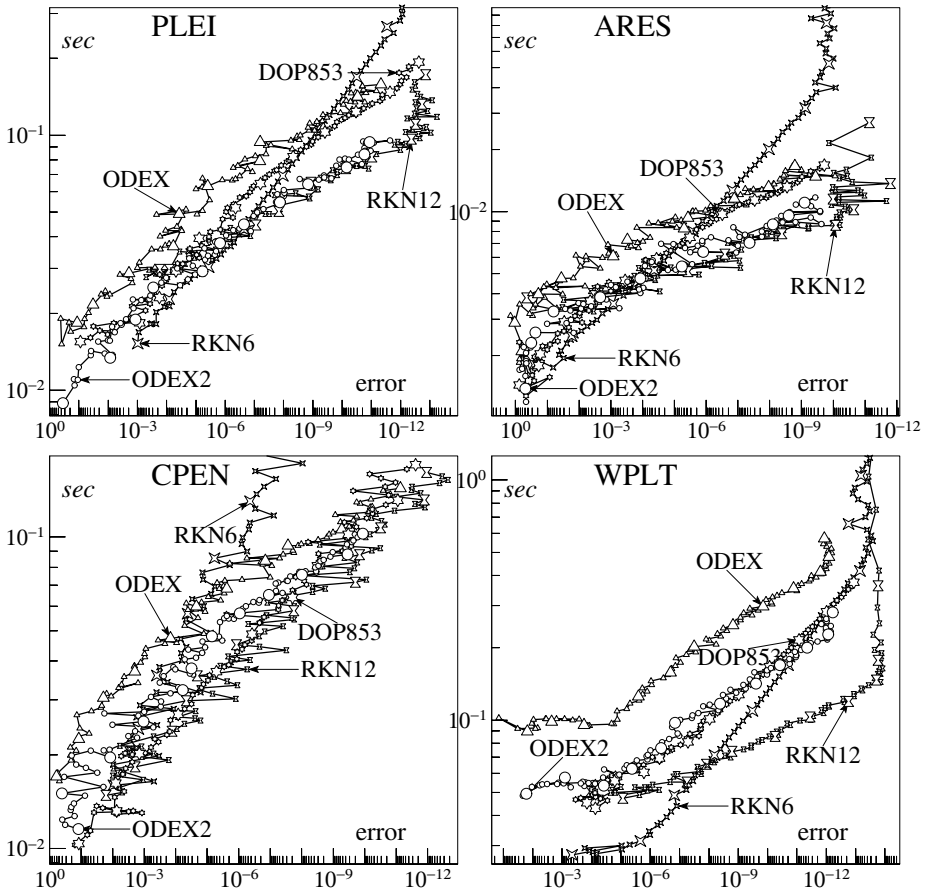


Fig. 14.6. Precision versus computing time

Exercises

1. Verify that the methods of Table 14.2 are of order 4 and 5, respectively.
2. The error coefficients of a p th order Nyström method are defined by

$$\begin{aligned} e(t) &= 1 - (\varrho(t) + 1)\gamma(t) \sum_i \bar{b}_i \Phi_i(t) & \text{for } \varrho(t) = p, \\ e'(t) &= 1 - \gamma(t) \sum_i b_i \Phi_i(t) & \text{for } \varrho(t) = p + 1. \end{aligned} \quad (14.38)$$

- a) The assumption (14.26) implies that

$$e(t) = -\varrho(t)e'(u) \quad \text{for } \varrho(t) = p,$$

where u is the N-tree obtained from t by adding a branch with a meagre vertex to the root of t .

- b) Compute the error coefficients of Nyström's method (Table 14.1) and compare them to those of the classical Runge-Kutta method.
3. Show that the order conditions for Runge-Kutta methods (Theorem 2.13) are a subset of the conditions (14.21). They correspond to the N-trees, all of whose vertices are fat.
4. Sometimes the definition of order of Nyström methods (14.8) is relaxed to

$$\begin{aligned} y(x_0 + h) - y_1 &= \mathcal{O}(h^{p+1}) \\ y'(x_0 + h) - y'_1 &= \mathcal{O}(h^p) \end{aligned} \quad (14.39)$$

(see Nyström 1925). Show that the conditions (14.39) are not sufficient to obtain global convergence of order p .

Hint. Investigate the asymptotic expansion of the global error with the help of Theorem 8.1 and formula (8.8).

5. The numerical solutions T_{kk} and T'_{kk} of the extrapolation method of this section are equivalent to a Nyström method of order $p = 2k$ with $s = p^2/8 + p/4 + 1$ stages.
6. A *collocation method* for $y'' = f(x, y, y')$ (or $y'' = f(x, y)$) can be defined as follows: let $u(x)$ be a polynomial of degree $s + 1$ defined by

$$\begin{aligned} u(x_0) &= y_0, & u'(x_0) &= y'_0 \\ u''(x_0 + c_i h) &= f(x_0 + c_i h, u(x_0 + c_i h), u'(x_0 + c_i h)), & i &= 1, \dots, s, \end{aligned} \quad (14.40)$$

then the numerical solution is given by $y_1 = u(x_0 + h)$, $y'_1 = u'(x_0 + h)$.

- a) Prove that this collocation method is equivalent to the Nyström method (14.4) where

$$\begin{aligned} a_{ij} &= \int_0^{c_i} \ell_j(t) dt, & \bar{a}_{ij} &= \int_0^{c_i} (c_i - t) \ell_j(t) dt, \\ b_i &= \int_0^1 \ell_i(t) dt, & \bar{b}_i &= \int_0^1 (1 - t) \ell_i(t) dt, \end{aligned} \quad (14.41)$$

and $\ell_j(t)$ are the Lagrange polynomials of (7.17).

- b) The a_{ij} satisfy $C(s)$ (see Theorem 7.8) and the \bar{a}_{ij} satisfy (14.28) for $q = 0, 1, \dots, s - 1$. These equations uniquely define a_{ij} and \bar{a}_{ij} .
- c) In general, a_{ij} and \bar{a}_{ij} do not satisfy (14.5).
- d) If $M(t) = \prod_{i=1}^s (t - c_i)$ is orthogonal to all polynomials of degree $r - 1$,

$$\int_0^1 M(t) t^{q-1} dt = 0, \quad q = 1, \dots, r,$$

then the collocation method (14.40) has order $p = s + r$.

- e) The polynomial $u(x)$ yields an approximation to the solution $y(x)$ on the whole interval $[x_0, x_0 + h]$. The following estimates hold:

$$y(x) - u(x) = \mathcal{O}(h^{s+2}), \quad y'(x) - u'(x) = \mathcal{O}(h^{s+1}).$$

II.15 P-Series for Partitioned Differential Equations

Divide ut regnes

(N. Machiavelli 1469-1527)

In the previous section we considered direct methods for second order differential equations $y'' = f(y, y')$. The idea was to write the equation as a partitioned differential system

$$\begin{pmatrix} y \\ y' \end{pmatrix}' = \begin{pmatrix} y' \\ f(y, y') \end{pmatrix} \quad (15.1)$$

and to discretize the two components, y and y' , by different formulas. There are many other situations where the problem possesses a natural partitioning. Typical examples are the Hamiltonian equations (I.6.26, I.14.26) and singular perturbation problems (see Chapter VI of Volume II). It may also be of interest to separate linear and nonlinear parts or the “non-stiff” and “stiff” components of a differential equation.

We suppose that the differential system is partitioned as

$$\begin{pmatrix} y_a \\ y_b \end{pmatrix}' = \begin{pmatrix} f_a(y_a, y_b) \\ f_b(y_a, y_b) \end{pmatrix} \quad (15.2)$$

where the solution vector is separated into two components y_a, y_b , each of which may itself be a vector. An extension to more components is straight-forward.

For the numerical solution of (15.2) we consider the *partitioned method*

$$\begin{aligned} k_i &= f_a \left(y_{a0} + h \sum_{j=1}^s a_{ij} k_j, y_{b0} + h \sum_{j=1}^s \hat{a}_{ij} \ell_j \right) \\ \ell_i &= f_b \left(y_{a0} + h \sum_{j=1}^s a_{ij} k_j, y_{b0} + h \sum_{j=1}^s \hat{a}_{ij} \ell_j \right) \\ y_{a1} &= y_{a0} + h \sum_{i=1}^s b_i k_i, \quad y_{b1} = y_{b0} + h \sum_{i=1}^s \hat{b}_i \ell_i \end{aligned} \quad (15.3)$$

where the coefficients a_{ij}, b_i and \hat{a}_{ij}, \hat{b}_i represent two different Runge-Kutta schemes. The first methods of this type are due to Hofer (1976) and Griepentrog (1978) who apply an explicit method to the nonstiff part and an implicit method to the stiff part of a differential equation. Later Rentrop (1985) modified this idea by combining explicit Runge-Kutta methods with Rosenbrock-type methods (Sec-

tion IV.7). Recent interest for partitioned methods came up when solving Hamiltonian systems (see Section II.16 below).

The subject of this section is the derivation of the order conditions for method (15.3). For order p it is necessary that each of the two Runge-Kutta schemes under consideration be of order p . This can be seen by applying the method to $y'_a = f_a(y_a)$, $y'_b = f_b(y_b)$. But this is not sufficient, the coefficients have to satisfy certain *coupling conditions*. In order to understand this, we first look at the derivatives of the exact solution of (15.2). Then we generalize the theory of B-series (see Section II.12) to the new situation (Hairer 1981) and derive the order conditions in the same way as in II.12 for Runge-Kutta methods.

Derivatives of the Exact Solution, P-Trees

In order to avoid sums and unnecessary indices we assume that y_a and y_b in (15.2) are scalar quantities. All subsequent formulas remain valid for vectors if the derivatives are interpreted as multi-linear mappings. Differentiating (15.2) and inserting (15.2) again for the derivatives we obtain for the first component y_a

$$y_a^{(1)} = f_a \quad (15.4;1)$$

$$y_a^{(2)} = \frac{\partial f_a}{\partial y_a} f_a + \frac{\partial f_a}{\partial y_b} f_b \quad (15.4;2)$$

$$y_a^{(3)} = \frac{\partial^2 f_a}{\partial y_a^2} (f_a, f_a) + \frac{\partial^2 f_a}{\partial y_b \partial y_a} (f_b, f_a) + \frac{\partial f_a}{\partial y_a} \frac{\partial f_a}{\partial y_a} f_a + \frac{\partial f_a}{\partial y_a} \frac{\partial f_a}{\partial y_b} f_b \\ + \frac{\partial^2 f_a}{\partial y_a \partial y_b} (f_a, f_b) + \frac{\partial^2 f_a}{\partial y_b^2} (f_b, f_b) + \frac{\partial f_a}{\partial y_b} \frac{\partial f_b}{\partial y_a} f_a + \frac{\partial f_a}{\partial y_b} \frac{\partial f_b}{\partial y_b} f_b. \quad (15.4;3)$$

Similar formulas hold for the derivatives of y_b .

For a graphical representation of these formulas we need two different kinds of vertices. As in Section II.14 we use “meagre” and “fat” vertices, which will correspond to f_a and f_b , respectively. Formulas (15.4) can then be represented as shown in Fig. 15.1.

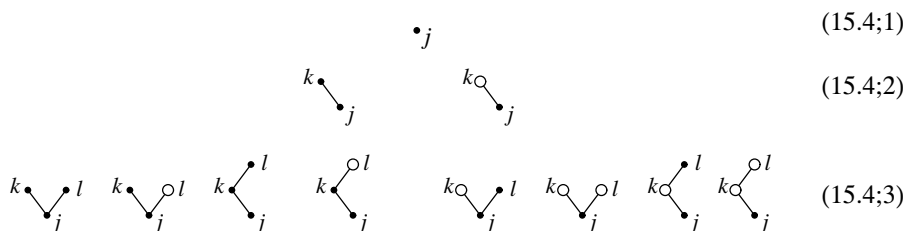


Fig. 15.1. The derivatives of the exact solution y_a

Definition 15.1. A *labelled P-tree* of order q is a labelled tree (see Definition 2.2)

$$t : A_q \setminus \{j\} \rightarrow A_q$$

together with a mapping

$$t' : A_q \rightarrow \{\text{“meagre”}, \text{“fat”}\}.$$

We denote by LP_q^a the set of those labelled P-trees of order q , whose root is meagre (i.e., $t'(j) = \text{“meagre”}$). Similarly, LP_q^b is the set of q th order labelled P-trees with a “fat” root.

Due to the symmetry of the second derivative the 2nd and 5th expressions in (15.4;3) are equal. We therefore define:

Definition 15.2. Two labelled P-trees (t, t') and (u, u') are *equivalent*, if they have the same order, say q , and if there exists a bijection $\sigma : A_q \rightarrow A_q$ such that $\sigma(j) = j$ and the following diagram commutes:

$$\begin{array}{ccccc} A_q \setminus \{j\} & \xrightarrow{t} & A_q & \xrightarrow{t'} & \{\text{“meagre”}, \text{“fat”}\} \\ \sigma \downarrow & & \sigma \downarrow & \nearrow & \\ A_q \setminus \{j\} & \xrightarrow{u} & A_q & \nearrow u' & \end{array}$$

Definition 15.3. An equivalence class of q th order labelled P-trees is called a *P-tree* of order q . The set of all P-trees of order q with a meagre root is denoted by TP_q^a , that with a fat root by TP_q^b . For a P-tree t we denote by $\varrho(t)$ the *order* of t , and by $\alpha(t)$ the number of elements in the equivalence class t .

Examples of P-trees together with the numbers $\varrho(t)$ and $\alpha(t)$ are given in Table 15.1 below. We first discuss a recursive representation of P-trees (extension of Definition 2.12), which is fundamental for the following theory.

Definition 15.4. Let t_1, \dots, t_m be P-trees. We then denote by

$$t = {}_a[t_1, \dots, t_m] \quad (15.5)$$

the unique P-tree t such that the root is “meagre” and the P-trees t_1, \dots, t_m remain if the root and the adjacent branches are chopped off. Similarly, we denote by ${}_b[t_1, \dots, t_m]$ the P-tree whose new root is “fat” (see Fig. 15.2). We further denote by τ_a and τ_b the meagre and fat P-trees of order one.

Our next aim is to make precise the connection between P-trees and the expressions of the formulas (15.4). For this we use the notation

$$w(t) = \begin{cases} a & \text{if the root of } t \text{ is meagre,} \\ b & \text{if the root of } t \text{ is fat.} \end{cases} \quad (15.6)$$

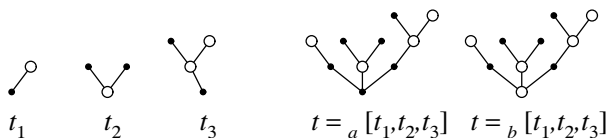


Fig. 15.2. Recursive definition of P-trees

Definition 15.5. The *elementary differentials*, corresponding to (15.2), are defined recursively by $(y = (y_a, y_b))$

$$F(\tau_a)(y) = f_a(y), \quad F(\tau_b)(y) = f_b(y)$$

and

$$F(t)(y) = \frac{\partial^m f_{w(t)}(y)}{\partial y_{w(t_1)} \dots \partial y_{w(t_m)}} \cdot (F(t_1)(y), \dots, F(t_m)(y))$$

for $t = {}_a[t_1, \dots, t_m]$ or $t = {}_b[t_1, \dots, t_m]$.

Elementary differentials for P-trees up to order 3 are given explicitly in Table 15.1.

We now return to the starting-point of this section and continue the differentiation of formulas (15.4). Using the notation of labelled P-trees, one sees that a differentiation of $F(t)(y_a, y_b)$ can be interpreted as an addition of a new branch with a meagre or fat vertex and a new summation letter to each vertex of the labelled P-tree t . In the same way as we proved Theorem 2.6 for non-partitioned differential equations, we arrive at

Theorem 15.6. *The derivatives of the exact solution of (15.2) satisfy*

$$\begin{aligned} y_a^{(q)} &= \sum_{t \in LTP_q^a} F(t)(y_a, y_b) = \sum_{t \in TTP_q^a} \alpha(t) F(t)(y_a, y_b) \\ y_b^{(q)} &= \sum_{t \in LTP_q^b} F(t)(y_a, y_b) = \sum_{t \in TTP_q^b} \alpha(t) F(t)(y_a, y_b). \end{aligned} \quad (15.4;q)$$

□

Table 15.1. P-trees and their elementary differentials

P-tree	repr. (15.5)	$\varrho(t)$	$\alpha(t)$	elem. differential	$\Phi_j(t)$
	τ_a	1	1	f_a	1
	$a[\tau_a]$	2	1	$\frac{\partial f_a}{\partial y_a} f_a$	$\sum_k a_{jk}$
	$a[\tau_b]$	2	1	$\frac{\partial f_a}{\partial y_b} f_b$	$\sum_k \hat{a}_{jk}$
	$a[\tau_a, \tau_a]$	3	1	$\frac{\partial^2 f_a}{\partial y_a^2} (f_a, f_a)$	$\sum_{k,l} a_{jk} a_{jl}$
	$a[\tau_a, \tau_b]$	3	2	$\frac{\partial^2 f_a}{\partial y_a \partial y_b} (f_a, f_b)$	$\sum_{k,l} a_{jk} \hat{a}_{jl}$
	$a[\tau_b, \tau_b]$	3	1	$\frac{\partial^2 f_a}{\partial y_b^2} (f_b, f_b)$	$\sum_{k,l} \hat{a}_{jk} \hat{a}_{jl}$
	$a[a[\tau_a]]$	3	1	$\frac{\partial f_a}{\partial y_a} \frac{\partial f_a}{\partial y_a} f_a$	$\sum_{k,l} a_{jk} a_{kl}$
	$a[a[\tau_b]]$	3	1	$\frac{\partial f_a}{\partial y_a} \frac{\partial f_a}{\partial y_b} f_b$	$\sum_{k,l} a_{jk} \hat{a}_{kl}$
	$a[b[\tau_a]]$	3	1	$\frac{\partial f_a}{\partial y_b} \frac{\partial f_b}{\partial y_a} f_a$	$\sum_{k,l} \hat{a}_{jk} a_{kl}$
	$a[b[\tau_b]]$	3	1	$\frac{\partial f_a}{\partial y_b} \frac{\partial f_b}{\partial y_b} f_b$	$\sum_{k,l} \hat{a}_{jk} \hat{a}_{kl}$
...
	τ_b	1	1	f_b	1
	$b[\tau_a]$	2	1	$\frac{\partial f_b}{\partial y_a} f_a$	$\sum_k a_{jk}$
	$b[\tau_b]$	2	1	$\frac{\partial f_b}{\partial y_b} f_b$	$\sum_k \hat{a}_{jk}$
...

P-Series

In Section II.12 we saw the importance of the key-lemma Corollary 12.7 for the derivation of the order conditions for Runge-Kutta methods. Therefore we extend this result also to partitioned ordinary differential equations.

