c. $y' = (y^2 + y)/t, \quad 1 \leq t \leq 3, \quad y(1) = -2,$ with $h = 0.2$

d. $y' = -ty + 4t/y, \quad 0 \leq t \leq 1, \quad y(0) = 1,$ with $h = 0.1$

7. Repeat Exercise 5 using Taylor's method of order four.

8. Repeat Exercise 6 using Taylor's method of order four.

9. Given the initial-value problem

$$y' = \frac{2}{t}y + t^2 e^t, \quad 1 \leq t \leq 2, \quad y(1) = 0,$$

with exact solution $y(t) = t^2(e^t - e)$:

a. Use Taylor's method of order two with $h = 0.1$ to approximate the solution, and compare it with the actual values of y.

b. Use the answers generated in part (a) and linear interpolation to approximate y at the following values, and compare them to the actual values of y.

i. $y(1.04)$ **ii.** $y(1.55)$ **iii.** $y(1.97)$

c. Use Taylor's method of order four with $h = 0.1$ to approximate the solution, and compare it with the actual values of y.

d. Use the answers generated in part (c) and piecewise cubic Hermite interpolation to approximate y at the following values, and compare them to the actual values of y.

i. $y(1.04)$ **ii.** $y(1.55)$ **iii.** $y(1.97)$

10. Given the initial-value problem

$$y' = \frac{1}{t^2} - \frac{y}{t} - y^2, \quad 1 \leq t \leq 2, \quad y(1) = -1,$$

with exact solution $y(t) = -1/t$:

a. Use Taylor's method of order two with $h = 0.05$ to approximate the solution, and compare it with the actual values of y.

b. Use the answers generated in part (a) and linear interpolation to approximate the following values of y, and compare them to the actual values.

i. $y(1.052)$ **ii.** $y(1.555)$ **iii.** $y(1.978)$

c. Use Taylor's method of order four with $h = 0.05$ to approximate the solution, and compare it with the actual values of y.

d. Use the answers generated in part (c) and piecewise cubic Hermite interpolation to approximate the following values of y, and compare them to the actual values.

i. $y(1.052)$ **ii.** $y(1.555)$ **iii.** $y(1.978)$

11. A projectile of mass $m = 0.11$ kg shot vertically upward with initial velocity $v(0) = 8$ m/s is slowed due to the force of gravity, $F_g = -mg$, and due to air resistance, $F_r = -kv|v|$, where $g = 9.8$ m/s^2 and $k = 0.002$ kg/m. The differential equation for the velocity v is given by

$$mv' = -mg - kv|v|.$$

a. Find the velocity after $0.1, 0.2, \ldots, 1.0$ s.

b. To the nearest tenth of a second, determine when the projectile reaches its maximum height and begins falling.

12. Use the Taylor method of order two with $h = 0.1$ to approximate the solution to

$$y' = 1 + t\sin(ty), \quad 0 \leq t \leq 2, \quad y(0) = 0.$$

5.4 Runge-Kutta Methods

The Taylor methods outlined in the previous section have the desirable property of high-order local truncation error, but the disadvantage of requiring the computation and evaluation of the derivatives of $f(t, y)$. This is a complicated and time-consuming procedure for most problems, so the Taylor methods are seldom used in practice.

In the later 1800s, Carl Runge (1856–1927) used methods similar to those in this section to derive numerous formulas for approximating the solution to initial-value problems.

Runge-Kutta methods have the high-order local truncation error of the Taylor methods but eliminate the need to compute and evaluate the derivatives of $f(t, y)$. Before presenting the ideas behind their derivation, we need to consider Taylor's Theorem in two variables. The proof of this result can be found in any standard book on advanced calculus (see, for example, [Fu], p. 331).

Theorem 5.13 Suppose that $f(t, y)$ and all its partial derivatives of order less than or equal to $n + 1$ are continuous on $D = \{(t, y) \mid a \le t \le b, c \le y \le d\}$, and let $(t_0, y_0) \in D$. For every $(t, y) \in D$, there exists ξ between t and t_0 and μ between y and y_0 with

$$f(t, y) = P_n(t, y) + R_n(t, y),$$

In 1901, Martin Wilhelm Kutta (1867–1944) generalized the methods that Runge developed in 1895 to incorporate systems of first-order differential equations. These techniques differ slightly from those we currently call Runge-Kutta methods.

where

$$\begin{aligned} P_n(t, y) &= f(t_0, y_0) + \left[(t - t_0)\frac{\partial f}{\partial t}(t_0, y_0) + (y - y_0)\frac{\partial f}{\partial y}(t_0, y_0)\right] \\ &+ \left[\frac{(t - t_0)^2}{2}\frac{\partial^2 f}{\partial t^2}(t_0, y_0) + (t - t_0)(y - y_0)\frac{\partial^2 f}{\partial t \partial y}(t_0, y_0)\right. \\ &+ \left.\frac{(y - y_0)^2}{2}\frac{\partial^2 f}{\partial y^2}(t_0, y_0)\right] + \cdots \\ &+ \left[\frac{1}{n!}\sum_{j=0}^{n}\binom{n}{j}(t - t_0)^{n-j}(y - y_0)^j\frac{\partial^n f}{\partial t^{n-j}\partial y^j}(t_0, y_0)\right] \end{aligned}$$

and

$$R_n(t, y) = \frac{1}{(n + 1)!}\sum_{j=0}^{n+1}\binom{n + 1}{j}(t - t_0)^{n+1-j}(y - y_0)^j\frac{\partial^{n+1} f}{\partial t^{n+1-j}\partial y^j}(\xi, \mu).$$

The function $P_n(t, y)$ is called the ***n*th Taylor polynomial in two variables** for the function f about (t_0, y_0), and $R_n(t, y)$ is the remainder term associated with $P_n(t, y)$. ■

Example 1 Use Maple to determine $P_2(t, y)$, the second Taylor polynomial about $(2, 3)$ for the function

$$f(t, y) = \exp\left[-\frac{(t - 2)^2}{4} - \frac{(y - 3)^2}{4}\right]\cos(2t + y - 7)$$

Solution To determine $P_2(t, y)$ we need the values of f and its first and second partial derivatives at $(2, 3)$. The evaluation of the function is easy

$$f(2, 3) = e^{\left(-0^2/4 - 0^2/4\right)}\cos(4 + 3 - 7) = 1,$$

but the computations involved with the partial derivatives are quite tedious. However, higher dimensional Taylor polynomials are available in the *MultivariateCalculus* subpackage of the *Student* package, which is accessed with the command

with(*Student*[*MultivariateCalculus*])

The first option of the *TaylorApproximation* command is the function, the second specifies the point (t_0, y_0) where the polynomial is centered, and the third specifies the degree of the polynomial. So we issue the command

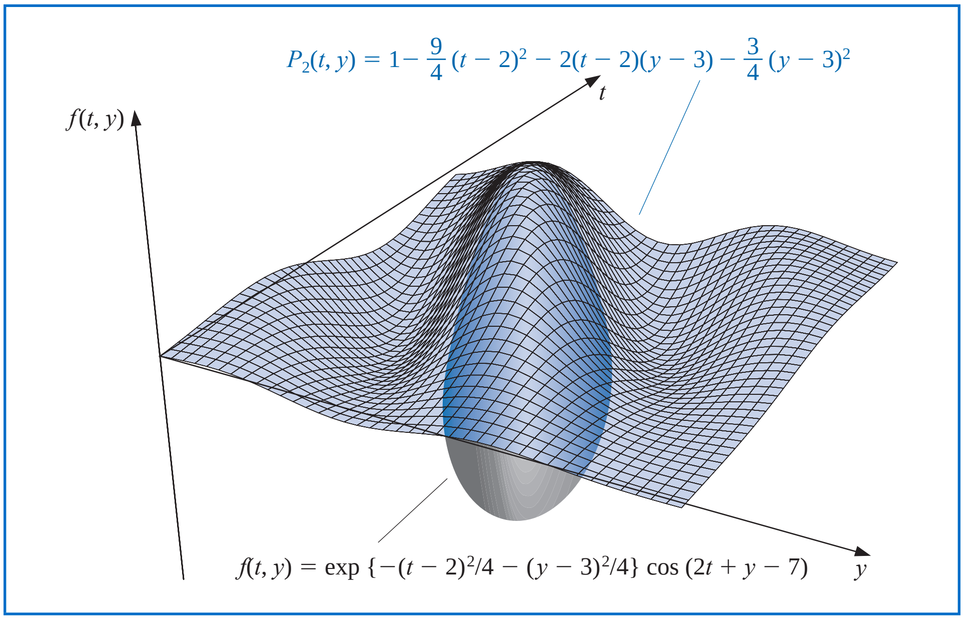
$$TaylorApproximation\left(e^{-\frac{(t-2)^2}{4}-\frac{(y-3)^2}{4}}\cos(2t+y-7), [t,y]=[2,3], 2\right)$$

The response from this Maple command is the polynomial

$$1-\frac{9}{4}(t-2)^2-2(t-2)(y-3)-\frac{3}{4}(y-3)^2$$

A plot option is also available by adding a fourth option to the *TaylorApproximation* command in the form *output* = *plot*. The plot in the default form is quite crude, however, because not many points are plotted for the function and the polynomial. A better illustration is seen in Figure 5.5.

