Chapter 17

Initial Value Problems for Ordinary Differential Equations

17.1 The Problem

For n first order differential equations in one variable involving n functions y_r, $r = 1, \ldots, n$, we shall consider the *initial value problem*

$$(17.1) \qquad \begin{cases} y' &= f(x, y) \quad \text{for } x \in [x_0, \beta], \text{ where} \\ y(x_0) &= y_0 \end{cases}$$

with

$$y = \begin{pmatrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_n(x) \end{pmatrix}, \quad f = \begin{pmatrix} f_1(x, y_1, y_2, \ldots, y_n) \\ f_2(x, y_1, y_2, \ldots, y_n) \\ \vdots \\ f_n(x, y_1, y_2, \ldots, y_n) \end{pmatrix}.$$

The interval $I = [x_0, \beta]$ is called the *interval of integration* for the differential equation.

For $n = 1$ the initial value problem is one dimensional:

$$(17.2) \qquad \begin{cases} y'(x) = f(x, y) \quad \text{for } x \in [x_0, \beta], \text{ where} \\ y(x_0) = y_0. \end{cases}$$

Existence and uniqueness conditions.

For the initial value problem (17.1), the following conditions insure a unique solution of the problem:

(1) The functions f_r, $r = 1, \ldots, n$, are continuous in a region D of the $(x, y_1, \ldots, y_n)$-parameter space.

(2) The functions f_r satisfy a Lipschitz condition for all $(x, y), (x, \bar{y}) \in D$:

$$\|f(x,y) - f(x,\tilde{y})\| \le L\|y - \tilde{y}\|.$$

Then there exists a region $\tilde{D} \subset D$, where the initial value problem (17.1) has exactly one solution y for the initial condition $y(x_0) = y_0$. Condition (2) is fulfilled if for example the f_r possess bounded partial derivatives with respect to y_k in D, in which case

$$L = \max_{\substack{1\le r,k\le n \\ (x,y)\in D}} \left|\frac{\partial f_r}{\partial y_k}\right|.$$

Each initial value problem for a single differential equation of n^{th} order in y with n initial conditions

$$\begin{cases} y^{(n)}(x) &= f(x, y, y', \ldots, y^{(n-1)}), \text{ where} \\ y(x_0) &= y_0,\ y'(x_0) = y_0', \ldots, y^{(n-1)}(x_0) = y_0^{(n-1)} \end{cases}$$

can be reduced to a system (17.1) by setting

$$y^{(k)}(x) =: y_{k+1}(x) \text{ for } k = 0, \ldots, n-1.$$

The associated initial value conditions become

$$y^{(k)}(x_0) = y_{k+1}(x_0) \quad \text{for} \quad k = 0, \ldots, n-1.$$

Thus all methods of this chapter can be employed equally for the solution of first order initial value problems and n^{th} order initial value problems.

Bibliography for Section 17.1

[ENGE87], 10.1; [FRIE79]; [GEAR71/1], 1.1, 3; [HAIR87] 1; [LUTH87], 1,2; [STET76], 9; [STOE91], 7; [WERN79], §1.

17.2 Principles of the Numerical Methods

The interval $[x_0, \beta]$ of integration for (17.1) shall be partitioned

$$x_0 < x_1 < x_2 < \ldots < x_N = \beta \tag{17.3}$$

with local step sizes $h_i := x_{i+1} - x_i > 0$ for $i = 0, \ldots, N-1$.

At the discrete grid points x_i we have to find approximate values Y_i for the exact solution $y(x_i)$

$$Y(x_i) = Y_i \approx y(x_i) = y_i.$$

Let us first consider the case $n = 1$ and integrate (17.2) from x_i to x_{i+1}:

$$\int_{x_i}^{x_{i+1}} y'(x)dx = \int_{x_i}^{x_{i+1}} f(x, y(x))dx.$$

By the fundamental theorem of calculus

$$y(x_{i+1}) = y(x_i) + \int_{x_i}^{x_{i+1}} f(x, y(x))dx \quad \text{for } i = 0, \ldots, N-1. \tag{17.4}$$

(If $n > 1$, we need to evaluate a vector valued integral, one component at a time).

All numerical procedures for solving (17.1) or (17.2) differ only in the choice of method used to approximate the integral in (17.4). They can be classified into three types:

1) one-step methods,
2) multi-step methods, and
3) extrapolation algorithms.

One-step methods use only *one* preceding value Y_i when calculating the next approximate value Y_{i+1}.

Multi-step methods employ $s+1$, $s \ge 1$, preceding values $Y_{i-s}, Y_{i-s+1}, \ldots, Y_{i-1}, Y_i$ to calculate Y_{i+1}.

Extrapolation algorithms use the Romberg quadrature method for the numerical solution of initial value problems.

Specialized methods are the so-called *predictor-corrector methods*. These are procedures which first determine an approximate value $Y_{i+1}^{(0)}$ with a one-step or a multi-step method in a *predictor step*. Then the value of $Y_{i+1}^{(0)}$ is improved by a so-called *corrector step*. The corrections are called $Y_{i+1}^{(1)}, Y_{i+1}^{(2)}, \ldots$.

The difference

$$\varepsilon_{i+1} := y(x_{i+1}) - Y(x_{i+1}) \quad \text{with} \quad Y(x_{i+1}) \approx y(x_{i+1}), \quad Y(x_i) = y(x_i)$$

is called the *local procedural error* at the grid point x_{i+1}. Here $y(x)$ is assumed to be the exact solution of the problem, and we assume that $Y(x_i)$ has been computed without rounding errors. Thus the local procedural error measures the error associated with integrating (17.4) from x_i to x_{i+1}. $O(h_i^{q_l})$ is the order of the *local procedural error*.

The difference

$$e_{i+1} := y(x_{i+1}) - Y(x_{i+1})$$

is the *global procedural error* at the grid point x_{i+1}. It measures the error at x_{i+1} taking into account all previously made errors. $O(h_{max}^{q_g})$ is the order of the global procedural error, where h_{max} is the maximal step size: $h_{max} = \max\limits_{0\le i\le N-1} h_i$. A global error order of q_g is achieved by a procedure if the solution y of (17.1) is (q_g+1)-times continuously differentiable.

Bibliography for Section 17.2

[ENGE87], 10.2; [GEAR71/1], 1.2; [HAIR87] 1; [HALL76], 1; [HENR68], 0.3; [LAPI71], 1; [LUTH87], 3; [STET73], 1; [STOE91], 7.2; [STUM82], 11.1, 12.1; [WERN86], chap.1, 2.

17.3 One-Step Methods

17.3.1 The Euler-Cauchy Polygonal Method

One can approximate the integral in (17.4) by the area of a rectangle:

$$\int_{x_i}^{x_{i+1}} f(x)dx = h_i f(x_i) + \frac{h_i^2}{2} f'(\xi_i) \quad \text{for } \xi_i \in [x_i, x_{i+1}],\ x_{i+1} = x_i + h_i.$$

Thus for $Y_0 = y_0$ we obtain $y_{i+1} = y(x_{i+1})$ as

$$(17.5)\begin{cases} y_{i+1} &= Y_{i+1} + \varepsilon_{i+1}^{EC} \quad \text{with} \\ Y_{i+1} &= Y_i + h_i f(x_i, Y_i) \quad \text{if } Y_i = y(x_i),\ i = 0,\dots,N-1, \\ \varepsilon_{i+1}^{EC} &= \frac{h_i^2}{2} y''(\xi_i) = O(h_i^2) \quad \text{locally for some } \xi_i \in [x_i, x_{i+1}]. \end{cases}$$

Y_{i+1} is the approximate value for $y(x_{i+1})$, and ε_{i+1}^{EC} is the *local procedural error*. It measures the error of the single Euler-Cauchy step from x_i to x_{i+1} under the assumption that $Y_i = y(x_i)$ and that y is twice continuously differentiable. The errors of preceding steps are taken into consideration only by the *global procedural error*:

$$e_{i+1}^{EC} := y_{i+1} - Y_{i+1} = O(h_{max}),\ i = 0,\dots,N-1,\ h_{max} = \max_{0\le i\le N-1}(x_{i+1} - x_i).$$

For systems (17.1) we can formulate the following algorithm that uses (17.5):

ALGORITHM 17.1 (*Euler-Cauchy method*).
For the solution of (17.1) calculate an approximate value Y_i for $y(x_i)$ at each grid point x_i. With $Y_0 = y(x_0)$ the Y_i are given as

$$Y_{i+1} = Y_i + h_i f(x_i, Y_i), \quad i = 0,\dots,N-1.$$

The global error order is $O(h_{max})$ for $h_{max} = \max\limits_{0\le i\le N-1}\{h_i\}$, i.e.,

$$y(x_{i+1}) = Y_{i+1} + O(h_{max}),$$

if y is twice continuously differentiable. For choosing a suitable grid (17.3) and controlling the step sizes, see section 17.3.7.

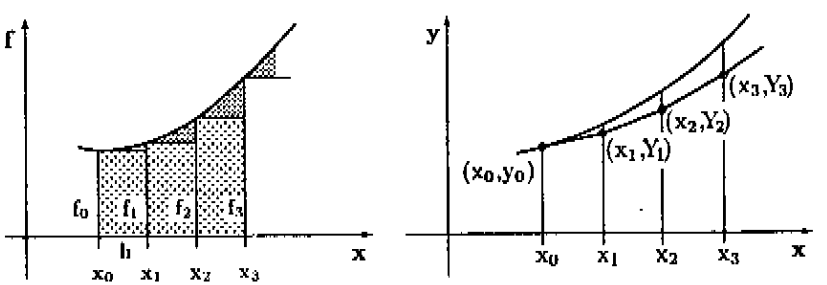

Figure 17.1: The polygonal method

17.3.2 The Improved Euler-Cauchy Method

The improved Euler-Cauchy method uses the direction field at $(x_i + h_i/2, Y_{i+1/2})$ in order to calculate Y_{i+1}. It thus obtains a much improved fitting of the solution to the direction field of the given differential equation.

ALGORITHM 17.2 (*Improved Euler-Cauchy method*).
For the solution of (17.1) and a suitably chosen grid (17.3), we set $Y_0 = y(x_0)$ and define

$$\begin{aligned} Y_{i+1} &= Y_i + h_i f(x_i + \left(\tfrac{h_i}{2}\right), Y_{i+1/2}) \\ &= Y_i + h_i f(x_i + \left(\tfrac{h_i}{2}\right), Y_i + \left(\tfrac{h_i}{2}\right) f(x_i, Y_i)), \; i = 0, \ldots, N-1, \end{aligned}$$

The global error order is $O(h_{\max}^2)$ with $h_{max} := \max\limits_{0 \le i \le N-1} (x_{i+1} - x_i)$, provided that y is three times continuously differentiable. For choosing the proper step size, see section 17.3.7 .

17.3.3 The Predictor-Corrector Method of Heun

Using the trapezoidal rule for evaluating the integral in (17.4) gives an implicit equation for Y_{i+1}, which must be solved iteratively. A first value for $Y_{i+1}^{(0)}$ is determined here by of the Euler-Cauchy method. For a grid (17.3) we have

Heun's method for $n = 1$:

Starting with $Y_0 = y(x_0)$ we calculate for each $i = 0, \ldots, N-1$:

Predictor: $Y_{i+1}^{(0)} = Y_i + h_i f(x_i, Y_i)$,

Corrector: $Y_{i+1}^{(\nu+1)} = Y_i + \frac{h_i}{2}(f(x_i, Y_i) + f(x_{i+1}, Y_{i+1}^{(\nu)})), \; \nu = 0, 1, 2 \ldots$

The local procedural error satisfies:

$$\varepsilon_{i+1}^H = -\frac{h_i^3}{12} y'''(\xi_i) \quad \text{for some } \xi_i \in [x_i, x_{i+1}],$$

provided the solution $y \in C^3[x_0, \beta]$.

Since we cannot compute Y_{i+1} directly, but only the iterates $Y_{i+1}^{(\nu+1)}$, there is an additional *iteration error*

$$\delta_{i+1,\nu}^H := Y_{i+1} - Y_{i+1}^{(\nu+1)}.$$

Thus the *proper local procedural error is*

$$E_{i+1,\nu}^H := y_{i+1} - Y_{i+1}^{(\nu+1)} = \varepsilon_{i+1}^H + \delta_{i+1,\nu}^H.$$

If $K = h_i L_i < 1$, we can estimate

$$|E_{i+1,\nu}^H| \le \frac{1 - h_i L_i + \left(\frac{h_i L_i}{2}\right)^{\nu+1}}{1 - h_i L_i} \frac{h_i^3}{12} |y'''(\xi_1)| + \frac{\left(\frac{h_i L_i}{2}\right)^{\nu+1}}{1 - h_i L_i} \frac{h_i^2}{2} |y''(\xi_2)|,$$

for some $\xi_1, \xi_2 \in [x_i, x_{i+1}]$, so that we have

$$E_{i+1,0}^H = O(h_i^3),$$

already for $\nu = 0$, provided $y \in C^3[x_0, \beta]$. Here L_i are the local Lipschitz constants for $x \in [x_i, x_{i+1}]$.

The local error order of the corrector step is already attained after one iteration step. Experience shows that, for sufficiently small step sizes h_i, one or two iterations suffice to make $|E_{i+1,\nu}^H|$ essentially equal to $|\varepsilon_{i+1}^H|$. This can be assured if one chooses the local step sizes h_i so that

$$0.05 \le K = h_i L_i \le 0.20.$$

The global procedural error e_{i+1}^H, which takes the errors of previous steps into account, is

$$e_{i+1}^H := y_{i+1} - Y_{i+1} = O(h_{\max}^2) \quad \text{for } h_{\max} = \max_{0 \le i \le N-1} h_i, \; h_i = x_{i+1} - x_i.$$

ALGORITHM 17.3 (*Heun's method*).
For solving (17.1) we must perform the following steps with a suitable grid (17.3) and $Y_0 = y(x_0)$ for each $i = 0, \ldots, N-1$:

1. $Y_{i+1}^{(0)} = Y_i + h_i f(x_i, Y_i)$ (predictor).
2. For $\nu = 0, 1$: $Y_{i+1}^{(\nu+1)} = Y_i + \frac{h_i}{2}\left(f(x_i, Y_i) + f(x_{i+1}, Y_{i+1}^{(\nu)})\right)$ (corrector).

The local step size h_i should be chosen so that $K = h_i L_i \le 0.20$. After step 2 we set for each i

$$Y_{i+1} := Y_{i+1}^{(2)}.$$

The global procedural error is

$$e_{i+1}^H = O(h_{\max}^2)$$

with $h_{\max} = \max\limits_{0 \le i \le N-1} h_i, \; h_i = x_{i+1} - x_i$, provided $y \in C^3[x_0, \beta]$.

17.3.4 Explicit Runge-Kutta Methods

17.3.4.1 Construction of Runge-Kutta Methods

The most important category of one-step methods are the Runge-Kutta methods. An explicit Runge-Kutta method of order m has the general form

$$(17.6)\quad \begin{cases} Y_{i+1} = Y_i + h_i \sum_{j=1}^{m} A_j k_j(x_i, Y_i, h_i) \text{ with} \\ k_1(x, Y, h) := f(x, Y), \\ k_j(x, Y, h) := f\left(x + a_j h, Y + h \sum_{s=1}^{j-1} b_{js} k_s(x, Y, h)\right),\ j = 2, \ldots, m. \end{cases}$$

This approach defines the *explicit* Runge-Kutta formulas.

For explicit m^{th} order Runge-Kutta methods with $m \leq 4$, one can achieve a local error order of $q_l = m+1$ and a global error order of $q_g = m$. For $m > 4$, we have $q_g < m$.

Here is a table of the global error order for the m^{th} order explicit Runge-Kutta methods:

m	1	2	3	4	5	6	7	8	9
q_g	1	2	3	4	4	5	6	6	7

If $m = 1$, the Runge-Kutta method is identical with the Euler-Cauchy method. For $m = 2$ the improved Euler-Cauchy method and Heun's method are obtained. Thus they are Runge-Kutta methods of order two. For $m = 4$ we shall find the so-called *classical Runge-Kutta method:*

17.3.4.2 The Classical Runge-Kutta Method

A Runge-Kutta procedure for $m = 4$ of the form (17.6) is called the *classical Runge-Kutta method.*

ALGORITHM 17.4 (*Classical Runge-Kutta method*).
To solve (17.1) one has to choose a suitable grid (17.3), and starting with $Y_0 = y(x_0)$, one evaluates

$$Y_{i+1} = Y_i + h_i \left\{ \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4 \right\} \text{ with}$$

$$\begin{aligned} k_1 &= f(x_i, Y_i), \\ k_2 &= f(x_i + (h_i/2),\ Y_i + h_i(k_1/2)), \\ k_3 &= f(x_i + (h_i/2),\ Y_i + h_i(k_2/2)), \\ k_4 &= f(x_i + h_i,\ Y_i + h_i k_3). \end{aligned}$$

At each grid point x_i, Y_i is an approximate value for the exact value $y(x_i)$. The local error order is $O(h_i^5)$, the global error order is $O(h_{max}^4)$ with $h_{max} := \max_{0 \leq i \leq N-1} (x_{i+1} - x_i)$ if $y \in C^5[a, b]$.

For $n = 1$ one can follow the procedure in the following calculation scheme. For $n > 1$ one has to proceed analogously for each component.

CALCULATION SCHEME 17.5 (*Classical Runge-Kutta method for* $n = 1$).

i	x	y	$k_j(x_i, Y_i, h_i),\ j = 1, \ldots, 4$	$k^{(i)}$
0	x_0	y_0	$k_1 = f(x_0, Y_0)$	k_1
	$x_0 + \frac{h_0}{2}$	$y_0 + h_0 \frac{k_1}{2}$	$k_2 = f(x_0 + \frac{h_0}{2}, y_0 + h_0 \frac{k_1}{2})$	$2k_2$
	$x_0 + \frac{h_0}{2}$	$y_0 + h_0 \frac{k_2}{2}$	$k_3 = f(x_0 + \frac{h_0}{2}, y_0 + h_0 \frac{k_2}{2})$	$2k_3$
	$x_0 + h_0$	$y_0 + h_0 k_3$	$k_4 = f(x_0 + h_0, y_0 + h_0 k_3)$	k_4
	$x_1 = x_0 + h_0$	$Y_1 = y_0 + h_0 k^{(0)}$		$k^{(0)} = \frac{1}{6}\sum$
1	x_1	Y_1	$k_1 = f(x_1, Y_1)$	k_1
	$x_1 + \frac{h_1}{2}$	$Y_1 + h_1 \frac{k_1}{2}$	$k_2 = f(x_1 + \frac{h_1}{2}, Y_1 + h_1 \frac{k_1}{2})$	$2k_2$
	$x_1 + \frac{h_1}{2}$	$Y_1 + h_1 \frac{k_2}{2}$	$k_3 = f(x_1 + \frac{h_1}{2}, Y_1 + h_1 \frac{k_2}{2})$	$2k_3$
	$x_1 + h_1$	$Y_1 + h_1 k_3$	$k_4 = f(x_1 + h_1, Y_1 + h_1 k_3)$	k_4
	$x_2 = x_1 + h_1$	$Y_2 = y_1 + h_1 k^{(1)}$		$k^{(1)} = \frac{1}{6}\sum$
2	x_2	Y_2	$k_1 = f(x_2, Y_2)$	k_1
	$\vdots$	$\vdots$	$\vdots$	$\vdots$

The classical Runge-Kutta method is expensive in computational time. Per Runge-Kutta step, one has to compute four functional values of f since $m = 4$. For higher m more evaluations are necessary. When comparing the calculation time and degree of exactness of the classical Runge-Kutta method or one of higher order with the previously mentioned methods, it is obvious that Runge-Kutta is preferrable to the Euler-Cauchy, the improved Euler-Cauchy method and to Heun's method. A detailed survey can be found in [LUTH87], p.68 .