It is convenient to introduce two new P-trees of order 0, namely \emptyset_a and \emptyset_b . The corresponding elementary differentials are $F(\emptyset_a)(y) = y_a$ and $F(\emptyset_b)(y) = y_b$. We further set

$$\begin{aligned}
 TP^a &= \{\emptyset_a\} \cup TP_1^a \cup TP_2^a \cup \dots & LTP^a &= \{\emptyset_a\} \cup LTP_1^a \cup LTP_2^a \cup \dots \\
 TP^b &= \{\emptyset_b\} \cup TP_1^b \cup TP_2^b \cup \dots & LTP^b &= \{\emptyset_b\} \cup LTP_1^b \cup LTP_2^b \cup \dots
 \end{aligned}
 \tag{15.7}$$

Definition 15.7. Let $\mathbf{c}(\emptyset_a)$, $\mathbf{c}(\emptyset_b)$, $\mathbf{c}(\tau_a)$, $\mathbf{c}(\tau_b)$, ... be real coefficients defined for all P-trees, i.e., $\mathbf{c} : T P^a \cup T P^b \rightarrow \mathbb{R}$. The series

$$P(\mathbf{c}, y) = (P_a(\mathbf{c}, y), P_b(\mathbf{c}, y))^T$$

where

$$P_a(\mathbf{c}, y) = \sum_{t \in LTP^a} \frac{h^{\varrho(t)}}{\varrho(t)!} \mathbf{c}(t) F(t)(y), \quad P_b(\mathbf{c}, y) = \sum_{t \in LTP^b} \frac{h^{\varrho(t)}}{\varrho(t)!} \mathbf{c}(t) F(t)(y)$$

is then called a *P-series*.

Theorem 15.6 simply states that the exact solution of (15.2) is a P-series

$$(y_a(x_0 + h), y_b(x_0 + h))^T = P(\mathbf{y}, (y_a(x_0), y_b(x_0))) \quad (15.8)$$

with $\mathbf{y}(t) = 1$ for all P-trees t .

Theorem 15.8. Let $\mathbf{c} : T P^a \cup T P^b \rightarrow \mathbb{R}$ be a sequence of coefficients such that $\mathbf{c}(\emptyset_a) = \mathbf{c}(\emptyset_b) = 1$. Then

$$h \begin{pmatrix} f_a(P(\mathbf{c}, (y_a, y_b))) \\ f_b(P(\mathbf{c}, (y_a, y_b))) \end{pmatrix} = P(\mathbf{c}', (y_a, y_b)) \quad (15.9)$$

with

$$\begin{aligned} \mathbf{c}'(\emptyset_a) = \mathbf{c}'(\emptyset_b) &= 0, & \mathbf{c}'(\tau_a) = \mathbf{c}'(\tau_b) &= 1 \\ \mathbf{c}'(t) = \varrho(t) \mathbf{c}(t_1) \dots \mathbf{c}(t_m) & \quad \text{if } t = {}_a[t_1, \dots, t_m] \text{ or } t = {}_b[t_1, \dots, t_m]. \end{aligned} \quad (15.10)$$

The *proof* is related to that of Theorem 12.6. It is given with more details in Hairer (1981). \square

Order Conditions for Partitioned Runge-Kutta Methods

With the help of Theorem 15.8 the order conditions for method (15.3) can readily be obtained. For this we denote the arguments in (15.3) by

$$g_i = y_{a0} + h \sum_{j=1}^s a_{ij} k_j, \quad \widehat{g}_i = y_{b0} + h \sum_{j=1}^s \widehat{a}_{ij} \ell_j, \quad (15.11)$$

and we assume that $G_i = (g_i, \widehat{g}_i)^T$ and $K_i = h(k_i, \ell_i)^T$ are P-series with coefficients $\mathbf{G}_i(t)$ and $\mathbf{K}_i(t)$, respectively. The formulas (15.11) then yield $\mathbf{G}_i(\emptyset_a) = 1$, $\mathbf{G}_i(\emptyset_b) = 1$ and

$$\mathbf{G}_i(t) = \begin{cases} \sum_{j=1}^s a_{ij} \mathbf{K}_j(t) & \text{if the root of } t \text{ is meagre,} \\ \sum_{j=1}^s \widehat{a}_{ij} \mathbf{K}_j(t) & \text{if the root of } t \text{ is fat.} \end{cases} \quad (15.12)$$

Application of Theorem 15.8 to the relations $k_j = f_a(G_j)$, $\ell_j = f_b(G_j)$ shows that $\mathbf{K}_j(t) = \mathbf{G}'_j(t)$ which, together with (15.10) and (15.12), recursively defines the values $\mathbf{K}_j(t)$.

It is usual to write $\mathbf{K}_j(t) = \gamma(t)\Phi_j(t)$ where $\gamma(t)$ is the integer given in Definition 2.10 (see also (2.17)). The coefficient $\Phi_j(t)$ is then obtained in the same way as the corresponding value of standard Runge-Kutta methods (see Definition 2.9) with the exception that a factor a_{ik} has to be replaced by \hat{a}_{ik} , if the vertex with label “ k ” is fat. A comparison of the P-series for the numerical solution $(y_{1a}, y_{1b})^T$ with that for the exact solution (15.8) yields the desired order conditions.

Theorem 15.9. *A partitioned Runge-Kutta method (15.3) is of order p iff*

$$\sum_{j=1}^s b_j \Phi_j(t) = \frac{1}{\gamma(t)} \quad \text{and} \quad \sum_{j=1}^s \hat{b}_j \Phi_j(t) = \frac{1}{\gamma(t)} \quad (15.13)$$

for all P-trees of order $\leq p$. □

Example. A partitioned method (15.3) is of order 2, if and only if each of the two Runge-Kutta schemes has order 2 and if the coupling conditions

$$\sum_{i,j} b_i \hat{a}_{ij} = \frac{1}{2}, \quad \sum_{i,j} \hat{b}_i a_{ij} = \frac{1}{2},$$

which correspond to trees ${}_a[\tau_b]$ and ${}_b[\tau_a]$ of Table 15.1 respectively, are satisfied. This happens if

$$c_i = \hat{c}_i \quad \text{for all } i.$$

This last assumption simplifies the order conditions considerably (the “thickness” of terminating vertices then has no influence). The resulting conditions for order up to 4 have been tabulated by Griepentrog (1978).

Further Applications of P-Series

Runge-Kutta methods violating (1.9). For the non-autonomous differential equation $y' = f(x, y)$ we consider, as in Exercise 6 of Section II.1, the Runge-Kutta method

$$k_i = f\left(x_0 + \hat{c}_i h, y_0 + h \sum_{j=1}^s a_{ij} k_j\right), \quad y_1 = y_0 + h \sum_{i=1}^s b_i k_i, \quad (15.14)$$

where \widehat{c}_i is not necessarily equal to $c_i = \sum_j a_{ij}$. Therefore, the x and y components in

$$\begin{aligned} y' &= f(x, y) \\ x' &= 1. \end{aligned} \quad (15.15)$$

are integrated differently. This system is of the form (15.2), if we put $y_a = y$, $y_b = x$, $f_a(y_a, y_b) = f(x, y)$ and $f_b(y_a, y_b) = 1$. Since f_b is constant, all elementary differentials that involve derivatives of f_b vanish identically. Thus, P-trees where at least one fat vertex is not an end-vertex need not be considered. It remains to treat the set

$$T_x = \{t \in TP_a; \text{ all fat vertices are end-vertices}\}. \quad (15.16)$$

Each tree of T_x gives rise to an order condition which is exactly that of Theorem 15.9. It is obtained in the usual way (Section II.2) with the exception that c_k has to be replaced by \widehat{c}_k , if the corresponding vertex is a fat one.

Fehlberg methods. The methods of Fehlberg, introduced in Section II.13, are equivalent to (15.14). However, it is known that the exact solution of the differential equation $y' = f(x, y)$ satisfies $y(x_0) = 0$, $y'(x_0) = 0, \dots, y^{(m)}(x_0) = 0$ at the initial value $x = x_0$. As explained in II.13, this implies that the expressions $f, \partial f / \partial x, \dots, \partial^{m-1} f / \partial x^{m-1}$ vanish at (x_0, y_0) and consequently also many of the elementary differentials disappear. The elements of T_x which remain to be considered are given in Fig. 15.3.

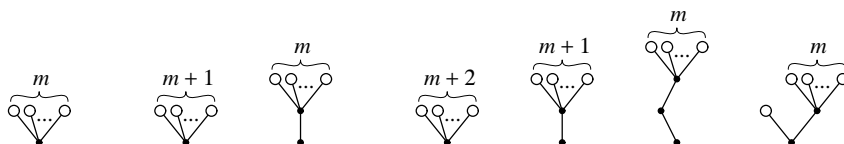


Fig. 15.3. P-trees for the methods of Fehlberg

Nyström methods. As a last application of Theorem 15.8 we present a new derivation of the order conditions for Nyström methods (Section II.14). The second order differential equation $y'' = f(y, y')$ can be written in partitioned form as

$$\begin{pmatrix} y \\ y' \end{pmatrix}' = \begin{pmatrix} y' \\ f(y, y') \end{pmatrix}. \quad (15.17)$$

In the notation of (15.2) we have $y_a = y$, $y_b = y'$, $f_a(y_a, y_b) = y_b$, $f_b(y_a, y_b) = f(y_a, y_b)$. The special structure of f_a implies that only P-trees which satisfy the condition (see Definition 14.2)

$$\text{“meagre vertices have at most one son and this son has to be fat”} \quad (15.18)$$

have to be considered. The essential P-trees are thus

$$TN_q^a = \{t \in TP_q^a ; t \text{ satisfies (15.18)}\}$$

$$TN_q^b = \{t \in TP_q^b ; t \text{ satisfies (15.18)}\}.$$

It follows that each element of TN_{q+1}^a can be written as $t = {}_a[u]$ with $u \in TN_q^b$. This implies a one-to-one correspondence between TN_{q+1}^a and TN_q^b , leaving the elementary differentials invariant:

$$F({}_a[u])(y_a, y_b) = \frac{\partial y_b}{\partial y_b} \cdot F(u)(y_a, y_b) = F(u)(y_a, y_b).$$

From this property it follows that

$$hP_b(\mathbf{c}, (y_a, y_b)) = P_a(\mathbf{c}', (y_a, y_b)) \quad (15.19)$$

where $\mathbf{c}'(\emptyset_a) = 0$, $\mathbf{c}'(\tau_a) = \mathbf{c}(\emptyset_b)$ and

$$\mathbf{c}'(t) = \varrho(t)\mathbf{c}(u) \quad \text{if } t = {}_a[u]. \quad (15.20)$$

This notation is in agreement with (15.10).

The order conditions of method (14.13) can now be derived as follows: assume g_i, g'_i to be P-series

$$g_i = P_a(\mathbf{c}_i, (y_0, y'_0)), \quad g'_i = P_b(\mathbf{c}_i, (y_0, y'_0)).$$

Theorem 15.8 then implies that

$$hf(g_i, g'_i) = P_b(\mathbf{c}'_i, (y_0, y'_0)). \quad (15.21)$$

Multiplying this relation by h it follows from (15.19) that

$$h^2 f(g_i, g'_i) = P_a(\mathbf{c}''_i, (y_0, y'_0)). \quad (15.22)$$

Here $\mathbf{c}''_i = (\mathbf{c}'_i)'$, i.e.,

$$\begin{aligned} \mathbf{c}''_i(t) &= 0 \quad \text{for } t = \emptyset_a \text{ and } t = \tau_a, & \mathbf{c}''_i({}_a[\tau_b]) &= 1, \\ \mathbf{c}''_i(t) &= \varrho(t)(\varrho(t) - 1)\mathbf{c}_i(t_1) \dots \mathbf{c}_i(t_m) & \text{if } t = {}_a[b[t_1, \dots, t_m]]. \end{aligned}$$

The relations (15.21) and (15.22), when inserted into (14.13), yield

$$\begin{aligned} \mathbf{c}_i(\tau_a) &= c_i, \\ \mathbf{c}_i(t) &= \begin{cases} \sum_j \bar{a}_{ij} \mathbf{c}''_j(t) & \text{if the root of } t \text{ is meagre,} \\ \sum_j a_{ij} \mathbf{c}'_j(t) & \text{if the root of } t \text{ is fat.} \end{cases} \end{aligned}$$

Finally, a comparison of the P-series for the exact and numerical solutions gives the order conditions (for order p)

$$\begin{aligned} \sum_i \bar{b}_i \mathbf{c}''_i(t) &= 1 \quad \text{for } t \in TN_q^a, \quad q = 2, \dots, p \\ \sum_i b_i \mathbf{c}'_i(t) &= 1 \quad \text{for } t \in TN_q^b, \quad q = 1, \dots, p. \end{aligned} \quad (15.23)$$

Exercises

1. Denote the number of elements of TP_q^a (P-trees with meagre root of order q) by α_q (see Table 15.2). Prove that

$$\alpha_1 + \alpha_2 x + \alpha_3 x^2 + \dots = (1-x)^{-2\alpha_1} (1-x^2)^{-2\alpha_2} (1-x^3)^{-2\alpha_3} \dots$$

Compute the first α_q and compare them with the a_q of Table 2.1.

Table 15.2. Number of elements of TP_q^a

q	1	2	3	4	5	6	7	8	9	10
α_q	1	2	7	26	107	458	2058	9498	44947	216598

2. There is no explicit, 4-stage Runge-Kutta method of order 4, which does not satisfy condition (1.9).

Hint. Use the techniques of the proof of Lemma 1.4.

3. Show that the order conditions (15.23) are the same as those given in Theorem 14.10.

4. Show that the partitioned method of Griepentrog (1978)

0	a_{ij}			0	0			\hat{a}_{ij}
1/2	1/2			1/2	$-\beta/2$	$(1+\beta)/2$		
1	-1	2		1	$(3+5\beta)/2$	$-(1+3\beta)$	$(1+\beta)/2$	
	1/6	2/3	1/6		1/6	2/3	1/6	

with $\beta = \sqrt{3}/3$ is of order 3 (the implicit method to the right is A -stable and is provided for the stiff part of the problem).

II.16 Symplectic Integration Methods

It is natural to look forward to those discrete systems which preserve as much as possible the intrinsic properties of the continuous system. (Feng Kang 1985)

Y.V. Rakitskii proposed . . . a requirement of the most complete conformity between two dynamical systems: one resulting from the original differential equations and the other resulting from the difference equations of the computational method. (Y.B. Suris 1989)

Hamiltonian systems, given by

$$\dot{p}_i = -\frac{\partial H}{\partial q_i}(p, q), \quad \dot{q}_i = \frac{\partial H}{\partial p_i}(p, q), \quad (16.1)$$

have been seen to possess two remarkable properties:

- a) the solutions preserve the Hamiltonian $H(p, q)$ (Ex. 5 of Section I.6);
- b) the corresponding flow is symplectic, i.e., preserves the differential 2-form

$$\omega^2 = \sum_{i=1}^n dp_i \wedge dq_i \quad (16.2)$$

(see Theorem I.14.12). In particular, the flow is volume preserving.

Both properties are usually destroyed by a numerical method applied to (16.1).

After some pioneering papers (de Vogelaere 1956, Ruth 1983, and Feng Kang (冯康 1985) an enormous avalanche of research started around 1988 on the characterization of existing numerical methods which preserve symplecticity or on the construction of new classes of symplectic methods. An excellent overview is presented by Sanz-Serna (1992).

Example 16.1. We consider the harmonic oscillator

$$H(p, q) = \frac{1}{2} (p^2 + k^2 q^2). \quad (16.3)$$

Here (16.1) becomes

$$\dot{p} = -k^2 q, \quad \dot{q} = p \quad (16.4)$$

and we study the action of several steps of a numerical method on a well-known set of initial data (p_0, q_0) (see Fig. 16.1):

- a) The explicit Euler method (I.7.3)

$$\begin{pmatrix} p_m \\ q_m \end{pmatrix} = \begin{pmatrix} 1 & -hk^2 \\ h & 1 \end{pmatrix} \begin{pmatrix} p_{m-1} \\ q_{m-1} \end{pmatrix}, \quad h = \frac{\pi}{8k}, \quad m = 1, \dots, 16; \quad (16.5a)$$

b) the implicit (or backward) Euler method (7.3)

$$\begin{pmatrix} p_m \\ q_m \end{pmatrix} = \frac{1}{1 + h^2 k^2} \begin{pmatrix} 1 & -hk^2 \\ h & 1 \end{pmatrix} \begin{pmatrix} p_{m-1} \\ q_{m-1} \end{pmatrix}, \quad h = \frac{\pi}{8k}, \quad m = 1, \dots, 16; \quad (16.5b)$$

c) Runge's method (1.4) of order 2

$$\begin{pmatrix} p_m \\ q_m \end{pmatrix} = \begin{pmatrix} 1 - \frac{h^2 k^2}{2} & -hk^2 \\ h & 1 - \frac{h^2 k^2}{2} \end{pmatrix} \begin{pmatrix} p_{m-1} \\ q_{m-1} \end{pmatrix}, \quad h = \frac{\pi}{4k}, \quad m = 1, \dots, 8; \quad (16.5c)$$

d) the implicit midpoint rule (7.4) of order 2

$$\begin{pmatrix} p_m \\ q_m \end{pmatrix} = \frac{1}{1 + \frac{h^2 k^2}{4}} \begin{pmatrix} 1 - \frac{h^2 k^2}{4} & -hk^2 \\ h & 1 - \frac{h^2 k^2}{4} \end{pmatrix} \begin{pmatrix} p_{m-1} \\ q_{m-1} \end{pmatrix}, \quad h = \frac{\pi}{4k}, \quad m = 1, \dots, 8. \quad (16.5d)$$

For the exact flow, the last of all these cats would precisely coincide with the first one and all cats would have the same area. Only the last method appears to be area preserving. It also preserves the Hamiltonian in this example.

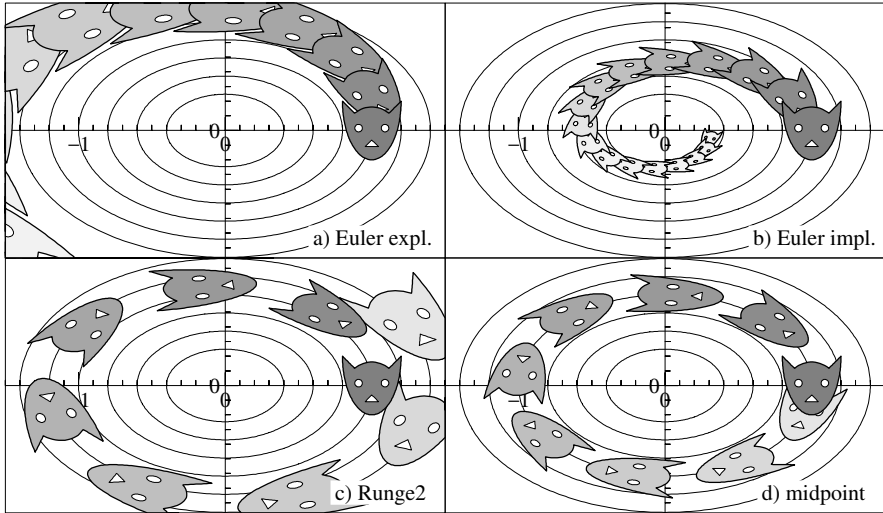


Fig. 16.1. Destruction of symplecticity of a Hamiltonian flow, $k = (\sqrt{5} + 1)/2$

Example 16.2. For a nonlinear problem we choose

$$H(p, q) = \frac{p^2}{2} - \cos(q) \left(1 - \frac{p}{6}\right) \quad (16.6)$$

which is similar to the Hamiltonian of the pendulum (I.14.25), but with some of the pendulum's symmetry destroyed. Fig. 16.2 presents 12000 consecutive solution values (p_i, q_i) for

- a) Runge's method of order 2 (see (1.4));
- b) the implicit Radau method with $s = 2$ and order 3 (see Exercise 6 of Section II.7);
- c) the implicit midpoint rule (7.4) of order 2.