Figure 5.5



The final parameter in this command indicates that we want the second multivariate Taylor polynomial, that is, the quadratic polynomial. If this parameter is 2, we get the quadratic polynomial, and if it is 0 or 1, we get the constant polynomial 1, because there are no linear terms. When this parameter is omitted, it defaults to 6 and gives the sixth Taylor polynomial. ■

Runge-Kutta Methods of Order Two

The first step in deriving a Runge-Kutta method is to determine values for a_1, α_1, and β_1 with the property that $a_1 f(t+\alpha_1, y+\beta_1)$ approximates

$$T^{(2)}(t,y)=f(t,y)+\frac{h}{2}f'(t,y),$$

with error no greater than $O(h^2)$, which is same as the order of the local truncation error for the Taylor method of order two. Since

$$f'(t,y)=\frac{df}{dt}(t,y)=\frac{\partial f}{\partial t}(t,y)+\frac{\partial f}{\partial y}(t,y)\cdot y'(t) \quad \text{and} \quad y'(t)=f(t,y),$$

we have

$$T^{(2)}(t,y) = f(t,y) + \frac{h}{2}\frac{\partial f}{\partial t}(t,y) + \frac{h}{2}\frac{\partial f}{\partial y}(t,y)\cdot f(t,y). \tag{5.18}$$

Expanding $f(t+\alpha_1, y+\beta_1)$ in its Taylor polynomial of degree one about (t,y) gives

$$\begin{aligned} a_1 f(t+\alpha_1, y+\beta_1) = a_1 f(t,y) + a_1\alpha_1 \frac{\partial f}{\partial t}(t,y) \\ + a_1\beta_1\frac{\partial f}{\partial y}(t,y) + a_1\cdot R_1(t+\alpha_1, y+\beta_1), \end{aligned} \tag{5.19}$$

where

$$R_1(t+\alpha_1, y+\beta_1) = \frac{\alpha_1^2}{2}\frac{\partial^2 f}{\partial t^2}(\xi,\mu) + \alpha_1\beta_1\frac{\partial^2 f}{\partial t\partial y}(\xi,\mu) + \frac{\beta_1^2}{2}\frac{\partial^2 f}{\partial y^2}(\xi,\mu), \tag{5.20}$$

for some ξ between t and $t+\alpha_1$ and μ between y and $y+\beta_1$.

Matching the coefficients of f and its derivatives in Eqs. (5.18) and (5.19) gives the three equations

$$f(t,y):\ a_1 = 1;\quad \frac{\partial f}{\partial t}(t,y):\ a_1\alpha_1 = \frac{h}{2};\quad \text{and}\quad \frac{\partial f}{\partial y}(t,y):\ a_1\beta_1 = \frac{h}{2}f(t,y).$$

The parameters a_1, α_1, and β_1 are therefore

$$a_1 = 1,\quad \alpha_1 = \frac{h}{2},\quad \text{and}\quad \beta_1 = \frac{h}{2}f(t,y),$$

so

$$T^{(2)}(t,y) = f\left(t+\frac{h}{2}, y+\frac{h}{2}f(t,y)\right) - R_1\left(t+\frac{h}{2}, y+\frac{h}{2}f(t,y)\right),$$

and from Eq. (5.20),

$$\begin{aligned} R_1\left(t+\frac{h}{2}, y+\frac{h}{2}f(t,y)\right) = \frac{h^2}{8}\frac{\partial^2 f}{\partial t^2}(\xi,\mu) + \frac{h^2}{4}f(t,y)\frac{\partial^2 f}{\partial t\partial y}(\xi,\mu) \\ + \frac{h^2}{8}(f(t,y))^2\frac{\partial^2 f}{\partial y^2}(\xi,\mu). \end{aligned}$$

If all the second-order partial derivatives of f are bounded, then

$$R_1\left(t+\frac{h}{2}, y+\frac{h}{2}f(t,y)\right)$$

is $O(h^2)$. As a consequence:

- The order of error for this new method is the same as that of the Taylor method of order two.

The difference-equation method resulting from replacing $T^{(2)}(t,y)$ in Taylor's method of order two by $f(t+(h/2), y+(h/2)f(t,y))$ is a specific Runge-Kutta method known as the *Midpoint method*.

Midpoint Method

$$w_0 = \alpha,$$

$$w_{i+1} = w_i + hf\left(t_i + \frac{h}{2}, w_i + \frac{h}{2}f(t_i, w_i)\right), \quad \text{for } i = 0, 1, \ldots, N-1.$$

Only three parameters are present in $a_1 f(t + \alpha_1, y + \beta_1)$ and all are needed in the match of $T^{(2)}$. So a more complicated form is required to satisfy the conditions for any of the higher-order Taylor methods.

The most appropriate four-parameter form for approximating

$$T^{(3)}(t, y) = f(t, y) + \frac{h}{2}f'(t, y) + \frac{h^2}{6}f''(t, y)$$

is

$$a_1 f(t, y) + a_2 f(t + \alpha_2, y + \delta_2 f(t, y)); \tag{5.21}$$

and even with this, there is insufficient flexibility to match the term

$$\frac{h^2}{6}\left[\frac{\partial f}{\partial y}(t, y)\right]^2 f(t, y),$$

resulting from the expansion of $(h^2/6)f''(t, y)$. Consequently, the best that can be obtained from using (5.21) are methods with $O(h^2)$ local truncation error.

The fact that (5.21) has four parameters, however, gives a flexibility in their choice, so a number of $O(h^2)$ methods can be derived. One of the most important is the *Modified Euler method*, which corresponds to choosing $a_1 = a_2 = \frac{1}{2}$ and $\alpha_2 = \delta_2 = h$. It has the following difference-equation form.

Modified Euler Method

$$w_0 = \alpha,$$

$$w_{i+1} = w_i + \frac{h}{2}[f(t_i, w_i) + f(t_{i+1}, w_i + hf(t_i, w_i))], \quad \text{for} \quad i = 0, 1, \ldots, N-1.$$

Example 2 Use the Midpoint method and the Modified Euler method with $N = 10$, $h = 0.2$, $t_i = 0.2i$, and $w_0 = 0.5$ to approximate the solution to our usual example,

$$y' = y - t^2 + 1, \quad 0 \le t \le 2, \quad y(0) = 0.5.$$

Solution The difference equations produced from the various formulas are

$$\text{Midpoint method:} \quad w_{i+1} = 1.22w_i - 0.0088i^2 - 0.008i + 0.218;$$

$$\text{Modified Euler method:} \quad w_{i+1} = 1.22w_i - 0.0088i^2 - 0.008i + 0.216,$$

for each $i = 0, 1, \ldots, 9$. The first two steps of these methods give

$$\text{Midpoint method:} \quad w_1 = 1.22(0.5) - 0.0088(0)^2 - 0.008(0) + 0.218 = 0.828;$$

$$\text{Modified Euler method:} \quad w_1 = 1.22(0.5) - 0.0088(0)^2 - 0.008(0) + 0.216 = 0.826,$$

and

$$\text{Midpoint method:} \quad w_2 = 1.22(0.828) - 0.0088(0.2)^2 - 0.008(0.2) + 0.218 = 1.21136;$$

$$\text{Modified Euler method:} \quad w_2 = 1.22(0.826) - 0.0088(0.2)^2 - 0.008(0.2) + 0.216 = 1.20692,$$