17.3.4.3 A List of Explicit Runge-Kutta Formulas

In the following we give a coefficient table for explicit Runge-Kutta methods of the form (17.6) for orders $m = 1, \ldots, 8$. For a local error order q_t, the global error order always is $q_g = q_t - 1$.

The formulas (17.6) can be written out in detail as:

$$Y_{i+1} = Y_i + h_i(A_1 k_1 + A_2 k_2 + \ldots + A_m k_m) \text{ with}$$

$$\begin{aligned}
k_1(x_i, Y_i, h_i) &= f(x_i, Y_i) \\
k_2(x_i, Y_i, h_i) &= f(x_i + a_2 h_i, Y_i + h_i b_{21} k_1) \\
k_3(x_i, Y_i, h_i) &= f(x_i + a_3 h_i, Y_i + h_i(b_{31} k_1 + b_{32} k_2)) \\
&\vdots \\
k_m(x_i, Y_i, h_i) &= f(x_i + a_m h_i, Y_i + h_i(b_{m1} k_1 + b_{m2} k_2 + \ldots + b_{m,m-1} k_{m-1})).
\end{aligned}$$

The coefficients A_j, a_j and b_{js} are listed for $j = 1, \ldots, m$, $s = 1, \ldots, m-1$ and $m = 1, \ldots, 8$ in the following table.

Further explicit Runge-Kutta formulas can be found in [FEHL60] and in [FEHL66]; see also section 17.3.4.4 for Runge-Kutta embedding formulas.

TABLE 17.6 (*Coefficient table for explicit Runge-Kutta formulas*).

m	j	A_j	a_j	b_{js} for $s = 1, \ldots, m-1$	q_g	name of the method
1	1	1	0		1	Euler-Cauchy
2	1	0	0		2	Improved
	2	1	1/2	1/2		E.-C. method
2	1	1/2	0		2	Heun's
	2	1/2	1	1		method
3	1	1/6	0		3	RK I
	2	2/3	1/2	1/2		
	3	1/6	1	−1 2		3^{rd} order
3	1	1/4	0		3	RK II
	2	0	1/3	1/3		
	3	3/4	2/3	0 2/3		3^{rd} order
4	1	1/8	0		4	3/8-
	2	3/8	1/3	1/3		
	3	3/8	2/3	−1/3 1		formula
	4	1/8	1	1 −1 1		
4	1	1/6	0		4	Classical
	2	1/3	1/2	1/2		
	3	1/3	1/2	0 1/2		RK-method
	4	1/6	1	0 0 1		

4	1	$\frac{1}{6}$	0						4	RK–Gill method
	2	$\frac{2-\sqrt{2}}{6}$	$\frac{1}{2}$	$\frac{1}{2}$						
	3	$\frac{2+\sqrt{2}}{6}$	$\frac{1}{2}$	$-\frac{1}{2}+\frac{\sqrt{2}}{2}$	$\frac{1-\sqrt{2}}{2}$					
	4	$\frac{1}{6}$	1	0	$-\frac{\sqrt{2}}{2}$	$1+\frac{\sqrt{2}}{2}$				
4	1	1/6	0						4	England I
	2	0	1/2	1/2						
	3	4/6	1/2	1/4	1/4					
	4	1/6	1	0	−1	2				
5	1	$\frac{25}{216}$	0						4	RK-Fehlberg method, 4^{th} order
	2	0	$\frac{1}{4}$	$\frac{1}{4}$						
	3	$\frac{1408}{2565}$	$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$					
	4	$\frac{2197}{4104}$	$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$				
	5	$-\frac{1}{5}$	1	$\frac{439}{216}$	−8	$\frac{3680}{513}$	$-\frac{845}{4104}$			
6	1	$\frac{16}{135}$	0						5	RK-Fehlberg method, 5^{th} order
	2	0	$\frac{1}{4}$	$\frac{1}{4}$						
	3	$\frac{6656}{12825}$	$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$					
	4	$\frac{28561}{56430}$	$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$				
	5	$-\frac{9}{50}$	1	$\frac{439}{216}$	−8	$\frac{3680}{519}$	$-\frac{845}{4104}$			
	6	$\frac{2}{55}$	$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$		
6	1	$\frac{14}{336}$	0						5	England II
	2	0	$\frac{1}{2}$	$\frac{1}{2}$						
	3	0	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$					
	4	$\frac{35}{336}$	1	0	−1	2				
	5	$\frac{162}{336}$	$\frac{2}{3}$	$\frac{7}{27}$	$\frac{10}{27}$	0	$\frac{1}{27}$			
	6	$\frac{125}{336}$	$\frac{1}{5}$	$\frac{28}{625}$	$-\frac{125}{625}$	$\frac{546}{625}$	$\frac{54}{625}$	$-\frac{378}{625}$		

6	1	$\frac{23}{192}$	0						5	Kutta-Nyström method
	2	0	$\frac{1}{3}$	$\frac{1}{3}$						
	3	$\frac{125}{192}$	$\frac{2}{5}$	$\frac{4}{25}$	$\frac{6}{25}$					
	4	0	1	$\frac{1}{4}$	$-\frac{12}{4}$	$\frac{15}{4}$				
	5	$-\frac{81}{192}$	$\frac{2}{3}$	$\frac{6}{81}$	$\frac{90}{81}$	$-\frac{50}{81}$	$\frac{8}{81}$			
	6	$\frac{125}{192}$	$\frac{4}{5}$	$\frac{6}{75}$	$\frac{36}{75}$	$\frac{10}{75}$	$\frac{8}{75}$	0		
6	1	$\frac{31}{384}$	0						5	RK-Fehlberg I (F I)
	2	0	$\frac{1}{6}$	$\frac{1}{6}$						
	3	$\frac{1125}{2816}$	$\frac{4}{15}$	$\frac{4}{75}$	$\frac{16}{75}$					
	4	$\frac{9}{32}$	$\frac{2}{3}$	$\frac{5}{6}$	$-\frac{8}{3}$	$\frac{5}{2}$				
	5	$\frac{125}{768}$	$\frac{4}{5}$	$-\frac{8}{5}$	$\frac{144}{25}$	−4	$\frac{16}{25}$			
	6	$\frac{5}{66}$	1	$\frac{361}{320}$	$-\frac{18}{5}$	$\frac{407}{128}$	$-\frac{11}{80}$	$\frac{55}{128}$		
7	1	$\frac{11}{120}$	0						6	Butcher method
	2	0	$\frac{1}{3}$	$\frac{1}{3}$						
	3	$\frac{27}{40}$	$\frac{2}{3}$	0	$\frac{2}{3}$					
	4	$\frac{27}{40}$	$\frac{1}{3}$	$\frac{1}{12}$	$\frac{1}{3}$	$-\frac{1}{12}$				
	5	$-\frac{4}{15}$	$\frac{1}{2}$	$-\frac{1}{16}$	$\frac{9}{8}$	$-\frac{3}{16}$	$-\frac{3}{8}$			
	6	$-\frac{4}{15}$	$\frac{1}{2}$	0	$\frac{9}{8}$	$-\frac{3}{8}$	$-\frac{3}{4}$	$\frac{1}{2}$		
	7	$\frac{11}{120}$	1	$\frac{9}{44}$	$-\frac{9}{11}$	$\frac{63}{44}$	$\frac{18}{11}$	0	$-\frac{16}{11}$	
8	1	$\frac{7}{1408}$	0						6	RK-Fehlberg II (F II)
	2	0	$\frac{1}{6}$	$\frac{1}{6}$						
	3	$\frac{1125}{2816}$	$\frac{4}{15}$	$\frac{4}{75}$	$\frac{16}{75}$					
	4	$\frac{9}{32}$	$\frac{2}{3}$	$\frac{5}{6}$	$-\frac{8}{3}$	$\frac{5}{2}$				
	5	$\frac{125}{768}$	$\frac{4}{5}$	$-\frac{8}{5}$	$\frac{144}{25}$	−4	$\frac{16}{25}$			
	6	0	1	$\frac{361}{320}$	$-\frac{18}{5}$	$\frac{407}{128}$	$-\frac{11}{80}$	$\frac{55}{128}$		
	7	$\frac{5}{66}$	0	$-\frac{11}{640}$	0	$\frac{11}{256}$	$-\frac{11}{160}$	$\frac{11}{256}$	0	
	8	$\frac{5}{66}$	1	$\frac{93}{640}$	$-\frac{18}{5}$	$\frac{803}{256}$	$-\frac{11}{160}$	$\frac{99}{256}$	0	1

17.3.4.4 Embedding Formulas

If for two explicit Runge-Kutta formulas of orders m and $\bar{m} > m$, the values for k_j coincide for $j = 1, \ldots, m$, one can use both formulas as a pair. The resulting formula is called an *embedding formula*.

In this sense the improved Euler-Cauchy method ($q_g = 2$) together with the Runge-Kutta method of order three; the Runge-Kutta Fehlberg methods with global error orders 4 and 5; the England I and England II formulas with $q_g = 4$ and $q_g = 5$; as well as the Fehlberg I- and Fehlberg II-formulas with $q_g = 5$ or $q_g = 6$ are embedding formulas. Each of the two formulas of a pair supplies an approximate value Y and $\bar{Y}$ for $y(x_i + h)$. These two approximate values can be used for controlling the step size effectively. In general we have:

$$
\begin{cases}
Y & = Y_i + h \sum_{j=1}^{m} A_j k_j \qquad \text{with global error order } q_g \\
\bar{Y} & = Y_i + h \sum_{j=1}^{\bar{m}} \bar{A}_j k_j \qquad \text{with global error order } \bar{q}_g \\
\text{for} & \\
k_1 & = f(x_i, Y_i) \\
k_2 & = f(x_i + a_2 h, Y_i + h b_{21} k_1) \\
\vdots & \\
k_j & = f(x_i + a_j h, Y_i + \sum_{s=1}^{j-1} h b_{js} k_s), \quad j = 3, \ldots, \bar{m}.
\end{cases}
$$

In the following list, we shall give each embedding formula a short code such as *"rk3(2)"* for the Runge-Kutta embedding formula of 3^{rd} and 2^{nd} order.

RK - embedding formula of 2^{nd} and 3^{rd} order, **rk3(2).**
($q_g = 2, \bar{q}_g = 3, m = 2, \bar{m} = 3$)

List of coefficients:

j	$\bar{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, \bar{m} - 1$	
1	$\frac{1}{6}$	0	0		
2	$\frac{2}{3}$	1	$\frac{1}{2}$	$\frac{1}{2}$	
3	$\frac{1}{6}$		1	-1	2

$$
\text{(17.7)} \quad
\begin{cases}
Y & = Y_i + h k_2 \\
\bar{Y} & = Y_i + h \left\{ \frac{1}{6} k_1 + \frac{2}{3} k_2 + \frac{1}{6} k_3 \right\} \\
\text{with} & \\
k_1 & = f(x_i, Y_i) \\
k_2 & = f(x_i + \frac{1}{2} h, Y_i + \frac{1}{2} h k_1) \\
k_3 & = f(x_i + h, Y_i - h k_1 + 2 h k_2).
\end{cases}
$$

RK - Fehlberg - embedding formula of 4^{th} and 5^{th} order, **rkf5(4).**
($q_g = 4, \bar{q}_g = 5, m = 5, \bar{m} = 6$)

Coefficient scheme:

j	$\bar{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, \bar{m} - 1$				
1	$\frac{16}{135}$	$\frac{25}{216}$	0					
2	0	0	$\frac{1}{4}$	$\frac{1}{4}$				
3	$\frac{6656}{12825}$	$\frac{1408}{2565}$	$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$			
4	$\frac{28561}{56430}$	$\frac{2197}{4104}$	$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$		
5	$-\frac{9}{50}$	$-\frac{1}{5}$	1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$-\frac{845}{4104}$	
6	$\frac{2}{55}$		$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$

$$
\text{(17.8)} \quad
\begin{cases}
Y & = Y_i + h \left\{ \frac{25}{216} k_1 + \frac{1408}{2565} k_3 + \frac{2197}{4104} k_4 - \frac{1}{5} k_5 \right\} \\
\bar{Y} & = Y_i + h \left\{ \frac{16}{135} k_1 + \frac{6656}{12825} k_3 + \frac{28561}{56430} k_4 - \frac{9}{50} k_5 + \frac{2}{55} k_6 \right\} \\
\text{with} & \\
k_1 & = f(x_i, Y_i) \\
k_2 & = f(x_i + \frac{1}{4} h,\ Y_i + \frac{1}{4} h k_1) \\
k_3 & = f(x_i + \frac{3}{8} h, Y_i + \frac{3}{32} h k_1 + \frac{9}{32} h k_2) \\
k_4 & = f(x_i + \frac{12}{13} h, Y_i + \frac{1932}{2197} h k_1 - \frac{7200}{2197} h k_2 + \frac{7296}{2197} h k_3) \\
k_5 & = f(x_i + h, Y_i + \frac{439}{216} h k_1 - 8 h k_2 + \frac{3680}{513} h k_3 - \frac{845}{4104} h k_4) \\
k_6 & = f(x_i + \frac{1}{2} h, Y_i - \frac{8}{27} h k_1 + 2 h k_2 - \frac{3544}{2565} h k_3 + \frac{1859}{4104} h k_4 - \frac{11}{40} h k_5).
\end{cases}
$$

England - formula of 4^{th} and 5^{th} order, **rke5(4).**
($q_g = 4$, $\tilde{q}_g = 5$, $m = 4$, $\tilde{m} = 6$)

Coefficient scheme:

j	$\tilde{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, \tilde{m}-1$				
1	$\frac{14}{336}$	$\frac{1}{6}$	0					
2	0	0	$\frac{1}{2}$	$\frac{1}{2}$				
3	0	$\frac{4}{6}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$			
4	$\frac{35}{336}$	$\frac{1}{6}$	1	0	-1	2		
5	$\frac{162}{336}$		$\frac{2}{3}$	$\frac{7}{27}$	$\frac{10}{27}$	0	$\frac{1}{27}$	
6	$\frac{125}{336}$		$\frac{1}{5}$	$\frac{28}{625}$	$-\frac{125}{625}$	$\frac{546}{625}$	$\frac{54}{625}$	$-\frac{378}{625}$

$$
(17.9)\quad \left\{
\begin{aligned}
Y &= Y_i + h\left\{\tfrac{1}{6}k_1 + \tfrac{4}{6}k_3 + \tfrac{1}{6}k_4\right\}\\
\bar{Y} &= Y_i + h\left\{\tfrac{14}{336}k_1 + \tfrac{35}{336}k_4 + \tfrac{162}{336}k_5 + \tfrac{125}{336}k_6\right\}\\
&\text{with}\\
k_1 &= f(x_i, Y_i)\\
k_2 &= f(x_i + \tfrac{h}{2}, Y_i + \tfrac{h}{2}k_1)\\
k_3 &= f(x_i + \tfrac{h}{2}, Y_i + \tfrac{h}{4}k_1 + \tfrac{h}{4}k_2)\\
k_4 &= f(x_i + h, Y_i - hk_2 + 2hk_3)\\
k_5 &= f(x_i + \tfrac{2}{3}h, Y_i + \tfrac{7}{27}hk_1 + \tfrac{10}{27}hk_2 + \tfrac{1}{27}hk_4)\\
k_6 &= f(x_i + \tfrac{h}{5}, Y_i + \tfrac{28}{625}hk_1 - \tfrac{125}{625}hk_2 + \tfrac{546}{625}hk_3 +\\
&\quad + \tfrac{54}{625}hk_4 - \tfrac{378}{625}hk_5).
\end{aligned}
\right.
$$

RK - Fehlberg - embedding formula of 5^{th} and 6^{th} order, **rkf6(5).**
($q_g = 5$, $\tilde{q}_g = 6$, $m = 6$, $\tilde{m} = 8$)

Coefficient scheme:

j	$\tilde{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, \tilde{m}-1$						
1	$\frac{7}{1408}$	$\frac{31}{384}$	0							
2	0	0	$\frac{1}{6}$	$\frac{1}{6}$						
3	$\frac{1125}{2816}$	$\frac{1125}{2816}$	$\frac{4}{15}$	$\frac{4}{75}$	$\frac{16}{75}$					
4	$\frac{9}{32}$	$\frac{9}{32}$	$\frac{2}{3}$	$\frac{5}{6}$	$-\frac{8}{3}$	$\frac{5}{2}$				
5	$\frac{125}{768}$	$\frac{125}{768}$	$\frac{4}{5}$	$-\frac{8}{5}$	$\frac{144}{25}$	-4	$\frac{16}{25}$			
6	0	$\frac{5}{66}$	1	$\frac{361}{320}$	$-\frac{18}{5}$	$\frac{407}{128}$	$-\frac{11}{80}$	$\frac{55}{128}$		
7	$\frac{5}{66}$		0	$-\frac{11}{640}$	0	$\frac{11}{256}$	$-\frac{11}{160}$	$\frac{11}{256}$	0	
8	$\frac{5}{66}$		1	$\frac{93}{640}$	$-\frac{18}{5}$	$\frac{803}{256}$	$-\frac{11}{160}$	$\frac{99}{256}$	0	1

$$
(17.10)\quad \left\{
\begin{aligned}
Y &= Y_i + h\left\{\tfrac{31}{384}k_1 + \tfrac{1125}{2816}k_3 + \tfrac{9}{32}k_4 + \tfrac{125}{768}k_5 + \tfrac{5}{66}k_6\right\}\\
\bar{Y} &= Y_i + h\left\{\tfrac{7}{1408}k_1 + \tfrac{1125}{2816}k_3 + \tfrac{9}{32}k_4 + \tfrac{125}{768}k_5 +\right.\\
&\quad \left. + \tfrac{5}{66}k_7 + \tfrac{5}{66}k_8\right\}\\
&\text{with}\\
k_1 &= f(x_i, Y_i)\\
k_2 &= f(x_i + \tfrac{h}{6}, Y_i + \tfrac{h}{6}k_1)\\
k_3 &= f(x_i + \tfrac{4}{15}h, Y_i + \tfrac{4}{75}hk_1 + \tfrac{16}{75}hk_2)\\
k_4 &= f(x_i + \tfrac{2}{3}h, Y_i + \tfrac{5}{6}hk_1 - \tfrac{8}{3}hk_2 + \tfrac{5}{2}hk_3)\\
k_5 &= f(x_i + \tfrac{4}{5}h, Y_i - \tfrac{8}{5}hk_1 + \tfrac{144}{25}hk_2 - 4hk_3 + \tfrac{16}{25}hk_4)\\
k_6 &= f(x_i + h, Y_i + \tfrac{361}{320}hk_1 - \tfrac{18}{5}hk_2 + \tfrac{407}{128}hk_3 +\\
&\quad - \tfrac{11}{80}hk_4 + \tfrac{55}{128}hk_5);\\
k_7 &= f(x_i, Y_i - \tfrac{11}{640}hk_1 + \tfrac{11}{256}hk_3 - \tfrac{11}{160}hk_4 + \tfrac{11}{256}hk_5)\\
k_8 &= f(x_i + h, Y_i + \tfrac{93}{640}hk_1 - \tfrac{18}{5}hk_2 + \tfrac{803}{256}hk_3 +\\
&\quad - \tfrac{11}{160}hk_4 + \tfrac{99}{256}hk_5 + hk_7).
\end{aligned}
\right.
$$

Further embedding formulas can be composed in complete analogy from the following coefficient schemes.