The initial values are

$$p_0 = 0, \quad q_0 = \begin{cases} \arccos(0.5) = \pi/3 & \text{for case (a)} \\ \arccos(-0.8) & \text{for cases (b) and (c).} \end{cases}$$

The computation is done with fixed step sizes

$$h = \begin{cases} 0.15 & \text{for case (a)} \\ 0.3 & \text{for cases (b) and (c).} \end{cases}$$

The solution of method (a) spirals out, that of method (b) spirals in and both by no means preserve the Hamiltonian. Method (c) behaves differently. Although the Hamiltonian is not precisely preserved (see picture (d)), its error remains bounded for long-scale computations.

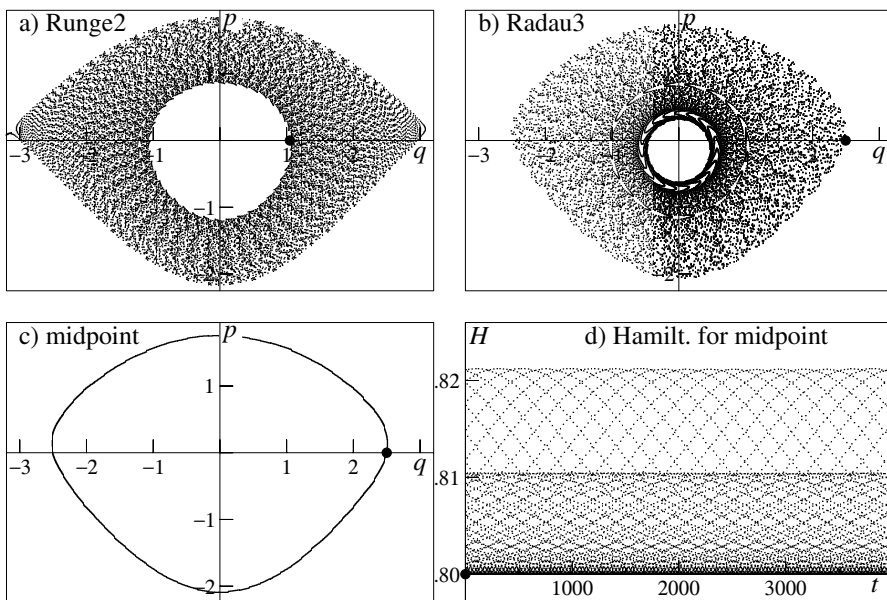


Fig. 16.2. A nonlinear pendulum and behaviour of H
 (• ... indicates the initial position)

Symplectic Runge-Kutta Methods

For a given Hamiltonian system (16.1), for a chosen one-step method (in particular a Runge-Kutta method) and a chosen step size h we denote by

$$\begin{aligned} \psi_h : \mathbb{R}^{2n} &\longrightarrow \mathbb{R}^{2n} \\ (p_0, q_0) &\longmapsto (p_1, q_1) \end{aligned} \quad (16.7)$$

the transformation defined by the method.

Remark. For implicit methods the numerical solution (p_1, q_1) need not exist for all h and all initial values (p_0, q_0) nor need it be uniquely determined (see Exercise 2). Therefore we usually will have to restrict the domain where ψ_h is defined and we will have to select a solution of the nonlinear system such that ψ_h is differentiable on this domain. The subsequent results hold for all possible choices of ψ_h .

Definition 16.4. A one-step method is called *symplectic* if for every smooth Hamiltonian H and for every step size h the mapping ψ_h is symplectic (see Definition I.14.11), i.e., preserves the differential 2-form ω^2 of (16.2).

We start with the easiest result.

Theorem 16.5. *The implicit s -stage Gauss methods of order $2s$ (Kuntzmann & Butcher methods of Section II.7) are symplectic for all s .*

Proof. We simplify the notation by putting $h = 1$ and $t_0 = 0$ and use the fact that the methods under consideration are collocation methods, i.e., the numerical solution after one step is defined by $(u(1), v(1))$ where $(u(t), v(t))$ are polynomials of degree s such that

$$\begin{aligned} u(0) &= p_0, & u'(c_i) &= -\frac{\partial H}{\partial q}(u(c_i), v(c_i)) \\ v(0) &= q_0, & v'(c_i) &= \frac{\partial H}{\partial p}(u(c_i), v(c_i)) \end{aligned} \quad i = 1, \dots, s. \quad (16.8)$$

The polynomials $u(t)$ and $v(t)$ are now considered as functions of the initial values. For arbitrary variations ξ_1^0 and ξ_2^0 of the initial point we denote the corresponding variations of u and v as

$$\xi_1^t = \frac{\partial(u(t), v(t))}{\partial(p_0, q_0)} \cdot \xi_1^0, \quad \xi_2^t = \frac{\partial(u(t), v(t))}{\partial(p_0, q_0)} \cdot \xi_2^0.$$

Symplecticity of the method means that the expression

$$\omega^2(\xi_1^1, \xi_2^1) - \omega^2(\xi_1^0, \xi_2^0) = \int_0^1 \frac{d}{dt} \omega^2(\xi_1^t, \xi_2^t) dt \quad (16.9)$$

should vanish. Since ξ_1^t and ξ_2^t are polynomials in t of degree s , the expression $\frac{d}{dt} \omega^2(\xi_1^t, \xi_2^t)$ is a polynomial of degree $2s - 1$. We can thus exactly integrate (16.9)

by the Gaussian quadrature formula and so obtain

$$\omega^2(\xi_1^1, \xi_2^1) - \omega^2(\xi_1^0, \xi_2^0) = \sum_{i=1}^s b_i \frac{d}{dt} \omega^2(\xi_1^t, \xi_2^t) \Big|_{t=c_i}. \quad (16.9')$$

Differentiation of (16.8) with respect to (p_0, q_0) shows that (ξ_1^t, ξ_2^t) satisfies the variational equation (I.14.27) at the collocation points $t = c_i$, $i = 1, \dots, s$. Therefore, the computations of the proof of Theorem I.14.12 imply that

$$\frac{d}{dt} \omega^2(\xi_1^t, \xi_2^t) \Big|_{t=c_i} = 0 \quad \text{for } i = 1, \dots, s. \quad (16.10)$$

This, introduced into (16.9'), completes the proof of symplecticity. \square

The following theorem, discovered independently by at least three authors (F. Lasagni 1988, J.M. Sanz-Serna 1988, Y.B. Suris 1989) characterizes the class of all symplectic Runge-Kutta methods:

Theorem 16.6. *If the $s \times s$ matrix M with elements*

$$m_{ij} = b_i a_{ij} + b_j a_{ji} - b_i b_j, \quad i, j = 1, \dots, s \quad (16.11)$$

satisfies $M = 0$, then the Runge-Kutta method (7.7) is symplectic.

Proof. The matrix M has been known from nonlinear stability theory for many years (see Theorem IV.12.4). Both theorems have very similar proofs, the one works with the *inner* product, the other with the *exterior* product.

We write method (7.7) applied to problem (16.1) as

$$P_i = p_0 + h \sum_j a_{ij} k_j \quad Q_i = q_0 + h \sum_j a_{ij} \ell_j \quad (16.12a)$$

$$p_1 = p_0 + h \sum_i b_i k_i \quad q_1 = q_0 + h \sum_i b_i \ell_i \quad (16.12b)$$

$$k_i = -\frac{\partial H}{\partial q}(P_i, Q_i) \quad \ell_i = \frac{\partial H}{\partial p}(P_i, Q_i), \quad (16.12c)$$

denote the J th component of a vector by an upper index J and introduce the linear maps (one-forms)

$$\begin{aligned} dp_1^J : \mathbb{R}^{2n} &\rightarrow \mathbb{R}, & dP_i^J : \mathbb{R}^{2n} &\rightarrow \mathbb{R}, \\ \xi &\mapsto \frac{\partial p_1^J}{\partial(p_0, q_0)} \xi & \xi &\mapsto \frac{\partial P_i^J}{\partial(p_0, q_0)} \xi \end{aligned} \quad (16.13)$$

and similarly also dp_0^J , dk_i^J , dq_0^J , dq_1^J , dQ_i^J , $d\ell_i^J$ (the one-forms dp_0^J and dq_0^J correspond to dp_J and dq_J of Section I.14). Using the notation (16.13),

symplecticity of the method is equivalent to

$$\sum_{J=1}^n dp_1^J \wedge dq_1^J = \sum_{J=1}^n dp_0^J \wedge dq_0^J. \quad (16.14)$$

To check this relation we differentiate (16.12) with respect to the initial values and obtain

$$dP_i^J = dp_0^J + h \sum_j a_{ij} dk_j^J \quad dQ_i^J = dq_0^J + h \sum_j a_{ij} d\ell_j^J \quad (16.15a)$$

$$dp_1^J = dp_0^J + h \sum_i b_i dk_i^J \quad dq_1^J = dq_0^J + h \sum_i b_i d\ell_i^J \quad (16.15b)$$

$$dk_i^J = - \sum_{L=1}^n \frac{\partial^2 H}{\partial q^J \partial p^L}(P_i, Q_i) \cdot dP_i^L - \sum_{L=1}^n \frac{\partial^2 H}{\partial q^J \partial q^L}(P_i, Q_i) \cdot dQ_i^L \quad (16.15c)$$

$$d\ell_i^J = \sum_{L=1}^n \frac{\partial^2 H}{\partial p^J \partial p^L}(P_i, Q_i) \cdot dP_i^L + \sum_{L=1}^n \frac{\partial^2 H}{\partial p^J \partial q^L}(P_i, Q_i) \cdot dQ_i^L. \quad (16.15d)$$

We now compute

$$\begin{aligned} dp_1^J \wedge dq_1^J - dp_0^J \wedge dq_0^J &= h \sum_i b_i dp_0^J \wedge d\ell_i^J + h \sum_i b_i dk_i^J \wedge dq_0^J + h^2 \sum_{i,j} b_i b_j dk_i^J \wedge d\ell_j^J \end{aligned} \quad (16.16)$$

by using (16.15b) and the multilinearity of the wedge product. This formula corresponds precisely to (IV.12.6). Exactly as in the proof of Theorem IV.12.5, we now eliminate in (16.16) the quantities dp_0^J and dq_0^J with the help of (16.15a) to obtain

$$\begin{aligned} dp_1^J \wedge dq_1^J - dp_0^J \wedge dq_0^J &= h \sum_i b_i dP_i^J \wedge d\ell_i^J + h \sum_i b_i dk_i^J \wedge dQ_i^J - h^2 \sum_{i,j} m_{ij} dk_i^J \wedge d\ell_j^J, \end{aligned} \quad (16.17)$$

the formula analogous to (IV.12.7). Equations (16.15c,d) are perfect analogues of the variational equation (I.14.27). Therefore the same computations as in (I.14.39) give

$$\sum_{J=1}^n dP_i^J \wedge d\ell_i^J + \sum_{J=1}^n dk_i^J \wedge dQ_i^J = 0 \quad (16.18)$$

and the first two terms in (16.17) disappear. The last term vanishes by hypothesis (16.11) and we obtain (16.14). \square

Remark. F. Lasagni (1990) has proved in an unpublished manuscript that for *irreducible* methods (see Definitions IV.12.15 and IV.12.17) the condition $M = 0$ is also *necessary* for symplecticity. For a publication see Abia & Sanz-Serna (1993, Theorem 5.1), where this proof has been elaborated and adapted to a more general setting.

Remarks. a) Explicit Runge-Kutta methods are never symplectic (Ex. 1).

b) Equations (16.11) imply a substantial simplification of the order conditions (Sanz-Serna & Abia 1991). We shall return to this when treating partitioned methods (see (16.40)).

c) An important tool for the construction of symplectic methods is the W -transformation (see Section IV.5, especially Theorem IV.5.6). As can be seen from formula (IV.12.10), the method under consideration is symplectic if and only if the matrix X is skew-symmetric (with the exception of $x_{11} = 1/2$). Sun Geng (孙耿 1992) constructed several new classes of symplectic Runge-Kutta methods. One of his methods, based on Radau quadrature, is given in Table 16.1.

d) An inspection of Table IV.5.14 shows that all Radau IA, Radau IIA, Lobatto IIIA (in particular the trapezoidal rule), and Lobatto IIIC methods are not symplectic.

Table 16.1. Sun's symplectic Radau method of order 5

$\frac{4 - \sqrt{6}}{10}$	$\frac{16 - \sqrt{6}}{72}$	$\frac{328 - 167\sqrt{6}}{1800}$	$\frac{-2 + 3\sqrt{6}}{450}$
$\frac{4 + \sqrt{6}}{10}$	$\frac{328 + 167\sqrt{6}}{1800}$	$\frac{16 + \sqrt{6}}{72}$	$\frac{-2 - 3\sqrt{6}}{450}$
1	$\frac{85 - 10\sqrt{6}}{180}$	$\frac{85 + 10\sqrt{6}}{180}$	$\frac{1}{18}$
	$\frac{16 - \sqrt{6}}{36}$	$\frac{16 + \sqrt{6}}{36}$	$\frac{1}{9}$

Preservation of the Hamiltonian and of first integrals. In Exercise 5 of Section I.6 we have seen that the Hamiltonian $H(p, q)$ is a *first integral* of the system (16.1). This means that every solution $p(t), q(t)$ of (16.1) satisfies $H(p(t), q(t)) = \text{Const}$. The numerical solution of a symplectic integrator does not share this property in general (see Fig. 16.2). However, we will show that every *quadratic* first integral will be preserved.

Denote $y = (p, q)$ and let G be a symmetric $2n \times 2n$ matrix. We suppose that the quadratic functional

$$\langle y, y \rangle_G := y^T G y$$

is a first integral of the system (16.1). This means that

$$\langle y, J^{-1} \text{grad } H(y) \rangle_G = 0 \quad \text{with} \quad J = \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix} \quad (16.19)$$

for all $y \in \mathbb{R}^{2n}$.

Theorem 16.7 (Sanz-Serna 1988). *A symplectic Runge-Kutta method (i.e., a method satisfying (16.11)) leaves all quadratic first integrals of the system (16.1) invariant, i.e., the numerical solution $y_n = (p_n, q_n)$ satisfies*

$$\langle y_1, y_1 \rangle_G = \langle y_0, y_0 \rangle_G \quad (16.20)$$

for all symmetric matrices G satisfying (16.19).

Proof (Cooper 1987). The Runge-Kutta method (7.7) applied to problem (16.1) is given by

$$\begin{aligned} y_1 &= y_0 + \sum_i b_i k_i, & Y_i &= y_0 + \sum_j a_{ij} k_j, \\ k_i &= J^{-1} \text{grad } H(Y_i). \end{aligned} \quad (16.21)$$

As in the proof of Theorem 16.6 (see also Theorem IV.12.4) we obtain

$$\langle y_1, y_1 \rangle_G - \langle y_0, y_0 \rangle_G = 2h \sum_i b_i \langle Y_i, k_i \rangle_G - h^2 \sum_{i,j} m_{ij} \langle k_i, k_j \rangle_G.$$

The first term on the right-hand side vanishes by (16.19) and the second one by (16.11). \square

An Example from Galactic Dynamics

Always majestic, usually spectacularly beautiful, galaxies
are ... (Binney & Tremaine 1987)

While the theoretical meaning of symplecticity of numerical methods is clear, its importance for practical computations is less easy to understand. Numerous numerical experiments have shown that symplectic methods, in a fixed step size mode, show an excellent behaviour for long-scale scientific computations of Hamiltonian systems. We shall demonstrate this on the following example chosen from galactic dynamics and give a theoretical justification later in this section. However, Calvo & Sanz-Serna (1992c) have made the interesting discovery that *variable step size* implementation can *destroy* the advantages of symplectic methods. In order to illustrate this phenomenon we shall include in our computations violent step changes; one with a random number generator and one with the step size changing in function of the solution position.