Table 5.6 lists all the results of the calculations. For this problem, the Midpoint method is superior to the Modified Euler method. ■

Table 5.6

t_i	$y(t_i)$	Midpoint Method	Error	Modified Euler Method	Error
0.0	0.5000000	0.5000000	0	0.5000000	0
0.2	0.8292986	0.8280000	0.0012986	0.8260000	0.0032986
0.4	1.2140877	1.2113600	0.0027277	1.2069200	0.0071677
0.6	1.6489406	1.6446592	0.0042814	1.6372424	0.0116982
0.8	2.1272295	2.1212842	0.0059453	2.1102357	0.0169938
1.0	2.6408591	2.6331668	0.0076923	2.6176876	0.0231715
1.2	3.1799415	3.1704634	0.0094781	3.1495789	0.0303627
1.4	3.7324000	3.7211654	0.0112346	3.6936862	0.0387138
1.6	4.2834838	4.2706218	0.0128620	4.2350972	0.0483866
1.8	4.8151763	4.8009586	0.0142177	4.7556185	0.0595577
2.0	5.3054720	5.2903695	0.0151025	5.2330546	0.0724173

Runge-Kutta methods are also options within the Maple command *InitialValueProblem*. The form and output for Runge-Kutta methods are the same as available under the Euler's and Taylor's methods, as discussed in Sections 5.1 and 5.2.

Higher-Order Runge-Kutta Methods

The term $T^{(3)}(t, y)$ can be approximated with error $O(h^3)$ by an expression of the form

$$f(t + \alpha_1, y + \delta_1 f(t + \alpha_2, y + \delta_2 f(t, y))),$$

involving four parameters, the algebra involved in the determination of $\alpha_1, \delta_1, \alpha_2$, and δ_2 is quite involved. The most common $O(h^3)$ is Heun's method, given by

Karl Heun (1859–1929) was a professor at the Technical University of Karlsruhe. He introduced this technique in a paper published in 1900. [Heu]

$$w_0 = \alpha$$

$$w_{i+1} = w_i + \frac{h}{4}\left(f(t_i, w_i) + 3f\left(t_i + \frac{2h}{3}, w_i + \frac{2h}{3}f\left(t_i + \frac{h}{3}, w_i + \frac{h}{3}f(t_i, w_i)\right)\right)\right),$$

$$\text{for} \quad i = 0, 1, \ldots, N - 1.$$

Illustration Applying Heun's method with $N = 10$, $h = 0.2$, $t_i = 0.2i$, and $w_0 = 0.5$ to approximate the solution to our usual example,

$$y' = y - t^2 + 1, \quad 0 \le t \le 2, \quad y(0) = 0.5.$$

gives the values in Table 5.7. Note the decreased error throughout the range over the Midpoint and Modified Euler approximations. □

Table 5.7

t_i	$y(t_i)$	Heun's Method	Error
0.0	0.5000000	0.5000000	0
0.2	0.8292986	0.8292444	0.0000542
0.4	1.2140877	1.2139750	0.0001127
0.6	1.6489406	1.6487659	0.0001747
0.8	2.1272295	2.1269905	0.0002390
1.0	2.6408591	2.6405555	0.0003035
1.2	3.1799415	3.1795763	0.0003653
1.4	3.7324000	3.7319803	0.0004197
1.6	4.2834838	4.2830230	0.0004608
1.8	4.8151763	4.8146966	0.0004797
2.0	5.3054720	5.3050072	0.0004648

Runge-Kutta methods of order three are not generally used. The most common Runge-Kutta method in use is of order four in difference-equation form, is given by the following.

Runge-Kutta Order Four

$$
\begin{aligned}
w_0 &= \alpha,\\
k_1 &= hf(t_i, w_i),\\
k_2 &= hf\left(t_i + \frac{h}{2}, w_i + \frac{1}{2}k_1\right),\\
k_3 &= hf\left(t_i + \frac{h}{2}, w_i + \frac{1}{2}k_2\right),\\
k_4 &= hf(t_{i+1}, w_i + k_3),\\
w_{i+1} &= w_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),
\end{aligned}
$$

for each $i = 0, 1, \ldots, N-1$. This method has local truncation error $O(h^4)$, provided the solution $y(t)$ has five continuous derivatives. We introduce the notation k_1, k_2, k_3, k_4 into the method is to eliminate the need for successive nesting in the second variable of $f(t, y)$. Exercise 32 shows how complicated this nesting becomes.

Algorithm 5.2 implements the Runge-Kutta method of order four.

ALGORITHM 5.2

Runge-Kutta (Order Four)

To approximate the solution of the initial-value problem

$$y' = f(t, y), \quad a \le t \le b, \quad y(a) = \alpha,$$

at $(N+1)$ equally spaced numbers in the interval $[a, b]$:

INPUT endpoints a, b; integer N; initial condition α.

OUTPUT approximation w to y at the $(N+1)$ values of t.

```
Step 1  Set $h = (b - a)/N$;
            $t = a$;
            $w = \alpha$;
        OUTPUT $(t, w)$.
Step 2  For $i = 1, 2, \ldots, N$ do Steps 3–5.
    Step 3  Set $K_1 = hf(t, w)$;
                $K_2 = hf(t + h/2, w + K_1/2)$;
                $K_3 = hf(t + h/2, w + K_2/2)$;
                $K_4 = hf(t + h, w + K_3)$.
    Step 4  Set $w = w + (K_1 + 2K_2 + 2K_3 + K_4)/6$; (Compute $w_i$.)
                $t = a + ih$. (Compute $t_i$.)
    Step 5  OUTPUT $(t, w)$.
Step 6  STOP.
```
■

Example 3 Use the Runge-Kutta method of order four with $h = 0.2$, $N = 10$, and $t_i = 0.2i$ to obtain approximations to the solution of the initial-value problem

$$y' = y - t^2 + 1, \quad 0 \le t \le 2, \quad y(0) = 0.5.$$

Solution The approximation to $y(0.2)$ is obtained by

$$w_0 = 0.5$$

$$k_1 = 0.2f(0, 0.5) = 0.2(1.5) = 0.3$$

$$k_2 = 0.2f(0.1, 0.65) = 0.328$$

$$k_3 = 0.2f(0.1, 0.664) = 0.3308$$

$$k_4 = 0.2f(0.2, 0.8308) = 0.35816$$

$$w_1 = 0.5 + \frac{1}{6}(0.3 + 2(0.328) + 2(0.3308) + 0.35816) = 0.8292933.$$

The remaining results and their errors are listed in Table 5.8. ■

Table 5.8

t_i	Exact $y_i = y(t_i)$	Runge-Kutta Order Four w_i	Error $\lvert y_i - w_i \rvert$
0.0	0.5000000	0.5000000	0
0.2	0.8292986	0.8292933	0.0000053
0.4	1.2140877	1.2140762	0.0000114
0.6	1.6489406	1.6489220	0.0000186
0.8	2.1272295	2.1272027	0.0000269
1.0	2.6408591	2.6408227	0.0000364
1.2	3.1799415	3.1798942	0.0000474
1.4	3.7324000	3.7323401	0.0000599
1.6	4.2834838	4.2834095	0.0000743
1.8	4.8151763	4.8150857	0.0000906
2.0	5.3054720	5.3053630	0.0001089

To obtain Runge-Kutta order 4 method results with *InitialValueProblem* use the option *method* = *rungekutta*, *submethod* = *rk4*. The results produced from the following call for out standard example problem agree with those in Table 5.6.