Further embedding formulas:

RK - Fehlberg - embedding formulas of 3^{rd} *and* 4^{th} *order*, **rkf4(3)**. ($q_g = 3$, $\tilde{q}_g = 4$, $m = 4$, $\tilde{m} = 5$)

Coefficient scheme:

j	$\tilde{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, \tilde{m}-1$			
1	$\frac{229}{1470}$	$\frac{79}{490}$	0				
2	0	0	$\frac{2}{7}$	$\frac{2}{7}$			
3	$\frac{1125}{1813}$	$\frac{2175}{3626}$	$\frac{7}{15}$	$\frac{77}{900}$	$\frac{343}{900}$		
4	$\frac{13718}{81585}$	$\frac{2166}{9065}$	$\frac{35}{38}$	$\frac{805}{1444}$	$-\frac{77175}{54872}$	$\frac{97125}{54872}$	
5	$\frac{1}{18}$		1	$\frac{79}{490}$	0	$\frac{2175}{3626}$	$\frac{2166}{9065}$

Prince-Dormand-embedding formulas of 4^{th} *and* 5^{th} *order*, **rk5(4)6 m**. ($q_g = 4$, $\tilde{q}_g = 5$, $m = 6$, $\tilde{m} = 6$)

Coefficient scheme:

j	$\tilde{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, \tilde{m}-1$				
1	$\frac{19}{216}$	$\frac{31}{540}$	0					
2	0	0	$\frac{1}{5}$	$\frac{1}{5}$				
3	$\frac{1000}{2079}$	$\frac{190}{297}$	$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$			
4	$-\frac{125}{216}$	$-\frac{145}{108}$	$\frac{3}{5}$	$\frac{3}{10}$	$-\frac{9}{10}$	$\frac{6}{5}$		
5	$\frac{81}{88}$	$\frac{351}{220}$	$\frac{2}{3}$	$\frac{226}{729}$	$-\frac{25}{27}$	$\frac{880}{729}$	$\frac{55}{729}$	
6	$\frac{5}{56}$	$\frac{1}{20}$	1	$-\frac{181}{270}$	$\frac{5}{2}$	$-\frac{266}{297}$	$-\frac{91}{27}$	$\frac{189}{55}$

Prince-Dormand-embedding formulas of 4^{th} *and* 5^{th} *order*, **rk5(4)7 m̃**. ($q_g = 4$, $\tilde{q}_g = 5$, $m = 7$, $\tilde{m} = 7$)

Coefficient scheme:

j	$\tilde{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, m-1$					
1	$\frac{35}{384}$	$\frac{5179}{57600}$	0						
2	0	0	$\frac{1}{5}$	$\frac{1}{5}$					
3	$\frac{500}{1113}$	$\frac{7571}{16695}$	$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$				
4	$\frac{125}{192}$	$\frac{393}{640}$	$\frac{4}{5}$	$\frac{44}{45}$	$-\frac{56}{15}$	$\frac{32}{9}$			
5	$-\frac{2187}{6784}$	$-\frac{92097}{339200}$	$\frac{8}{9}$	$\frac{19372}{6561}$	$-\frac{25360}{2187}$	$\frac{64448}{6561}$	$-\frac{212}{729}$		
6	$\frac{11}{84}$	$\frac{187}{2100}$	1	$\frac{9017}{3168}$	$-\frac{355}{33}$	$\frac{46732}{5247}$	$\frac{49}{176}$	$-\frac{5103}{18656}$	
7	0	$\frac{1}{40}$	1	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$

Prince-Dormand-embedding formulas of 4^{th} *and* 5^{th} *order*, **rk5(4)7 m̄**. ($q_g = 4$, $\tilde{q}_g = 5$, $m = 7$, $\tilde{m} = 7$)

Coefficient scheme:

j	$\tilde{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, m-1$					
1	$\frac{19}{200}$	$\frac{431}{5000}$	0						
2	0	0	$\frac{2}{9}$	$\frac{2}{9}$					
3	$\frac{3}{5}$	$\frac{333}{500}$	$\frac{1}{3}$	$\frac{1}{12}$	$\frac{1}{4}$				
4	$-\frac{243}{400}$	$-\frac{7857}{10000}$	$\frac{5}{9}$	$\frac{55}{324}$	$-\frac{25}{108}$	$\frac{50}{81}$			
5	$\frac{33}{40}$	$\frac{957}{1000}$	$\frac{2}{3}$	$\frac{83}{330}$	$-\frac{13}{22}$	$\frac{61}{66}$	$\frac{9}{110}$		
6	$\frac{7}{80}$	$\frac{193}{2000}$	1	$-\frac{19}{28}$	$\frac{9}{4}$	$\frac{1}{7}$	$-\frac{27}{7}$	$\frac{22}{7}$	
7	0	$-\frac{1}{50}$	1	$\frac{19}{200}$	0	$\frac{3}{5}$	$-\frac{243}{400}$	$\frac{33}{40}$	$\frac{7}{80}$

Prince-Dormand-embedding formula of 5^{th} and 6^{th} order, **rk6(5)8 m̃.**
($q_g = 5$, $\tilde{q}_g = 6$, $m = 7$, $\tilde{m} = 8$)

Coefficient scheme:

j	$\bar{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, m-1$						
1	$\frac{61}{864}$	$\frac{821}{10800}$	0							
2	0	0	$\frac{1}{10}$	$\frac{1}{10}$						
3	$\frac{98415}{321776}$	$\frac{19683}{71825}$	$\frac{2}{9}$	$-\frac{2}{81}$	$\frac{20}{81}$					
4	$\frac{16807}{146016}$	$\frac{175273}{912600}$	$\frac{3}{7}$	$\frac{615}{1372}$	$-\frac{270}{343}$	$\frac{1053}{1372}$				
5	$\frac{1375}{7344}$	$\frac{395}{3672}$	$\frac{3}{5}$	$\frac{3243}{5500}$	$-\frac{54}{55}$	$\frac{50949}{71600}$	$\frac{4998}{17875}$			
6	$\frac{1375}{5408}$	$\frac{785}{2704}$	$\frac{4}{5}$	$-\frac{26492}{37125}$	$\frac{72}{55}$	$\frac{2808}{23375}$	$-\frac{24206}{37125}$	$\frac{338}{495}$		
7	$-\frac{37}{1120}$	$\frac{3}{50}$	1	$\frac{5561}{2376}$	$-\frac{35}{11}$	$-\frac{24117}{31603}$	$\frac{899983}{200772}$	$-\frac{5225}{1836}$	$\frac{3925}{4056}$	
8	$\frac{1}{10}$		1	$\frac{466457}{266112}$	$-\frac{2946}{1232}$	$\frac{10513573}{3212352}$	$-\frac{5610201}{14158144}$	$-\frac{424325}{205632}$	$\frac{376225}{454272}$	0

Verner - embedding formula of 5^{th} and 6^{th} order, **rkv6(5).**
($q_g = 5$, $\tilde{q}_g = 6$, $m = 6$, $\tilde{m} = 8$)

Coefficient scheme:

j	$\bar{A}_j$	A_j	a_j	b_{js} for $s = 1, \ldots, m-1$						
1	$\frac{57}{640}$	$\frac{3}{80}$	0							
2	0	0	$\frac{1}{18}$	$\frac{1}{18}$						
3	$-\frac{16}{65}$	$\frac{4}{25}$	$\frac{1}{6}$	$-\frac{1}{12}$	$\frac{1}{4}$					
4	$\frac{1377}{2240}$	$\frac{243}{1120}$	$\frac{2}{9}$	$-\frac{2}{81}$	$\frac{4}{27}$	$\frac{8}{81}$				
5	$\frac{212}{320}$	$\frac{77}{160}$	$\frac{2}{3}$	$\frac{40}{33}$	$-\frac{4}{11}$	$-\frac{56}{11}$	$\frac{54}{11}$			
6	0	$\frac{73}{700}$	1	$-\frac{369}{73}$	$\frac{72}{73}$	$\frac{5380}{219}$	$-\frac{12285}{584}$	$\frac{2695}{1752}$		
7	$\frac{891}{8320}$		$\frac{8}{9}$	$-\frac{8716}{891}$	$\frac{656}{297}$	$\frac{39520}{891}$	$-\frac{416}{11}$	$\frac{52}{27}$	0	
8	$\frac{2}{35}$		1	$\frac{3015}{256}$	$-\frac{9}{4}$	$-\frac{4219}{78}$	$\frac{5985}{128}$	$-\frac{539}{384}$	0	$\frac{693}{3328}$

Verner - embedding formula of 6^{th} and 7^{th} order.
($q_g = 6, \tilde{q}_g = 7, m = 8, \tilde{m} = 10$)

Coefficient scheme of the embedding formula **rkv7(6)**:

j	$\bar{A}_j$	A_j	a_j	b_{js} for $s = 1(1)m-1$									
1	$\frac{2881}{40320}$	$\frac{7}{90}$	0										
2	0	0	$\frac{1}{12}$	$\frac{1}{12}$									
3	0	0	$\frac{1}{6}$	0	$\frac{1}{6}$								
4	$\frac{1216}{2961}$	$\frac{16}{45}$	$\frac{1}{4}$	$\frac{1}{16}$	0	$\frac{3}{16}$							
5	$-\frac{2634}{4085}$	$\frac{16}{45}$	$\frac{3}{4}$	$\frac{21}{16}$	0	$-\frac{81}{16}$	$\frac{9}{2}$						
6	$\frac{24137569}{57482380}$	0	$\frac{16}{17}$	$\frac{1344688}{250563}$	0	$-\frac{1709184}{83521}$	$\frac{1365632}{83521}$	$-\frac{73208}{250563}$					
7	$-\frac{4}{21}$	$\frac{2}{15}$	$\frac{1}{2}$	$-\frac{559}{384}$	0	6	$-\frac{204}{47}$	$\frac{14}{39}$	$-\frac{4913}{78208}$				
8	0	$\frac{7}{90}$	1	$-\frac{625}{224}$	0	12	$-\frac{456}{47}$	$\frac{48}{91}$	$\frac{14739}{136864}$	$\frac{6}{7}$			
9	$\frac{4131}{3920}$		$\frac{2}{3}$	$-\frac{12253}{99144}$	0	$\frac{16}{27}$	$\frac{16}{459}$	$\frac{29072}{161109}$	$-\frac{2023}{73816}$	$\frac{112}{12393}$	0		
10	$-\frac{157}{1260}$		1	$\frac{30517}{2512}$	0	$-\frac{7296}{157}$	$\frac{268728}{7379}$	$\frac{2472}{2041}$	$-\frac{3522621}{10743824}$	$\frac{132}{157}$	0	$-\frac{12393}{4396}$	

Coefficient scheme of the embedding formula rkv8(7):

j	$\hat{A}_j$	A_j	a_j	b_{js} for $s = 1(1)m-1$												
1	$\frac{31}{720}$	$\frac{13}{288}$	0													
2	0	0	$\frac{1}{4}$	$\frac{1}{4}$												
3	0	0	$\frac{1}{12}$	$\frac{5}{72}$	$\frac{1}{72}$											
4	0	0	$\frac{1}{8}$	$\frac{1}{32}$	0	$\frac{3}{32}$										
5	0	0	$\frac{2}{5}$	$\frac{106}{125}$	0	$-\frac{408}{125}$	$\frac{352}{125}$									
6	$\frac{16}{75}$	$\frac{32}{125}$	$\frac{1}{2}$	$\frac{1}{48}$	0	0	$\frac{8}{33}$	$\frac{125}{528}$								
7	$\frac{16807}{79200}$	$\frac{31213}{144000}$	$\frac{6}{7}$	$-\frac{1283}{2401}$	0	0	$\frac{39936}{26411}$	$-\frac{64125}{26411}$	$\frac{5520}{2401}$							
8	$\frac{16807}{79200}$	$\frac{2401}{12375}$	$\frac{1}{7}$	$\frac{37}{392}$	0	0	0	$\frac{1625}{9408}$	$-\frac{2}{15}$	$\frac{61}{6720}$						
9	$\frac{243}{1760}$	$\frac{1701}{14080}$	$\frac{2}{3}$	$\frac{17176}{25515}$	0	0	$-\frac{47104}{25515}$	$\frac{1325}{504}$	$-\frac{41792}{25515}$	$\frac{20237}{145800}$	$\frac{4312}{6075}$					
10	0	$\frac{2401}{19200}$	$\frac{2}{7}$	$-\frac{23824}{180075}$	0	0	$-\frac{77824}{1980825}$	$-\frac{636635}{633864}$	$\frac{254048}{300125}$	$-\frac{183}{7000}$	$\frac{8}{11}$	$-\frac{324}{3773}$				
11	0	$\frac{19}{450}$	1	$\frac{12733}{7600}$	0	0	$-\frac{20032}{5225}$	$\frac{456485}{80256}$	$-\frac{42599}{7125}$	$\frac{339227}{912000}$	$-\frac{1029}{4180}$	$\frac{1701}{1408}$	$\frac{5145}{2432}$			
12	$\frac{243}{1760}$		$\frac{1}{3}$	$-\frac{27081}{204120}$	0	0	$\frac{40448}{280665}$	$-\frac{1353775}{1197504}$	$\frac{17662}{25515}$	$-\frac{71887}{1166400}$	$\frac{98}{225}$	$\frac{1}{16}$	$\frac{3773}{11664}$	0		
13	$\frac{31}{720}$		1	$\frac{11203}{8680}$	0	0	$-\frac{38144}{11935}$	$\frac{2354425}{458304}$	$-\frac{84046}{16275}$	$\frac{673309}{1636800}$	$\frac{4704}{8525}$	$\frac{9477}{10912}$	$-\frac{1029}{992}$	0	$\frac{729}{341}$	

Verner - embedding formula of 7^{th} and 8^{th} order.
$(q_9 = 7, \tilde{q}_9 = 8, m = 11, \tilde{m} = 13)$

Coefficient scheme of the embedding formula rk8(7)13m:

j	$\hat{A}_j$	A_j	a_j	b_{js} for $s = 1(1)m-1$												
1	$\frac{140054451}{3364480064}$	$\frac{12451937}{455176473}$	0													
2	0	0	$\frac{1}{18}$	$\frac{1}{18}$												
3	0	0	$\frac{1}{12}$	$\frac{1}{48}$	$\frac{1}{16}$											
4	0	0	$\frac{1}{8}$	$\frac{1}{32}$	0	$\frac{3}{32}$										
5	0	0	$\frac{5}{16}$	$\frac{5}{16}$	0	$-\frac{75}{64}$	$\frac{75}{64}$									
6	$-\frac{59235493}{10685277825}$	$-\frac{5057110846}{976300145}$	$\frac{3}{8}$	$\frac{3}{80}$	0	0	$\frac{3}{16}$	$\frac{3}{20}$								
7	$\frac{181606767}{758867731}$	$\frac{17570044468}{3445150321}$	$\frac{59}{400}$	$\frac{29443841}{614563906}$	0	0	$\frac{77734538}{692538347}$	$-\frac{28693883}{1125000000}$	$\frac{23124283}{1800000000}$							
8	$\frac{561292985}{797845732}$	$\frac{45604539}{265991186}$	$\frac{93}{200}$	$\frac{16016141}{946892911}$	0	0	$\frac{61564190}{1337722637}$	$\frac{2278713}{633445777}$	$\frac{545815736}{2771057229}$	$-\frac{180191667}{1043307555}$						
9	$-\frac{12418914201}{13713445529}$	$-\frac{2567574721}{75196517206}$	$\frac{34800233248}{37119169821}$	$\frac{39632705}{573591083}$	0	0	$-\frac{413436366}{643702515}$	$-\frac{471739975}{2624392201}$	$\frac{100307431}{723423059}$	$\frac{790204164}{839813087}$	$\frac{800635310}{3783071287}$					
10	$\frac{760417239}{1151165299}$	$\frac{45588588}{322736333}$	$\frac{13}{20}$	$\frac{246121993}{1340847787}$	0	0	$-\frac{37595012795}{152268765246}$	$\frac{30912137}{1069237803}$	$-\frac{12932083}{456754933}$	$\frac{6005943493}{2106947869}$	$\frac{393006217}{1396673457}$	$\frac{123872331}{1001029789}$				
11	$\frac{118820643}{751138087}$	$\frac{53011738}{667516719}$	$\frac{1201146811}{1299019798}$	$-\frac{1028468189}{846180014}$	0	0	$\frac{8478235783}{508512852}$	$\frac{13117732495}{14324721323}$	$-\frac{1003041299935}{4701304382}$	$-\frac{487777925059}{3047939560}$	$\frac{15336726248}{1032824649}$	$-\frac{45442564141}{3398467696}$	$\frac{1065993473}{597172653}$			
12	$-\frac{528747749}{2220607170}$	$\frac{2}{45}$	1	$\frac{1858923177}{718118043}$	0	0	$-\frac{31350945171}{6471077341}$	$-\frac{4777554414}{1098033517}$	$-\frac{703435178}{230739211}$	$\frac{5731566737}{1027545527}$	$\frac{5232866602}{850066563}$	$-\frac{4093664535}{808653257}$	$\frac{3962137247}{1805957418}$	$\frac{65686358}{487910083}$		
13	$\frac{1}{9}$		1	$\frac{403863854}{491083209}$	0	0	$-\frac{30684932393}{4347140067}$	$-\frac{411431997}{543043805}$	$\frac{6527843427}{914398604}$	$\frac{111730625}{925320556}$	$-\frac{13158990841}{6184727034}$	$\frac{3936647629}{1878049440}$	$-\frac{160529052}{685175525}$	$\frac{2486385103}{1413531060}$	0		

Prince-Dormand-embedding formulas of 7^{th} and 8^{th} order, **rk8(7)13m.**
$(q_9 = 7, \tilde{q}_9 = 8, m = 12, \tilde{m} = 13)$

RK-Fehlberg - embedding formulas of 7^{th} and 8^{th} order.
($q_g = 7, \tilde{q}_g = 8, m = 11, \tilde{m} = 13$)