A galaxy is a set of N stars which are mutually attracted by Newton's law. A relatively easy way to study them is to perform a long-scale computation of the orbit of *one* of its stars in the potential formed by the $N - 1$ remaining ones (see Binney & Tremaine 1987, Chapter 3); this potential is assumed to perform a uniform rotation with time, but not to change otherwise. The potential is determined

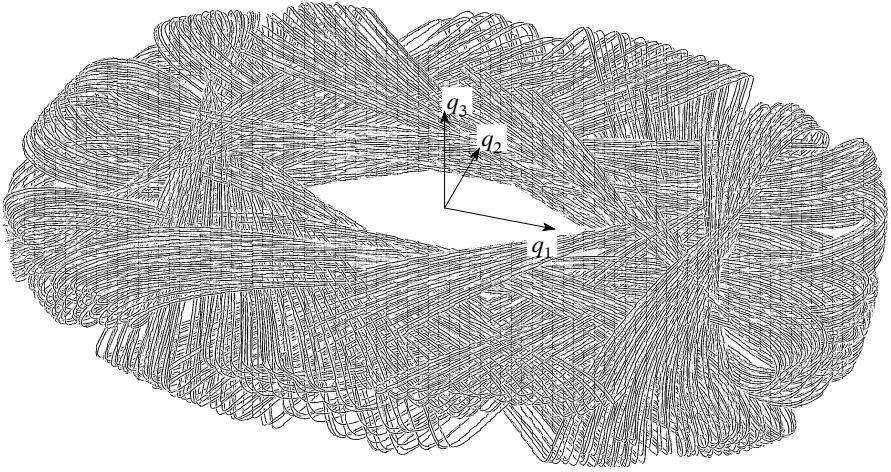


Fig. 16.3. Galactic orbit

by Poisson's differential equation $\Delta V = 4G\pi\rho$, where ρ is the density distribution of the galaxy, and real-life potential-density pairs are difficult to obtain (e.g., de Zeeuw & Pfenniger 1988). A popular issue is to choose a simple formula for V in such a way that the resulting ρ corresponds to a reasonable galaxy, for example (Binney 1981, Binney & Tremaine 1987, p. 45f, Pfenniger 1990)

$$V = A \ln \left(C + \frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} \right). \quad (16.22)$$

The Lagrangian for a coordinate system rotating with angular velocity Ω becomes

$$\mathcal{L} = \frac{1}{2} \left((\dot{x} - \Omega y)^2 + (\dot{y} + \Omega x)^2 + \dot{z}^2 \right) - V(x, y, z). \quad (16.23)$$

This gives with the coordinates (see (I.6.23))

$$\begin{aligned} p_1 &= \frac{\partial \mathcal{L}}{\partial \dot{x}} = \dot{x} - \Omega y, & p_2 &= \frac{\partial \mathcal{L}}{\partial \dot{y}} = \dot{y} + \Omega x, & p_3 &= \frac{\partial \mathcal{L}}{\partial \dot{z}} = \dot{z}, \\ q_1 &= x, & q_2 &= y, & q_3 &= z, \end{aligned}$$

the Hamiltonian

$$\begin{aligned} H &= p_1 \dot{q}_1 + p_2 \dot{q}_2 + p_3 \dot{q}_3 - \mathcal{L} \\ &= \frac{1}{2} (p_1^2 + p_2^2 + p_3^2) + \Omega (p_1 q_2 - p_2 q_1) + A \ln \left(C + \frac{q_1^2}{a^2} + \frac{q_2^2}{b^2} + \frac{q_3^2}{c^2} \right). \end{aligned} \quad (16.24)$$

We choose the parameters and initial values as

$$\begin{aligned} a &= 1.25, \quad b = 1, \quad c = 0.75, \quad A = 1, \quad C = 1, \quad \Omega = 0.25, \\ q_1(0) &= 2.5, \quad q_2(0) = 0, \quad q_3(0) = 0, \quad p_1(0) = 0, \quad p_3(0) = 0.2, \end{aligned} \quad (16.25)$$

and take for $p_2(0)$ the larger of the roots for which $H = 2$. Our star then sets out for its voyage through the galaxy, the orbit is represented in Fig. 16.3 for $0 \leq t \leq 15000$. We are interested in its Poincaré sections with the half-plane $q_2 = 0$, $q_1 > 0$, $\dot{q}_2 > 0$ for $0 \leq t \leq 1000000$. These consist, for the exact solution, in 47101 cut points which are presented in Fig. 16.6l. These points were computed with the (non-symplectic) code DOP853 with $Tol = 10^{-17}$ in quadruple precision on a VAX 8700 computer.

Fig. 16.4, Fig. 16.5, and Fig. 16.6 present the obtained numerical results for the methods and step sizes summarized in Table 16.2.

Table 16.2. Methods for numerical experiments

item	method	order	h	points $t \leq 1000000$	impl.	symplec.	symmet.
a)	Gauss	6	1/5	47093	yes	yes	yes
b)	"	"	2/5	46852	"	"	"
c)	Gauss	6	random	46717	yes	yes	yes
d)	Gauss	6	partially halved	46576	yes	yes	yes
e)	Radau	5	1/10	46597	yes	no	no
f)	"	"	1/5	46266	"	"	"
g)	RK44	4	1/40	47004	no	no	no
h)	"	"	1/10	46192	"	"	"
i)	Lobatto	6	1/5	47091	yes	no	yes
j)	"	"	2/5	46839	"	"	"
k)	Sun Geng	5	1/5	47092	yes	yes	no
l)	exact	—	—	47101	—	—	—

Remarks.

- ad a): the Gauss6 method (Kuntzmann & Butcher method based on Gaussian quadrature with $s = 3$ and $p = 6$, see Table 7.4) for $h = 1/5$ is nearly identical to the exact solution;
- ad b): Gauss6 for $h = 2/5$ is much better than Gauss6 with random or partially halved step sizes (see item (c) and (d)) where $h \leq 2/5$.
- ad c): h was chosen at random uniformly distributed on $(0, 2/5)$;
- ad d): h was chosen “partially halved” in the sense that

$$h = \begin{cases} 2/5 & \text{if } q_1 > 0, \\ 1/5 & \text{if } q_1 < 0. \end{cases}$$

This produced the worst result for the 6th order Gauss method. We thus

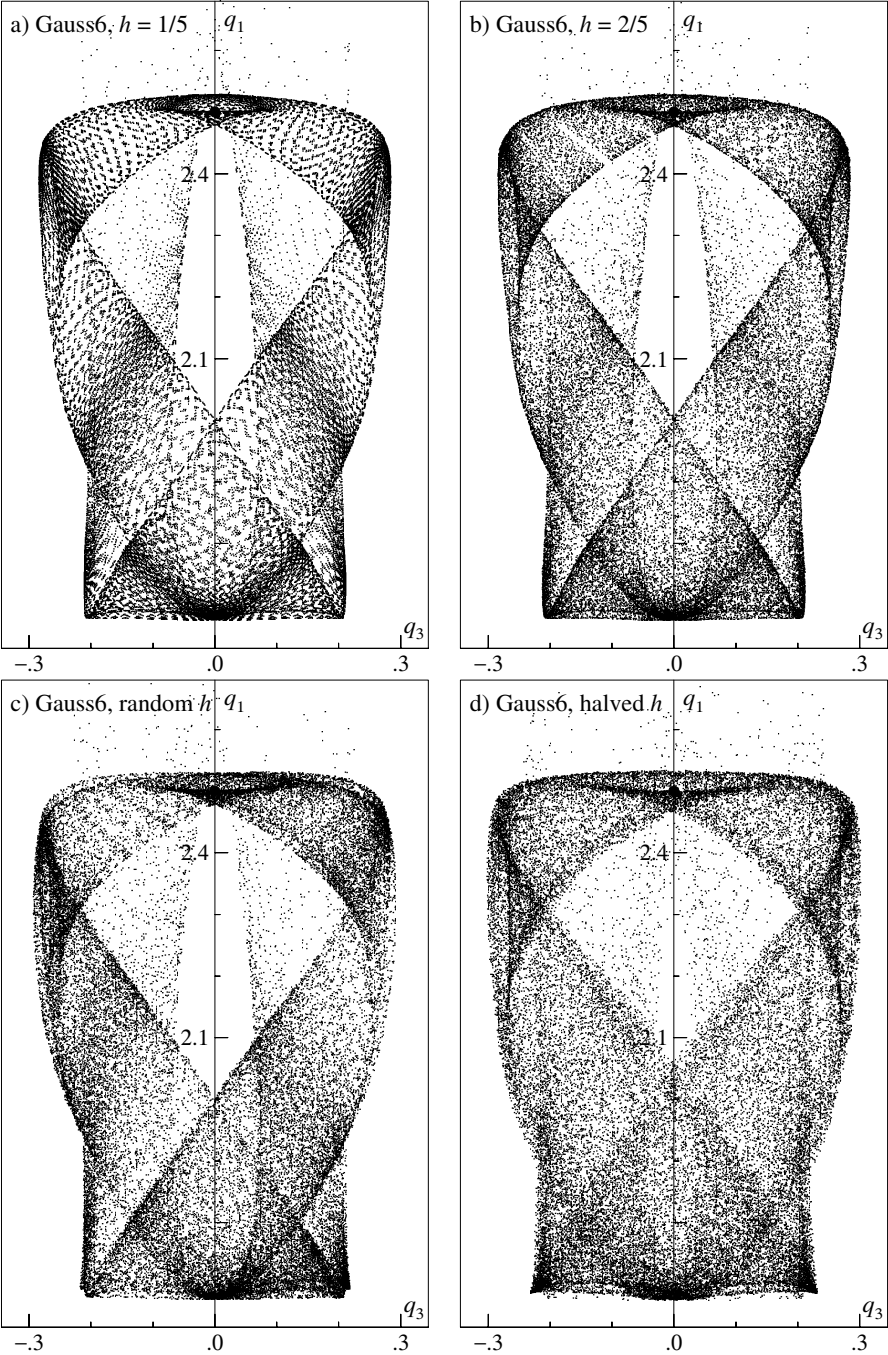


Fig. 16.4. Poincaré cuts for $0 \leq t \leq 1000000$; methods (a)-(d)

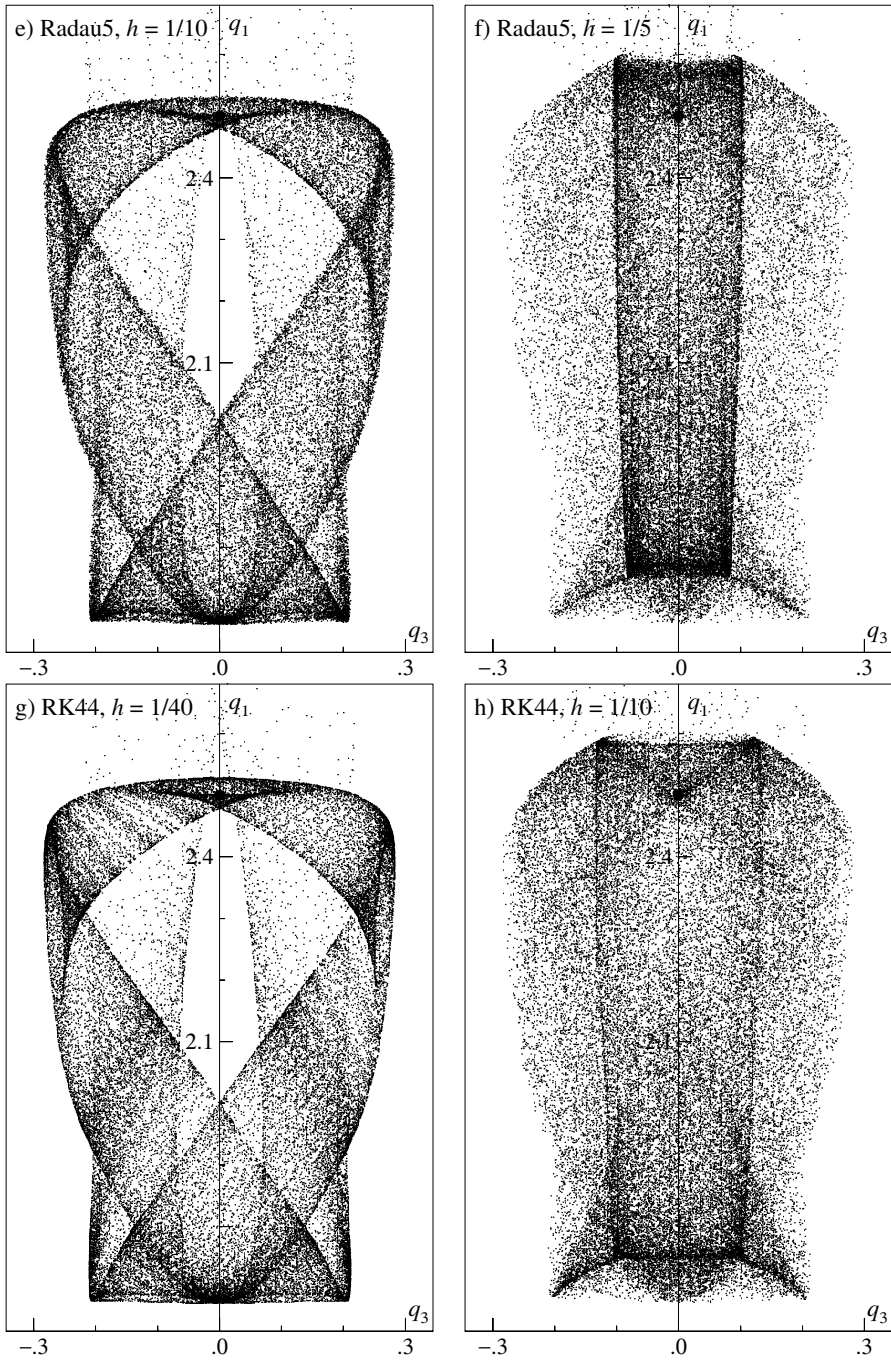


Fig. 16.5. Poincaré cuts for $0 \leq t \leq 1000000$; methods (e)-(h)

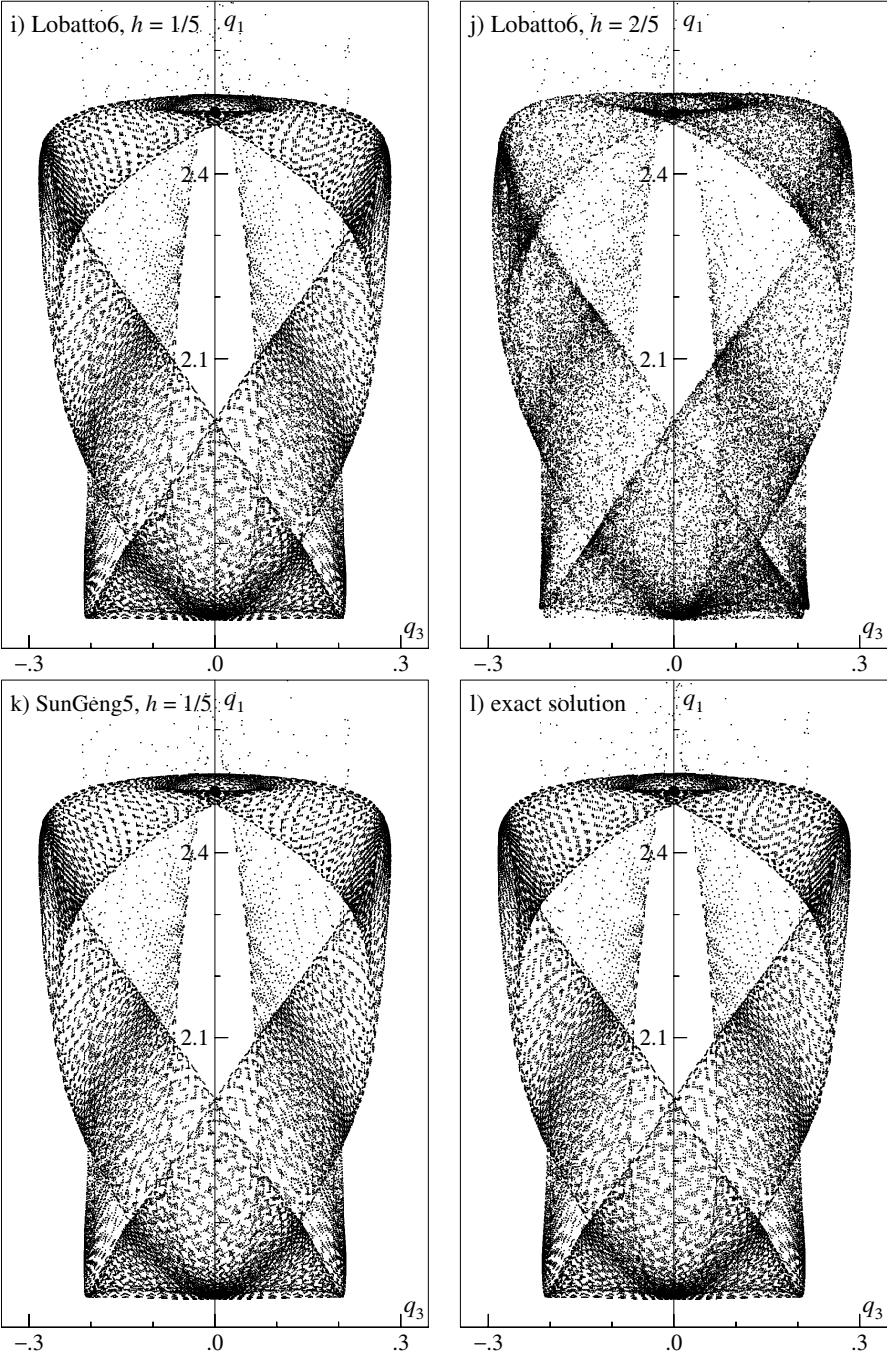


Fig. 16.6. Poincaré cuts for $0 \leq t \leq 1000000$; methods (i)-(l)

see that symplectic and symmetric methods compensate on the way back the errors committed on the outward journey.

- ad e), f): Radau5 (method of Ehle based on Radau quadrature with $s = 3$ and $p = 5$, see Table 7.7) is here not at all satisfactory;
- ad g): The explicit method RK44 (Runge-Kutta method with $s = p = 4$, see Table 1.2, left) is evidently much faster than the implicit methods, even with a smaller step size;
- ad h): With increasing step size RK44 deteriorates drastically;
- ad i): this is a non-symplectic but symmetric collocation method based on Lobatto quadrature with $s = 4$ of order 6 (see Table IV.5.8); its good performance on this nonlinear Hamiltonian problem is astonishing;
- ad j): with increasing h Lobatto6 is less satisfactory (see also Fig. 16.7);
- ad k): this is the symplectic non-symmetric method based on Radau quadrature of order 5 due to Sun Geng 孙耿 (Table 16.1).

The preservation of the Hamiltonian (correct value $H = 2$) during the computation for $0 \leq t \leq 1000000$ is shown in Fig. 16.7. While the errors for the symplectic and symmetric methods in constant step size mode remain bounded, random h (case c) results in a sort of Brownian motion, and the nonsymplectic methods as well as Gauss6 with partially halved step size result in permanent deterioration.

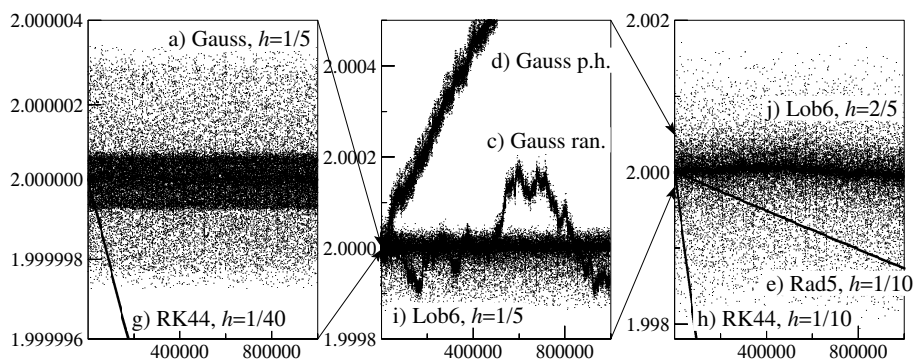


Fig. 16.7. Evolution of the Hamiltonian

Partitioned Runge-Kutta Methods

The fact that the system (16.1) possesses a natural partitioning suggests the use of partitioned Runge-Kutta methods as discussed in Section II.15. The main interest of such methods is for separable Hamiltonians where it is possible to obtain explicit symplectic methods.

A partitioned Runge-Kutta method for system (16.1) is defined by

$$P_i = p_0 + h \sum_j a_{ij} k_j \quad Q_i = q_0 + h \sum_j \hat{a}_{ij} \ell_j \quad (16.26a)$$

$$p_1 = p_0 + h \sum_i b_i k_i \quad q_1 = q_0 + h \sum_i \hat{b}_i \ell_i \quad (16.26b)$$

$$k_i = -\frac{\partial H}{\partial q}(P_i, Q_i) \quad \ell_i = \frac{\partial H}{\partial p}(P_i, Q_i) \quad (16.26c)$$

where b_i, a_{ij} and \hat{b}_i, \hat{a}_{ij} represent two different Runge-Kutta schemes.

