$$K_1 = hf(t_n, y_n)$$
$$| K_1 - K_1^* | = h \, | f(t_n, y_n) - f(t_n, y_n^*) | \leqslant hL \, | y_n - y_n^* |$$
$$K_2 = hf(t_n + c_2h, y_n + a_{21}K_1)$$
$$| K_2 - K_2^* | = h \, | f(t_n + c_2h, y_n + a_{21}K_1) - f(t_n + c_2h, y_n^* + a_{21}K_1^*) |$$
$$\leqslant hL \, | y_n + a_{21}K_1 - y_n^* - a_{21}K_1^* |$$
$$\leqslant hL(1 + hLa_{21}) \, | y_n - y_n^* |$$
$$K_3 = hf(t_n + c_3h, y_n + a_{31}K_1 + a_{32}K_2)$$
$$| K_3 - K_3^* | \leqslant hL \, | y_n + a_{31}K_1 + a_{32}K_2 - y_n^* - a_{31}K_1^* - a_{32}K_2^* |$$
$$\leqslant hL(1 + a_{31}hL + a_{32}hL(1 + hLa_{21})) \, | y_n - y_n^* |$$

when we use (2.27), the increment function satisfies

$$| \phi(t_n, y_n, h) - \phi(t_n, y_n^*, h) |$$
$$= h^{-1} \, | w_1K_1 + w_2K_2 + w_3K_3 - w_1K_1^* - w_2K_2^* - w_3K_3^* |$$
$$\leqslant h^{-1}(w_1 \, | K_1 - K_1^* | + w_2 \, | K_2 - K_2^* | + w_3 \, | K_3 - K_3^* |)$$
$$\leqslant h^{-1}[w_1hL \, | y_n - y_n^* | + w_2hL(1 + hLa_{21}) \, | y_n - y_n^* |$$
$$+ w_3hL(1 + a_{31}hL + a_{32}hL(1 + hLa_{21})) \, | y_n - y_n^* |]$$
$$\leqslant L[(w_1 + w_2 + w_3) + (w_2a_{21} + w_3(a_{31} + a_{32})) hL$$
$$+ w_3a_{21}a_{32}(hL)^2] \, | y_n - y_n^* |$$

The use of Equation (2.24) yields

$$| \phi(t_n, y_n, h) - \phi(t_n, y_n^*, h) | \leqslant L \left(1 + \frac{1}{2}hL + \frac{1}{6}(hL)^2 \right) | y_n - y_n^* |$$

Therefore the increment function ϕ satisfies a Lipschitz condition in y and it is also continuous in h. Thus, we conclude that the third order Runge-Kutta method is convergent.

The Runge-Kutta methods are widely used for solving initial value problems. These methods provide approximations which converge to the true solution as $h \to 0$ and also have the advantage of self starting. The disadvantages of the Runge-Kutta methods are that they involve considerably more computation per step.

2.3.7 Approximation of truncation error

In the numerical solution of differential equations, it is desirable to have estimates of the local discretization (or truncation) errors of the solutions at each step. The estimate may be used not only to provide some idea of the errors, but also to indicate when to adjust the step size. If the magnitude of the estimate is greater than the preassigned upper bound, the step size is reduced to achieve smaller local errors. If the magnitude of the estimate is less than the preassigned lower bound, the step size is increased to save the computing time. For our discussion the rounding error will be ignored. A scheme for estimating the discretization error is called *extrapolation* or, sometimes *Richardson's extrapolation*. It is useful for calculation of the total (not local) truncation error for any method. If the function $f(t, y)$ is sufficiently differentiable and if p is the order of the numerical method, then from Theorem 2.3 we have

$$\epsilon_n = h^p\delta(t_n)+0(h^{p+1}) \tag{2.28}$$

where $\delta(t)$ is called the *magnified error function* and $\epsilon_n = y_n - y(t_n)$. Suppose we calculate $y(t)$ using a certain h and get $y_n(h)$. Then we repeat the calculation using $h/2$, and obtain $y_n(h/2)$. It follows from (2.28) that

$$y_n(h)-y(t_n) = h^p\delta(t_n)+0(h^{p+1})$$

$$y_n\left(\frac{h}{2}\right)-y(t_n) = \left(\frac{h}{2}\right)^p\delta(t_n)+0(h^{p+1}) \tag{2.29}$$

Hence
$$y_n(h)-y_n\left(\frac{h}{2}\right) = \left(1-\frac{1}{2^p}\right)h^p\delta(t_n)+0(h^{p+1}) \tag{2.30}$$

From the equations in (2.28) and (2.30) we obtain the equation

$$\epsilon_n = \frac{2^p}{2^p-1}\left[y_n(h)-y_n\left(\frac{h}{2}\right)\right]+0(h^{p+1})$$

Thus, we obtain the *Richardson* extrapolation to the true solution at the mesh point t_n

$$y(t_n) = \frac{2^p y_n\left(\frac{h}{2}\right)-y_n(h)}{2^p-1}+0(h^{p+1})$$

The right sides of the relations of the following

$$\epsilon_n \cong \frac{2^p}{2^p-1}\left(y_n(h)-y_n\left(\frac{h}{2}\right)\right) \tag{2.31}$$

$$y(t_n) \cong \frac{2^p y_n\left(\frac{h}{2}\right)-y_n(h)}{2^p-1} \tag{2.32}$$

help, respectively, to determine the estimate of the accumulated truncation error and the true solution at t_n with an error whose order exceeds the order of the singlestep method by one.