Coefficient scheme of the embedding formula rkf8(7):

j	$\tilde{A}_j$	A_j	a_j	b_{js} for $s = 1(1)m-1$										
1	0	$\frac{41}{840}$	0											
2	0	0	$\frac{2}{27}$	$\frac{2}{27}$										
3	0	0	$\frac{1}{9}$	$\frac{1}{36}$	$\frac{1}{12}$									
4	0	0	$\frac{1}{6}$	$\frac{1}{24}$	0	$\frac{1}{8}$								
5	0	0	$\frac{5}{12}$	$\frac{5}{12}$	0	$-\frac{25}{16}$	$\frac{25}{16}$							
6	$\frac{34}{105}$	$\frac{34}{105}$	$\frac{1}{2}$	$\frac{1}{20}$	0	0	$\frac{1}{4}$	$\frac{1}{5}$						
7	$\frac{9}{35}$	$\frac{9}{35}$	$\frac{5}{6}$	$-\frac{25}{108}$	0	0	$\frac{125}{108}$	$-\frac{65}{27}$	$\frac{125}{54}$					
8	$\frac{9}{35}$	$\frac{9}{35}$	$\frac{1}{6}$	$\frac{31}{300}$	0	0	0	$\frac{61}{225}$	$-\frac{2}{9}$	$\frac{13}{900}$				
9	$\frac{9}{280}$	$\frac{9}{280}$	$\frac{2}{3}$	2	0	0	$-\frac{53}{6}$	$\frac{704}{45}$	$-\frac{107}{9}$	$\frac{67}{90}$	3			
10	$\frac{9}{280}$	$\frac{9}{280}$	$\frac{1}{3}$	$-\frac{91}{108}$	0	0	$\frac{23}{108}$	$-\frac{976}{135}$	$\frac{311}{54}$	$-\frac{19}{60}$	$\frac{17}{6}$	$-\frac{1}{12}$		
11	0	$\frac{41}{840}$	1	$\frac{2383}{4100}$	0	0	$-\frac{314}{164}$	$\frac{4496}{1025}$	$-\frac{301}{82}$	$\frac{2133}{4100}$	$\frac{45}{82}$	$\frac{45}{164}$	$\frac{18}{41}$	
12	$\frac{41}{840}$		0	$\frac{3}{205}$	0	0	0	0	$-\frac{6}{41}$	$-\frac{3}{205}$	$-\frac{3}{41}$	$\frac{3}{41}$	$\frac{6}{41}$	0
13	$\frac{41}{840}$		1	$-\frac{1777}{4100}$	0	0	$-\frac{341}{164}$	$\frac{4496}{1025}$	$-\frac{289}{82}$	$\frac{2193}{4100}$	$\frac{51}{82}$	$\frac{33}{164}$	$\frac{12}{41}$	0	1

Verner - embedding formulas of 8^{th} and 9^{th} order.
($q_g = 8, \tilde{q}_g = 9, m = 14, \tilde{m} = 16$)

Coefficient scheme of the embedding formula rkv9(8):

j	$\tilde{A}_j$	A_j	a_j	b_{js} for $s = 1(1)m-1$												
1	$\frac{33}{525}$	$\frac{103}{1680}$	0													
2	0	0	$\frac{1}{12}$	$\frac{1}{12}$												
3	0	0	$\frac{1}{9}$	$\frac{1}{27}$	$\frac{2}{27}$											
4	0	0	$\frac{1}{6}$	$\frac{1}{24}$	0	$\frac{1}{8}$										
5	0	0	$\frac{2,2}{15}$	$\frac{4,04}{375}$	0	$\frac{-94,-84}{125}$	$\frac{328,208}{375}$									
6	0	0	$\frac{6,1}{15}$	$\frac{9,-1}{150}$	0	0	$\frac{312,32}{1425}$	$\frac{69,29}{570}$								
7	0	0	$\frac{6,-1}{15}$	$\frac{927,-347}{1250}$	0	0	$\frac{-16248,7328}{9375}$	$\frac{-489,179}{3750}$	$\frac{14268,-3798}{9375}$							
8	$\frac{171}{1400}$	$-\frac{27}{140}$	$\frac{2}{3}$	$\frac{2}{27}$	0	0	0	0	$\frac{16,-1}{54}$	$\frac{16,1}{54}$						
9	$\frac{86}{525}$	$\frac{76}{105}$	$\frac{1}{2}$	$\frac{19}{256}$	0	0	0	0	$\frac{118,-23}{512}$	$\frac{118,23}{512}$	$\frac{-9}{256}$					
10	$\frac{93}{280}$	$-\frac{201}{280}$	$\frac{1}{3}$	$\frac{11}{144}$	0	0	0	0	$\frac{266,-1}{864}$	$\frac{266,1}{864}$	$-\frac{1}{16}$	$-\frac{8}{27}$				
11	$-\frac{2048}{6825}$	$\frac{1024}{1365}$	$\frac{1}{4}$	$\frac{5034,-271}{61440}$	0	0	0	0	0	$\frac{7859,-1626}{10240}$	$\frac{-2232,813}{20480}$	$\frac{-594,271}{960}$	$\frac{657,-813}{5120}$			
12	$-\frac{3}{18200}$	$\frac{3}{7280}$	$\frac{4}{3}$	$\frac{5996,-3794}{405}$	0	0	0	0	$\frac{-4342,-338}{9}$	$\frac{154922,-40458}{135}$	$\frac{-4176,3794}{45}$	$\frac{-340864,242816}{405}$	$\frac{26304,-15176}{45}$	$-\frac{26624}{81}$		
13	$\frac{39}{175}$	$\frac{12}{35}$	$\frac{5}{6}$	$\frac{3793,2168}{103680}$	0	0	0	0	$\frac{4042,2263}{13824}$	$\frac{-231278,40717}{69120}$	$\frac{7947,-2168}{11520}$	$\frac{1048,-542}{405}$	$\frac{-1383,542}{720}$	$\frac{2624}{1053}$	$\frac{3}{1664}$	
14	0	$\frac{9}{280}$	1	$-\frac{137}{1296}$	0	0	0	0	$\frac{5642,-337}{864}$	$\frac{5642,337}{864}$	$-\frac{299}{48}$	$\frac{184}{81}$	$-\frac{44}{9}$	$-\frac{5120}{1053}$	$-\frac{11}{468}$	$\frac{16}{9}$
15	$\frac{9}{25}$		$\frac{1}{6}$	$\frac{33617,-2168}{518400}$	0	0	0	0	$\frac{-3846,31}{13824}$	$\frac{155338,-52807}{345600}$	$\frac{-12537,2168}{57600}$	$\frac{92,542}{2025}$	$\frac{-1797,-542}{3600}$	$\frac{320}{567}$	$-\frac{1}{1920}$	$\frac{4}{105}$ 0
16	$\frac{233}{4200}$		1	$\frac{-34457,-30352}{279600}$	0	0	0	0	$\frac{-29666,-4499}{7456}$	$\frac{2779182,-615973}{186400}$	$\frac{-94329,91056}{93200}$	$\frac{-232192,121408}{17475}$	$\frac{101226,-22764}{5825}$	$-\frac{169984}{9087}$	$-\frac{87}{30290}$	$\frac{492}{1165}$ 0 $\frac{1260}{233}$

$$\frac{a,b}{c} \to \frac{a+b\sqrt{6}}{c}$$

Each of the formulas of such a compatible embedding pair supplies an approximate value Y and $\tilde{Y}$ for y at $x_i + h$. Since the two formulas of one pair have different error orders and since we have to calculate only one set of k_j values for both formulas, embedding formulas are well suited for step size control and thus for adaptive procedures. Section 17.3.7.2, and algorithms 17.12 and 17.13 indicate procedures with an automatic step size control which relies on the difference $Y - \tilde{Y}$ from the two approximations for y of a compatible pair.

The following expressions for the difference $Y - \tilde{Y}$ can be derived from the embedding formulas (17.7) to (17.10):

For the Runge-Kutta embedding formula of 2^{nd} and 3^{rd} order we have

$$(17.7') \qquad Y - \tilde{Y} = h\left\{-\tfrac{1}{6}k_1 + \tfrac{1}{3}k_2 - \tfrac{1}{6}k_3\right\} ;$$

for the Runge-Kutta-Fehlberg embedding formula of 4^{th} and 5^{th} order we have

$$(17.8') \qquad Y - \tilde{Y} = h\left\{-\tfrac{1}{360}k_1 + \tfrac{128}{4275}k_3 + \tfrac{2197}{75240}k_4 - \tfrac{1}{50}k_5 - \tfrac{2}{55}k_6\right\};$$

for the England embedding formula of 4^{th} and 5^{th} order we have

$$(17.9') \qquad Y - \tilde{Y} = h\left\{\tfrac{42}{336}k_1 + \tfrac{224}{336}k_3 + \tfrac{21}{336}k_4 - \tfrac{162}{336}k_5 - \tfrac{125}{336}k_6\right\} ;$$

while for the Fehlberg I/II-embedding formula of 5^{th} and 6^{th} order we obtain the especially simple formula

$$(17.10') \qquad Y - \tilde{Y} = \tfrac{5}{66}h\left\{k_1 + k_6 - k_7 - k_8\right\} .$$

Bibliography for Section 17.3.4

[DORM80]; [DORM81]; [FEHL60]; [FEHL66]; [FEHL69]; [FEHL70]; [FEHL75]; [HAIR87] 2.6; [HULL72]; [LUTH87]; [VERN78].

17.3.5 Implicit Runge-Kutta Methods of Gaussian Type

With an *explicit* Runge-Kutta method for the initial value problem (17.1) and the partition (17.3), one can obtain a local error order of maximally $q_l = m+1$ if $m \leq 4$, and of at most $q_l = m$, if $m > 4$. Such explicit Runge-Kutta methods use m functional evaluations f_j, $j = 1, \ldots, m$, per step.

With an *implicit* Runge-Kutta method one can obtain a local error order of $q_l = 2m+1$ from m functional evaluations per step if the grid points $x_i + a_j h_i$ are identical with the nodes of the Gaussian quadrature formulas for the interval $[x_i, x_{i+1}]$, see section 15.7. Implicit Runge-Kutta formulas use (17.6) for $s = 1, \ldots, m$ instead of $s = 1, \ldots, j-1$ for the explicit formulas. In the following table, we describe implicit Runge-Kutta formulas of the Gaussian type for $m = 1, 2, 3$:

TABLE 17.7 (*Implicit Runge-Kutta formulas of Gaussian type*).

m		q_l
1	$Y_{i+1} = Y_i + h_i k_1(x_i, Y_i, h_i)$ with $k_1 = f(x_i + h_i/2, Y_i + k_1/2)$	3
2	$Y_{i+1} = Y_i + (h_i/2)(k_1 + k_2)$ with $k_1 = f(x_i + (1/2)(1 - 1/\sqrt{3})h_i,$ $\quad Y_i + (1/4)k_1 + (1/2)(1/2 - 1/\sqrt{3})k_2),$ $k_2 = f(x_i + (1/2)(1 + 1/\sqrt{3})h_i,$ $\quad Y_i + (1/2)(1/2 + 1/\sqrt{3})k_1 + (1/4)k_2).$	5
3	$Y_{i+1} = Y_i + h_i((5/18)k_1 + (4/9)k_2 + (5/18)k_3)$ with $k_1 = f(x_i + (1/2)(1 - \sqrt{3/5})h_i,\ Y_i + (5/36)k_1 +$ $\quad (2/9 - 1/\sqrt{15})k_2 + (5/36 - 1/(2\sqrt{15}))k_3),$ $k_2 = f(x_i + h_i/2,\ Y_i + (5/36 + \sqrt{15/24})k_1 +$ $\quad (2/9)k_2 + (5/36 - \sqrt{15/24})k_3),$ $k_3 = f(x_i + (1/2)(1 + \sqrt{3/5})h_i,\ Y_i + (5/36 +$ $\quad 1/(2\sqrt{15}))k_1 + (2/9 + 1/\sqrt{15})k_2 + (5/36)k_3).$	7

For all $2 \le m \le 20$, [GLAS66] gives tables for the coefficients A_j, a_j, b_{js}. The systems of equations given above for the k_j are nonlinear and must be solved iteratively. Similar systems result for $m > 3$.

The iterative solution of such nonlinear system is demonstrated here for $m = 2$. For this purpose, we use an upper index on k_j as the iteration index. The initial values are

$$k_1^{(0)} = k_2^{(0)} = f(x_i, Y_i).$$

The *iteration rule* is given by:

$$(17.11)\quad \begin{cases} k_1^{(\nu+1)} = f\left(x_i + \frac{1}{2}\left(1 - \frac{1}{\sqrt{3}}\right)h_i,\ Y_i + \frac{1}{4}k_1^{(\nu)} + \frac{1}{2}\left(\frac{1}{2} - \frac{1}{\sqrt{3}}\right)k_2^{(\nu)}\right), \\ k_2^{(\nu+1)} = f\left(x_i + \frac{1}{2}\left(1 + \frac{1}{\sqrt{3}}\right)h_i,\ Y_i + \frac{1}{2}\left(\frac{1}{2} + \frac{1}{\sqrt{3}}\right)k_1^{(\nu)} + \frac{1}{4}k_2^{(\nu)}\right), \\ \qquad\qquad \nu = 1, 2, \ldots \end{cases}$$

The iteration will converge for any initial values $k_1^{(0)}$, $k_2^{(0)}$, see [GRIG77], p. 40, and [SOMM67], p. 31, provided h_i is chosen so that

$$(17.12)\quad \max_{1\le j\le m} h_i L_i \sum_{s=1}^{m} |b_{js}| < 1, \text{ where } L_i = \max_{x\in[x_i, x_{i+1}]} \left|\frac{\partial f_r}{\partial y_k}\right|,\ 1 \le r, k \le n.$$

To obtain a local error of order $O(h_i^{2m+1}), 2m-1$ iteration steps are needed. The step size h_i satisfying (17.12) and the number m of functional evaluations per integration step can be chosen at will. As is shown in [SOMM67], one can minimize the needed computational time $T(\epsilon, m)$ for a given error threshold ϵ by a proper choice of m. If one knows the optimal m, the step size $h_i = x_{i+1} - x_i = h_i(\epsilon, m)$ can be found for each integration step.

Corresponding formulas and coefficients for implicit Runge-Kutta methods in which the m arguments $x_i + \alpha_j h_i$ coincide with the nodes of other quadrature formulas such as Newton-Cotes, Maclaurin, etc., can be found in [SOMM67]. Another method of step size control which is based upon two different quadrature formulas is given in [GRIG77], p. 69/70.

Bibliography for Section 17.3.5

[GLAS66]; [GRIG77]; [HAIR87] 2.7; [SOMM67].

17.3.6 Consistence and Convergence of One-Step Methods

When we consider the algorithms of one-step methods for solving the initial value problem (17.1) on a grid (17.3), we note that each procedure can be described by a recursion of the following form:

$$(17.13)\quad Y_{i+1} = Y_i + h_i \Phi(x_i, Y_i, h_i), \quad i = 0, \ldots, N-1.$$

Each one-step method is uniquely determined by the associated function Φ. For example for the improved Euler-Cauchy method this function is

$$\Phi(x_i, Y_i, h_i) = f(x_i + \frac{h_i}{2}, Y_i + h_i/2\ f(x_i, Y_i)).$$

And the one for the m^{th} order Runge-Kutta method is:

$$\Phi(x_i, Y_i, h_i) = \sum_{j=1}^{m} A_j k_j(x_i, Y_i, h_i)$$

with the A_j and k_j from the table in section 17.3.4 .

DEFINITION 17.8 (*Local discretization error, truncation error*).
For the defining function $\Phi(x, Y, h)$ of a one-step method using (17.4) and (17.13), we define the *local discretization error* at the grid point x_i as:

$$\tau_i := \frac{1}{h_i}(y(x_{i+1}) - y(x_i)) - \Phi(x_i, y(x_i), h_i).$$

τ_i is also called the *truncation error* at x_i. Here $y(x)$ is the exact solution of the intial value problem $y'(x) = f(x, y)$, $y(x_0) = y_0$.

DEFINITION 17.9 (*Consistency*).
A one-step method is called *consistent*, if the weighted sum of the local discretization errors at all grid points x_i, $i = 0, \ldots, N$,

$$\sum_{i=0}^{N-1} h_i \|\tau_i\|$$

converges to 0 as $h_{\max} \to 0$ with $h_{\max} = \max_{0\le i\le N-1} h_i$ and $h_i := x_{i+1} - x_i > 0$.

A one-step method is consistent if the maximum local discretization error satisfies

$$\max_{0\le i\le N-1}\{\|\tau_i\|\}\to 0 \quad \text{as} \quad h_{\max}\to 0.$$

Thus, all the one-step methods of the previous sections are consistent, provided the solution y of (17.1) is sufficiently often continuously differentiable.

DEFINITION 17.10 (*Consistency order*) .
The order $O(h_{max}^q)$, with which the truncation error or local discretization error tends towards zero, is called the *consistency order* of the method.

The consistency order is equal to the global error order. To achieve the consistency order, the exact solution y must be sufficiently often continuously differentiable. To achieve consistency, y must be twice continuously differentiable.

THEOREM 17.11 (*Convergence*).
A consistent one-step method with a consistency order $q > 0$ whose associated function Φ fulfills a Lipschitz condition relative to y is *convergent* of order q. Therefore

$$\lim_{h_{\max}\to 0}\|Y_i - y(x_i)\| = 0.$$

For a proof see [STOE91], 7.2.2; [WERN79] .

17.3.7 Error Estimation and Step Size Control

17.3.7.1 Error Estimation

If $Y_h(x)$ and $Y_{\check h}(x)$ are approximate values for y at a point $x \in [x_0, \beta]$ computed for step sizes h and $\check h$ using a method with global error order q_g, then we can estimate the global procedural error as

$$e_h := y(x) - Y_h(x) \approx \frac{Y_h(x) - Y_{\check h}(x)}{(\check h/h)^{q_g} - 1} = e_h^* \tag{17.14}$$

And

$$Y_h^*(x) = Y_h(x) + e_h^* = \frac{(\check h/h)^{q_g} Y_h(x) - Y_{\check h}(x)}{(\check h/h)^{q_g} - 1}$$

gives an improved approximate value for the exact solution $y(x)$ when compared with $Y_h(x)$. For sufficiently often differentiable solutions y,

$$y(x) = Y_h^*(x) + O(h^{q_g+1}).$$

The global error order is increased by at least one by adding the estimated error to the approximate value, see [STUM82], p.253.

For $\check h = 2h$ we have

$$e_h(x) \approx \frac{Y_h(x) - Y_{2h}(x)}{2^{q_g} - 1} = e_h^*,$$
$$Y_h^*(x) = \frac{2^{q_g} Y_h(x) - Y_{2h}(x)}{2^{q_g} - 1},$$

where Y_h denotes the approximation for $y(x)$ for the step size h and Y_{2h} the one for the step size $2h$. And the improved approximate value $Y_h^*(x)$ is better by at least one h-power than $Y_h(x)$.

For example, we can obtain the following estimation formulas and improved approximate values for specific one-step procedures with $\check h = 2h$.