Theorem 16.10 (Sanz-Serna 1992b, Suris 1990). *a) If the coefficients of (16.26) satisfy*

$$b_i = \hat{b}_i, \quad i = 1, \dots, s \quad (16.27)$$

$$b_i \hat{a}_{ij} + \hat{b}_j a_{ji} - b_i \hat{b}_j = 0, \quad i, j = 1, \dots, s \quad (16.28)$$

then the method (16.26) is symplectic.

b) If the Hamiltonian is separable (i.e., $H(p, q) = T(p) + U(q)$) then the condition (16.28) alone implies symplecticity of the method.

Proof. Following the lines of the proof of Theorem 16.6 we obtain

$$\begin{aligned} dp_1^J \wedge dq_1^J - dp_0^J \wedge dq_0^J &= h \sum_i \hat{b}_i dP_i^J \wedge d\ell_i^J + h \sum_i b_i dk_i^J \wedge dQ_i^J \\ &\quad - h^2 \sum_{i,j} (b_i \hat{a}_{ij} + \hat{b}_j a_{ji} - b_i \hat{b}_j) dk_i^J \wedge d\ell_j^J, \end{aligned} \quad (16.29)$$

instead of (16.17). The last term vanishes by (16.28). If $b_i = \hat{b}_i$ for all i , symplecticity of the method follows from (16.18). If the Hamiltonian is separable (the mixed derivatives $\partial^2 H / \partial q^J \partial p^L$ and $\partial^2 H / \partial p^J \partial q^L$ are not present in (16.15c,d)) then each of the two terms in (16.18) vanishes separately and the method is symplectic without imposing (16.27). \square

Remark. If (16.28) is satisfied and if the Hamiltonian is separable, it can be assumed without loss of generality that

$$b_i \neq 0, \quad \hat{b}_i \neq 0 \quad \text{for all } i. \quad (16.30)$$

Indeed, the stage values P_i (for i with $\widehat{b}_i = 0$) and Q_j (for j with $b_j = 0$) don't influence the numerical solution (p_1, q_1) and can be removed from the scheme. Notice however that in the resulting scheme the number of stages P_i may be different from that of Q_j .

Explicit methods for separable Hamiltonians. Let the Hamiltonian be of the form $H(p, q) = T(p) + U(q)$ and consider a partitioned Runge-Kutta method satisfying

$$\begin{aligned} a_{ij} &= 0 & \text{for } i < j & \quad (\text{diagonally implicit}) \\ \widehat{a}_{ij} &= 0 & \text{for } i \leq j & \quad (\text{explicit}). \end{aligned} \quad (16.31)$$

Since $\partial H / \partial q$ depends only on q , the method (16.26) is explicit for such a choice of coefficients. Under the assumption (16.30), the symplecticity condition (16.28) then becomes

$$a_{ij} = b_j \quad \text{for } i \geq j, \quad \widehat{a}_{ij} = \widehat{b}_j \quad \text{for } i > j, \quad (16.32)$$

so that the method (16.26) is characterized by the two schemes

$$\left| \begin{array}{cccccc} b_1 & & & & & \\ b_1 & b_2 & & & & \\ b_1 & b_2 & b_3 & & & \\ \vdots & \vdots & \ddots & \ddots & & \\ b_1 & b_2 & \cdots & b_{s-1} & b_s \\ \hline b_1 & b_2 & \cdots & b_{s-1} & b_s \end{array} \right| \quad \left| \begin{array}{cccccc} 0 & & & & & \\ \widehat{b}_1 & 0 & & & & \\ \widehat{b}_1 & \widehat{b}_2 & 0 & & & \\ \vdots & \vdots & \ddots & \ddots & & \\ \widehat{b}_1 & \widehat{b}_2 & \cdots & \widehat{b}_{s-1} & 0 \\ \hline \widehat{b}_1 & \widehat{b}_2 & \cdots & \widehat{b}_{s-1} & \widehat{b}_s \end{array} \right| \quad (16.33)$$

If we admit the cases $b_1 = 0$ and/or $\widehat{b}_s = 0$, it can be shown (Exercise 6) that this scheme already represents the most general method (16.26) which is symplectic and explicit. We denote this scheme by

$$\begin{aligned} b: & \quad b_1 & b_2 & \cdots & b_s \\ \widehat{b}: & \quad \widehat{b}_1 & \widehat{b}_2 & \cdots & \widehat{b}_s. \end{aligned} \quad (16.34)$$

This method is particularly easy to implement:

$$\begin{aligned} P_0 &= p_0, \quad Q_1 = q_0 \\ \text{for } i &:= 1 \text{ to } s \text{ do} \\ P_i &= P_{i-1} - h b_i \partial U / \partial q(Q_i) \\ Q_{i+1} &= Q_i + h \widehat{b}_i \partial T / \partial p(P_i) \\ p_1 &= P_s, \quad q_1 = Q_{s+1} \end{aligned} \quad (16.35)$$

Special case $s = 1$. The combination of the implicit Euler method ($b_1 = 1$) with the explicit Euler method ($\widehat{b}_1 = 1$) gives the following symplectic method of order 1:

$$p_1 = p_0 - h \frac{\partial U}{\partial q}(q_0), \quad q_1 = q_0 + h \frac{\partial T}{\partial p}(p_1). \quad (16.36a)$$

By interchanging the roles of p and q we obtain the method

$$q_1 = q_0 + h \frac{\partial T}{\partial p}(p_0), \quad p_1 = p_0 - h \frac{\partial U}{\partial q}(q_1) \quad (16.36b)$$

which is also symplectic. Methods (16.36a) and (16.36b) are mutually adjoint (see Section II.8).

Construction of higher order methods. The order conditions for general partitioned Runge-Kutta methods applied to general problems (15.2) are derived in Section II.15 (Theorem 15.9). Let us here discuss how these conditions simplify in our special situation.

A) We consider the system (16.1) with separable Hamiltonian. In the notation of Section II.15 this means that $f_a(y_a, y_b)$ depends only on y_b and $f_b(y_a, y_b)$ depends only on y_a . Therefore, many elementary differentials vanish and only P-trees whose meagre and fat vertices alternate in each branch have to be considered. This is a considerable reduction of the order conditions.

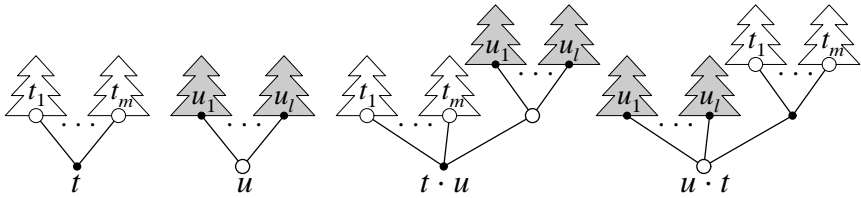


Fig. 16.8. Product of P-trees

B) As observed by Abia & Sanz-Serna (1993) the condition (16.28) acts as a simplifying assumption. Indeed, multiplying (16.28) by $\Phi_i(t) \cdot \Phi_j(u)$ (where $t = {}_a[t_1, \dots, t_m] \in TP^a$, $u = {}_b[u_1, \dots, u_l] \in TP^b$) and summing up over all i and j yields

$$\sum_i b_i \Phi_i(t \cdot u) + \sum_j \hat{b}_j \Phi_j(u \cdot t) - \left(\sum_i b_i \Phi_i(t) \right) \left(\sum_j \hat{b}_j \Phi_j(u) \right) = 0. \quad (16.37)$$

Here we have used the notation of Butcher (1987)

$$t \cdot u = {}_a[t_1, \dots, t_m, u], \quad u \cdot t = {}_b[u_1, \dots, u_l, t], \quad (16.38)$$

illustrated in Fig. 16.8. Since

$$\frac{1}{\gamma(t \cdot u)} + \frac{1}{\gamma(u \cdot t)} - \frac{1}{\gamma(t)} \cdot \frac{1}{\gamma(u)} = 0 \quad (16.39)$$

(this relation follows from (16.37) by inserting the coefficients of a symplectic Runge-Kutta method of sufficiently high order, e.g., a Gauss method) we obtain the following fact:

let $\varrho(t) + \varrho(u) = p$ and assume that all order conditions for P-trees of order $< p$ are satisfied, then

$$\sum_i b_i \Phi_i(t \cdot u) = \frac{1}{\gamma(t \cdot u)} \quad \text{iff} \quad \sum_j \hat{b}_j \Phi_j(u \cdot t) = \frac{1}{\gamma(u \cdot t)}. \quad (16.40)$$

From Fig. 16.8 we see that the P-trees $t \cdot u$ and $u \cdot t$ have the same geometrical structure. They differ only in the position of the root. Repeated application of this property implies that of all P-trees with identical geometrical structure only one has to be considered.

A method of order 3 (Ruth 1983). The above reductions leave five order conditions for a method of order 3 which, for $s = 3$, are the following:

$$\begin{aligned} b_1 + b_2 + b_3 &= 1, & \hat{b}_1 + \hat{b}_2 + \hat{b}_3 &= 1, & b_2 \hat{b}_1 + b_3(\hat{b}_1 + \hat{b}_2) &= 1/2, \\ b_2 \hat{b}_1^2 + b_3(\hat{b}_1 + \hat{b}_2)^2 &= 1/3, & \hat{b}_1 b_1^2 + \hat{b}_2(b_1 + b_2)^2 + \hat{b}_3(b_1 + b_2 + b_3)^2 &= 1/3. \end{aligned}$$

This nonlinear system possesses many solutions. A particularly simple solution, proposed by Ruth (1983), is

$$\begin{aligned} b: & \quad 7/24 \quad 3/4 \quad -1/24 \\ \hat{b}: & \quad 2/3 \quad -2/3 \quad 1. \end{aligned} \quad (16.41)$$

Concatenation of a method with its adjoint. The adjoint method of (16.26) is obtained by replacing h by $-h$ and by exchanging the roles of p_0, q_0 and p_1, q_1 (see Section II.8). This results in a partitioned Runge-Kutta method with coefficients (compare Theorem 8.3)

$$\begin{aligned} a_{ij}^* &= b_{s+1-j} - a_{s+1-i, s+1-j}, & b_i^* &= b_{s+1-i}, \\ \hat{a}_{ij}^* &= \hat{b}_{s+1-j} - \hat{a}_{s+1-i, s+1-j}, & \hat{b}_i^* &= \hat{b}_{s+1-i}. \end{aligned}$$

For the adjoint of (16.33) the first method is explicit and the second one is diagonally implicit, but otherwise it has the same structure. Adding dummy stages, it becomes of the form (16.33) with coefficients

$$\begin{aligned} b^*: & \quad 0 \quad b_s \quad b_{s-1} \quad \dots \quad b_1 \\ \hat{b}^*: & \quad \hat{b}_s \quad \hat{b}_{s-1} \quad \dots \quad \hat{b}_1 \quad 0. \end{aligned} \quad (16.42)$$

The following idea of Sanz-Serna (1992b) allows one to improve a method of odd order p : one considers the composition of method (16.33) (step size $h/2$) with its adjoint (again with step size $h/2$). The resulting method, which is represented by the coefficients

$$\begin{aligned} b_1/2 \quad b_2/2 \quad \dots \quad b_{s-1}/2 \quad b_s/2 \quad b_s/2 \quad b_{s-1}/2 \quad \dots \quad b_1/2 \\ \hat{b}_1/2 \quad \hat{b}_2/2 \quad \dots \quad \hat{b}_{s-1}/2 \quad \hat{b}_s \quad \hat{b}_{s-1}/2 \quad \dots \quad \hat{b}_1/2 \quad 0, \end{aligned}$$

is symmetric and therefore has an even order which is $\geq p + 1$. Concatenating

Ruth's method (16.41) with its adjoint yields the fourth order method

$$\begin{array}{cccccc} b: & 7/48 & 3/8 & -1/48 & -1/48 & 3/8 & 7/48 \\ \widehat{b}: & 1/3 & -1/3 & 1 & -1/3 & 1/3 & 0. \end{array} \quad (16.43)$$

Symplectic Nyström Methods

A frequent special case of a separable Hamiltonian $H(p, q) = T(p) + U(q)$ is when $T(p)$ is a quadratic functional $T(p) = p^T M p / 2$ (with M a constant symmetric matrix). In this situation the Hamiltonian system becomes

$$\dot{p} = -\frac{\partial U}{\partial q}(q), \quad \dot{q} = M p,$$

which is equivalent to the second order equation

$$\ddot{q} = -M \frac{\partial U}{\partial q}(q). \quad (16.44)$$

It is therefore natural to consider Nyström methods (Section II.14) which for the system (16.44) are given by

$$\begin{aligned} Q_i &= q_0 + c_i h \dot{q}_0 + h^2 \sum_j \bar{a}_{ij} k'_j, & k'_j &= -M \frac{\partial U}{\partial q}(Q_j), \\ q_1 &= q_0 + h \dot{q}_0 + h^2 \sum_i \bar{b}_i k'_i, & \dot{q}_1 &= \dot{q}_0 + h \sum_i b_i k'_i. \end{aligned}$$

Replacing the variable \dot{q} by $M p$ and k'_i by $M \ell_i$, this method reads

$$\begin{aligned} Q_i &= q_0 + c_i h M p_0 + h^2 \sum_{j=1}^s \bar{a}_{ij} M \ell_j, & \ell_j &= -\frac{\partial U}{\partial q}(Q_j), \\ q_1 &= q_0 + h M p_0 + h^2 \sum_{i=1}^s \bar{b}_i M \ell_i, & p_1 &= p_0 + h \sum_{i=1}^s b_i \ell_i. \end{aligned} \quad (16.45)$$

Theorem 16.11 (Suris 1989). *Consider the system (16.44) where M is a symmetric matrix. Then, the s -stage Nyström method (16.45) is symplectic if the following two conditions are satisfied:*

$$\bar{b}_i = b_i(1 - c_i), \quad i = 1, \dots, s \quad (16.46a)$$

$$b_i(\bar{b}_j - \bar{a}_{ij}) = b_j(\bar{b}_i - \bar{a}_{ji}), \quad i, j = 1, \dots, s. \quad (16.46b)$$

Proof (Okunbor & Skeel 1992). As in the proof of Theorem 16.6 we differentiate the formulas (16.45) and compute

$$\begin{aligned} dp_1^J \wedge dq_1^J - dp_0^J \wedge dq_0^J \\ = h \sum_i b_i d\ell_i^J \wedge dq_0^J + h \sum_K M_{JK} dp_0^J \wedge dp_0^K \end{aligned} \quad (16.47)$$

$$\begin{aligned}
 & + h^2 \sum_i b_i \sum_K M_{JK} d\ell_i^J \wedge dp_0^K + h^2 \sum_i \bar{b}_i \sum_K M_{JK} dp_0^J \wedge d\ell_i^K \\
 & + h^3 \sum_{i,j} b_i \bar{b}_j \sum_K M_{JK} d\ell_i^J \wedge d\ell_j^K.
 \end{aligned}$$

Next we eliminate dq_0^J with the help of the differentiated equation of Q_i , sum over all J and so obtain

$$\begin{aligned}
 & \sum_{J=1}^n dp_1^J \wedge dq_1^J - \sum_{J=1}^n dp_0^J \wedge dq_0^J \\
 & = h \sum_i b_i \sum_J d\ell_i^J \wedge dQ_i^J + h \sum_{J,K} M_{JK} dp_0^J \wedge dp_0^K \\
 & \quad + h^2 \sum_i (b_i - \bar{b}_i - b_i c_i) \sum_{J,K} M_{JK} d\ell_i^J \wedge dp_0^K \\
 & \quad + h^3 \sum_{i < j} (b_i \bar{b}_j - b_j \bar{b}_i - b_i \bar{a}_{ij} + b_j \bar{a}_{ji}) \sum_{J,K} M_{JK} d\ell_i^J \wedge d\ell_j^K.
 \end{aligned}$$

The last two terms disappear by (16.46) whereas the first two terms vanish due to the symmetry of M and of the second derivatives of $U(q)$. \square

We have already encountered condition (16.46a) in Lemma 14.13. There, it was used as a simplifying assumption. It implies that only the order conditions for \dot{q}_1 have to be considered.

For Nyström methods satisfying both conditions of (16.46), one can assume without loss of generality that

$$b_i \neq 0 \quad \text{for } i = 1, \dots, s. \quad (16.48)$$

Let $I = \{i \mid b_i = 0\}$, then $\bar{b}_i = 0$ for $i \in I$ and $\bar{a}_{ij} = 0$ for $i \notin I, j \in I$. Hence, the stage values Q_i ($i \in I$) don't influence the numerical result (p_1, q_1) and can be removed from the scheme.

Explicit methods. Our main interest is in methods which satisfy

$$\bar{a}_{ij} = 0 \quad \text{for } i \leq j. \quad (16.49)$$

Under the assumption (16.48) the condition (16.46) then implies that the remaining coefficients are given by

$$\bar{a}_{ij} = b_j(c_i - c_j) \quad \text{for } i > j. \quad (16.50)$$

In this situation we may also suppose that

$$c_i \neq c_{i-1} \quad \text{for } i = 2, 3, \dots, s,$$

because equal consecutive c_i lead (via condition (16.50)) to equal stage values Q_i . Therefore the method is equivalent to one with a smaller number of stages.

The particular form of the coefficients \bar{a}_{ij} allows the following simple implementation (Okunbor & Skeel 1992b)

$$\begin{aligned}
 & Q_0 = q_0, \quad P_0 = p_0 \\
 & \text{for } i := 1 \text{ to } s \text{ do} \\
 & \quad Q_i = Q_{i-1} + h(c_i - c_{i-1})MP_{i-1} \quad (\text{with } c_0 = 0) \\
 & \quad P_i = P_{i-1} - hb_i \partial U / \partial q(Q_i) \\
 & \quad q_1 = Q_s + h(1 - c_s)MP_s, \quad p_1 = P_s.
 \end{aligned} \tag{16.51}$$

Special case $s = 1$. Putting $b_1 = 1$ (c_1 is a free parameter) yields a symplectic, explicit Nyström method of order 1. For the choice $c_1 = 1/2$ it has order 2.