$C := InitialValueProblem(deq, y(0) = 0.5, t = 2, method = rungekutta, submethod = rk4, numsteps = 10, output = information, digits = 8)$

Computational Comparisons

The main computational effort in applying the Runge-Kutta methods is the evaluation of f. In the second-order methods, the local truncation error is $O(h^2)$, and the cost is two function evaluations per step. The Runge-Kutta method of order four requires 4 evaluations per step, and the local truncation error is $O(h^4)$. Butcher (see [But] for a summary) has established the relationship between the number of evaluations per step and the order of the local truncation error shown in Table 5.9. This table indicates why the methods of order less than five with smaller step size are used in preference to the higher-order methods using a larger step size.

Table 5.9

Evaluations per step	2	3	4	$5 \le n \le 7$	$8 \le n \le 9$	$10 \le n$
Best possible local truncation error	$O(h^2)$	$O(h^3)$	$O(h^4)$	$O(h^{n-1})$	$O(h^{n-2})$	$O(h^{n-3})$

One measure of comparing the lower-order Runge-Kutta methods is described as follows:

- The Runge-Kutta method of order four requires four evaluations per step, whereas Euler's method requires only one evaluation. Hence if the Runge-Kutta method of order four is to be superior it should give more accurate answers than Euler's method with one-fourth the step size. Similarly, if the Runge-Kutta method of order four is to be superior to the second-order Runge-Kutta methods, which require two evaluations per step, it should give more accuracy with step size h than a second-order method with step size $h/2$.

The following illustrates the superiority of the Runge-Kutta fourth-order method by this measure for the initial-value problem that we have been considering.

Illustration For the problem

$$y' = y - t^2 + 1, \quad 0 \le t \le 2, \quad y(0) = 0.5,$$

Euler's method with $h = 0.025$, the Midpoint method with $h = 0.05$, and the Runge-Kutta fourth-order method with $h = 0.1$ are compared at the common mesh points of these methods $0.1, 0.2, 0.3, 0.4,$ and 0.5. Each of these techniques requires 20 function evaluations to determine the values listed in Table 5.10 to approximate $y(0.5)$. In this example, the fourth-order method is clearly superior. □

Table 5.10

t_i	Exact	Euler $h = 0.025$	Modified Euler $h = 0.05$	Runge-Kutta Order Four $h = 0.1$
0.0	0.5000000	0.5000000	0.5000000	0.5000000
0.1	0.6574145	0.6554982	0.6573085	0.6574144
0.2	0.8292986	0.8253385	0.8290778	0.8292983
0.3	1.0150706	1.0089334	1.0147254	1.0150701
0.4	1.2140877	1.2056345	1.2136079	1.2140869
0.5	1.4256394	1.4147264	1.4250141	1.4256384

EXERCISE SET 5.4

1. Use the Modified Euler method to approximate the solutions to each of the following initial-value problems, and compare the results to the actual values.

a. $y' = te^{3t} - 2y$, $0 \le t \le 1$, $y(0) = 0$, with $h = 0.5$; actual solution $y(t) = \frac{1}{5}te^{3t} - \frac{1}{25}e^{3t} + \frac{1}{25}e^{-2t}$.

b. $y' = 1 + (t - y)^2$, $2 \le t \le 3$, $y(2) = 1$, with $h = 0.5$; actual solution $y(t) = t + \frac{1}{1-t}$.

c. $y' = 1 + y/t$, $1 \le t \le 2$, $y(1) = 2$, with $h = 0.25$; actual solution $y(t) = t \ln t + 2t$.

d. $y' = \cos 2t + \sin 3t$, $0 \le t \le 1$, $y(0) = 1$, with $h = 0.25$; actual solution $y(t) = \frac{1}{2}\sin 2t - \frac{1}{3}\cos 3t + \frac{4}{3}$.

2. Use the Modified Euler method to approximate the solutions to each of the following initial-value problems, and compare the results to the actual values.

a. $y' = e^{t-y}$, $0 \le t \le 1$, $y(0) = 1$, with $h = 0.5$; actual solution $y(t) = \ln(e^t + e - 1)$.

b. $y' = \dfrac{1+t}{1+y}$, $1 \le t \le 2$, $y(1) = 2$, with $h = 0.5$; actual solution $y(t) = \sqrt{t^2 + 2t + 6} - 1$.

c. $y' = -y + ty^{1/2}$, $2 \le t \le 3$, $y(2) = 2$, with $h = 0.25$; actual solution $y(t) = \left(t - 2 + \sqrt{2}ee^{-t/2}\right)^2$.

d. $y' = t^{-2}(\sin 2t - 2ty)$, $1 \le t \le 2$, $y(1) = 2$, with $h = 0.25$; actual solution $y(t) = \frac{1}{2}t^{-2}(4 + \cos 2 - \cos 2t)$.

3. Use the Modified Euler method to approximate the solutions to each of the following initial-value problems, and compare the results to the actual values.

a. $y' = y/t - (y/t)^2$, $1 \le t \le 2$, $y(1) = 1$, with $h = 0.1$; actual solution $y(t) = t/(1 + \ln t)$.

b. $y' = 1 + y/t + (y/t)^2$, $1 \le t \le 3$, $y(1) = 0$, with $h = 0.2$; actual solution $y(t) = t \tan(\ln t)$.

c. $y' = -(y + 1)(y + 3)$, $0 \le t \le 2$, $y(0) = -2$, with $h = 0.2$; actual solution $y(t) = -3 + 2(1 + e^{-2t})^{-1}$.

d. $y' = -5y + 5t^2 + 2t$, $0 \le t \le 1$, $y(0) = \frac{1}{3}$, with $h = 0.1$; actual solution $y(t) = t^2 + \frac{1}{3}e^{-5t}$.

4. Use the Modified Euler method to approximate the solutions to each of the following initial-value problems, and compare the results to the actual values.

a. $y' = \dfrac{2 - 2ty}{t^2 + 1}$, $0 \le t \le 1$, $y(0) = 1$, with $h = 0.1$; actual solution $y(t) = \dfrac{2t + 1}{t^2 + 1}$.

b. $y' = \dfrac{y^2}{1 + t}$, $1 \le t \le 2$, $y(1) = -(\ln 2)^{-1}$, with $h = 0.1$; actual solution $y(t) = \dfrac{-1}{\ln(t + 1)}$.

c. $y' = (y^2 + y)/t$, $1 \le t \le 3$, $y(1) = -2$, with $h = 0.2$; actual solution $y(t) = \dfrac{2t}{1 - 2t}$.

d. $y' = -ty + 4t/y$, $0 \le t \le 1$, $y(0) = 1$, with $h = 0.1$; actual solution $y(t) = \sqrt{4 - 3e^{-t^2}}$.

5. Repeat Exercise 1 using the Midpoint method.

6. Repeat Exercise 2 using the Midpoint method.

7. Repeat Exercise 3 using the Midpoint method.

8. Repeat Exercise 4 using the Midpoint method.

9. Repeat Exercise 1 using Heun's method.

10. Repeat Exercise 2 using Heun's method.

11. Repeat Exercise 3 using Heun's method.

12. Repeat Exercise 4 using Heun's method.

13. Repeat Exercise 1 using the Runge-Kutta method of order four.

14. Repeat Exercise 2 using the Runge-Kutta method of order four.

15. Repeat Exercise 3 using the Runge-Kutta method of order four.

16. Repeat Exercise 4 using the Runge-Kutta method of order four.

17. Use the results of Exercise 3 and linear interpolation to approximate values of $y(t)$, and compare the results to the actual values.

a. $y(1.25)$ and $y(1.93)$ **b.** $y(2.1)$ and $y(2.75)$

c. $y(1.3)$ and $y(1.93)$ **d.** $y(0.54)$ and $y(0.94)$

18. Use the results of Exercise 4 and linear interpolation to approximate values of $y(t)$, and compare the results to the actual values.

a. $y(0.54)$ and $y(0.94)$ **b.** $y(1.25)$ and $y(1.93)$

c. $y(1.3)$ and $y(2.93)$ **d.** $y(0.54)$ and $y(0.94)$

19. Repeat Exercise 17 using the results of Exercise 7.

20. Repeat Exercise 18 using the results of Exercise 8.

21. Repeat Exercise 17 using the results of Exercise 11.

22. Repeat Exercise 18 using the results of Exercise 12.

23. Repeat Exercise 17 using the results of Exercise 15.

24. Repeat Exercise 18 using the results of Exercise 16.

25. Use the results of Exercise 15 and Cubic Hermite interpolation to approximate values of $y(t)$, and compare the approximations to the actual values.

a. $y(1.25)$ and $y(1.93)$ **b.** $y(2.1)$ and $y(2.75)$

c. $y(1.3)$ and $y(1.93)$ **d.** $y(0.54)$ and $y(0.94)$

26. Use the results of Exercise 16 and Cubic Hermite interpolation to approximate values of $y(t)$, and compare the approximations to the actual values.

a. $y(0.54)$ and $y(0.94)$ **b.** $y(1.25)$ and $y(1.93)$

c. $y(1.3)$ and $y(2.93)$ **d.** $y(0.54)$ and $y(0.94)$

27. Show that the Midpoint method and the Modified Euler method give the same approximations to the initial-value problem

$$y' = -y + t + 1, \quad 0 \le t \le 1, \quad y(0) = 1,$$

for any choice of h. Why is this true?