1. Euler-Cauchy method:

$$e_h^{EC}(x) \approx Y_h^{EC}(x) - Y_{2h}^{EC}(x)$$
$$Y_h^*(x) = 2Y_h^{EC}(x) - Y_{2h}^{EC}(x).$$

2. Heun's method and improved Euler-Cauchy method:

$$e_h^{H}(x) \approx \tfrac{1}{3}\left(Y_h^{H}(x) - Y_{2h}^{H}(x)\right)$$
$$Y_h^{*H}(x) = \tfrac{1}{3}\left(4Y_h^{H}(x) - Y_{2h}^{H}(x)\right).$$

3. Classical Runge-Kutta method:

$$e_h^{RK}(x) \approx \tfrac{1}{15}\left(Y_h^{RK}(x) - Y_{2h}^{RK}(x)\right)$$
$$Y_h^{*RK}(x) = \tfrac{1}{15}\left(16Y_h^{RK}(x) - Y_{2h}^{RK}(x)\right).$$

17.3.7.2 Automatic Step Size Control, Adaptive Methods for Initial Value Problems

Generally, it is not appropriate to calculate with a constant step size, i.e., with an equidistant grid. We recommend to adapt the chosen step size locally to the behavior of the solution, for examples see [LUTH87], 4.3. In regions of little change in the solution, one can choose relatively big steps, in regions of large changes of the solution one should proceed with relatively small steps. It is possible to control the step size automatically by using error estimation. For this we shall describe several possibilities and procedures below.

Method A) to control the step sizes.

By using the error estimate (17.14), one can proceed as follows: after every two steps with a step size h from x_{i-1} to x_i and x_{i+1}, one carries out a step with the double step size $2h$ from x_{i-1} using the same method. If the estimated error is far below the given error bound one can increase the step size for the next step. If the estimated error is larger, one integrates over the last half once more with a smaller step size. If the step size has been chosen correctly one can use the error estimate in (17.14) to improve the approximate value, one then continues with the improved value.

Method B) to control the step size.

The following method of automatic step size control is still more effective: One uses two one-step methods with associated functions Φ and $\tilde{\Phi}$, one of which has the global convergence order q_g, and the other at least the order q_g+1. One calculates the approximate values Y and $\tilde{Y}$ at the point $x = x_i+h$ with both methods, starting from an approximate value Y_i for the grid point x_i and a step size h. Depending on the outcome of the error estimation, one can accept the chosen step size h and accept $x_{i+1} := x$ as a new grid point, or one must repeat the integration with a smaller step size. Thus, one works adaptively, as in the following algorithm, see [LUTH87], 4.3.

ALGORITHM 17.12 (*Automatic step size control, adaptive initial value solver*).

Choose two one-step methods with associated functions Φ and $\tilde{\Phi}$ with error orders q_g and q_g+1 at least. Let Y_i be an approximate value for the exact solution y at the grid point x_i . Then proceed with a chosen step size h as follows:

1. Calculate an approximate solution Y with the first method and an approximate solution $\tilde{Y}$ with the second method for $x_i + h$:

$$Y = Y_i + h\Phi(x_i, Y_i, h), \quad \tilde{Y} = Y_i + h\tilde{\Phi}(x_i, Y_i, h).$$

For a given error bound $\varepsilon > 0$ set

$$S := \left(\frac{h\varepsilon}{\|Y - \tilde{Y}\|}\right)^{1/q_g}.$$

2. If $S \geq 1$ then $Y_{i+1} := \tilde{Y}$ is accepted as a new approximation at the grid point $x_{i+1} := x_i + h$. For the next step, carried out as in 1, choose the new step size as

$$h := \min\{2; S\} \cdot h.$$

If $S < 1$, one has to repeat the first step using the step size

$$h := \max\{\frac{1}{2}; S\} \cdot h.$$

ALGORITHM 17.13 (*Automatic step size control according to* it [HULL72]).

Choose two one-step methods of orders q_g and $\tilde{q}_g \geq q_g + 1$. Let $\tilde{Y}_i$ be an approximate value for the exact solution y at the grid point x_i.

1. Compute approximate solutions Y and $\tilde{Y}$ at $x_i + h$ by using the two one-step methods.
2. Compute

$$S := 0.9h\left(\frac{\varepsilon}{\|Y - \tilde{Y}\|}\right)^{1/(q_g+1)},$$

where $\varepsilon = \|\tilde{Y}\|$ RELERR + ABSERR.

3. If $\|Y - \tilde{Y}\| < \varepsilon$, then $Y_{i+1} := Y$ is accepted as a new approximation for the grid point $x_{i+1} := x_i + h$. The next step uses the step size

$$h := \min\{S; 4h\}.$$

If $\|Y - \tilde{Y}\| > \varepsilon$, the first step has to be repeated with the new step size

$$h := \max\{S; \frac{1}{4}h\}.$$

REMARK concerning the error estimates when using embedding formulas:
Section 17.3.4.4 described embedding formulas. They are especially suited for adaptive methods using step size control as done in algorithm 17.12 or 17.13, because the approximate value $\tilde{Y}$ can be found with very little computational effort once the approximate value Y has been computed: All the k_j-values necessary for computing Y can be used for finding $\tilde{Y}$.
As examples, in section 17.3.4.4 we have given the Runge-Kutta formulas of 2^{nd} and 3^{rd} order, the Runge-Kutta-Fehlberg formulas of 4^{th} and 5^{th} order, the England formulas of 4^{th} and 5^{th} order and the Fehlberg formulas of 5^{th} and 6^{th} order. The differences $Y - \tilde{Y}$ that are needed for algorithms 17.12 or 17.13 are given in the formulas (17.7') up to (17.10').

REMARK: In the program section of this book we include a program IVP which works adaptively using automatic step size control, giving the user the choice between the Runge-Kutta embedding formulas of second and third order, or the England formulas of fourth and fifth order. If for an initial value problem (17.1) one wants to calculate approximate values for the solution $y(x)$ at the points $x_k = x_0 + kh$ for $k = 1, \ldots, k_{end}$, it is useful to call the program IVP in a loop in such a way that the solution is calculated at the point x_{k+1}, i.e., in each loop the initial values x_k and $Y(x_k)$ are used to compute $Y(x_{k+1})$.

When integrating (17.1) from x_k to x_{k+1}, one can use a step size control according to algorithm 17.12. The mixed error test (1.6) is to be used and the error bound ε should be set as

$$\varepsilon = \text{ABSERR} + \text{RELERR}\,\|\tilde{Y}\|,$$

so that the S of algorithm 17.12 has the form

$$S = \left(h_k \frac{\text{ABSERR} + \text{RELERR}\,\|\tilde{Y}\|}{\|Y - \tilde{Y}\|} \right)^{1/q}.$$

REMARK concerning the choice of a suitable embedding formula:

Naturally, it is possible to use any other pair of embedding formulas of section 17.3.4.4 adaptively. Section 17.8 will deal with those as well as give test results on algorithms 17.12 and 17.13 for automatic step size control and decision hints.

Bibliography for Section 17.3

[BJÖR74], 8.1-8.3; [COLL66], II, §2; [CONT80], 6; [ENGE87], 10.3, 11.1; [GEAR71/1], 2; [GRIG77], vol.1; [HAIR87] 2; [HENR68], part 1; [LAPI71], 2,3; [LUTH87], 4.1-4.3; [NOBL65], II, 10.2-10.5; [RALS79] vol.1, 9; [RICE77], p.257-276; [SCHW89], 9.1; [STET73], 3; [STUM82], 11; [WERN79], IV, §6,7; [ZURM65], §25, 27 .

17.4 Multi-Step Methods

17.4.1 The Principle of Multi-Step Methods

Multi-step methods use $s+1$ preceding values $Y_{i-s}, Y_{i-s+1}, \ldots, Y_{i-1}, Y_i$ and calculate an approximate value Y_{i+1} for $y(x_{i+1})$ on a given grid.

One considers the initial value problem

$$(17.15)\quad \begin{cases} y'(x) = f(x,y) = f(x,y(x)) = f(x,y_1,y_2,\ldots,y_n) \text{ for } x \in [x_{-s},\beta] \\ y(x_{-s}) = y_{-s}. \end{cases}$$

On the interval $[x_{-s},\beta]$ of integration for the differential equation we define a partition

$$x_{-s} < x_{-s+1} < \ldots < x_{N-s} = \beta$$

with local step sizes $h_i := x_{i+1} - x_i > 0$ for $i = -s, \ldots, N-s$, where $N > s$.

Initially, we assume that the values of y and those of $f(x,y)$ are known at the points $x_{-s}, x_{-s+1}, \ldots, x_{-1}, x_0$. The points $(x_i, f(x_i,y_i))$ for $i = -s, \ldots, 0$ form the *starting values* for computing the approximate values $Y_i = Y(x_i)$ for $y_i = y(x_i), i = 1, \ldots, N-s$, at the remaining $N-s$ grid points $x_1, x_2, \ldots, x_{N-s}$. The starting values of y are either given (exactly or approximately) or they must be calculated approximately by means of a one-step method (e.g. by the classical Runge-Kutta method). In the following we denote the starting values by $(x_i, f(x_i,y_i)) = (x_i, f_i)$ for $i = -s, \ldots, 0$.

We will proceed from the integral equation (17.4) associated with the differential equation (17.15) in $[x_i, x_{i+1}]$. With one class of multi-step methods, the function f in (17.4) is replaced by the interpolating polynomial Φ_s of degree s through the $s+1$ interpolation points $(x_j, f_j), j = i-s, \ldots, i$, and Φ_s is integrated over the interval $[x_i, x_{i+1}]$. Thus one can obtain an approximate value Y_{i+1} for y_{i+1}. If $i = 0$, these interpolation points are identical with the starting points, for $i > 0$ some starting points and points (x_j, f_j) for $j = 1, \ldots, i$ jointly determine the approximate value Y_{i+1} using the computed approximations $Y_1, Y_2, \ldots, Y_i$. Since the right hand side of (17.4) contains

only the values Y_{i-s} to Y_i, this yields an *explicit formula* for calculating the approximate value Y_{i+1}, and the integration step is an extrapolation step.

Analogously we can obtain an *implicit formula* if we use the node x_{i+1} in the interpolation polynomial for f together with the nodes $x_{i-s}, x_{i-s+1}, \ldots, x_i$. Then the right hand side of (17.4) also contains Y_{i+1}, besides $Y_{i-s}, Y_{i-s+1}, \ldots, Y_i$. A formula of this type is the corrector formula of Heun's method.

If one uses an explicit and an implicit formula as a pair, the explicit formula is called a *predictor*, and the implicit one a *corrector*, while the procedure is called a *predictor-corrector method*.

References: For multi-step methods we recommend particularly the book of Shampine and Gordon, see [SHAM75]. It emphasizes the Adams methods and gives very efficient algorithms and FORTRAN programs.

17.4.2 The Adams-Bashforth Method

The *Adams-Bashforth method* results from (17.4) by replacing $f(x, y(x))$ by its interpolation polynomial $\Phi_s(x)$ and the associated remainder $R_{s+1}(x)$ at the $s+1$ interpolation points (x_j, f_j), $j = (i-s), \ldots, i$. Integrating from x_i to x_{i+1} yields

$$y_{i+1} = Y_{i+1} + \epsilon_{i+1}^{AB} \quad \text{with} \quad Y_{i+1} = Y_i + \int_{x_i}^{x_{i+1}} \Phi_s(x)dx \quad \text{with}$$

$$\epsilon_{i+1}^{AB} := y_{i+1} - Y_{i+1} = \int_{x_i}^{x_{i+1}} R_{s+1}(x)dx.$$

ϵ_{i+1}^{AB} is the local procedural error which results from integrating over $[x_i, x_{i+1}]$, assuming that Y_i is exact.

Thus for every s and given points (x_j, f_j), $j = i-s, \ldots, i$, one obtains an Adams-Bashforth formula that computes Y_{i+1} by integrating from x_i to x_{i+1} with an associated local procedural error of $\epsilon_{i+1}^{AB} = O(h^{q_\ell})$.
Next we give the Adams-Bashforth formulas for $s = 3, 4, 5, 6$ and equidistant grid points: With $h_i = h = \text{const}$ we have

$$\begin{aligned}
s=3\ (q_\ell=5): \quad Y_{i+1} &= Y_i + \tfrac{h}{24}(55f_i - 59f_{i-1} + 37f_{i-2} - 9f_{i-3}),\\
&\quad i = 0, \ldots, n-4,\\
\epsilon_{i+1}^{AB} &= \tfrac{251}{720}h^5 y^{(5)}(\eta_i) = O(h^5),\ \eta_i \in [x_i, x_{i+1}];\\
s=4\ (q_\ell=6): \quad Y_{i+1} &= Y_i + \tfrac{h}{720}(1901f_i - 2774f_{i-1} + 2616f_{i-2} +\\
&\quad -1274f_{i-3} + 251f_{i-4}),\\
\epsilon_{i+1}^{AB} &= \tfrac{95}{288}h^6 y^{(6)}(\eta_i) = O(h^6),\ \eta_i \in [x_i, x_{i+1}],\\
&\quad i = 0, \ldots, n-5;\\
s=5\ (q_\ell=7): \quad Y_{i+1} &= Y_i + \tfrac{h}{1440}(4277f_i - 7923f_{i-1} + 9982f_{i-2} +\\
&\quad -7298f_{i-3} + 2877f_{i-4} - 475f_{i-5}),\\
\epsilon_{i+1}^{AB} &= \tfrac{19087}{60480}h^7 y^{(7)}(\eta_i) = O(h^7),\ \eta_i \in [x_i, x_{i+1}],\\
&\quad i = 0, \ldots, n-6;\\
s=6\ (q_\ell=8): \quad Y_{i+1} &= Y_i + \tfrac{h}{60480}(198721f_i - 447288f_{i-1} +\\
&\quad +705549f_{i-2} - 688256f_{i-3} + 407139f_{i-4} +\\
&\quad -134472f_{i-5} + 19087f_{i-6}),\\
\epsilon_{i+1}^{AB} &= \tfrac{5257}{17280}h^8 y^{(8)}(\eta_i) = O(h^8),\ \eta_i \in [x_i, x_{i+1}],\\
&\quad i = 0, \ldots, n-7.
\end{aligned}$$

The global error order is $O(h^{q_g})$ with $q_g = q_\ell - 1$.

For the Adams-Bashforth formulas, one always needs $s+1$ given points (x_j, f_j), which have to be determined by another method. This method should have the same local error order. This could be achieved by a suitable Runge-Kutta method and would be a good reason to employ the Runge-Kutta method for the entire interval $[x_{-s}, \beta]$, instead of combining the Adams-Bashforth formula with the Runge-Kutta formula.
Since for an Adams-Bashforth step from x_i to x_{i+1}, one has to calculate only one new functional value f_i in contrast to m new functional values for a Runge-Kutta step of order m, the Adams-Bashforth formula works much faster than the Runge-Kutta formula. This would be a good reason to combine the Runge-Kutta method with the Adams-Bashforth formula.

However, the Adams-Bashforth formula should not be employed by itself, but rather as a predictor together with an implicit formula as a corrector. The reason for this is that in the Adams-Bashforth formula the interpolation interval is $[x_{i-s}, x_i]$ for Φ_s, while $[x_i, x_{i+1}]$ is the integration interval of Φ_s, so that the integration is evaluated in an extrapolation step. As is well known, the remainder R_{s+1} of an interpolation has large values for points outside of the interpolation interval, see section 9.6. We therefore have to be aware

that the local procedural error ε_{i+1}^{AB} will increase strongly for large h and will become bigger than the local procedural error of the Runge-Kutta method of the same error order. For error estimates, consult section 17.4.5.

One can construct other multi-step formulas by again replacing $f(x,y(x))$ in (17.4) with an interpolating polynomial Φ_s for $s+1$ interpolation points $(x_j,f_j), j = i-s,\ldots,i$, and by integrating over $[x_{i-r},x_{i+1}]$ for integers $0 \le r \le s$. The case $r = 0$ gives the above Adams-Bashforth formulas. For further such methods see [COLL66], p.86-88; [HENR68], p.199-201, 241; [SHAM75]; [STUM82], p.273-276; [WERN79], p.290-294.

17.4.3 The Predictor-Corrector Method of Adams-Moulton

This method combines an Adams-Bashforth extrapolation formula with an implicit corrector formula of at least the same error order. We recommend to choose a corrector formula with an error order one higher than that of the predictor formula. In this case we obtain a predictor-corrector method. One can obtain a corrector of higher error order if one replaces $f(x,y(x))$ in (17.4) by its interpolation polynomial for the $s+2$ interpolation points (x_j,f_j), $j = i-s,\ldots,i+1$, and then proceeds in an analogous way to section 17.4.2.

If $s = 3$, we obtain for an integration step from x_i to x_{i+1} and an equidistant partition:

$$\begin{aligned} y_{i+1} &= Y_{i+1} + \varepsilon_{i+1}^{AM_3} \quad \text{with} \\ Y_{i+1} &= Y_i + \tfrac{h}{720}(251f_{i+1} + 646f_i - 264f_{i-1} + 106f_{i-2} - 19f_{i-3}), \\ \varepsilon_{i+1}^{AM_3} &= -\tfrac{3}{160}h^6y^{(6)}(\eta_i) = O(h^6) \text{ for some } \eta_i \in [x_i,x_{i+1}]. \end{aligned}$$

Since $f_{i+1} = f(x_{i+1},Y_{i+1})$, the formula for Y_{i+1} is implicit so that Y_{i+1} must be determined iteratively. This iteration shall be labelled with an upper index ν. Then the *Adams-Moulton formula* for $s = 3$ becomes:

$$\begin{aligned} (17.16)\quad Y_{i+1}^{(\nu+1)} = Y_i \;+\; &\tfrac{h}{720}(251f(x_{i+1},Y_{i+1}^{(\nu)}) + 646f_i - 264f_{i-1} + \\ &+106f_{i-2} - 19f_{i-3}). \end{aligned}$$

It is employed as a corrector formula together with the Adams-Bashforth formula for $s = 3$ as a predictor. The corrector formula will converge if

$$\frac{251}{720}hL = \kappa < 1, \quad \text{where} \quad L = \max_{1\le k,r\le n}\left|\frac{\partial f_r}{\partial y_k}\right|.$$

If the step size h is sufficiently small, one or two iterations in (17.16) will suffice.

ALGORITHM 17.14 (*Predictor-corrector method of Adams-Moulton for* $s = 3$).

Given: The differential equation $y'(x) = f(x,y), x \in [x_{-3},\beta = x_{N-3}]$, with the initial condition $y(x_{-3}) = y_{-3}$, the step size $h > 0$, the nodes $x_i = x_0 + ih$, $i = -3,\ldots,N-3$, and the starting values (x_i,f_i), $i = -3,\ldots,0$.

Task: Compute approximations Y_i for $y(x_i), i = 1,\ldots,N-3$ by performing the following steps for each integration step from x_i to x_{i+1}:

1^{st} step: Calculate $Y_{i+1}^{(0)}$ from the Adams-Bashforth formula (predictor-formula with $q_\ell = 5$)

$$Y_{i+1}^{(0)} = Y_i + \tfrac{h}{24}(55f_i - 59f_{i-1} + 37f_{i-2} - 9f_{i-3}).$$

2^{nd} step: Calculate $f(x_{i+1},Y_{i+1}^{(0)})$.

3^{rd} step: Calculate $Y_{i+1}^{(\nu+1)}$ for $\nu = 0$ and $\nu = 1$ from the Adams-Moulton formula (17.16) (corrector formula with $q_\ell = 6$).

In general two iteration steps will suffice if h is chosen so that $K = hL \le 0.20$. Then one can accept

$$Y_{i+1}^{(\nu+1)} = Y_{i+1} \approx y_{i+1}.$$

If in the course of computations, it is necessary to decrease the step size before reaching the node x_j, it is generally recommended to halve h. In this case, one has to recalculate the initial values needed for the ensuing calculations for $i = j-2, j-\frac{3}{2}, j-1$ and $j-\frac{1}{2}$.