Special case $s = 3$. To obtain order 3, four order conditions have to be satisfied (see Table 14.3). The first three mean that (b_i, c_i) is a quadrature formula of order 3. They allow us to express b_1, b_2, b_3 in terms of c_1, c_2, c_3 . The last condition then becomes (Okunbor & Skeel 1992b)

$$\begin{aligned}
 1 + 24\left(c_1 - \frac{1}{2}\right)\left(c_2 - \frac{1}{2}\right) + 24(c_2 - c_1)(c_3 - c_1)(c_3 - c_2) \\
 + 144\left(c_1 - \frac{1}{2}\right)\left(c_2 - \frac{1}{2}\right)\left(c_3 - \frac{1}{2}\right)\left(c_1 + c_3 - c_2 - \frac{1}{2}\right) = 0.
 \end{aligned} \tag{16.52}$$

We thus get a two-parameter family of third order methods. Okunbor & Skeel (1992b) suggest taking

$$c_2 = \frac{1}{2}, \quad c_1 = 1 - c_3 = \frac{1}{6}\left(2 + \sqrt[3]{2} + \frac{1}{\sqrt[3]{2}}\right) \tag{16.53}$$

(the real root of $12c_1(2c_1 - 1)^2 = 1$). This method is symmetric and thus of order 4. Another 3-stage method of order 4 has been found by Qin Meng-Zhao & Zhu Wen-jie (1991).

Higher order methods. For the construction of methods of order ≥ 4 it is worthwhile to investigate the effect of the condition (16.46b) on the order conditions. As for partitioned Runge-Kutta methods one can show that SN-trees with the same geometrical structure lead to equivalent order conditions. For details we refer to Calvo & Sanz-Serna (1992). With the notation of Table 14.3, the SN-trees t_6 and t_7 as well as the pairs t_9, t_{12} and t_{10}, t_{13} give rise to equivalent order conditions. Consequently, for order 5, one has to consider 10 conditions. Okunbor & Skeel (1992c) present explicit, symplectic Nyström methods of orders 5 and 6 with 5 and 7 stages, respectively. A 7th order method is given by Calvo & Sanz-Serna (1992b).

Conservation of the Hamiltonian; Backward Analysis

The differential equation actually solved by the difference scheme will be called the modified equation.

(Warming & Hyett 1974, p. 161)

The *wrong* solution of the *right* equation; the *right* solution of the *wrong* equation.

(Feng Kang, Beijing Sept. 1, 1992)

We have observed above (Example 16.2 and Fig. 16.6) that for the numerical solution of symplectic methods the Hamiltonian H remained between fixed bounds over any long-term integration, i.e., so-called secular changes of H were absent. Following several authors (Yoshida 1993, Sanz-Serna 1992, Feng Kang 1991b) this phenomenon is explained by interpreting the numerical solution as the *exact* solution of a *perturbed Hamiltonian system*, which is obtained as the formal expansion (16.56) in powers of h . The *exact* conservation of the perturbed Hamiltonian \tilde{H} then involves the quasi-periodic behaviour of H along the computed points. This resembles Wilkinson's famous idea of backward error analysis in linear algebra and, in the case of differential equations, seems to go back to Warming & Hyett (1974). We demonstrate this idea for the symplectic Euler method (see (16.36b))

$$\begin{aligned} p_1 &= p_0 - hH_q(p_0, q_1) \\ q_1 &= q_0 + hH_p(p_0, q_1) \end{aligned} \quad (16.54)$$

which, when expanded around the point (p_0, q_0) , gives

$$\begin{aligned} p_1 &= p_0 - hH_q - h^2 H_{qq} H_p - \frac{h^3}{2} H_{qqq} H_p H_p - h^3 H_{qq} H_{pq} H_p - \dots \Big|_{p_0, q_0} \\ q_1 &= q_0 + hH_p + h^2 H_{pq} H_p + \frac{h^3}{2} H_{pqq} H_p H_p + h^3 H_{pq} H_{pq} H_p + \dots \Big|_{p_0, q_0}. \end{aligned} \quad (16.54')$$

In the case of non-scalar equations the p 's and q 's must here be equipped with various summation indices. We suppress these in the sequel for the sake of simplicity and think of scalar systems only. The exact solution of a perturbed Hamiltonian

$$\begin{aligned} \dot{p} &= -\tilde{H}_q(p, q) \\ \dot{q} &= \tilde{H}_p(p, q) \end{aligned}$$

has a Taylor expansion analogous to Theorem 2.6 as follows

$$\begin{aligned} p_1 &= p_0 - h\tilde{H}_q + \frac{h^2}{2} (\tilde{H}_{qp}\tilde{H}_q - \tilde{H}_{qq}\tilde{H}_p) + \dots \\ q_1 &= q_0 + h\tilde{H}_p + \frac{h^2}{2} (-\tilde{H}_{pp}\tilde{H}_q + \tilde{H}_{pq}\tilde{H}_p) + \dots \end{aligned} \quad (16.55)$$

We now set

$$\tilde{H} = H + hH^{(1)} + h^2H^{(2)} + h^3H^{(3)} + \dots \quad (16.56)$$

with unknown functions $H^{(1)}, H^{(2)}, \dots$, insert this into (16.55) and compare the

resulting formulas with (16.54'). Then the comparison of the h^2 terms gives

$$H_q^{(1)} = \frac{1}{2} H_{qq} H_p + \frac{1}{2} H_{qp} H_q, \quad H_p^{(1)} = \frac{1}{2} H_{pp} H_q + \frac{1}{2} H_{pq} H_p$$

which by miracle (the "miracle" is in fact a consequence of the symplecticity of method (16.54)) allow the common primitive

$$H^{(1)} = \frac{1}{2} H_p H_q. \quad (16.56;1)$$

The h^3 terms lead to

$$H^{(2)} = \frac{1}{12} (H_{pp} H_q^2 + H_{qq} H_p^2 + 4H_{pq} H_p H_q) \quad (16.56;2)$$

and so on.

Connection with the Campbell-Baker-Hausdorff formula. An elegant access to the expansion (16.56), which works for separable Hamiltonians $H(p, q) = T(p) + U(q)$, has been given by Yoshida (1993). We interpret method (16.54) as composition of the two symplectic maps

$$z_0 = \begin{pmatrix} p_0 \\ q_0 \end{pmatrix} \xrightarrow{S_T} z = \begin{pmatrix} p_0 \\ q_1 \end{pmatrix} \xrightarrow{S_U} z_1 = \begin{pmatrix} p_1 \\ q_1 \end{pmatrix} \quad (16.57)$$

which consist, respectively, in solving exactly the Hamiltonian systems

$$\begin{aligned} \dot{p} &= 0 & \text{and} & & \dot{p} &= -U_q(q) \\ \dot{q} &= T_p(p) & & & \dot{q} &= 0 \end{aligned} \quad (16.58)$$

and apply some Lie theory. If we introduce for these equations the differential operators given by (13.2')

$$D_T \Psi = \frac{\partial \Psi}{\partial q} T_p(p), \quad D_U \Psi = -\frac{\partial \Psi}{\partial p} U_q(q), \quad (16.59)$$

the formulas (13.3) allow us to write the Taylor series of the map S_T as

$$z = \sum_{i=0}^{\infty} \frac{h^i}{i!} D_T^i z \Big|_{z=z_0}. \quad (16.60)$$

If now $F(z)$ is an arbitrary function of the solution $z(t) = (p(t), q(t))$ (left equation of (16.58)), we find, as in (13.2), that

$$F(z)' = D_T F, \quad F(z)'' = D_T^2 F, \dots$$

and (16.60) extends to (Gröbner 1960)

$$F(z) = \sum_{i=0}^{\infty} \frac{h^i}{i!} D_T^i F(z) \Big|_{z=z_0}. \quad (16.60')$$

We now insert S_U for F and insert for S_U the formula analogous to (16.60) to

obtain for the composition (16.57)

$$\begin{aligned} z_1 = (p_1, q_1) &= \sum_{i=0}^{\infty} \frac{h^i}{i!} D_T^i \sum_{j=0}^{\infty} \frac{h^j}{j!} D_U^j z \Big|_{z=z_0} \\ &= \exp(hD_T) \exp(hD_U)(p, q) \Big|_{p=p_0, q=q_0}. \end{aligned} \quad (16.61)$$

But the product $\exp(hD_T) \exp(hD_U)$ is *not* $\exp(hD_T + hD_U)$, as we have all learned in school, because the operators D_T and D_U do not commute. This is precisely the content of the famous Campbell-Baker-Hausdorff Formula (claimed in 1898 by J.E. Campbell and proved independently by Baker (1905) and in the “kleine Untersuchung” of Hausdorff (1906)) which states, for our problem, that

$$\exp(hD_T) \exp(hD_U) = \exp(h\tilde{D}) \quad (16.62)$$

where

$$\begin{aligned} \tilde{D} &= D_T + D_U + \frac{h}{2} [D_T, D_U] + \frac{h^2}{12} ([D_T, [D_T, D_U]] + [D_U, [D_U, D_T]]) \\ &\quad + \frac{h^3}{24} [D_T, [D_U, [D_U, D_T]]] + \dots \end{aligned} \quad (16.63)$$

and $[D_A, D_B] = D_A D_B - D_B D_A$ is the commutator. Equation (16.62) shows that the map (16.57) is the exact solution of the differential equation corresponding to the differential operator \tilde{D} . A straightforward calculation now shows: If

$$D_A \Psi = -\frac{\partial \Psi}{\partial p} A_q + \frac{\partial \Psi}{\partial q} A_p \quad \text{and} \quad D_B \Psi = -\frac{\partial \Psi}{\partial p} B_q + \frac{\partial \Psi}{\partial q} B_p \quad (16.64)$$

are differential operators corresponding to Hamiltonians A and B respectively, then

$$[D_A, D_B] \Psi = D_C \Psi = -\frac{\partial \Psi}{\partial p} C_q + \frac{\partial \Psi}{\partial q} C_p$$

where

$$C = A_p B_q - A_q B_p. \quad (16.65)$$

A repeated application of (16.65) now allows us to obtain for all brackets in (16.63) a corresponding Hamiltonian which finally leads to

$$\tilde{H} = T + U + \frac{h}{2} T_p U_q + \frac{h^2}{12} (T_{pp} U_q^2 + U_{qq} T_p^2) + \frac{h^3}{12} T_{pp} U_{qq} T_p U_q + \dots \quad (16.66)$$

which is the specialization of (16.56) to separable Hamiltonians.

Example 16.12 (Yoshida 1993). For the mathematical pendulum

$$H(p, q) = \frac{p^2}{2} - \cos q \quad (16.67)$$

series (16.66) becomes

$$\tilde{H} = \frac{p^2}{2} - \cos q + \frac{h}{2} p \sin q + \frac{h^2}{12} (\sin^2 q + p^2 \cos q) + \frac{h^3}{12} p \cos q \sin q + \mathcal{O}(h^4). \quad (16.68)$$

Fig. 16.9 presents for various step sizes h and for various initial points ($p_0=0, q_0=-1.5$; $p_0=0, q_0=-2.5$; $p_0=1.5, q_0=-\pi$; $p_0=2.5, q_0=-\pi$) the numerically computed points for method (16.54) compared to the contour lines of $\tilde{H} = \text{Const}$ given by the terms up to order h^3 in (16.68). The excellent agreement of the results with theory for $h \leq 0.6$ leaves nothing to be desired, while for h beyond 0.9 the dynamics of the numerical method turns rapidly into chaotic behaviour.

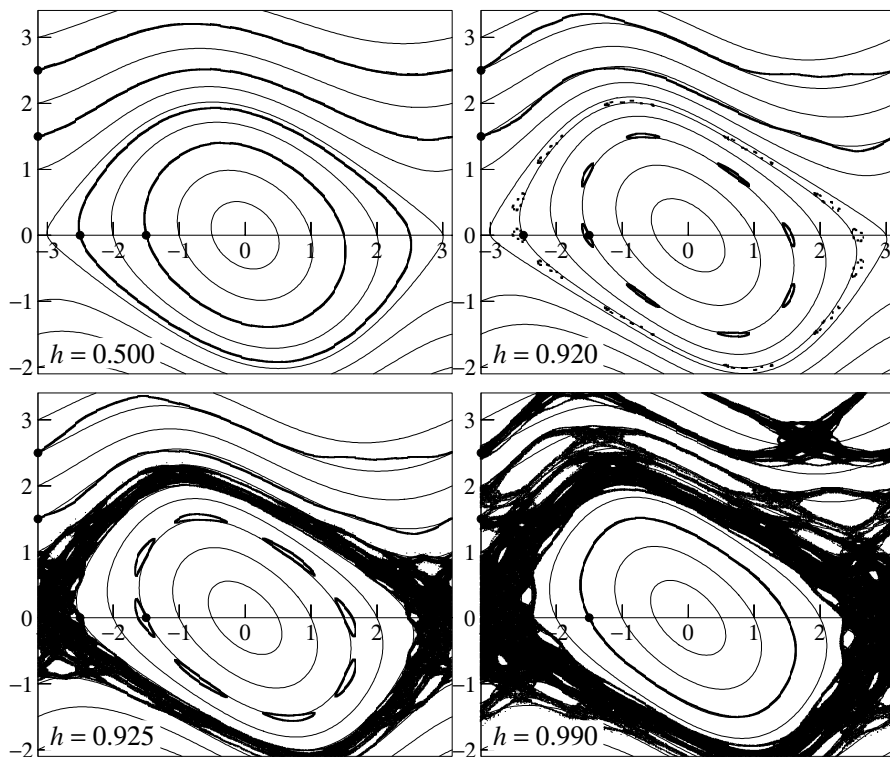


Fig. 16.9. Symplectic method compared to perturbed Hamiltonian
($\bullet \dots$ indicate the initial positions)

Remark. For much research, especially in the beginning of the “symplectic era”, the central role for the construction of canonical difference schemes is played by the Hamilton-Jacobi theory and generating functions. For this, the reader may consult the papers Feng Kang (1986), Feng Kang, Wu Hua-mo, Qin Meng-zhao & Wang Dao-liu (1989), Channell & Scovel (1990) and Miesbach & Pesch (1992). Many additional numerical experiments can be found in Channell & Scovel (1990), Feng Kang (1991), and Pullin & Saffman (1991).

Exercises

1. Show that explicit Runge-Kutta methods are never symplectic.

Hint. Compute the diagonal elements of M .

2. Study the existence and uniqueness of the numerical solution for the implicit mid-point rule when applied to the Hamiltonian system

$$\dot{p} = -q^2, \quad \dot{q} = p.$$

Show that the method possesses no solution at all for $h^2 q_0 + h^3 p_0/2 < -1$ and two solutions for $h^2 q_0 + h^3 p_0/2 > -1$ ($h \neq 0$). Only one of the solutions tends to (p_0, q_0) for $h \rightarrow 0$.

3. A Runge-Kutta method is called *linearly symplectic* if it is symplectic for all linear Hamiltonian systems

$$\dot{y} = J^{-1} C y$$

(J is given in (16.19) and C is a symmetric matrix). Prove (Feng Kang 1985) that a Runge-Kutta method is linearly symplectic if and only if its stability function satisfies

$$R(-z)R(z) = 1 \quad \text{for all } z \in \mathbb{C}. \quad (16.69)$$

Hint. For the definition of the stability function see Section IV.2 of Volume II. Then by Theorem I.14.14, linear symplecticity is equivalent to

$$R(hJ^{-1}C)^T J R(hJ^{-1}C) = J.$$

Furthermore, the matrix $B := J^{-1}C$ is seen to verify $B^T J = -JB$ and hence also $(B^k)^T J = J(-B)^k$ for $k = 0, 1, 2, \dots$. This implies that

$$R(hJ^{-1}C)^T J = J R(-hJ^{-1}C).$$

4. Prove that the stability function of a symmetric Runge-Kutta method satisfies (16.69).
5. Compute all quadratic first integrals of the Hamiltonian system (16.4).
6. For a separable Hamiltonian consider the method (16.26) where $a_{ij} = 0$ for $i < j$, $\hat{a}_{ij} = 0$ for $i < j$ and for every i either $a_{ii} = 0$ or $\hat{a}_{ii} = 0$. If the method satisfies (16.28) then it is equivalent to one given by scheme (16.33).

Hint. Remove first all stages which don't influence the numerical result (see the remark after Theorem 16.10). Then deduce from (16.28) relations similar to (16.32). Finally, remove identical stages and add, if necessary, a dummy stage in order that both methods have the same number of stages.

7. (Lasagni 1990). Characterize symplecticity for multi-derivative Runge-Kutta methods. Show that the s -stage q -derivative method of Definition 13.1 is symplectic if its coefficients satisfy

$$b_i^{(r)} b_j^{(m)} - b_i^{(r)} a_{ij}^{(m)} - b_j^{(m)} a_{ji}^{(r)} = \begin{cases} b_i^{(r+m)} & \text{if } i = j \text{ and } r + m \leq q, \\ 0 & \text{otherwise.} \end{cases} \quad (16.70)$$

Hint. Denote $k^{(r)} = D_H^r p$, $\ell^{(r)} = D_H^r q$, where D_H is the differential operator as in (16.59) and (16.64), so that the exact solution of (16.1) is given by

$$p(x_0+h) = p_0 + \sum_{r \geq 1} \frac{h^r}{r!} k^{(r)}(p_0, q_0), \quad q(x_0+h) = q_0 + \sum_{r \geq 1} \frac{h^r}{r!} \ell^{(r)}(p_0, q_0).$$

Then deduce from the symplecticity of the exact solution that

$$\frac{1}{\varrho!} (dp \wedge d\ell^{(\varrho)} + dk^{(\varrho)} \wedge dq) + \sum_{r+m=\varrho} \frac{1}{r!} \frac{1}{m!} dk^{(r)} \wedge d\ell^{(m)} = 0. \quad (16.71)$$

This, together with a modification of the proof of Theorem 16.6, allows us to obtain the desired result.

8. (Yoshida 1990, Qin Meng-Zhao & Zhu Wen-Jie 1992). Let $y_1 = \psi_h(y_0)$ denote a symmetric numerical scheme of order $p = 2k$. Prove that the composed method

$$\psi_{c_1 h} \circ \psi_{c_2 h} \circ \psi_{c_1 h}$$

is symmetric and has order $p + 2$ if

$$2c_1 + c_2 = 1, \quad 2c_1^{2k+1} + c_2^{2k+1} = 0. \quad (16.72)$$

Hence there exist, for separable Hamiltonians, explicit symplectic partitioned methods of arbitrarily high order.

Hint. Proceed as for (4.1)-(4.2) and use Theorem 8.10 (the order of a symmetric method is even).

9. The Hamiltonian function (16.24) for the galactic problem is *not* separable. Nevertheless, both methods (16.36a) and (16.36b) can be applied explicitly. Explain.

II.17 Delay Differential Equations

Detailed studies of the real world impel us, albeit reluctantly, to take account of the fact that the rate of change of physical systems depends not only on their present state, but also on their past history. (Bellman & Cooke 1963)

Delay differential equations are equations with “retarded arguments” or “time lags” such as

$$y'(x) = f(x, y(x), y(x - \tau)) \quad (17.1)$$

or

$$y'(x) = f(x, y(x), y(x - \tau_1), y(x - \tau_2)) \quad (17.2)$$

or of even more general form. Here the derivative of the solutions depends also on its values at previous points.