We denote the predicted accumulated error by P_n and the actual error in the extrapolated solution by T_n where

$$P_n = \frac{2^p}{2^p-1}\left(y_n(h)-y_n\left(\frac{h}{2}\right)\right)$$

and
$$T_n = \frac{2^p y_n\left(\frac{h}{2}\right)-y_n(h)}{2^p-1}-y(t_n)$$

We have estimated the truncation error at $t=5$ for the differential equation

$$\frac{dy}{dt} = -y^2, \quad y(0) = 1$$

when solved by the various order Runge-Kutta methods with step sizes $h = 2^{-m}$, $m = 4\,(1)\,8$.

Using the following methods:

(a) the second order Euler-Cauchy method, $p = 2$,
(b) the nearly optimal third order method, $p = 3$,
(c) the classical fourth order method, $p = 4$,

we have tabulated the error ϵ_n, the predicted error P_n, the extrapolated error T_n and the magnified error function $\delta(t_n)$ in Table 2.3.

TABLE 2.3 ESTIMATION OF THE TRUNCATION ERROR IN $y' = -y^2$, $y(0) = 1$, AT $t = 5$

Second order Euler-Cauchy method

h	ϵ_n	P_n	T_n	$\delta(t_n)$
2^{-4}	468629−10	471382−10	−275291−12	119969−07
2^{-5}	115093−10	115437−10	−344374−13	117855−07
2^{-6}	285149−11	285579−11	−430039−14	116797−07
2^{-7}	709647−12	710184−12	−537111−15	116269−07
2^{-8}	177009−12	177076−12	−671063−16	116005−07

Nearly optimal third order method

h	ϵ_n	P_n	T_n	$\delta(t_n)$
2^{-4}	−117753−11	−118324−11	570577−14	−482317−08
2^{-5}	−142199−12	−142547−12	348358−15	−465957−08
2^{-6}	−174700−13	−174916−13	215137−16	−457967−08
2^{-7}	−216493−14	−216627−14	133650−17	−454019−08
2^{-8}	−269447−15	−269530−15	832775−19	−452057−08

Classical fourth order method

h	ϵ_n	P_n	T_n	$\delta(t_n)$
2^{-4}	581973−14	58170[illegible]−14	269744−17	381402−09
2^{-5}	366262−15	365588−15	674144−18	384053−09
2^{-6}	235234−16	228794−16	644000−18	394657−09
2^{-7}	207396−17	143042−17	643540−18	556725−09
2^{-8}	732941−18	894086−19	643532−18	314796−08

From the numerical results we can draw the following conclusions:

(a) The predicted error P_n gives good estimate for the error value ϵ_n.
(b) The extrapolated error T_n is smaller than the corresponding predicted error P_n and compares favourably with the predicted error P_n of one order higher method.
(c) The value of the magnified error function $\delta(t_n)$ tends to a constant value as h decreases.

For $h_i = h_0 b^i$ with $0 < b < 1$, (2.38) becomes

$$Y_m^{(k)} = \frac{Y_{m-1}^{(k+1)} - b^{mr} Y_{m-1}^{(k)}}{1 - b^{mr}} \tag{2.39}$$

Equation (2.39) for $b = 1/2$ simplifies to

$$Y_m^{(k)} = \frac{2^m Y_{m-1}^{(k+1)} - Y_{m-1}^{(k)}}{2^m - 1}, \quad r = 1 \tag{2.40}$$

and

$$Y_m^{(k)} = \frac{4^m Y_{m-1}^{(k+1)} - Y_{m-1}^{(k)}}{4^m - 1}, \quad r = 2 \tag{2.41}$$

From Equation (2.38), we notice that each $Y_m^{(k)}$ is a linear combination of $y(t, h_i)$, $i = k, k+1, \ldots, k+m$, which can be written in the form

$$Y_m^{(k)} = \sum_{j=0}^{m} c_{m,\, m-j}\, Y_0^{(k+j)} \tag{2.42}$$

where $c_{m,\, m-j}$ are constant coefficients.