CALCULATION SCHEME 17.15 (*Adams-Moulton method for* $s = 3$ *and* $n = 1$).

	i	x_i	$Y_i = Y(x_i)$	$f_i = f(x_i, Y_i)$
Starting Values	-3 -2 -1 0	x_{-3} x_{-2} x_{-1} x_0	$Y_{-3} = y_{-3}$ Y_{-2} Y_{-1} Y_0	f_{-3} f_{-2} f_{-1} f_0
Extrapolation with Adams-Bashforth	1	x_1	$Y_1^{(0)}$	$f(x_1, Y_1^{(0)})$
Interpolation with Adams-Moulton	1 1	x_1 x_1	$Y_1^{(1)}$ $Y_1^{(2)} =: Y_1$	$f(x_1, Y_1^{(1)})$ $f(x_1, Y_1)$
Extrapolation with A.–B.	2	x_2	$Y_2^{(0)}$	$f(x_2, Y_2^{(0)})$
Interpolation with A.–M.	2 2	x_2 x_2	$Y_2^{(1)}$ $Y_2^{(2)} =: Y_2$	$f(x_2, Y_2^{(1)})$

Further Adams-Moulton methods.

In the following we shall indicate further Adams-Moulton methods in which the error order of the predictor is one less than that of the corrector. We shall use the abbreviation $f_{i+1}^{(\nu)} := f(x_{i+1}, Y_{i+1}^{(\nu)})$.

$$
\begin{aligned}
s = 4: \quad Y_{i+1}^{(0)} &= Y_i + \tfrac{h}{720}(1901 f_i - 2774 f_{i-1} + 2616 f_{i-2} + \\
&\quad -1274 f_{i-3} + 251 f_{i-4}), \\
Y_{i+1}^{(\nu+1)} &= Y_i + \tfrac{h}{1440}(475 f_{i+1}^{(\nu)} + 1427 f_i - 798 f_{i-1} + 482 f_{i-2} + \\
&\quad -173 f_{i-3} + 27 f_{i-4}), \\
\varepsilon_{i+1}^{AM_4} &= -\tfrac{863}{60480} h^7 y^{(7)}(\eta_i) = O(h^7), \ \eta_i \in [x_i, x_{i+1}], \\
&\quad i = 0, \ldots, n-4.
\end{aligned}
$$

$$
\begin{aligned}
s = 5: \quad Y_{i+1}^{(0)} &= Y_i + \tfrac{h}{1440}(4277 f_i - 7923 f_{i-1} + 9982 f_{i-2} + \\
&\quad -7298 f_{i-3} + 2877 f_{i-4} - 475 f_{i-5}), \\
Y_{i+1}^{(\nu+1)} &= Y_i + \tfrac{h}{60480}(19087 f_{i+1}^{(\nu)} + 65112 f_i - 46461 f_{i-1} + \\
&\quad +37504 f_{i-2} - 20211 f_{i-3} + 6312 f_{i-4} - 863 f_{i-5}), \\
\varepsilon_{i+1}^{AM_5} &= -\tfrac{275}{24192} h^8 y^{(8)}(\eta_i) = O(h^8), \ \eta_i \in [x_i, x_{i+1}], \\
&\quad i = 0, \ldots, n-5.
\end{aligned}
$$

$$
\begin{aligned}
s = 6: \quad Y_{i+1}^{(0)} &= Y_i + \tfrac{h}{60480}(198721 f_i - 447288 f_{i-1} + 705549 f_{i-2} + \\
&\quad -688256 f_{i-3} + 407139 f_{i-4} - 134472 f_{i-5} + \\
&\quad +19087 f_{i-6}), \\
Y_{i+1}^{(\nu+1)} &= Y_i + \tfrac{h}{120960}(36799 f_{i+1}^{(\nu)} + 139849 f_i - 121797 f_{i-1} + \\
&\quad +123133 f_{i-2} - 88547 f_{i-3} + 41499 f_{i-4} + \\
&\quad -11351 f_{i-5} + 1375 f_{i-6}), \\
\varepsilon_{i+1}^{AM_6} &= -\tfrac{33953}{3628800} h^9 y^{(9)}(\eta_i) = O(h^9), \ \eta_i \in [x_i, x_{i+1}], \\
&\quad i = 0, \ldots, n-6.
\end{aligned}
$$

Since the error order of the corrector is always one larger than that of the predictor, one or two iteration steps are sufficient in most cases. In general, a predictor-corrector method whose predictor part has the error order r_1 and whose corrector has the error order r_2, has the following local procedural error E_{i+1}^{PK} after $\nu + 1$ iteration steps:

$$E_{i+1}^{PK} := y_{i+1} - Y_{i+1}^{(\nu+1)} = O(h^{\min(r_2, r_1+\nu+1)}).$$

For $r_1 = r_2 - 1$ we thus attain the error order of the corrector after one iteration step. For an arbitrary $r_1 < r_2$, we can achieve the error order $O(h^{r_2})$ after $\nu = r_2 - r_1 - 1$ iteration steps. Since, however, the error of the predictor exceeds that of the corrector for $s \geq 3$ by a factor greater than 10, one or more iterations can be required in practice to reduce the total error to the error of the corrector. If one is, however, satisfied with obtaining an overall error order equal to the one for the corrector, then if $r_1 = r_2 - 1$ only one iteration is required. If $r_1 = r_2$, one must be satisfied with one iteration, see also [HENR68], p.196; [STUM82], p.271; [WERN79], p.299. If still more iterations are needed, it is better to decrease the step size than to continue iterating.

In the following we give an Adams-Moulton method whose predictor formula is the Adams-Bashforth formula for $s = 3$ and whose corrector is the Adams-Moulton formula for $s = 2$. They have the same local error order $O(h^5)$:

Predictor: ($Adams - Bashforth$ for $s = 3$)
$$Y_{i+1}^{(0)} = Y_i + \tfrac{h}{24}(55f_i - 59f_{i-1} + 37f_{i-2} - 9f_{i-3}),$$

Corrector: ($Adams - Moulton$ for $s = 2$)
$$Y_{i+1}^{(\nu+1)} = Y_i + \tfrac{h}{24}(9f_{i+1}^{(\nu)} + 19f_i - 5f_{i-1} + f_{i-2}).$$

This procedure requires only one iteration step for each grid point and thus it saves computing time. This predictor-corrector pair, has an especially simple error estimate, see also section 17.4.5, formulas (17.18), (17.19), so that without much calculation time and without requiring additional calculations with different step sizes, each value Y_i can be improved immediately.

Instead of the Adams-Moulton formulas as corrector, one can also use formulas with an especially advantageous error propagation. For this purpose, we define the corrector with $q_\ell = m + 3$ in the form

$$Y_{i+1} = \sum_{k=0}^{m} a_{i-k} Y_{i-k} + h \sum_{k=-1}^{m} b_{i-k} f(x_{i-k}, Y_{i-k}). \tag{17.17}$$

If e_{i+1}^F stands for the global procedural error of a formula (17.17), and e_{i+1}^{AM} for the same error for the Adams-Moulton formula of the same error order, then $\|e_{i+1}^F\|/\|e_{i+1}^{AM}\|$ is a measure for the quality of the corrector (17.17) with respect to error propagation. According to [FEHL61],

$$\begin{aligned} Y_{i+1}^{(\nu+1)} = {} & \tfrac{243}{1000} Y_i + \tfrac{1}{8} Y_{i-2} + \tfrac{79}{125} Y_{i-5} + \tfrac{h}{400}\Big(120 f(x_{i+1}, Y_{i+1}^{(\nu)}) + \\ & + 567 f(x_i, Y_i) + 600 f(x_{i-2}, Y_{i-2}) + 405 f(x_{i-4}, Y_{i-4}) + \\ & + 72 f(x_{i-5}, Y_{i-5})\Big) \end{aligned}$$

is a corrector with $q_\ell = 7$ for which $\|e_{i+1}^F\|/\|e_{i+1}^{AM}\|$ amounts to only about 8% of the global procedural error of the Adam-Moulton formula of the same error order. To be used as a predictor, one would need an extrapolation formula with $q_\ell = 6$. Hence one can use the Adams-Bashforth formula for $s = 4$. Due to the very small error terms in (17.17), we recommend to iterate more than twice.

17.4.4 The Adams-Störmer Method

In this section we describe a multi-step method that treats an initial value problem of the form

$$\begin{aligned} y''(x) &= g(x, y, y') \quad \text{with} \\ y(x_0) &= y_0, \quad y'(x_0) = y_0' \end{aligned}$$

directly without reducing it to an initial value problem (17.1).

ALGORITHM 17.16 (*Adams-Störmer method*).
Assume an initial value problem $y''(x) = g(x, y, y')$ is given with $y(x_{-3}) = y_{-3}$, $y'(x_{-3}) = y'_{-3}$. If $x_i = x_0 + ih, \quad i = -3, \ldots, N-3$, are the nodes of the integration interval $[x_{-3}, x_{N-3} = \beta]$, one carries out the following steps in order to calculate the approximate value Y_{i+1} for y_{i+1} for each $i = 1, \ldots, N-2$, after having calculated the starting values from the triples $(x_i, Y_i, Y_i'), i = -3, \ldots, 0$, possibly by using a Runge-Kutta method:

1^{st} step: Calculate the values $Y_{i+1}^{(0)}$, $Y_{i+1}^{\prime(0)}$ from the predictor formula of local error order $O(h^5)$:

$$Y_{i+1}^{(0)} = Y_i + hY_i' + \tfrac{h^2}{360}(323g_i - 264g_{i-1} + 159g_{i-2} - 38g_{i-3}),$$

$$Y_{i+1}^{\prime(0)} = Y_i' + \tfrac{h}{24}(55g_i - 59g_{i-1} + 37g_{i-2} - 9g_{i-3}),$$

with $g_i := g(x_i, Y_i, Y_i')$.

2^{nd} step: Calculate $g(x_{i+1}, Y_{i+1}^{(0)}, Y_{i+1}^{\prime(0)})$.

3^{rd} step: Calculate $Y_{i+1}^{(\nu+1)}$ and $Y_{i+1}^{\prime(\nu+1)}$ for $\nu = 0$ and $\nu = 1$ according to the corrector formulas ($q_\ell = 6$):

$$\begin{aligned} Y_{i+1}^{(\nu+1)} = Y_i + hY_i' + \tfrac{h^2}{1440}\Big(135 g(x_{i+1}, Y_{i+1}^{(\nu)}, Y_{i+1}^{\prime(\nu)}) + 752 g_i - 246 g_{i-1} + \\ + 96 g_{i-2} - 17 g_{i-3}\Big), \end{aligned}$$

$$\begin{aligned} Y_{i+1}^{\prime(\nu+1)} = Y_i' + \tfrac{h}{720}\Big(251 g(x_{i+1}, Y_{i+1}^{(\nu)}, Y_{i+1}^{\prime(\nu)}) + 646 g_i - 264 g_{i-1} + \\ + 106 g_{i-2} - 19 g_{i-3}\Big). \end{aligned}$$

NOTE. There is no easy rule to decide whether it is more advantageous to treat an initial value problem of a second or higher order differential equation by:

(1) using the Adams-Störmer method (direct procedure) directly or another direct method for differential equations of higher order, or

(2) reducing the problem to an initial value problem for a system of first order differential equations (indirect procedure).

According to [RUTI60], to proceed with (1) for problems with many integration steps can lead to a detrimental accumulation of rounding errors, see also [ENGE87], 11.4. Thus the approach (2) is generally preferred.

For high order differential equations, it has been shown that the direct methods that correspond to the classical Runge-Kutta method and the Adams-Moulton method achieve a smaller global error only if the derivative $y^{(n-1)}$ does not occur in f. For a problem of the form $y^{(n)} = f(x, y, y', \ldots, y^{(n-1)})$, the indirect methods generally have the smaller global error.

17.4.5 Error Estimates for Multi-Step Methods

The error estimates given in section 17.3.7 can also be used for multi-step methods. For instance, the estimates and improved approximations from (17.14) are for:

1) the Adams-Bashforth method for $s = 3$:

$$\begin{aligned} e_h^{AB}(x) &\approx \frac{1}{15}\left(Y_h(x) - Y_{2h}(x)\right), \\ Y_h^*(x) &= \frac{1}{15}\left(16Y_h(x) - Y_{2h}(x)\right); \end{aligned}$$

2) the Adams-Moulton method for $s = 3$:

$$\begin{aligned} e_h^{AM}(x) &\approx \frac{1}{31}\left(Y_h(x) - Y_{2h}(x)\right), \\ Y_h^*(x) &= \frac{1}{31}\left(32Y_h(x) - Y_{2h}(x)\right). \end{aligned}$$

If the Adams-Bashforth formula for $s = 3$ (with a local error order of $O(h^5)$) is used as a predictor, and the Adams-Moulton formula for $s = 2$ (with local error order of $O(h^5)$) is used as a corrector, then the following estimate for the global procedural error holds, see [CONT80] p.237:

$$\begin{aligned} q_\ell = 5: \quad e_h^{AM}(x) &:= y(x) - Y_h^{(1)}(x) \approx -\tfrac{1}{14}\left(Y_h^{(1)}(x) - Y_h^{(0)}(x)\right). \\ Y^*(x) &= Y_h^{(1)}(x) - \tfrac{1}{14}\left(Y_h^{(1)}(x) - Y_h^{(0)}(x)\right) \\ &= \tfrac{1}{14}\left(13Y_h^{(1)}(x) + Y_h^{(0)}(x)\right). \end{aligned}$$

This estimate is very simple to use because it does not require any calculations for a doubled step size. It also can be used to determine whether the chosen step size is sufficient for the desired precision.

Analogously, one can combine an Adams-Bashforth formula with an Adams-Moulton formula of equal error order for $q_\ell = 6, 7, 8$ to form a predictor-corrector pair. Then only one iteration step is required, and the following estimates for the global procedural error hold:

$$\begin{aligned} q_\ell = 6: \quad e_h^{AM}(x) &:= y(x) - Y_h^{(1)}(x) \approx -\tfrac{1}{18}\left(Y_h^{(1)}(x) - Y_h^{(0)}(x)\right), \\ q_\ell = 7: \quad e_h^{AM}(x) &:= y(x) - Y_h^{(1)}(x) \approx -\tfrac{1}{22}\left(Y_h^{(1)}(x) - Y_h^{(0)}(x)\right), \\ q_\ell = 8: \quad e_h^{AM}(x) &:= y(x) - Y_h^{(1)}(x) \approx -\tfrac{1}{26}\left(Y_h^{(1)}(x) - Y_h^{(0)}(x)\right). \end{aligned}$$

REMARK: An effective automatic step size control, like the one in section 17.3.7 for one-step methods, is also possible for multi-step methods by using much more involved procedures. Further information can be found in [SHAM75] with its quite elaborate FORTRAN programs.

17.4.6 Computational Error of One-Step and Multi-Step Methods

While the global procedural error of one-step and multi-step methods decreases with $h \to 0$ of order q_g, the global computational error increases with decreasing step size. The total error, i.e., the sum of procedural error and computational error, can thus not be reduced arbitrarily. Hence the step size h should be chosen so that procedural and computational errors have about the same order of magnitude.

If $r_h(x)$ denotes the global computational error at x, the following crude estimate ($n = 1$) is valid for one-step methods:

$$|r_h(x)| \le \begin{cases} \frac{\varepsilon}{h_{\max}}(x - x_0) & \text{if} \quad C = 0, \\ \frac{\varepsilon}{h_{\max}}\left(e^{C(x-x_0)} - 1\right) & \text{otherwise.} \end{cases}$$

Here ε is the maximum of the absolute computational error of each calculation step and $C = L$, the Lipschitz constant, in case of the Euler-Cauchy method and $C \approx L$ for the classical Runge-Kutta method.

For multi-step and predictor-corrector methods we have

$$|r_h(x)| \le \frac{\varepsilon}{h_{\max}} \cdot \frac{x_i - x_0}{1 - C_2 h_{\max} L} \cdot e^{\frac{C_1 L (x - x_0)}{1 - C_2 h_{\max} L}},$$

where C_1 and C_2 depend on the coefficients of the formulas used, see [HENR68], 5.3, 5.4. The global computational error is thus of order $O(1/h_{max})$ for both one-step and multi-step methods.

Bibliography for Section 17.4

[CARN69], 6.8-6.12; [COLL73], II, 4.2; [CONT80], 6.6-6.8, 6.11; [ENGE87], 10.4; [GEAR71/1], 7-10; [HAIR87] 3; [HENR64], 14.6-14.7; [HENR68], 5,6; [LAPI71], 4.7; [LUTH87], 4.4; [McCA67], 9.2; [RALS67] vol.1, 8.; [SCHW89], 9.2; [STET73], 4; [STOE91], 7.2.6 - 13; [STUM82], 12; [WERN79], IV, §§8-10.

17.5 Bulirsch-Stoer-Gragg Extrapolation

We consider the initial value problem (17.1). W.l.o.g. we can assume $n = 1$:

$$y'(x) = f(x, y), \qquad y(x_0) = y_0.$$

The extrapolation method of Bulirsch-Stoer-Gragg can easily be expanded to systems.

We want to find an approximation $Y(\bar{x})$ for the exact solution $y(\bar{x})$ of the initial value problem at $\bar{x}$ with

$$\bar{x} := x_0 + Nh \quad \text{for} \quad h := \frac{\bar{x} - x_0}{N} \quad \text{and} \quad N > 0.$$

Gragg's function $S(\bar{x}; h)$ supplies an approximate value for $y(\bar{x})$ with a global error order of $O(h^2)$. It is calculated according to [STOE91], 7.2.14 as follows:

With

$$(17.18)\quad \begin{cases} z_0 & := y_0 \\ z_1 & := z_0 + hf(x_0, y_0), \quad x_1 := x_0 + h \\ z_{i+1} & := z_{i-1} + 2hf(x_i, z_i), \quad x_{i+1} := x_i + h \text{ for } i = 1, \ldots, N-1, \end{cases}$$

S is defined as

$$(17.19)\qquad S(\bar{x}; h) := \frac{1}{2}[z_N + z_{N-1} + hf(x_N, z_N)].$$

In the extrapolation method of Bulirsch and Stoer, one chooses a sequence of positive integers

$$(17.20)\qquad \{n_0, n_1, n_2, \ldots\}$$

with $0 < n_0 < n_1 < \ldots$ and computes Gragg's function $S(\bar{x}; h_j)$ for each

$$(17.21)\qquad h_j := \frac{\bar{x} - x_0}{n_j}.$$

The numbers n_j must all be even or all be odd. The values $S(\bar{x}; h_j)$ are computed as in (17.18) and (17.19). One can obtain the value

$$S(\bar{x}; h_j) := \frac{1}{2}[z_{n_j} + z_{n_j - 1} + h_j f(x_{n_j}, z_{n_j})].$$

for each $j = 0, 1, \ldots$ with

$$\begin{array}{lll} z_0 & := y_0 & \\ z_1 & := z_0 + h_j f(x_0, y_0), & x_1 := x_0 + h_j \\ z_{i+1} & := z_{i-1} + 2h_j f(x_i, z_i), & x_{i+1} := x_i + h_j, \quad i = 1, \ldots, n_j - 1. \end{array}$$

Then $y(\bar{x}) = S(\bar{x}; h_j) + O(h_j^2)$.