28. Water flows from an inverted conical tank with circular orifice at the rate

$$\frac{dx}{dt} = -0.6\pi r^2 \sqrt{2g}\frac{\sqrt{x}}{A(x)},$$

where r is the radius of the orifice, x is the height of the liquid level from the vertex of the cone, and $A(x)$ is the area of the cross section of the tank x units above the orifice. Suppose $r = 0.1$ ft, $g = 32.1$ ft/s^2, and the tank has an initial water level of 8 ft and initial volume of $512(\pi/3)$ ft^3. Use the Runge-Kutta method of order four to find the following.

a. The water level after 10 min with $h = 20$ s

b. When the tank will be empty, to within 1 min.

29. The irreversible chemical reaction in which two molecules of solid potassium dichromate ($K_2Cr_2O_7$), two molecules of water (H_2O), and three atoms of solid sulfur (S) combine to yield three molecules of the gas sulfur dioxide (SO_2), four molecules of solid potassium hydroxide (KOH), and two molecules of solid chromic oxide (Cr_2O_3) can be represented symbolically by the *stoichiometric equation*:

$$2K_2Cr_2O_7 + 2H_2O + 3S \longrightarrow 4KOH + 2Cr_2O_3 + 3SO_2.$$

If n_1 molecules of $K_2Cr_2O_7$, n_2 molecules of H_2O, and n_3 molecules of S are originally available, the following differential equation describes the amount $x(t)$ of KOH after time t:

$$\frac{dx}{dt} = k\left(n_1 - \frac{x}{2}\right)^2\left(n_2 - \frac{x}{2}\right)^2\left(n_3 - \frac{3x}{4}\right)^3,$$

where k is the velocity constant of the reaction. If $k = 6.22 \times 10^{-19}, n_1 = n_2 = 2 \times 10^3$, and $n_3 = 3 \times 10^3$, use the Runge-Kutta method of order four to determine how many units of potassium hydroxide will have been formed after 0.2 s?

30. Show that the difference method

$$w_0 = \alpha,$$
$$w_{i+1} = w_i + a_1 f(t_i, w_i) + a_2 f(t_i + \alpha_2, w_1 + \delta_2 f(t_i, w_i)),$$

for each $i = 0, 1, \ldots, N-1$, cannot have local truncation error $O(h^3)$ for any choice of constants a_1, a_2, α_2, and δ_2.

31. Show that Heun's method can be expressed in difference form, similar to that of the Runge-Kutta method of order four, as

$$w_0 = \alpha,$$
$$k_1 = hf(t_i, w_i),$$
$$k_2 = hf\left(t_i + \frac{h}{3}, w_i + \frac{1}{3}k_1\right),$$
$$k_3 = hf\left(t_i + \frac{2h}{3}, w_i + \frac{2}{3}k_2\right),$$
$$w_{i+1} = w_i + \frac{1}{4}(k_1 + 3k_3),$$

for each $i = 0, 1, \ldots, N-1$.

32. The Runge-Kutta method of order four can be written in the form

$$\begin{aligned} w_0 &= \alpha, \\ w_{i+1} &= w_i + \frac{h}{6}f(t_i, w_i) + \frac{h}{3}f(t_i + \alpha_1 h, w_i + \delta_1 hf(t_i, w_i)) \\ &\quad + \frac{h}{3}f(t_i + \alpha_2 h, w_i + \delta_2 hf(t_i + \gamma_2 h, w_i + \gamma_3 hf(t_i, w_i))) \\ &\quad + \frac{h}{6}f(t_i + \alpha_3 h, w_i + \delta_3 hf(t_i + \gamma_4 h, w_i + \gamma_5 hf(t_i + \gamma_6 h, w_i + \gamma_7 hf(t_i, w_i)))). \end{aligned}$$

Find the values of the constants

$$\alpha_1,\ \alpha_2,\ \alpha_3,\ \delta_1,\ \delta_2,\ \delta_3,\ \gamma_2,\ \gamma_3,\ \gamma_4,\ \gamma_5,\ \gamma_6,\ \text{and}\ \gamma_7.$$

5.5 Error Control and the Runge-Kutta-Fehlberg Method

In Section 4.6 we saw that the appropriate use of varying step sizes for integral approximations produced efficient methods. In itself, this might not be sufficient to favor these methods due to the increased complication of applying them. However, they have another feature

You might like to review the Adaptive Quadrature material in Section 4.6 before considering this material.

that makes them worthwhile. They incorporate in the step-size procedure an estimate of the truncation error that does not require the approximation of the higher derivatives of the function. These methods are called *adaptive* because they adapt the number and position of the nodes used in the approximation to ensure that the truncation error is kept within a specified bound.

There is a close connection between the problem of approximating the value of a definite integral and that of approximating the solution to an initial-value problem. It is not surprising, then, that there are adaptive methods for approximating the solutions to initial-value problems and that these methods are not only efficient, but also incorporate the control of error.

Any one-step method for approximating the solution, $y(t)$, of the initial-value problem

$$y' = f(t, y), \quad \text{for } a \le t \le b, \quad \text{with } y(a) = \alpha$$

can be expressed in the form

$$w_{i+1} = w_i + h_i\phi(t_i, w_i, h_i), \quad \text{for } i = 0, 1, \ldots, N-1,$$

for some function ϕ.

An ideal difference-equation method

$$w_{i+1} = w_i + h_i\phi(t_i, w_i, h_i), \quad i = 0, 1, \ldots, N-1,$$

for approximating the solution, $y(t)$, to the initial-value problem

$$y' = f(t, y), \quad a \le t \le b, \quad y(a) = \alpha,$$

would have the property that, given a tolerance $\varepsilon > 0$, a minimal number of mesh points could be used to ensure that the global error, $|y(t_i) - w_i|$, did not exceed ε for any $i = 0, 1, \ldots, N$. Having a minimal number of mesh points and also controlling the global error of a difference method is, not surprisingly, inconsistent with the points being equally spaced in the interval. In this section we examine techniques used to control the error of a difference-equation method in an efficient manner by the appropriate choice of mesh points.

Although we cannot generally determine the global error of a method, we will see in Section 5.10 that there is a close connection between the local truncation error and the global error. By using methods of differing order we can predict the local truncation error and, using this prediction, choose a step size that will keep it and the global error in check.

To illustrate the technique, suppose that we have two approximation techniques. The first is obtained from an nth-order Taylor method of the form

$$y(t_{i+1}) = y(t_i) + h\phi(t_i, y(t_i), h) + O(h^{n+1}),$$

and produces approximations with local truncation error $\tau_{i+1}(h) = O(h^n)$. It is given by

$$\begin{aligned} w_0 &= \alpha \\ w_{i+1} &= w_i + h\phi(t_i, w_i, h), \quad \text{for } i > 0. \end{aligned}$$

In general, the method is generated by applying a Runge-Kutta modification to the Taylor method, but the specific derivation is unimportant.