Time lags are present in many models of applied mathematics. They can also be the source of interesting mathematical phenomena such as instabilities, limit cycles, periodic behaviour.

Existence

For equations of the type (17.1) or (17.2), where the delay values $x - \tau$ are bounded away from x by a positive constant, the question of existence is an easy matter: suppose that the solution is known, say

$$y(x) = \varphi(x) \quad \text{for } x_0 - \tau \leq x \leq x_0.$$

Then $y(x - \tau)$ is a known function of x for $x_0 \leq x \leq x_0 + \tau$ and (17.1) becomes an ordinary differential equation, which can be treated by known existence theories. We then know $y(x)$ for $x_0 \leq x \leq x_0 + \tau$ and can compute the solution for $x_0 + \tau \leq x \leq x_0 + 2\tau$ and so on. This “method of steps” then yields existence and uniqueness results for all x . For more details we recommend the books of Bellman & Cooke (1963) and Driver (1977, especially Chapter V).

Example 1. We consider the equation

$$y'(x) = -y(x - 1), \quad y(x) = 1 \quad \text{for } -1 \leq x \leq 0. \quad (17.3)$$

Proceeding as described above, we obtain

$$\begin{aligned}
 y(x) &= 1 - x && \text{for } 0 \leq x \leq 1, \\
 y(x) &= 1 - x + \frac{(x-1)^2}{2!} && \text{for } 1 \leq x \leq 2, \\
 y(x) &= 1 - x + \frac{(x-1)^2}{2!} - \frac{(x-2)^3}{3!} && \text{for } 2 \leq x \leq 3, \text{ etc.}
 \end{aligned}$$

The solution is displayed in Fig. 17.1. We observe that despite the fact that the differential equation and the initial function are C^∞ , the solution has discontinuities in its derivatives. This results from the fact that the initial function does not satisfy the differential equation. With every time step τ , however, these discontinuities are smoothed out more and more.

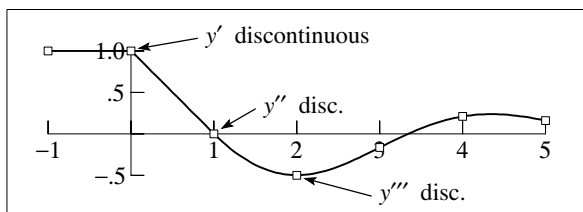


Fig. 17.1. Solution of (17.3)

Example 2. Our next example clearly illustrates the fact that the solutions of a delay equation depend on the entire history between $x_0 - \tau$ and x_0 , and not only on the initial value:

$$y'(x) = -1.4 \cdot y(x-1) \quad (17.4)$$

- a) $\varphi(x) = 0.8$ for $-1 \leq x \leq 0$,
- b) $\varphi(x) = 0.8 + x$ for $-1 \leq x \leq 0$,
- c) $\varphi(x) = 0.8 + 2x$ for $-1 \leq x \leq 0$.

The solutions are displayed in Fig. 17.2. An explanation for the oscillatory behaviour of the solutions will be given below.

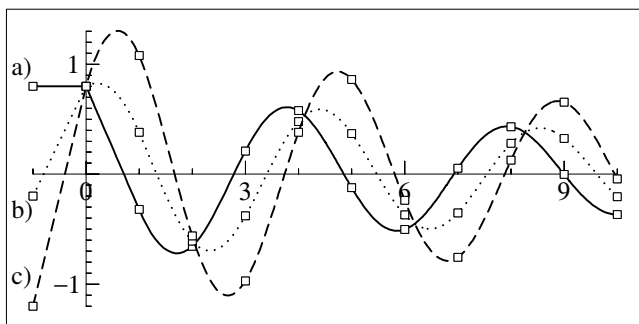


Fig. 17.2. Solutions of (17.4)

Constant Step Size Methods for Constant Delay

If we apply the Runge-Kutta method (1.8) (or (7.7)) to a delay equation (17.1) we obtain

$$g_i^{(n)} = y_n + h \sum_j a_{ij} f(x_n + c_j h, g_j^{(n)}, y(x_n + c_j h - \tau))$$

$$y_{n+1} = y_n + h \sum_j b_j f(x_n + c_j h, g_j^{(n)}, y(x_n + c_j h - \tau)).$$

But which values should we give to $y(x_n + c_j h - \tau)$? If the delay is constant and satisfies $\tau = kh$ for some integer k , the most natural idea is to use the back-values of the old solution

$$g_i^{(n)} = y_n + h \sum_j a_{ij} f(x_n + c_j h, g_j^{(n)}, \gamma_j^{(n)}) \quad (17.5a)$$

$$y_{n+1} = y_n + h \sum_j b_j f(x_n + c_j h, g_j^{(n)}, \gamma_j^{(n)}) \quad (17.5b)$$

where

$$\gamma_j^{(n)} = \begin{cases} \varphi(x_n + c_j h - \tau) & \text{if } n < k \\ g_j^{(n-k)} & \text{if } n \geq k. \end{cases} \quad (17.5c)$$

This can be interpreted as solving successively

$$y'(x) = f(x, y(x), \varphi(x - \tau)) \quad (17.1a)$$

for the interval $[x_0, x_0 + \tau]$, then

$$\begin{aligned} y'(x) &= f(x, y(x), z(x)) \\ z'(x) &= f(x - \tau, z(x), \varphi(x - 2\tau)) \end{aligned} \quad (17.1b)$$

for the interval $[x_0 + \tau, x_0 + 2\tau]$, then

$$\begin{aligned} y'(x) &= f(x, y(x), z(x)) \\ z'(x) &= f(x - \tau, z(x), v(x)) \\ v'(x) &= f(x - 2\tau, v(x), \varphi(x - 3\tau)) \end{aligned} \quad (17.1c)$$

for the interval $[x_0 + 2\tau, x_0 + 3\tau]$, and so on. This is the perfect numerical analog of the “method of steps” mentioned above.

Theorem 17.1. *If c_i, a_{ij}, b_j are the coefficients of a p -th order Runge-Kutta method, then (17.5) is convergent of order p .*

Proof. The sequence (17.1a), (17.1b), ... are ordinary differential equations normally solved by a p th order Runge-Kutta method. Therefore the result follows immediately from Theorem 3.6. \square

Remark. For the collocation method based on Gaussian quadrature formula, Theorem 17.1 yields superconvergence in spite of the use of the low order approximations $\gamma_j^{(n)}$ of (17.5c). Bellen (1984) generalizes this result to the situation where $\tau = \tau(x)$ and $\gamma_j^{(n)}$ is the value of the collocation polynomial at $x_n + c_j h - \tau(x_n + c_j h)$. He proves superconvergence if the grid-points are chosen such that every interval $[x_{n-1}, x_n]$ is mapped, by $x - \tau(x)$, into $[x_{j-1}, x_j]$ for some $j < n$.

Numerical Example. We have integrated the problem

$$y'(x) = (1.4 - y(x-1)) \cdot y(x)$$

(see (17.12) below) for $0 \leq x \leq 10$ with initial values $y(x) = 0$, $-1 \leq x < 0$, $y(0) = 0.1$, and step sizes $h = 1, 1/2, 1/4, 1/8, \dots, 1/128$ using Kutta's methods of order 4 (Table 1.2, left). The absolute value of the global errors (and the solution in grey) are presented in Fig. 17.3. The 4th order convergence can clearly be observed. The downward peaks are provoked by sign changes in the error.

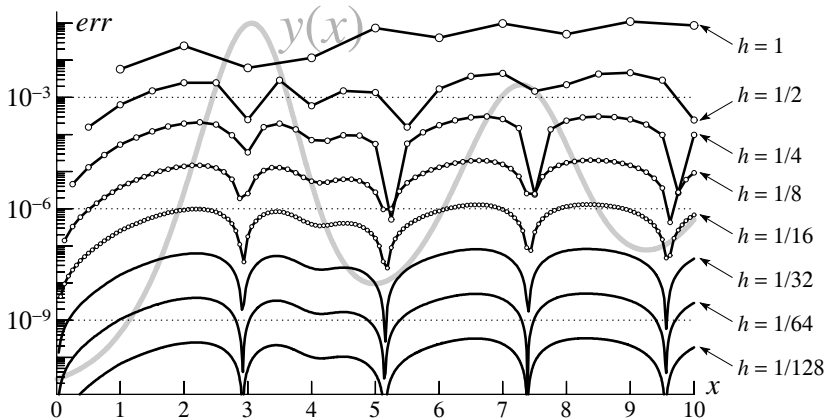


Fig. 17.3. Errors of RK44 with retarded stages (17.5)

Variable Step Size Methods

Although method (17.5) allows efficient and easy to code computations for simple problems with constant delays (such as all the examples of this section), it does not allow to change the step size arbitrarily, and an application to variable delay equations is not straightforward. If complete flexibility is desired, we need a *global* approximation to the solution. Such global approximations are furnished by multi-step methods of Adams or BDF type (see Chapter III.1) or the modern Runge-Kutta methods which are constructed together with a dense output. The code RETARD of the appendix is a modification of the code DOPRI5 (method of Dormand &

Prince in Table 5.2 with Shampine's dense output; see (6.12), (6.13) and the subsequent discussion) in such a way that after every successful step of integration the coefficients of the continuous solution are written into memory. Back-values of the solution are then available by calling the function YLAG(I,X,PHI). For example, for problem (17.4) the subroutine FCN would read as

$$F(1) = -1.4D0 * YLAG(1, X - 1.D0, PHI).$$

As we have seen, the solutions possess discontinuities in the derivatives at several points, e.g. for (17.1) at $x_0 + \tau$, $x_0 + 2\tau$, $x_0 + 3\tau, \dots$ etc. Therefore the code RETARD provides a possibility to match given points of discontinuities exactly (specify IWORK(6) and WORK(11), ...) which improves precision and computation time.

Earlier Runge-Kutta codes for delay equations have been written by Oppelstrup (1976), Oberle & Pesch (1981) and Bellen & Zennaro (1985). Bock & Schlöder (1981) exploited the natural dense output of multistep methods.

Stability

It can be observed from Fig. 17.1 and Fig. 17.2 that the solutions, after the initial phase, seem to tend to something like $e^{\alpha x} \cos \beta(x - \delta)$. We now try to determine α and β . We study the equation

$$y'(x) = \lambda y(x) + \mu y(x - 1). \quad (17.6)$$

There is no loss of generality in supposing the delay $\tau = 1$, since any delay $\tau \neq 1$ can be reduced to $\tau = 1$ by a coordinate change.

We search for a solution of the form

$$y(x) = e^{\gamma x} \quad \text{where } \gamma = \alpha + i\beta. \quad (17.7)$$

Introducing this into (17.6) we obtain the following "characteristic equation" for γ

$$\gamma - \lambda - \mu e^{-\gamma} = 0, \quad (17.8)$$

which, for $\mu \neq 0$, possesses an infinity of solutions: in fact, if $|\gamma|$ becomes large, we obtain from (17.8), since λ is fixed, that $\mu e^{-\gamma}$ must be large too and

$$\gamma \approx \mu e^{-\gamma}. \quad (17.8')$$

This implies that $\gamma = \alpha + i\beta$ is close to the imaginary axis. Hence $|\gamma| \approx |\beta|$ and from (17.8')

$$|\beta| \approx |\mu| e^{-\alpha}.$$

Therefore the roots of (17.8) lie asymptotically on the curves $-\alpha = \log |\beta| - \log |\mu|$. Again from (17.8'), we have a root whenever the argument of $\mu e^{-i\beta}$ is close to $\pi/2$ (for $\beta > 0$), i.e. if

$$\beta \approx \arg \mu - \frac{\pi}{2} + 2k\pi \quad k = 1, 2, \dots$$

There are thus two sequences of characteristic values which tend to infinity on logarithmic curves left of the imaginary axis, with 2π as asymptotic distance between two consecutive values.

The “general solution” of (17.6) is thus a Fourier-like superposition of solutions of type (17.7) (Wright 1946, see also Bellman & Cooke 1963, Chapter 4). The larger $-\operatorname{Re} \gamma$ is, the faster these solutions “die out” as $x \rightarrow \infty$. The dominant solutions are thus (provided that the corresponding coefficients are not zero) those which correspond to the largest real part, i.e., those closest to the origin. For equations (17.3) and (17.4) the characteristic equations are $\gamma + e^{-\gamma} = 0$ and $\gamma + 1.4e^{-\gamma} = 0$ with solutions $\gamma = -0.31813 \pm 1.33724i$ and $\gamma = -0.08170 \pm 1.51699i$ respectively, which explains nicely the behaviour of the asymptotic solutions of Fig. 17.1 and Fig. 17.2.

Remark. For the case of *matrix equations*

$$y'(x) = Ay(x) + By(x-1)$$

where A and B are not simultaneously diagonalizable, we set $y(x) = ve^{\gamma x}$ where $v \neq 0$ is a given vector. The equation now leads to

$$\gamma v = Av + Be^{-\gamma}v,$$

which has a nontrivial solution if

$$\det(\gamma I - A - Be^{-\gamma}) = 0, \quad (17.8'')$$

the characteristic equation for the more general case. The shape of the solutions of (17.8'') is similar to those of (17.8), there are just $r = \operatorname{rank}(B)$ points in each strip of width 2π instead of one.

All solutions of (17.6) remain *stable* for $x \rightarrow \infty$ if all characteristic roots of (17.8) remain in the negative half plane. This result follows either from the above expansion theorem or from the theory of Laplace transforms (e.g., Bellmann & Cooke (1963), Chapter 1), which, in fact, is closely related.

In order to study the boundary of the stability domain, we search for (λ, μ) values for which the first solution γ crosses the imaginary axis, i.e. $\gamma = i\theta$ for θ real. If we insert this into (17.8), we obtain

$$\begin{aligned} \lambda &= -\mu & \text{for } \theta = 0 \ (\gamma \text{ real}) \\ \lambda &= i\theta - \mu e^{-i\theta} & \text{for } \theta \neq 0 \end{aligned}$$

or, by separating real and imaginary parts,

$$\lambda = \frac{\cos \theta \cdot \theta}{\sin \theta}, \quad \mu = -\frac{\theta}{\sin \theta}$$

valid for real λ and μ . These paths are sketched in Fig. 17.4 and separate in the (λ, μ) -plane the domains of stability and instability for the solutions of (17.6) (a result of Hayes 1950).

If we put $\theta = \pi/2$, we find that the solutions of $y'(x) = \mu y(x-1)$ remain *stable* for

$$-\frac{\pi}{2} \leq \mu \leq 0 \quad (17.9a)$$

and are *unstable* for

$$\mu < -\frac{\pi}{2} \quad \text{as well as} \quad \mu > 0. \quad (17.9b)$$

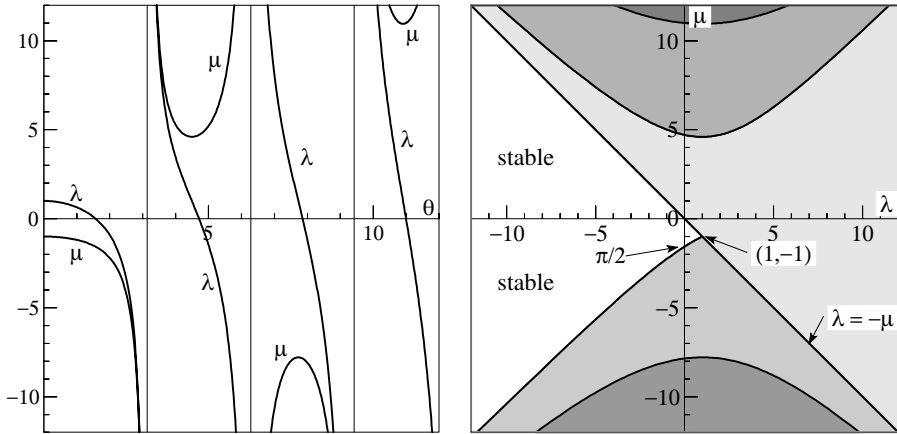


Fig. 17.4. Domain of stability for $y'(x) = \lambda y(x) + \mu y(x-1)$

An Example from Population Dynamics

Lord Cherwell drew my attention to an equation, equivalent to (8) (here: (17.12)) with $a = \log 2$, which he had encountered in his application of probability methods to the problem of distribution of primes. My thanks are due to him for thus introducing me to an interesting problem. (E.M. Wright 1945)

We now demonstrate the phenomena discussed above and the power of our programs on a couple of examples drawn from applications. For supplementary applications of delay equations to all sorts of sciences, consult the impressive list in Driver (1977, p. 239-240).

Let $y(x)$ represent the population of a certain species, whose development as a function of time is to be studied. The simple model of infinite exponential growth $y' = \lambda y$ was soon replaced by the hypothesis that the growth rate λ will decrease with increasing population y due to illness and lack of food and space. One then arrives at the model (Verhulst 1845, Pearl & Reed 1922)

$$y'(x) = k \cdot (a - y(x)) \cdot y(x). \quad (17.10)$$

“Nous donnerons le nom *logistique* à la courbe caractérisée par l’équation précédente” (Verhulst). It can be solved by elementary functions (Exercise 1). All solutions with initial value $y_0 > 0$ tend asymptotically to a as $x \rightarrow \infty$. If we assume the growth rate to depend on the population of the *preceding* generation, (17.10) becomes a delay equation (Cunningham 1954, Wright 1955, Kakutani & Markus 1958)

$$y'(x) = k \cdot (a - y(x - \tau)) \cdot y(x). \quad (17.11)$$

Introducing the new function $z(x) = k\tau y(\tau x)$ into (17.11) and again replacing z by y and $ka\tau$ by a we obtain

$$y'(x) = (a - y(x - 1)) \cdot y(x). \quad (17.12)$$

This equation has an equilibrium point at $y(x) = a$. The substitution $y(x) = a + z(x)$ and linearization leads to the equation $z'(x) = -az(x - 1)$, and condition (17.9) shows that this equilibrium point is locally stable if $0 < a \leq \pi/2$. Hence the characteristic equation, here $\gamma + ae^{-\gamma} = 0$, possesses two real solutions iff $a < 1/e = 0.368$, which makes monotonic solutions possible; otherwise they are oscillatory. For $a > \pi/2$ the equilibrium solution is unstable and gives rise to a periodic limit cycle.