Since S has an asymptotic expansion in powers of h_j^2, one can construct approximations with a higher error order by using Richardson extrapolation, just as in Romberg integration. For this one forms a "Romberg scheme" as follows, see section 14.10:

The numbers of its first column are given as

$$L_j^{(0)} := S(\bar{x}; h_j) \quad \text{for} \quad j = 0, 1, \ldots.$$

Then one calculates the values for the columns with $k = 1, 2, \ldots$ using the formula

$$(17.22)\qquad L_j^{(k)} = \frac{\left(\frac{h_j}{h_{j+k}}\right)^{2k} L_{j+1}^{(k-1)} - L_j^{(k-1)}}{\left(\frac{h_j}{h_{j+k}}\right)^2 - 1} \quad \text{for} \quad j = 0, 1, \ldots$$

The columns of this Romberg scheme converge towards $y(\bar{x})$ for all functions y that are sufficiently often differentiable:

$$\lim_{j\to\infty} L_j^{(k)} = y(\bar{x}) \quad \text{for} \quad k \quad \text{fixed;}$$

The convergence of the k^{th} column has the order $q_p = 2k+2$ for $k = 0, 1, 2, \ldots$.

NOTE: The number of columns used should be limited so that no oscillations occur. Oscillations can be triggered by the beginning influences of rounding errors as well as the possibility that f is not sufficiently smooth, i.e., not differentiable sufficiently often.

Romberg sequence.

If we choose $n_j = 2^j N$ for an even integer N, the sequence (17.20) becomes

$$N \cdot \{1, 2, 4, 8, 16, 32, \ldots\}.$$

This is called the *Romberg sequence*. For these n_j, (17.21) becomes

$$h_j := \frac{\bar{x} - x_0}{2^j N} = \frac{h}{2^j}. \tag{17.23}$$

The $L_j^{(k)}, k = 1, 2, \ldots$, are computed from (17.22) and (17.23) as

$$L_j^{(k)} = \frac{2^{2k} L_{j+1}^{(k-1)} - L_j^{(k-1)}}{2^{2k} - 1} \quad \text{for} \quad j = 0, 1, \ldots.$$

Bulirsch sequence.

If we choose the Bulirsch sequence $\{2, 4, 6, 8, 12, 16, \ldots\}$ in (17.20), we obtain for $j > 0$ and $h_0 := h$ with

$$h_j = \begin{cases} \dfrac{h_0}{2^{(j+1)/2}} & \text{for odd j} \\ \dfrac{h_0}{3 \cdot 2^{(j-2)/2}} & \text{for even j} \end{cases}$$

the h_j values as $\{h_0, h_1, \ldots\} = h \cdot \{1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{6}, \frac{1}{8}, \frac{1}{12}, \frac{1}{16}, \ldots\}$.

With these h_j, the computations in (17.22) are significantly simplified.

Step size control.

In extrapolation methods one should also utilize step size control. This can be done by means of one of the methods in section 17.3.7, see specifically the algorithms 17.12 and 17.13.

The program DESEXT in the program section uses the above extrapolation method with the Bulirsch sequence. Its step size control is realized with the method of [HALL76], p.113.

NOTE. Instead of the above method based upon polynomial extrapolation or the Richardson principle, see e.g. [BJÖR74], 7.22; [STUM82], p. 253; [WERN79] III, §7, one can also use algorithms based on rational extrapolation. Test examples for such procedures have given even better numerical results, see [BULI66], [GRAG65].

Bibliography for Section 17.5

[GEAR71/1], 6; [GRIG77], 5,2; [HAIR87] 2.9; [HALL76], 6; [LAPI71], 5; [STET73], 6.3; [STOE91], 7.2.14.

17.6 Stability

17.6.1 Preliminary Remarks

Integrating the initial value problem (17.1) numerically supplies approximate solutions $Y_i = Y(x_i)$ at the grid points $x_0 < x_1 < \ldots < x_n = \beta$ for the unknown exact solution $y_i = y(x_i)$. For all previously mentioned methods one can prove that the approximate values Y_i converge towards the exact values $y(x_i)$ as $h_{max} \to 0$ under the assumption that the associated function Φ satisfies a Lipschitz condition and that the calculations have no rounding errors.

However, it is necessary to examine a method in the presence of rounding errors as well. The only useful algorithms are so-called *stable algorithms*. According to section 1.4, an algorithm is *stable* if the error committed in one calculation step does not increase in the following steps. It is called *unstable* if, even for arbitrarily many steps, the approximations Y_i differ unboundedly from the exact values y_i, so that the computed solution is in essence useless.

The cause of an instability can lie within the differential equation itself or it can originate in the numerical procedure. In the first case, the instability is a result of the physical process described by the differential equation. In the second case, the instability can be avoided by chosing a more appropriate numerical method. All our investigations in this section will again only be made for $n = 1$, i.e., for an initial value problem $y'(x) = f(x, y)$ with $y(x_0) = y_0$.

17.6.2 Stability of Differential Equations

Let y be the solution of the initial value problem (17.1) with $n = 1$:

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

and u be a solution close to y that satisfies the same differential equation as y. However assume that the initial condition for u is slightly altered by rounding errors and procedural errors. Such errors can originate for example if the initial value y_0 of the problem has been calculated numerically.

We can express u as:

$$u(x) := y(x) + \varepsilon\eta(x).$$

Here η is a so-called *error function* and ε is a parameter with $0 < \varepsilon \ll 1$.

Then η satisfies the so-called *differential variational equation:*

$$\eta' = f_y\eta.$$

Under the assumption $f_y = c = \text{const}$, it has the solution

$$\eta(x) = \eta_0 e^{c(x-x_0)}, \qquad \eta(x_0) = \eta_0.$$

If $f_y = c < 0$, then $\eta(x)$ decreases for increasing x. In this case, the differential equation is called *stable*, otherwise it is *unstable*. For a stable differential equation, the distance of the solutions for different initial values diminishes for growing x, so that an error made at one point, such as a rounding error in the initial condition, will fade away.

17.6.3 Stability of the Numerical Method

For $n = 1$ one-step and multi-step methods have the general form

$$\sum_{k=0}^{M} a_{M-k} Y_{i+1-k} = h \sum_{k=0}^{M} b_{M-k} f_{i+1-k} \tag{17.24}$$

for an equidistant grid . They are explicit, if $b_M = 0$, and implicit for $b_M \neq 0$. Their coefficients must satisfy

$$\sum_{k=0}^{M} a_k = 0, \quad \sum_{k=0}^{M} b_k = 1.$$

For example for the Euler-Cauchy method and $M = 1$, we have seen:

$$a_0 = -1, \quad a_1 = 1, \quad b_0 = 1, \quad b_1 = 0,$$

while for Heun's method with $M = 2$:

$$a_0 = 0, \quad a_1 = -1, \quad a_2 = 1, \quad b_0 = 0, \quad b_1 = 1/2, \quad b_2 = 1/2.$$

An equation of the form (17.24) is called a *difference equation* of *order* M.

For multi-step methods we have $M > 1$, since several preceding values $Y_i, Y_{i-1}, \ldots$, are used to calculate Y_{i+1}. The Runge-Kutta methods are not multi-step procedures although $M = 3$, for the values Y_i, Y_{i-1} associated with b_1, b_2 are determined by $f(x_{i-2}, Y_{i-2})$. The reason for this is that in algorithm 17.4 the value Y_i corresponds to Y_{i-2}, the value $Y_i + \frac{h_i k_1}{2}$ to Y_{i-1}, and the value $Y_i + h_i k_2/2$ to Y_i.

The differential equation $y'(x) = f(x, y)$ can thus be replaced by a difference equation of the form (17.24), which gives rise to a *difference variational equation* with $U_i := Y_i + \varepsilon H_i$:

$$\sum_{k=0}^{M} a_{M-k} H_{i+1-k} = h \sum_{k=0}^{M} b_{M-k} H_{i+1-k} f_y(x_{i+1-k}, U_{i+1-k}). \tag{17.25}$$

Here U_i is another approximate solution close to the approximate solution Y_i, H_i is the error solution, and ε is a parameter with $0 < \varepsilon \ll 1$. To solve (17.25) we set

$$H(x_j) := H_j = \lambda^j \quad \text{for an integer } j.$$

(Strictly speaking, the following considerations hold only for linear differential equations with constant coefficients. For a more general theory see [DAHL74]).

Under the assumption $f_y = c = \text{const}$, we can insert the above expression for H_j into (17.25):

$$\sum_{k=0}^{M} a_{M-k}\lambda^{i+1-k} - hc\sum_{k=0}^{M} b_{M-k}\lambda^{i+1-k} = 0. \tag{17.26}$$

Clearly $\lambda \neq 0$, for if $\lambda = 0$ then $H = 0$, i.e., the approximate values Y_i would not be subject to any perturbations caused by rounding errors or truncation errors at preceding nodes. Multiplying (17.26) by λ^{M-1-i}, we obtain a polynomial equation for λ:

$$P(\lambda) := \sum_{k=0}^{M} a_{M-k}\lambda^{M-k} - hc\sum_{k=0}^{M} b_{M-k}\lambda^{M-k} = 0.$$

With

$$\begin{aligned} \varrho(\lambda) &:= \sum_{k=0}^{M} a_{M-k}\lambda^{M-k} = \sum_{k=0}^{M} a_k\lambda^k, \quad \text{and} \\ \sigma(\lambda) &:= \sum_{k=0}^{M} a_{M-k}\lambda^{M-k} = \sum_{k=0}^{M} b_k\lambda^k \end{aligned}$$

this can be written in the form

$$P(\lambda) = \varrho(\lambda) - hc\sigma(\lambda) = 0. \tag{17.27}$$

$P(\lambda) := \varrho(\lambda) - hc\sigma(\lambda)$ is called the *characteristic polynomial* of the multi-step method.

The M zeros of (17.27) are λ_ν, $\nu = 1, \ldots, M$. For each value of ν there is an *error solution* $(H_\nu)_j = (\lambda_\nu)^j$ of (17.25). The error solutions $(H_\nu)_j$ will not increase for increasing j precisely when $|\lambda_\nu| \leq 1$ for all ν.

If $|\lambda_\nu| < 1$ for all ν, we have *strong stability*. If there is an index ν with $|\lambda_\nu| = 1$, then λ_ν may only be a simple zero of (17.27) for asymptotic stability.

Since the equation (17.25) is linear in H_j, every linear combination of the functions $(H_\nu)_j$ is also a solution. If all λ_ν are different, then the M solutions $(H_\nu)_j = (\lambda_\nu)^j$, $\nu = 1, \ldots, M$, are linearly independent, and the general solution of (17.25) has the form

$$H_j = \sum_{\nu=1}^{M} c_\nu(\lambda_\nu)^j$$

with arbitrary real coefficients c_ν. The class of solutions of (17.25) thus has cardinality ∞^M, where M denotes the order of the difference equation.

One consequence of this is: If $M > 1$ (except for the Runge-Kutta methods), the order of the difference equation is higher than that of the differential equation. Thus the difference equation will have a larger solution space than that of the underlying differential equation. And the numerical method will produce "parasitic solutions". Only one of the solutions of the difference equation will converge for $h \to 0$ towards the solution of the initial value problem (17.1), provided that the multi-step procedure on which it is based does converge. The higher the order M of the difference equation is, the better its local error order $O(h^q)$. A difference equation (17.24) of M^{th} order can give rise to a numerically stable integration method with the local error order $O(h^{M+3})$ for even M, and with $O(h^{M+2})$ for odd M at most. Procedures with a higher error order must be numerically unstable ([WERN79],§9). In the following we shall only deal with local behavior, and locally we shall set $f_y = c = \text{const}$. We can distinguish the following types of stability:

- Asymptotic stability,
- Absolute stability ,
- A-stability, and
- Stiff stability .

DEFINITION 17.17 (*Asymptotic stability, stability for $h \to 0$*).
An algorithm for the numerical integration of an initial value problem (17.1) is called *asymptotically stable* for $h \to 0$, if the polynomial $\varrho(\lambda)$ fulfills the *root condition*, i.e., if it has only roots λ_ν with $|\lambda_\nu| \leq 1$ and if $|\lambda_\nu| = 1$, then λ_ν is a simple root of ϱ.

Consistent one-step methods are always asymptotically stable, if the associated function Φ is Lipschitz bounded. Since $\varrho(1) = \Sigma a_k = 0$, each method has at least one root $\lambda_\nu = 1$, thus one can never speak of a strongly asymptotically stable method here. During computations, however, it cannot be guaranteed that one works with a sufficiently small h for which the method satisfies the above indicated criterion. For this reason, one needs stability statements also for $h \neq 0$ in order to form a valid assessment of the stability behavior of the method used for the chosen step size h:

DEFINITION 17.18 (*Absolute stability, stability for $h \neq 0$*).
An algorithm for the numerical integration of (17.1) is *strongly absolutely stable* for a fixed $h \neq 0$, if all roots λ_ν of (17.27) satisfy $|\lambda_\nu| < 1$. It is called *weakly absolutely stable*, if the root condition of Definition 17.17 is fulfilled, otherwise it is *unstable*.
A curve in $\mathbb{R}^2$ which bounds the region of absolute stability of a method is called its *stability boundary*, see also [ENGE87], p. 458.

With absolutely stable procedures we are guaranteed that stable solutions of the given differential equation are approximated by stable solutions of the corresponding difference equation. The following phenomenon is, however, possible:

$$\lim_{x\to\infty} y(x) = 0, \quad \text{but} \quad \lim_{\substack{x\to\infty \\ h\neq 0}} Y(x_0 + ih) = d > 0.$$

If $\lim_{x\to\infty} y(x) = 0$ must imply $\lim_{\substack{i\to\infty \\ h\neq 0}} Y(x_0 + ih) = 0$ as well, then one has to require stability for arbitrary values hc with $Re(hc) < 0$ and $|hc| \to \infty$. Such a behavior is called *A-stability*.

DEFINITION 17.19 (*A-stability, stability for arbitrary hc with $|hc| \to \infty$*).
A procedure for the numerical integration of (17.1) is called *A-stable*, if for arbitrary hc with $Re(hc) < 0$, the root condition of Definition 17.17 is fulfilled for $P(\lambda) = \varrho(\lambda) + hc\sigma(\lambda) = 0$.

For A-stable methods the stability region must therefore contain the left half-plane of $\mathbf{C}$, i.e., the entire negative real axis of the hc-plane must be part of the stability region. The following statements hold about A-stability of one-step and multi-step methods:

(1) Explicit one-step methods are not A-stable.

(2) There exist A-stable implicit one-step method such as the implicit Runge-Kutta formulas of Gaussian type or the Rosenbrock methods, see [HALL76], p. 148, but not all implicit Runge-Kutta methods are A-stable.

(3) An explicit multi-step procedure cannot be A-stable.

(4) The global error order (or consistency order) of an A-stable multi-step method can be at most two.

(5) Among the A-stable multi-step methods with consistency order two, Heun's method has the smallest error.

The extrapolation method of section 17.5 is not A-stable. This summary shows that, except for the methods under (2), no method is known with a global error order $q_g > 2$ that fulfills the necessary condition of A-stability for all values c with $Re(hc) < 0$.

The term "stiff stability" will be defined in section 17.7.

Bibliography for Section 17.6

[ATKI78], 6.8; [HAIR87]; [HALL76], 2; [JELT76]; [JELT78]; [LINI77]; [LUTH87], 12; [RUTI52]; [STOE91], 7.2; [WERN79], §9-11 .

17.7 Stiff Systems of Differential Equations

17.7.1 The Problem

There is one class of initial value problems (17.1) for whose numerical treatment only very specific methods are useful. This is the class of *stiff differential equations* $y'(x) = f(x, y)$.

DEFINITION 17.20. A system of differential equations $y' = f(x, y)$ is called *stiff*, if the component functions $y_i(x)$ of the solution y of (17.1) have a very different growth behavior: For increasing x, there are strongly decreasing, weakly decreasing, as well as increasing solutions y_i and the increasing contributions grow much less quickly than the decreasing ones attenuate.

Requirements for a method to solve stiff systems of differential equations.

A method that is to be useful for stiff systems must meet the requirement that components of an approximate solution that have decreased below a certain threshold cannot have any more influence on the solution when the integration process is continued. This means that the stiff components must be integrated with a method in which for arbitrary $h > 0$ and all complex c with $Re(c) < 0$ we have

$$\lim_{\substack{i\to\infty \\ h\neq 0}} Y(x_0 + ih) = 0.$$

A-stability is (at least in a limited way) necessary for a numerical method to integrate stiff differential equations, since the numerical solution must reproduce correctly the required diminishing of the stiff components. In this respect the implicit Runge-Kutta methods of Gaussian type in section 17.3.5 are well suited for stiff systems.

17.7.2 Criteria for the Stiffness of a System

(A) For the system (17.1) of differential equations we assume at first that $f(x,y) = Ay$ for a constant (n,n) matrix $A = (a_{ik})$, $i,k = 1,\ldots,n$. Thus we assume that a system

$$y'(x) = Ay(x) \tag{17.28}$$

of linear differential equations with constant coefficients is given.
If A is diagonalizable, then the problem can be transformed to diagonal form. This is always possible if A has n distinct eigenvalues. The system (17.28) then separates into n scalar differential equations $y_i' = \lambda_i y_i$ for $i = 1,\ldots,n$. In general A can be reduced by similarity to its Jordan normal form. If the Jordan normal form of A is nondiagonal, then two eigenvalues of A would have to be equal, which is very improbable. So the assumption that A is diagonalizable is not as restrictive as it might seem, see chapter 7.2. Hence let us assume that the system (17.28) can be separated into n scalar differential equations. And in the remainder of this section we shall investigate scalar model problems for which we want to find integration methods with an appropriate stability behavior.

A system (17.28) is called *stiff* if for the eigenvalues λ_i of A:

$$\frac{\min\limits_{i,x\in I} Re(\lambda_i(x,y))}{|\max\limits_{i,x\in I} Re(\lambda_i(x,y))|} \gg 1. \tag{17.29}$$

Criterion (17.29) states that a system (17.28) can only be stiff if one of the eigenvalues of A lies in the left half-plane of $\mathbb{C}$ and moreover the real part of the left-most eigenvalue of A is significantly larger than the maximal real part of all the eigenvalues of A.

(B) To a given system of the form (17.1) we can locally associate a system of the form (17.28) for every $x \in I$. The matrix A can be taken as the Jacobi matrix of (17.1):

$$\begin{cases} A & := \left(\dfrac{\partial f_i(x,y)}{\partial y_k}\right) = A(x,y), \quad i,k = 1,\ldots,n, \\ f_i & = f_i(x,y) = f_i(x,y_1,y_2,\ldots,y_n). \end{cases} \tag{17.30}$$

If λ_i, $i = 1,\ldots,n$, are the eigenvalues of A in (17.30) at (x,y), then (17.29) is a criterion whether the system (17.1) is stiff in a neighborhood of (x,y). We note that the matrix A in (17.30) can vary strongly in the interval of integration.
A detailed analysis of stiffness is given in [HAIR91].