The second method is similar but one order higher; it comes from an $(n+1)$st-order Taylor method of the form

$$y(t_{i+1}) = y(t_i) + h\tilde{\phi}(t_i, y(t_i), h) + O(h^{n+2}),$$

and produces approximations with local truncation error $\tilde{\tau}_{i+1}(h) = O(h^{n+1})$. It is given by

$$\tilde{w}_0 = \alpha$$

$$\tilde{w}_{i+1} = \tilde{w}_i + h\tilde{\phi}(t_i, \tilde{w}_i, h), \quad \text{for } i > 0.$$

We first make the assumption that $w_i \approx y(t_i) \approx \tilde{w}_i$ and choose a fixed step size h to generate the approximations w_{i+1} and $\tilde{w}_{i+1}$ to $y(t_{i+1})$. Then

$$\begin{aligned}\tau_{i+1}(h) &= \frac{y(t_{i+1}) - y(t_i)}{h} - \phi(t_i, y(t_i), h)\\ &= \frac{y(t_{i+1}) - w_i}{h} - \phi(t_i, w_i, h)\\ &= \frac{y(t_{i+1}) - [w_i + h\phi(t_i, w_i, h)]}{h}\\ &= \frac{1}{h}(y(t_{i+1}) - w_{i+1}).\end{aligned}$$

In a similar manner, we have

$$\tilde{\tau}_{i+1}(h) = \frac{1}{h}(y(t_{i+1}) - \tilde{w}_{i+1}).$$

As a consequence, we have

$$\begin{aligned}\tau_{i+1}(h) &= \frac{1}{h}(y(t_{i+1}) - w_{i+1})\\ &= \frac{1}{h}[(y(t_{i+1}) - \tilde{w}_{i+1}) + (\tilde{w}_{i+1} - w_{i+1})]\\ &= \tilde{\tau}_{i+1}(h) + \frac{1}{h}(\tilde{w}_{i+1} - w_{i+1}).\end{aligned}$$

But $\tau_{i+1}(h)$ is $O(h^n)$ and $\tilde{\tau}_{i+1}(h)$ is $O(h^{n+1})$, so the significant portion of $\tau_{i+1}(h)$ must come from

$$\frac{1}{h}\left(\tilde{w}_{i+1} - w_{i+1}\right).$$

This gives us an easily computed approximation for the local truncation error of the $O(h^n)$ method:

$$\tau_{i+1}(h) \approx \frac{1}{h}\left(\tilde{w}_{i+1} - w_{i+1}\right).$$

The object, however, is not simply to estimate the local truncation error but to adjust the step size to keep it within a specified bound. To do this we now assume that since $\tau_{i+1}(h)$ is $O(h^n)$, a number K, independent of h, exists with

$$\tau_{i+1}(h) \approx Kh^n.$$

Then the local truncation error produced by applying the nth-order method with a new step size qh can be estimated using the original approximations w_{i+1} and $\tilde{w}_{i+1}$:

$$\tau_{i+1}(qh) \approx K(qh)^n = q^n(Kh^n) \approx q^n\tau_{i+1}(h) \approx \frac{q^n}{h}(\tilde{w}_{i+1} - w_{i+1}).$$

To bound $\tau_{i+1}(qh)$ by ε, we choose q so that

$$\frac{q^n}{h}|\tilde{w}_{i+1} - w_{i+1}| \approx |\tau_{i+1}(qh)| \le \varepsilon;$$

that is, so that

$$q \le \left(\frac{\varepsilon h}{|\tilde{w}_{i+1} - w_{i+1}|} \right)^{1/n}. \tag{5.22}$$

Runge-Kutta-Fehlberg Method

Erwin Fehlberg developed this and other error control techniques while working for the NASA facility in Huntsville, Alabama during the 1960s. He received the Exceptional Scientific Achievement Medal from NASA in 1969.

One popular technique that uses Inequality (5.22) for error control is the **Runge-Kutta-Fehlberg method**. (See [Fe].) This technique uses a Runge-Kutta method with local truncation error of order five,

$$\tilde{w}_{i+1} = w_i + \frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6,$$

to estimate the local error in a Runge-Kutta method of order four given by

$$w_{i+1} = w_i + \frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5,$$

where the coefficient equations are

$$\begin{aligned}
k_1 &= hf(t_i, w_i),\\
k_2 &= hf\left(t_i + \frac{h}{4}, w_i + \frac{1}{4}k_1\right),\\
k_3 &= hf\left(t_i + \frac{3h}{8}, w_i + \frac{3}{32}k_1 + \frac{9}{32}k_2\right),\\
k_4 &= hf\left(t_i + \frac{12h}{13}, w_i + \frac{1932}{2197}k_1 - \frac{7200}{2197}k_2 + \frac{7296}{2197}k_3\right),\\
k_5 &= hf\left(t_i + h, w_i + \frac{439}{216}k_1 - 8k_2 + \frac{3680}{513}k_3 - \frac{845}{4104}k_4\right),\\
k_6 &= hf\left(t_i + \frac{h}{2}, w_i - \frac{8}{27}k_1 + 2k_2 - \frac{3544}{2565}k_3 + \frac{1859}{4104}k_4 - \frac{11}{40}k_5\right).
\end{aligned}$$

An advantage to this method is that only six evaluations of f are required per step. Arbitrary Runge-Kutta methods of orders four and five used together (see Table 5.9 on page 290) require at least four evaluations of f for the fourth-order method and an additional six for the fifth-order method, for a total of at least ten function evaluations. So the Runge-Kutta-Fehlberg method has at least a 40% decrease in the number of function evaluations over the use of a pair of arbitrary fourth- and fifth-order methods.

In the error-control theory, an initial value of h at the ith step is used to find the first values of w_{i+1} and $\tilde{w}_{i+1}$, which leads to the determination of q for that step, and then the calculations were repeated. This procedure requires twice the number of function evaluations per step as without the error control. In practice, the value of q to be used is chosen somewhat differently in order to make the increased function-evaluation cost worthwhile. The value of q determined at the ith step is used for two purposes:

- When $q < 1$: to reject the initial choice of h at the ith step and repeat the calculations using qh, and

- When $q \ge 1$: to accept the computed value at the ith step using the step size h, but change the step size to qh for the $(i+1)$st step.

Because of the penalty in terms of function evaluations that must be paid if the steps are repeated, q tends to be chosen conservatively. In fact, for the Runge-Kutta-Fehlberg method with $n = 4$, a common choice is

$$q = \left(\frac{\varepsilon h}{2|\tilde{w}_{i+1} - w_{i+1}|} \right)^{1/4} = 0.84 \left(\frac{\varepsilon h}{|\tilde{w}_{i+1} - w_{i+1}|} \right)^{1/4}.$$

In Algorithm 5.3 for the Runge-Kutta-Fehlberg method, Step 9 is added to eliminate large modifications in step size. This is done to avoid spending too much time with small step sizes in regions with irregularities in the derivatives of y, and to avoid large step sizes, which can result in skipping sensitive regions between the steps. The step-size increase procedure could be omitted completely from the algorithm, and the step-size decrease procedure used only when needed to bring the error under control.

ALGORITHM
5.3

Runge-Kutta-Fehlberg

To approximate the solution of the initial-value problem

$$y' = f(t, y), \quad a \le t \le b, \quad y(a) = \alpha,$$

with local truncation error within a given tolerance:

INPUT endpoints a, b; initial condition α; tolerance TOL; maximum step size $hmax$; minimum step size $hmin$.