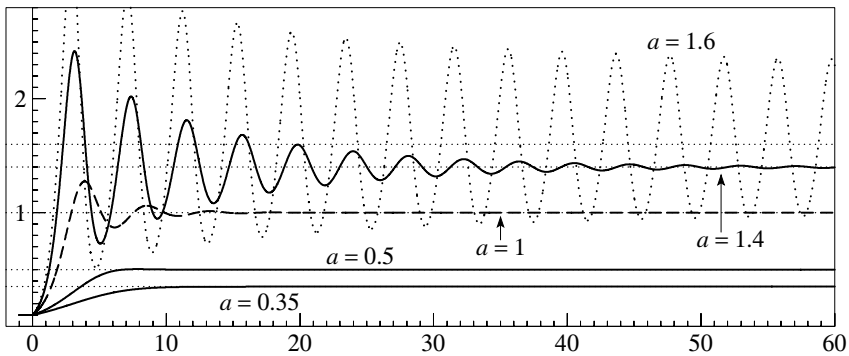


Fig. 17.5. Solutions of the population dynamics problem (17.12)

The solutions in Fig. 17.5 have been computed by the code RETARD of the appendix with subroutine FCN as

$$F(1) = (A - \text{YLAG}(1, X - 1.00, \text{PHI})) * Y(1), \quad A = 0.35, 0.5, 1., 1.4, \text{ and } 1.6.$$

Infectious Disease Modelling

De tous ceux qui ont traité cette matière, c'est sans contredit M. de la Condamine qui l'a fait avec plus de succès. Il est déjà venu à bout de persuader la meilleure partie du monde raisonnable de la grande utilité de l'inoculation: quant aux autres, il serait inutile de vouloir employer la raison avec eux: puisqu'ils n'agissent pas par principes. Il faut les conduire comme des enfants vers leur mieux ...
(Daniel Bernoulli 1760)

Daniel Bernoulli ("Docteur en medecine, Professeur de Physique en l'Université de Bâle, Associé étranger de l'Academie des Sciences") was the first to use differential calculus to model infectious diseases in his 1760 paper on smallpox vaccination. At the beginning of our century, mathematical modelling of epidemics gained new interest. This finally led to the classical model of Kermack & McKendrick (1927): let $y_1(x)$ measure the *susceptible* portion of the population, $y_2(x)$ the *infected*, and $y_3(x)$ the *removed* (e.g. immunized) one. It is then natural to assume that the number of newly infected people per time unit is proportional to the product $y_1(x)y_2(x)$, just as in bimolecular chemical reactions (see Section I.16). If we finally assume the number of newly removed persons to be proportional to the infected ones, we arrive at the model

$$y_1' = -y_1 y_2, \quad y_2' = y_1 y_2 - y_2, \quad y_3' = y_2 \quad (17.13)$$

where we have taken for simplicity all rate constants equal to one. This system can be integrated by elementary methods (divide the first two equations and solve $dy_2/dy_1 = -1 + 1/y_1$). The numerical solution with initial values $y_1(0) = 5$, $y_2(0) = 0.1$, $y_3(0) = 1$ is painted in gray color in Fig. 17.6: an epidemic breaks out, everybody finally becomes "removed" and nothing further happens.

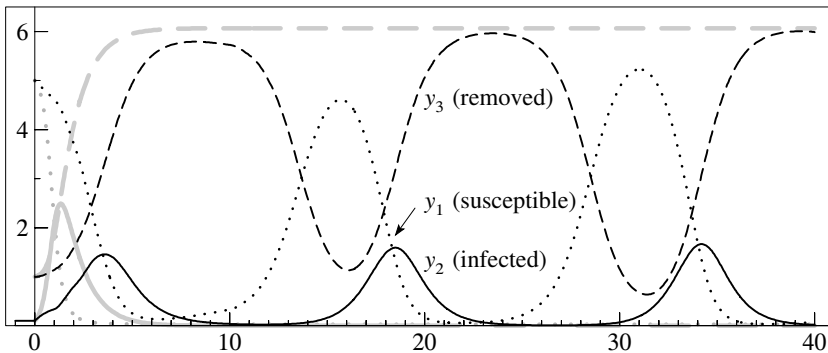


Fig. 17.6. Periodic outbreak of disease, model (17.14)
(in gray: Solution of Kermack - McKendrick model (17.13))

We arrive at a periodic outbreak of the disease, if we assume that immunized people become susceptible again, say after a fixed time τ ($\tau = 10$). If we also

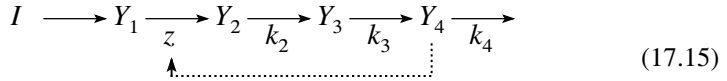
introduce an incubation period of, say, $\tau_2 = 1$, we arrive at the model

$$\begin{aligned}y_1'(x) &= -y_1(x)y_2(x-1) + y_2(x-10) \\y_2'(x) &= y_1(x)y_2(x-1) - y_2(x) \\y_3'(x) &= y_2(x) - y_2(x-10)\end{aligned}\tag{17.14}$$

instead of (17.13). The solutions of (17.14), for the initial phases $y_1(x) = 5$, $y_2(x) = 0.1$, $y_3(x) = 1$ for $x \leq 0$, are shown in Fig. 17.6 and illustrate the periodic outbreak of the disease.

An Example from Enzyme Kinetics

Our next example, more complicated than the preceding ones, is from enzyme kinetics (Okamoto & Hayashi 1984). Consider the following consecutive reactions



where I is an exogenous substrate supply which is maintained constant and n molecules of the end product Y_4 inhibit co-operatively the reaction step of $Y_1 \rightarrow Y_2$ as

$$z = \frac{k_1}{1 + \alpha(y_4(x))^n}.$$

It is generally expected that the inhibitor molecule must be moved to the position of the regulatory enzyme by forces such as diffusion or active transport. Thus, we consider this time consuming process causing time-delay and we arrive at the model

$$\begin{aligned}y_1'(x) &= I - zy_1(x) \\y_2'(x) &= zy_1(x) - y_2(x) \\y_3'(x) &= y_2(x) - y_3(x) \\y_4'(x) &= y_3(x) - 0.5y_4(x)\end{aligned}\quad z = \frac{1}{1 + 0.0005(y_4(x-4))^3}.\tag{17.16}$$

This system possesses an equilibrium at $zy_1 = y_2 = y_3 = I$, $y_4 = 2I$, $y_1 = I(1 + 0.004I^3) =: c_1$. When it is linearized in the neighbourhood of this equilibrium point, it becomes

$$\begin{aligned}y_1'(x) &= -c_1y_1(x) + c_2y_4(x-4) \\y_2'(x) &= c_1y_1(x) - y_2(x) - c_2y_4(x-4) \\y_3'(x) &= y_2(x) - y_3(x) \\y_4'(x) &= y_3(x) - 0.5y_4(x)\end{aligned}\tag{17.17}$$

where $c_2 = c_1 \cdot I^3 \cdot 0.006$. By setting $y(x) = v \cdot e^{\gamma x}$ we arrive at the characteristic equation (see (17.8')), which becomes after some simplifications

$$(c_1 + \gamma)(1 + \gamma)^2(0.5 + \gamma) + c_2\gamma e^{-4\gamma} = 0. \quad (17.18)$$

As in the paper of Okamoto & Hayashi, we put $I = 10.5$. Then (17.18) possesses one pair of complex solutions in \mathbb{C}^+ , namely

$$\gamma = 0.04246 \pm 0.47666i$$

and the equilibrium solution is unstable (see Fig. 17.7). The period of the solution of the linearized equation is thus $T = 2\pi/0.47666 = 13.18$. The solutions then tend to a limit cycle of approximately the same period.

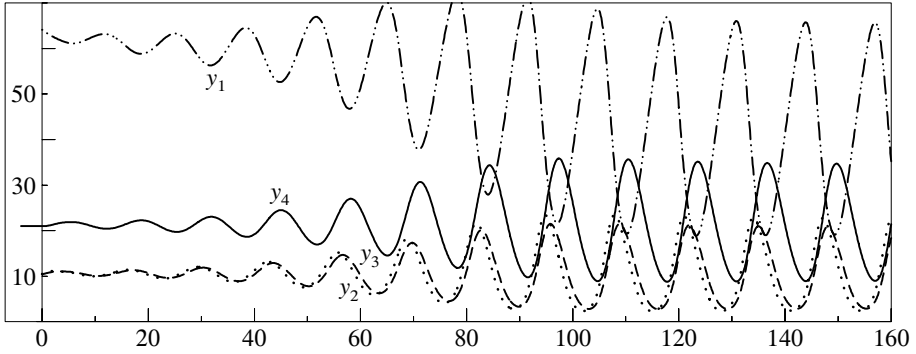


Fig. 17.7. Solutions of the enzyme kinetics problem (17.16), $I = 10.5$.
Initial values close to equilibrium position

A Mathematical Model in Immunology

We conclude our series of examples with Marchuk's model (Marchuk 1975) for the struggle of viruses $V(t)$, antibodies $F(t)$ and plasma cells $C(t)$ in the organism of a person infected by a viral disease. The equations are

$$\begin{aligned} \frac{dV}{dt} &= (h_1 - h_2 F)V \\ \frac{dC}{dt} &= \xi(m)h_3 F(t - \tau)V(t - \tau) - h_5(C - 1) \\ \frac{dF}{dt} &= h_4(C - F) - h_8 FV. \end{aligned} \quad (17.19)$$

The first is a Volterra - Lotka like predator-prey equation. The second equation describes the creation of new plasma cells with time lag due to infection, in the absence of which the second term creates an equilibrium at $C = 1$. The third equation models the creation of antibodies from plasma cells ($h_4 C$) and their

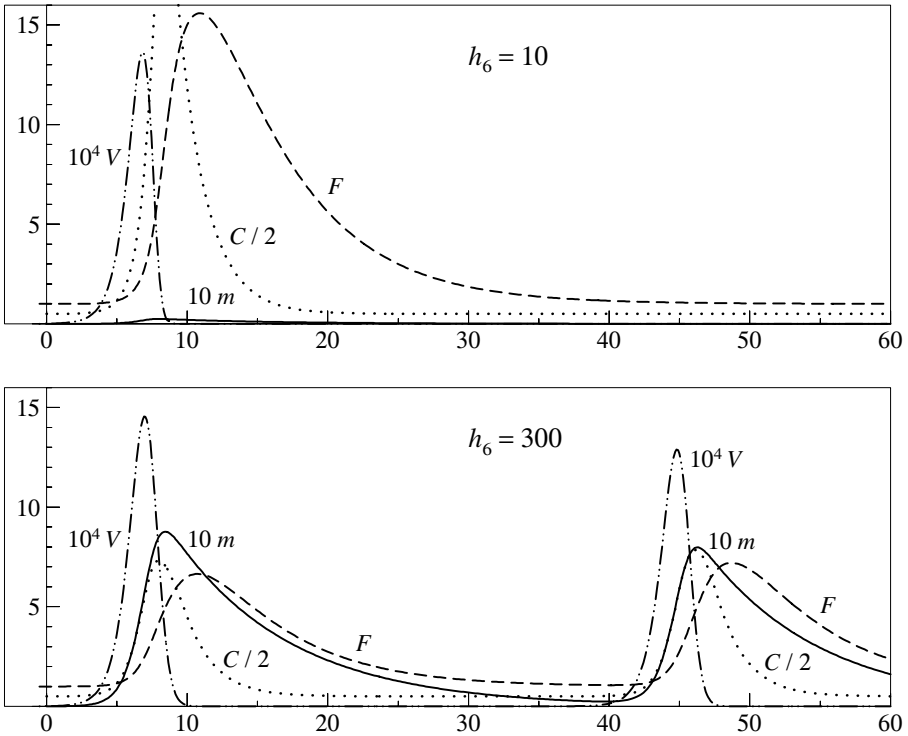


Fig. 17.8. Solutions of the Marchuk immunology model

decrease due to aging ($-h_4 F$) and binding with antigens ($-h_8 FV$). The term $\xi(m)$, finally, is defined by

$$\xi(m) = \begin{cases} 1 & \text{if } m \leq 0.1 \\ (1-m)\frac{10}{9} & \text{if } 0.1 \leq m \leq 1 \end{cases}$$

and expresses the fact that the creation of plasma cells slows down when the organism is damaged by the viral infection. The relative characteristic $m(t)$ of damaging is given by a fourth equation

$$\frac{dm}{dt} = h_6 V - h_7 m$$

where the first term expresses the damaging and the second recuperation.

This model allows us, by changing the coefficients h_1, h_2, \dots, h_8 , to model all sorts of behaviour of stable health, unstable health, acute form of a disease, chronic form etc. See Chapter 2 of Marchuk (1983). In Fig. 17.8 we plot the solutions of this model for $\tau = 0.5$, $h_1 = 2$, $h_2 = 0.8$, $h_3 = 10^4$, $h_4 = 0.17$, $h_5 = 0.5$, $h_7 = 0.12$, $h_8 = 8$ and initial values $V(t) = \max(0, 10^{-6} + t)$ if $t \leq 0$, $C(0) = 1$, $F(t) = 1$ if $t \leq 0$, $m(0) = 0$. In dependence of the value of h_6 ($h_6 = 10$

or $h_6 = 300$), we then observe either complete recovery (defined by $V(t) < 10^{-16}$), or periodic outbreak of the disease due to damaging ($m(t)$ becomes nearly 1).

Integro-Differential Equations

Often the hypothesis that a system depends on the time lagged solution at a specified fixed value $x - \tau$ is not very realistic, and one should rather suppose this dependence to be stretched out over a longer period of time. Then, instead of (17.1), we would have for example

$$y'(x) = f\left(x, y(x), \int_{x-\tau}^x K(x, \xi, y(\xi)) d\xi\right). \quad (17.20)$$

The numerical treatment of these problems becomes much more expensive (see Brunner & van der Houwen (1986) for a study of various discretization methods). If $K(x, \xi, y)$ is zero in the neighbourhood of the diagonal $x = \xi$, one can eventually use RETARD and call a quadrature routine for each function evaluation.

Fortunately, many integro-differential equations can be reduced to ordinary or delay differential equations by introducing new variables for the integral function.

Example (Volterra 1934). Consider the equation

$$y'(x) = \left(\varepsilon - \alpha y(x) - \int_0^x k(x - \xi)y(\xi) d\xi\right) \cdot y(x) \quad (17.21)$$

for population dynamics, where the integral term represents a decrease of the reproduction rate due to pollution. If now for example $k(x) = c$, we put

$$\int_0^x y(\xi) d\xi = v(x), \quad y(x) = v'(x)$$

and obtain

$$v''(x) = (\varepsilon - \alpha v'(x) - cv(x)) \cdot v'(x),$$

an ordinary differential equation.

The same method is possible for equations (17.20) with “degenerate kernel”; i.e., where

$$K(x, \xi, y) = \sum_{i=1}^m a_i(x)b_i(\xi, y). \quad (17.22)$$

If we insert this into (17.20) and put

$$v_i(x) = \int_{x-\tau}^x b_i(\xi, y(\xi)) d\xi, \quad (17.23)$$

we obtain

$$y'(x) = f\left(x, y(x), \sum_{i=1}^m a_i(x) v_i(x)\right) \quad (17.20')$$

$$v'_i(x) = b_i(x, y(x)) - b_i(x - \tau, y(x - \tau)) \quad i = 1, \dots, m,$$

a system of delay differential equations.

Exercises

1. Compute the solution of the Verhulst & Pearl equation (17.10).
2. Compute the equilibrium points of Marchuk's equation (17.19) and study their stability.
3. Assume that the kernel $k(x)$ in Volterra's equation (17.21) is given by

$$k(x) = p(x)e^{-\beta x}$$

where $p(x)$ is some polynomial. Show that this problem can be transformed into an ordinary differential equation.

4. Consider the integro-differential equation

$$y'(x) = f\left(x, y(x), \int_0^x K(x, \xi, y(\xi)) d\xi\right). \quad (17.24)$$

- a) For the degenerate kernel (17.22) problem (17.24) becomes equivalent to the ordinary differential equation

$$\begin{aligned} y'(x) &= f\left(x, y(x), \sum_{j=1}^m a_j(x) v_j(x)\right) \\ v'_j(x) &= b_j(x, y(x)). \end{aligned} \quad (17.25)$$

- b) Show that an application of an explicit (p th order) Runge-Kutta method to (17.25) yields the formulas (Pouzet 1963)

$$\begin{aligned} y_{n+1} &= y_n + h \sum_{i=1}^s b_i f(x_n + c_i h, g_i^{(n)}, u_i^{(n)}) \\ g_i^{(n)} &= y_n + h \sum_{j=1}^{i-1} a_{ij} f(x_n + c_j h, g_j^{(n)}, u_j^{(n)}) \\ u_i^{(n)} &= F_n(x_n + c_i h) + h \sum_{j=1}^{i-1} a_{ij} K(x_n + c_i h, x_n + c_j h, g_j^{(n)}) \end{aligned} \quad (17.26)$$

where

$$F_0(x) = 0, \quad F_{n+1}(x) = F_n(x) + h \sum_{i=1}^s b_i K(x, x_n + c_i h, g_i^{(n)}).$$

- c) If we apply method (17.26) to problem (17.24), where the kernel does not necessarily satisfy (17.22), we nevertheless have convergence of order p .

Hint. Approximate the kernel by a degenerate one.

5. (Zennaro 1986). For the delay equation (17.1) consider the method (17.5) where (17.5c) is replaced by

$$\gamma_j^{(n)} = \begin{cases} \varphi(x_n + c_j h - \tau) & \text{if } n < k \\ q_{n-k}(c_j) & \text{if } n \geq k. \end{cases} \quad (17.5c')$$

Here $q_n(\theta)$ is the polynomial given by a continuous Runge-Kutta method (Section II.6)

$$q_n(\theta) = y_n + h \sum_{j=1}^s b_j(\theta) f(x_n + c_j h, g_j^{(n)}, \gamma_j^{(n)}).$$

- a) Prove that the orthogonality conditions

$$\int_0^1 \theta^{q-1} \left(\gamma(t) \sum_{j=1}^s b_j(\theta) \Phi_j(t) - \theta^{\varrho(t)} \right) d\theta = 0 \quad \text{for } q + \varrho(t) \leq p \quad (17.27)$$

imply convergence of order p , if the underlying Runge-Kutta method is of order p for ordinary differential equations.

Hint. Use the theory of B-series and the Gröbner - Alekseev formula (I.14.18) of Section I.14.

- b) If for a given Runge-Kutta method the polynomials $b_j(\theta)$ of degree $\leq [(p+1)/2]$ are such that $b_j(0) = 0$, $b_j(1) = b_j$ and

$$\int_0^1 \theta^{q-1} b_j(\theta) d\theta = \frac{1}{q} b_j(1 - c_j^q), \quad q = 1, \dots, [(p-1)/2], \quad (17.28)$$

then (17.27) is satisfied. In addition one has the order conditions

$$\sum_{j=1}^s b_j(\theta) \Phi_j(t) = \frac{\theta^{\varrho(t)}}{\gamma(t)} \quad \text{for } \varrho(t) \leq [(p+1)/2].$$

- c) Show that the conditions (17.28) admit unique polynomials $b_j(\theta)$ of degree $[(p+1)/2]$.
6. Solve Volterra's equation (17.21) with $k(x) = c$ and compare the solution with the "pollution free" problem (17.10). Which population lives better, that *with* pollution, or that without?

<http://www.springer.com/978-3-642-05163-0>

Solving Ordinary Differential Equations I
Nonstiff Problems

Hairer, E.; Nørsett, S.P.; Wanner, G.

1993, XV, 528 p., Softcover

ISBN: 978-3-642-05163-0