17.7.3 Gear's Method for Integrating Stiff Systems

The stability region of an A-stable method includes the negative hc-half-plane. Demanding A-stability restricts the global error order of methods suitable for integrating stiff systems. For this reason, several modified stability notions have been introduced which are related to A-stability, but allow to increase the global error order of a method. See [GEAR71/2], [GEAR71/1], [GRIG77] vol. 2).

Gear's method is based on the characteristic polynomial

$$P(\lambda) = \varrho(\lambda) - hc\sigma(\lambda). \tag{17.31}$$

At first $\sigma(x)$ is chosen in such a way that $P(\lambda) \to 0$ as $|hc| \to \infty$: Division of (17.31) by hc and taking the limit $|hc| \to \infty$ leads to $\sigma(\lambda) = 0$. The simplest choice is

$$\sigma(\lambda) = \lambda^M. \tag{17.32}$$

For this reason, $P(\lambda)$ has the best possible stability property for $hc = \infty$: If $|hc| = \infty$, P has an M-fold root at $\lambda = 0$. The condition of strong absolute stability (definition 17.18) is fulfilled at $|hc| = \infty$. In order to determine the behavior at finite points of the hc-plane, one has to
1) find the polynomial $P(\lambda)$ and thus $\varrho(\lambda)$, and
2) to determine its stability region.

1) For (17.32) one can calculate the associated $\varrho(x)$ from the consistency conditions: A linear multi-step method has the consistency order q, if (see [GRIG77] vol. 2, p. 334)

$$(17.33)\quad \begin{cases} (i) & \sum_{k=0}^{M} \left(a_k k^j - j b_k k^{j-1}\right) = 0,\ j = 1, \ldots, q, \text{ and} \\ (ii) & \sum_{k=0}^{M} a_k = 0. \end{cases}$$

(17.33) is a system of $q+1$ linear equations for the coefficients a_k, $k = 0, \ldots, q$, of the multi-step procedure.

2) The stability region of the multi-step method obtained by solving the linear system in 1) is determined for $\lambda := re^{i\varphi}$ with $|\lambda| = r \leq 1$ from

$$hc = \frac{\varrho(\lambda)}{\sigma(\lambda)} = \frac{\varrho(re^{i\varphi})}{\sigma(re^{i\varphi})} = u + iv,\ \varphi \in [0, 2\pi].$$

The stability boundary is obtained by setting $r = 1$.

Corrector formulas of Gear's method for $q = 1, \ldots, 6$.

The iteration rules for the various methods are given below for $\sigma(\lambda) = \lambda^M$, $M = 1, \ldots, 6$, with $M = q$ as consistency order:

M	Corrector formulas of Gear's method for $q = 1, \ldots, 6$	
1	$Y_{i+1}^{(\nu+1)} = Y_i + hf_{i+1}^{(\nu)} = Y_i + hf(x_{i+1}, Y_{i+1}^{(\nu)})$,	$\nu = 0, 1, 2, \ldots$
2	$Y_{i+1}^{(\nu+1)} = \frac{1}{3}\left(4Y_i - Y_{i-1} + 2hf_{i+1}^{(\nu)}\right)$,	$\nu = 0, 1, 2, \ldots$
3	$Y_{i+1}^{(\nu+1)} = \frac{1}{11}\left(18Y_i - 9Y_{i-1} + 2Y_{i-2} + 6hf_{i+1}^{(\nu)}\right)$,	$\nu = 0, 1, 2, \ldots$
4	$Y_{i+1}^{(\nu+1)} = \frac{1}{25}\left(48Y_i - 36Y_{i-1} + 16Y_{i-2} + -3Y_{i-3} + 12hf_{i+1}^{(\nu)}\right)$,	$\nu = 0, 1, 2, \ldots$
5	$Y_{i+1}^{(\nu+1)} = \frac{1}{137}\left(300Y_i - 300Y_{i-1} + 200Y_{i-2} + -75Y_{i-3} + 12Y_{i-4} + 60hf_{i+1}^{(\nu)}\right)$,	$\nu = 0, 1, 2, \ldots$
6	$Y_{i+1}^{(\nu+1)} = \frac{1}{147}\left(360Y_i - 450Y_{i-1} + 400Y_{i-2} + -225Y_{i-3} + 72Y_{i-4} - 10Y_{i-5} + 60hf_{i+1}^{(\nu)}\right)$,	$\nu = 0, 1, 2, \ldots$

For $q = 1, \ldots, 6$ only one functional evaluation of f is necessary per iteration step. The stability regions for $q = 1, \ldots, 6$ are given in the following picture (see [GEAR71/1], p. 212). Every boundary curve of a stability region passes through the point $|hc| = \infty$ of the hc-plane, see figures 17.2 and 17.3. For $M = 2$ the method is weakly A-stable, as is Heun's method. The formulas are stiffly stable for $M = 3, \ldots, 6$.

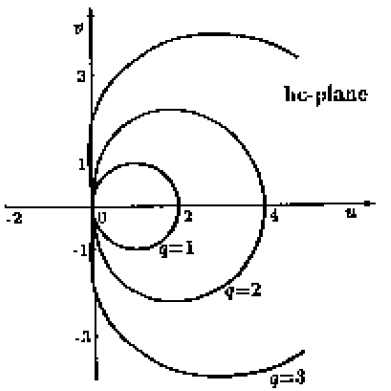

Figure 17.2: Stability regions for Gear's method for $q = 1, 2, 3$

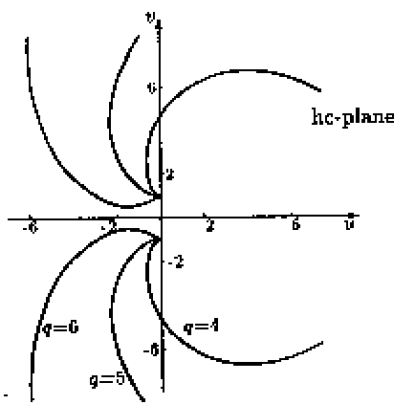

Figure 17.3: Stability regions for Gear's method for $q = 4, 5, 6$

One can represent the *stability regions* for the methods of Gear in a simplified way as follows:

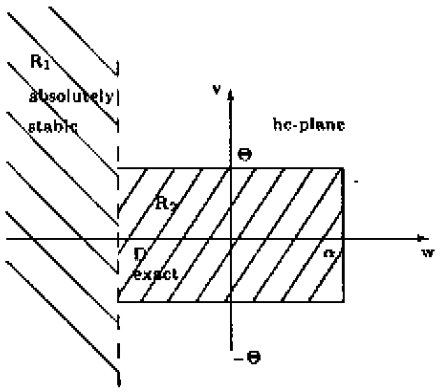

Figure 17.4: Regions of stiff stability
(for $q = 6$, for example, $D = -6.1$, $\Theta = 0.5$, and $\alpha \approx 0$)

The borders of the stability regions are tangents to the curves in the figures 17.2 and 17.3. Gear called the regions in figure 17.4 *regions of stiff stability*. He gave the following definition ([GEAR71/2]):

DEFINITION 17.21 (*Stiff stability*).
A procedure is called *stiff-stable* if it is absolutely stable in the region $R_1 = \{hc \mid (Re(hc) \leq D < 0)\}$ and is exact in the region $R_2 = \{hc \mid D \leq Re(hc) < \alpha,\ |Im(hc)| < \Theta\}$.

Convergence of the corrector formula.

In general, the corrector formula of Gear's method (see the earlier table of corrector formulas for $q = 1, \ldots, 6$) has the form

$$Y_{i+1}^{(\nu+1)} = \sum_{k=1}^{M} (hb_{M-k} f_{i+1-k} - a_{M-k} Y_{i+1-k}) + hb_M f_{i+1}^{(\nu)},$$

which can be derived from (17.24) by solving for Y_{i+1}. The corrector iteration converges if

$$\left\| hb_M \left(\frac{\partial f_i}{\partial y_k} \right) \right\| < 1,$$

where $\left(\frac{\partial f_i}{\partial y_k}\right)$ is the Jacobi matrix and $\|\cdot\|$ a matrix norm. For stiff single differential equations, as well as for separated differential equations, the convergence condition becomes

$$h \left| b_M \frac{\partial f}{\partial y} \right| < 1.$$

The convergence condition for the iteration might demand a smaller h than is required by the stiffly-stable method used. In this case, one should use Newton's method instead of the general iteration procedure for solving the corrector equation. To save computational time it may suffice to work with the simplified Newton's method - provided its convergence is not jeopardized, see section 6.2.

The predictor.

As initial values (predictors) for the corrector iteration, the values $Y_{i+1}^{(0)} = Y_i$ were suggested by Gear. The corrector iteration converges if h is chosen according to the convergence conditions just mentioned. In general, three iterations are suggested. Predictors such as the Adams-Bashforth formulas for the Adams-Moulton formulas can be constructed for the procedures of M^{th} order with $M = 3, \ldots, 6$: Starting with the equations (17.33), one constructs an explicit procedure ($b_M = 0$) by setting $\sigma(\lambda) = \lambda^{M-1}$.

Remarks for Gear's method.

The whole procedure was implemented by Gear, see SUBROUTINE DIFSUB in [GEAR71/3]. The program works with automatic step size control and automatically selects the global error order for a given error bound. Besides DIFSUB there exist newer versions Gear 3 or Gear Rev. 3 and a program package developed by Byrne and Hindmarsh in 1975 from DIFSUB. It uses the MS-methods as given by Nordsick, see [GRIG77], where both q_g and h can be controlled. At the same time, this package contains a numerical method with which one can avoid calculating the starting values. One begins the calculation with a one-step method of the class of chosen methods ([GRIG77], p. 90 ff.). When modifying step sizes one must be careful so that stability is not jeopardized (i.e., one should not routinely halve the step size). For details consult [GEAR74]. Further procedures for systems of stiff differential equations are given in [GRIG77], p. 236.

In general, Gear's method does not give useful results for stiff problems where the eigenvalues λ_i of the Jacobi matrix (f_y) are close to the imaginary axis of the hc-plane. For them some eigenvalues are outside of or close to the boundary of the region of absolute stability. In this case, Rosenbrock's or the modified Rosenbrock's methods have proven useful ([KAPS81]. Here, instead of the notion of stiff stability, other modifications of A-stability have to be used according to the structure of the procedure ([GRIG77, p.236 ss.; [JELT76];

[JELT78]). A-stable methods are the implicit Runge-Kutta formulas with Gaussian nodes (see section 17.3.5), as well as certain types of Rosenbrock methods. The amount of computational time, however, is considerably higher here than with Gear's method.

REMARK. Effective algorithms for solving stiff systems and their FORTRAN programs based on the work of Gear, Krogh and Hull are contained in the program package DEPAC which is available in coded form, see [SHAM75]. Another effective procedure for stiff systems is the Enright method, see [ENGE87], p. 491.

Bibliography for Section 17.7

[AHLB67]; [GEAR71/1]; [GEAR71/2]; [GEAR71/3]; [GEAR74]; [GRIG77] vol.2, 3.3; [HAIR91]; [HALL76], Part 2; [HULL72]; [HULL75]; [LAPI71], 6; [LIND77]; [LINI77]; [LUTH87], 13; [SHAM75]; [WERN86], chap.5.

17.8 Suggestions for Choosing among the Methods

None of the methods of this chapter will perform better than any other for every problem. One must, therefore, learn about the advantages and disadvantages of each method in order to decide for each particular problem which method to use. One cannot expect that the theory alone will be able to furnish strict criteria for choosing the optimal method for a desired accuracy. One can only formulate the following general guidelines on comparing the class of one-step methods and extrapolation methods on the one side with the multi-step methods on the other.

As one typical representative of one-step methods we shall choose the *classical Runge-Kutta method:*

Advantages: 1) Self-starting, fixed local error order $O(h^5)$, simple to handle.
2) Automatic step size control easily possible.

Disadvantage: Each integration step requires the calculation of four functional values.

Extrapolation method:

Advantages: 1) Self-starting, no fixed error order.
2) Step size control possible.

Disadvantage: Considerable computational time per step.

Multi-step methods:

Advantages: In general, each integration step requires only two or three functional evaluations per iteration step. Formulas of arbitrarily high order can easily be constructed.

Disadvantages: 1) Not self-starting. Calculation of starting values required.
2) Step size control possible ([GRIG77], vol.2, p.92, 98.), but very time consuming since the starting values must be recalculated if the step size is changed (see [GEAR80]).

These statements are independently true for all types of initial value problems. To help us decide on the merits of the methods we must moreover distinguish between non-stiff and stiff initial value problems:

Suggestions for non-stiff systems of differential equations.

The literature contains extensive numerical comparisons of the known numerical methods for integrating initial value problems for systems of differential equations of first order. A general test program DETEST was developed to investigate and compare all of the methods for solving initial value problems of ordinary differential equations in [HALL 73] by Hall, Enright, Hull and Sedgivida . A short description of this program and test results can be found in [HULL72] and [ENRI76].

There the test problems were divided into five classes:

1) Single first order differential equations.
2) Small systems of first order.
3) Medium sized systems of first order.
4) Systems of first order for calculating orbits.
5) Large order systems.

In each instance an error bound ε is given for the local procedural error so that rounding errors have no effect if computations are carried out in double precision. The step size is adjusted if the given bound ε demands it. The amount of

computational effort serves as a measure of the suitability of a method under the given conditions (i.e., type of problem, size of ε). The computational effort is subdivided into:

(a) Time for the calculation of functional values of f, which depends on their number per step and the complexity of the function f.

(b) Time for the other operations, apart from (c) (overhead).

(c) Time for necessary changes in the step size.

The following values for ε were used:

$$(i)\ \varepsilon = 10^{-3}, \quad (ii)\ \varepsilon = 10^{-6}, \quad (iii)\ \varepsilon = 10^{-9}.$$

Test results from [ENRI76], p. 626-635:

If the time for (a) is not significant such as it would be for a simple function f, the Bulirsch-Stoer-Gragg extrapolation method is best, see section 17.5. If, however, the needed number of f-values is large and their calculation is time consuming, then the Adams procedures are more advantageous, although the amount of calculation time in (b) is larger here. Implicit Adams type methods of variable order are particularly advantageous, see [KROG68], where the order of the method is automatically determined at the beginning of calculations from the given ε. h is chosen as large as possible while maintaining stability. Depending on whether the accuracy bound ε is exceeded or not, the error order of the method used locally is increased or decreased. Runge-Kutta methods are only of advantage if the amount of computational time in (a) is low and a modest accuracy is required, such as $\varepsilon = 10^{-3}$. Time tests for single differential equations are also included in [ENRI76].

Suggestions for Runge-Kutta embedding formulas.

Adaptive initial value solvers are the methods of choice here in every situation. Since adaptive methods require the calculation of two approximations Y and $\bar{Y}$ for the solution y in each step if step size control is to be used, we must try and minimize the computational effort to find $\bar{Y}$ in order to be competitive with other methods. When using Runge-Kutta embedding formulas this computational effort is minimal, since Y and $\bar{Y}$ are formed from the same k values and the computation of $\bar{Y}$ can be performed quickly, see section 17.3.4.4.
Our own extensive tests of embedding formulas have lead to the following results:

(1) The method **rk5(4)6m** (Prince-Dormand embedding formula of 4^{th} and 5^{th} order) can be especially recommended among the formulas of 4^{th} and 5^{th} order. **rk5(4)7m** and **rke5(4)** are less suitable. **rke5(4)** requires a rather large amount of computational time.

(2) Among the formulas of 6^{th} and 7^{th} order, **rkv6(5)** is to be preferred for its reliability and exactness. The formulas **rk6(5)8m** and **rkv7(6)** can be highly recommended due to their short computational times.

(3) Among the formulas of 8^{th} and 9^{th} order, **rk8(7)13m** is preferred due to its high reliability, its exactness and its short computational time; **rkv8(7)** is a close second. However, **rkv9(8)** requires a large amount of computations and has no other advantages over the above methods of 8^{th} order.

(4) The formulas **rkf6(5)** and **rkf8(7)** did not give comparably good results in the tests.

(5) For stiff problems none of the explicit Runge-Kutta methods gave acceptable results.

Suggestions for stiff systems.

Before deciding upon a method one should try to verify whether the given system is stiff. In general, one can decide this question with the criteria of section 17.7.2. For small systems, it is sometimes possible to recognize stiffness directly from the given differential equations. In general a system will reveal its stiffness if when using an explicit method the step size has to be chosen very small due to an already irrelevant component of the solution. A verification of the stiffness criteria in section 17.7.2 can, however, require considerable effort. But if one does not test for stiffness or falsely diagnoses a system as stiff and then uses one of the implicit methods for stiff systems of section 17.7, one might get very inexact results for a non-stiff system despite the long calculation time of the procedures there. For this reason, attempts are being made to develop numerical tests that find out easily whether a given problem is stiff or not.

Such a test has been proposed by L.F. Shampine. The given initial value problem is integrated first by an explicit Runge-Kutta method, and then by the Euler-Cauchy method with a prescribed level of exactness for the local procedural error. The Runge-Kutta method is normally chosen as the one given by Fehlberg with $m = 5$, and $q_\ell = 5$ with coefficients in table 17.6. Formula (17.8) provides a good approximation for the local procedural error at x_{i+1}.

If such a test indicates that the problem is stiff, then suggestions for the choice of the method can be drawn from the examination in [HULL75]. This paper describes systematic numerical tests in which five different methods that are suitable for solving stiff problems are applied to 25 specific known stiff problems. To test procedures for stiff systems, a test program STIFF DETEST was developed to test all known useful methods. The program can be found in a technical report of Bedet, Enright, Hull of the Department of Computer

Science of the University of Toronto. The results are published in [HULL75]. A measure of the usefulness of a method for a certain problem class is taken as the computational time required for obtaining a given accuracy bound. This time is composed of computer overhead, the time for functional evaluations and the computing of Jacobians as well as for matrix inversions. Here are some general suggestions: The stiff-stable methods of Enright and Gear have proved well suited for all problems in which the eigenvalues of every Jacobi matrix do not lie close to the imaginary axis of the hc-plane. The Enright procedure is stiffly-stable. The Enright method generally gives more accurate results, but it takes more time than Gear's method.

For problems in electrical engineering, where eigenvalues often appear close to the imaginary axis, Gear's method is nearly useless, while Enright's method has only limited use. A stiffly-stable procedure is obviously of no use here either, since $Im(hc) \gg 1$ and $Re(hc)$ is small. For this reason, one should uses an A-stable method. Implicit Runge-Kutta methods with Gaussian nodes have this property. It is, however, easier to deal with Rosenbrock methods for the integration of stiff systems. Suitable Rosenbrock methods are given in [KAPS81]. Kaps indicated modified Rosenbrock methods up to order 6 in 1977; they were implemented by Kaps and Wanner in [KAPS81].

A short description of the tests in [HULL75] can be found in [GRIG77], vol. 2. FORTRAN-Codes for Gear's method can be found in [SHAM75], see also our remark at the end of section 17.4.1 .

Bibliography for Section 17.8

[ENRI76]; [GEAR71/1], 12; [HAIR87]; [HAIR91]; [HULL72]; [HULL75]; [KAPS81]; [KROG66]; [KROG68]; [WERN86], 4.8, 5.5.