OUTPUT t, w, h where w approximates $y(t)$ and the step size h was used, or a message that the minimum step size was exceeded.

```
Step 1 Set $t = a$;
           $w = \alpha$;
           $h = hmax$;
           $FLAG = 1$;
       OUTPUT $(t, w)$.
Step 2 While ($FLAG = 1$) do Steps 3–11.
    Step 3 Set $K_1 = hf(t, w)$;
               $K_2 = hf\left(t + \frac{1}{4}h, w + \frac{1}{4}K_1\right)$;
               $K_3 = hf\left(t + \frac{3}{8}h, w + \frac{3}{32}K_1 + \frac{9}{32}K_2\right)$;
               $K_4 = hf\left(t + \frac{12}{13}h, w + \frac{1932}{2197}K_1 - \frac{7200}{2197}K_2 + \frac{7296}{2197}K_3\right)$;
               $K_5 = hf\left(t + h, w + \frac{439}{216}K_1 - 8K_2 + \frac{3680}{513}K_3 - \frac{845}{4104}K_4\right)$;
               $K_6 = hf\left(t + \frac{1}{2}h, w - \frac{8}{27}K_1 + 2K_2 - \frac{3544}{2565}K_3 + \frac{1859}{4104}K_4 - \frac{11}{40}K_5\right)$.
    Step 4 Set $R = \frac{1}{h}\left|\frac{1}{360}K_1 - \frac{128}{4275}K_3 - \frac{2197}{75240}K_4 + \frac{1}{50}K_5 + \frac{2}{55}K_6\right|$.
               (Note: $R = \frac{1}{h}|\tilde{w}_{i+1} - w_{i+1}|$.)
    Step 5 If $R \le TOL$ then do Steps 6 and 7.
        Step 6 Set $t = t + h$; (Approximation accepted.)
                   $w = w + \frac{25}{216}K_1 + \frac{1408}{2565}K_3 + \frac{2197}{4104}K_4 - \frac{1}{5}K_5$.
```

```
        Step 7   OUTPUT $(t, w, h)$.
    Step 8   Set $\delta = 0.84(TOL/R)^{1/4}$.
    Step 9   If $\delta \le 0.1$ then set $h = 0.1h$
                 else if $\delta \ge 4$ then set $h = 4h$
                     else set $h = \delta h$.   (Calculate new $h$.)
Step 10  If $h > hmax$ then set $h = hmax$.
Step 11  If $t \ge b$ then set $FLAG = 0$
             else if $t + h > b$ then set $h = b - t$
                 else if $h < hmin$ then
                     set $FLAG = 0$;
                     OUTPUT ('minimum h exceeded').
                     (Procedure completed unsuccessfully.)
Step 12  (The procedure is complete.)
         STOP.
```
■

Example 1 Use the Runge-Kutta-Fehlberg method with a tolerance $TOL = 10^{-5}$, a maximum step size $hmax = 0.25$, and a minimum step size $hmin = 0.01$ to approximate the solution to the initial-value problem

$$y' = y - t^2 + 1, \quad 0 \le t \le 2, \quad y(0) = 0.5,$$

and compare the results with the exact solution $y(t) = (t+1)^2 - 0.5e^t$.

Solution We will work through the first step of the calculations and then apply Algorithm 5.3 to determine the remaining results. The initial condition gives $t_0 = 0$ and $w_0 = 0.5$. To determine w_1 using $h = 0.25$, the maximum allowable stepsize, we compute

$$k_1 = hf(t_0, w_0) = 0.25\left(0.5 - 0^2 + 1\right) = 0.375;$$

$$k_2 = hf\left(t_0 + \frac{1}{4}h, w_0 + \frac{1}{4}k_1\right) = 0.25\left(\frac{1}{4}0.25, 0.5 + \frac{1}{4}0.375\right) = 0.3974609;$$

$$\begin{aligned} k_3 &= hf\left(t_0 + \frac{3}{8}h, w_0 + \frac{3}{32}k_1 + \frac{9}{32}k_2\right) \\ &= 0.25\left(0.09375, 0.5 + \frac{3}{32}0.375 + \frac{9}{32}0.3974609\right) = 0.4095383; \end{aligned}$$

$$\begin{aligned} k_4 &= hf\left(t_0 + \frac{12}{13}h, w_0 + \frac{1932}{2197}k_1 - \frac{7200}{2197}k_2 + \frac{7296}{2197}k_3\right) \\ &= 0.25\left(0.2307692, 0.5 + \frac{1932}{2197}0.375 - \frac{7200}{2197}0.3974609 + \frac{7296}{2197}0.4095383\right) \\ &= 0.4584971; \end{aligned}$$

$$\begin{aligned} k_5 &= hf\left(t_0 + h, w_0 + \frac{439}{216}k_1 - 8k_2 + \frac{3680}{513}k_3 - \frac{845}{4104}k_4\right) \\ &= 0.25\left(0.25, 0.5 + \frac{439}{216}0.375 - 8(0.3974609) + \frac{3680}{513}0.4095383 - \frac{845}{4104}0.4584971\right) \\ &= 0.4658452; \end{aligned}$$

$$\begin{aligned}
k_6 &= hf\left(t_0 + \frac{1}{2}h, w_0 - \frac{8}{27}k_1 + 2k_2 - \frac{3544}{2565}k_3 + \frac{1859}{4104}k_4 - \frac{11}{40}k_5\right) \\
&= 0.25\left(0.125, 0.5 - \frac{8}{27}0.375 + 2(0.3974609) - \frac{3544}{2565}0.4095383\right. \\
&\quad \left. + \frac{1859}{4104}0.4584971 - \frac{11}{40}0.4658452\right) \\
&= 0.4204789.
\end{aligned}$$

The two approximations to $y(0.25)$ are then found to be

$$\begin{aligned}
\tilde{w}_1 &= w_0 + \frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6 \\
&= 0.5 + \frac{16}{135}0.375 + \frac{6656}{12825}0.4095383 + \frac{28561}{56430}0.4584971 - \frac{9}{50}0.4658452 \\
&\quad + \frac{2}{55}0.4204789 \\
&= 0.9204870,
\end{aligned}$$

and

$$\begin{aligned}
w_1 &= w_0 + \frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5 \\
&= 0.5 + \frac{25}{216}0.375 + \frac{1408}{2565}0.4095383 + \frac{2197}{4104}0.4584971 - \frac{1}{5}0.4658452 \\
&= 0.9204886.
\end{aligned}$$

This also implies that

$$\begin{aligned}
R &= \frac{1}{0.25}\left|\frac{1}{360}k_1 - \frac{128}{4275}k_3 - \frac{2197}{75240}k_4 + \frac{1}{50}k_5 + \frac{2}{55}k_6\right| \\
&= 4\left|\frac{1}{360}0.375 - \frac{128}{4275}0.4095383 - \frac{2197}{75240}0.4584971 + \frac{1}{50}0.4658452 + \frac{2}{55}0.4204789\right| \\
&= 0.00000621388,
\end{aligned}$$

and

$$q = 0.84\left(\frac{\varepsilon}{R}\right)^{1/4} = 0.84\left(\frac{0.00001}{0.00000621388}\right)^{1/4} = 0.9461033291.$$

Since $q < 1$ we can accept the approximation 0.9204886 for $y(0.25)$ but we should adjust the step size for the next iteration to $h = 0.9461033291(0.25) \approx 0.2365258$. However, only the leading 5 digits of this result would be expected to be accurate because R has only about 5 digits of accuracy. Because we are effectively subtracting the nearly equal numbers w_i and $\tilde{w}_i$ when we compute R, there is a good likelihood of round-off error. This is an additional reason for being conservative when computing q.

The results from the algorithm are shown in Table 5.11. Increased accuracy has been used to ensure that the calculations are accurate to all listed places. The last two columns in Table 5.11 show the results of the fifth-order method. For small values of t, the error is less than the error in the fourth-order method, but the error exceeds that of the fourth-order method when t increases. ■

Table 5.11

		RKF-4				RKF-5	
t_i	$y_i = y(t_i)$	w_i	h_i	R_i	$\|y_i - w_i\|$	$\hat{w}_i$	$\|y_i - \hat{w}_i\|$
0	0.5	0.5			0.5		
0.2500000	0.9204873	0.9204886	0.2500000	6.2×10^{-6}	1.3×10^{-6}	0.9204870	2.424×10^{-7}
0.4865522	1.3964884	1.3964910	0.2365522	4.5×10^{-6}	2.6×10^{-6}	1.3964900	1.510×10^{-6}
0.7293332	1.9537446	1.9537488	0.2427810	4.3×10^{-6}	4.2×10^{-6}	1.9537477	3.136×10^{-6}
0.9793332	2.5864198	2.5864260	0.2500000	3.8×10^{-6}	6.2×10^{-6}	2.5864251	5.242×10^{-6}
1.2293332	3.2604520	3.2604605	0.2500000	2.4×10^{-6}	8.5×10^{-6}	3.2604599	7.895×10^{-6}
1.4793332	3.9520844	3.9520955	0.2500000	7×10^{-7}	1.11×10^{-5}	3.9520954	1.096×10^{-5}
1.7293332	4.6308127	4.6308268	0.2500000	1.5×10^{-6}	1.41×10^{-5}	4.6308272	1.446×10^{-5}
1.9793332	5.2574687	5.2574861	0.2500000	4.3×10^{-6}	1.73×10^{-5}	5.2574871	1.839×10^{-5}
2.0000000	5.3054720	5.3054896	0.0206668		1.77×10^{-5}	5.3054896	1.768×10^{-5}

An implementation of the Runge-Kutta-Fehlberg method is also available in Maple using the *InitialValueProblem* command. However, it differs from our presentation because it does not require the specification of a tolerance for the solution. For our example problem it is called with

$C := InitialValueProblem(deq, y(0) = 0.5, t = 2, method = rungekutta, submethod = rkf, numsteps = 10, output = information, digits = 8)$

As usual, the information is placed in a table that is accessed by double clicking on the output. The results can be printed in the method outlined in precious sections.

EXERCISE SET 5.5

1. Use the Runge-Kutta-Fehlberg method with tolerance $TOL = 10^{-4}$, $hmax = 0.25$, and $hmin = 0.05$ to approximate the solutions to the following initial-value problems. Compare the results to the actual values.
   a. $y' = te^{3t} - 2y, \quad 0 \le t \le 1, \quad y(0) = 0;$ actual solution $y(t) = \frac{1}{5}te^{3t} - \frac{1}{25}e^{3t} + \frac{1}{25}e^{-2t}$.
   b. $y' = 1 + (t - y)^2, \quad 2 \le t \le 3, \quad y(2) = 1;$ actual solution $y(t) = t + 1/(1 - t)$.
   c. $y' = 1 + y/t, \quad 1 \le t \le 2, \quad y(1) = 2;$ actual solution $y(t) = t \ln t + 2t$.
   d. $y' = \cos 2t + \sin 3t, \quad 0 \le t \le 1, \quad y(0) = 1;$ actual solution $y(t) = \frac{1}{2}\sin 2t - \frac{1}{3}\cos 3t + \frac{4}{3}$.
2. Use the Runge-Kutta Fehlberg Algorithm with tolerance $TOL = 10^{-4}$ to approximate the solution to the following initial-value problems.
   a. $y' = (y/t)^2 + y/t, \quad 1 \le t \le 1.2, \quad y(1) = 1,$ with $hmax = 0.05$ and $hmin = 0.02$.
   b. $y' = \sin t + e^{-t}, \quad 0 \le t \le 1, \quad y(0) = 0,$ with $hmax = 0.25$ and $hmin = 0.02$.
   c. $y' = (y^2 + y)/t, \quad 1 \le t \le 3, \quad y(1) = -2,$ with $hmax = 0.5$ and $hmin = 0.02$.
   d. $y' = t^2, \quad 0 \le t \le 2, \quad y(0) = 0,$ with $hmax = 0.5$ and $hmin = 0.02$.
3. Use the Runge-Kutta-Fehlberg method with tolerance $TOL = 10^{-6}$, $hmax = 0.5$, and $hmin = 0.05$ to approximate the solutions to the following initial-value problems. Compare the results to the actual values.
   a. $y' = y/t - (y/t)^2, \quad 1 \le t \le 4, \quad y(1) = 1;$ actual solution $y(t) = t/(1 + \ln t)$.
   b. $y' = 1 + y/t + (y/t)^2, \quad 1 \le t \le 3, \quad y(1) = 0;$ actual solution $y(t) = t\tan(\ln t)$.
   c. $y' = -(y + 1)(y + 3), \quad 0 \le t \le 3, \quad y(0) = -2;$ actual solution $y(t) = -3 + 2(1 + e^{-2t})^{-1}$.
   d. $y' = (t + 2t^3)y^3 - ty, \quad 0 \le t \le 2, \quad y(0) = \frac{1}{3};$ actual solution $y(t) = (3 + 2t^2 + 6e^{t^2})^{-1/2}$.

4. The Runge-Kutta-Verner method (see [Ve]) is based on the formulas

$$w_{i+1} = w_i + \frac{13}{160}k_1 + \frac{2375}{5984}k_3 + \frac{5}{16}k_4 + \frac{12}{85}k_5 + \frac{3}{44}k_6 \quad \text{and}$$
$$\tilde{w}_{i+1} = w_i + \frac{3}{40}k_1 + \frac{875}{2244}k_3 + \frac{23}{72}k_4 + \frac{264}{1955}k_5 + \frac{125}{11592}k_7 + \frac{43}{616}k_8,$$

where

$$k_1 = hf(t_i, w_i),$$
$$k_2 = hf\left(t_i + \frac{h}{6}, w_i + \frac{1}{6}k_1\right),$$
$$k_3 = hf\left(t_i + \frac{4h}{15}, w_i + \frac{4}{75}k_1 + \frac{16}{75}k_2\right),$$
$$k_4 = hf\left(t_i + \frac{2h}{3}, w_i + \frac{5}{6}k_1 - \frac{8}{3}k_2 + \frac{5}{2}k_3\right),$$
$$k_5 = hf\left(t_i + \frac{5h}{6}, w_i - \frac{165}{64}k_1 + \frac{55}{6}k_2 - \frac{425}{64}k_3 + \frac{85}{96}k_4\right),$$
$$k_6 = hf\left(t_i + h, w_i + \frac{12}{5}k_1 - 8k_2 + \frac{4015}{612}k_3 - \frac{11}{36}k_4 + \frac{88}{255}k_5\right),$$
$$k_7 = hf\left(t_i + \frac{h}{15}, w_i - \frac{8263}{15000}k_1 + \frac{124}{75}k_2 - \frac{643}{680}k_3 - \frac{81}{250}k_4 + \frac{2484}{10625}k_5\right),$$
$$k_8 = hf\left(t_i + h, w_i + \frac{3501}{1720}k_1 - \frac{300}{43}k_2 + \frac{297275}{52632}k_3 - \frac{319}{2322}k_4 + \frac{24068}{84065}k_5 + \frac{3850}{26703}k_7\right).$$

The sixth-order method $\tilde{w}_{i+1}$ is used to estimate the error in the fifth-order method w_{i+1}. Construct an algorithm similar to the Runge-Kutta-Fehlberg Algorithm, and repeat Exercise 3 using this new method.

5. In the theory of the spread of contagious disease (see [Ba1] or [Ba2]), a relatively elementary differential equation can be used to predict the number of infective individuals in the population at any time, provided appropriate simplification assumptions are made. In particular, let us assume that all individuals in a fixed population have an equally likely chance of being infected and once infected remain in that state. Suppose $x(t)$ denotes the number of susceptible individuals at time t and $y(t)$ denotes the number of infectives. It is reasonable to assume that the rate at which the number of infectives changes is proportional to the product of $x(t)$ and $y(t)$ because the rate depends on both the number of infectives and the number of susceptibles present at that time. If the population is large enough to assume that $x(t)$ and $y(t)$ are continuous variables, the problem can be expressed

$$y'(t) = kx(t)y(t),$$

where k is a constant and $x(t) + y(t) = m$, the total population. This equation can be rewritten involving only $y(t)$ as

$$y'(t) = k(m - y(t))y(t).$$

a. Assuming that $m = 100{,}000$, $y(0) = 1000$, $k = 2 \times 10^{-6}$, and that time is measured in days, find an approximation to the number of infective individuals at the end of 30 days.

b. The differential equation in part (a) is called a *Bernoulli equation* and it can be transformed into a linear differential equation in $u(t) = (y(t))^{-1}$. Use this technique to find the exact solution to the equation, under the same assumptions as in part (a), and compare the true value of $y(t)$ to the approximation given there. What is $\lim_{t\to\infty} y(t)$? Does this agree with your intuition?

6. In the previous exercise, all infected individuals remained in the population to spread the disease. A more realistic proposal is to introduce a third variable $z(t)$ to represent the number of individuals