Substituting (2.42) into (2.38) we get the recursion relation in the coefficients as

$$c_{m,\, m-j} = \frac{h_k^r\, c_{m-1,\, m-j} - h_{k+m}^r\, c_{m-1,\, m-1-j}}{h_k^r - h_{k+m}^r} \tag{2.43}$$

$$c_{m-1,\, m} = c_{m-1,\, -1} = 0$$

Using (2.43) and (2.42) we may write

$$\begin{bmatrix} Y_0^{(0)} \\ Y_1^{(0)} \\ \vdots \\ Y_m^{(0)} \end{bmatrix} = \begin{bmatrix} c_{00} & & & \\ c_{11} & c_{10} & & \\ c_{22} & c_{21} & c_{20} & \\ \vdots & & & \ddots \\ c_{mm} & & & c_{m0} \end{bmatrix} \begin{bmatrix} Y_0^{(0)} \\ Y_0^{(1)} \\ \vdots \\ Y_0^{(m)} \end{bmatrix}$$

We know that the numerical methods of interest converge as the step size tends to zero, i.e.

$$\lim_{k \to \infty} y(t, h_k) = Y_0^{(k)} = y(t) \tag{2.44}$$

The convergence of $Y_m^{(0)}$ to $y(t)$ can be seen from Equation (2.37). Whenever $\tau_m \neq 0$, $m = 1, 2, \ldots$, this equation states that each column of the Y-scheme converges to $y(t)$ faster than the preceding one; and, in fact, the principal diagonal converges faster than any column.

We illustrate the extrapolation method as applied to Euler's method.

2.4.1 Euler extrapolation method

The approximate value $y_n(h)$ is obtained from the algorithm

$$y_{n+1} = y_n + h f(t_n, y_n), \quad n = 0, 1, 2, \ldots$$

Since $p = 1$ for Euler's method, from (2.28) the approximate value $y_n(h)$ to $y(t_n)$ has the asymptotic expansion of the form

$$y(t_n, h) = y(t_n) + \tau_1(t_n)h + \tau_2(t_n)h^2 + \dots$$

We use step lengths h_0, $h_0/2$, $h_0/2^2, \dots, h_0/2^k$ and generate $Y_0^{(k)}$. In Euler's method, we know y_n at t_n and advance the computation from t_n to t_{n+1} to find y_{n+1}. We take $t_{n+1} - t_n = h_0$ and start the procedure by computing y_{n+1} with step length h_0 and denote it by $Y_0^{(0)}$, i.e.

$$Y_0^{(0)} = y_n + h_0 f_n$$

Next, we put $h_1 = h_0/2$ and apply Euler's method twice to obtain $y(t_{n+1}, h_1)$ at t_{n+1},

$$Y_0^{(1)} = y(t_{n+1}, h_1)$$

for $h_2 = h_0/2^2$, we apply Euler's method four times. Similarly, for $h_k = h_0/2^k$, we apply Euler's method 2^k times to obtain $Y_0^{(k)}$ The above procedure gets simplified if we consider the initial value problem

$$y' = \lambda y,\ y(t_0) = y_0$$

and we obtain

$$Y_0^{(0)} = (1 + \lambda h_0) y_n$$

$$Y_0^{(1)} = \left(1 + \frac{\lambda h_0}{2}\right)^2 y_n$$

$$\vdots$$

$$Y_0^{(k)} = \left(2 + \frac{\lambda h_0}{2^k}\right)^{2^k} y_n$$

The convergence of $Y_0^{(k)}$ to the exact value for $h_0 = 1$ and $\lambda = 1$ is shown in Figure 2.2.

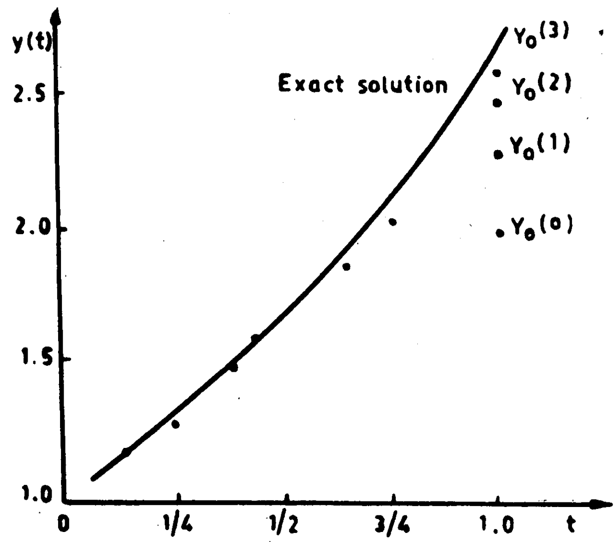

Fig. 2.2 Solution of $y' = y$, $y(0) = 1$ by Euler extrapolation method

After determining the first column $Y_0^{(k)}$ of the Y-scheme, we obtain the other columns with the help of relation (2.40)

$$Y_m^{(k)} = \frac{2^m\, Y_{m-1}^{(k+1)} - Y_{m-1}^{(k)}}{2^m - 1}, \quad m=1, 2, 3, \ldots$$

The value of m is chosen by comparing the two successive values $Y_{m-1}^{(0)}$ and $Y_m^{(0)}$ and we increase m till this difference is within the prescribed tolerance ϵ.

When the convergence is obtained, $Y_m^{(0)}$ is used as y_{n+1} and the procedure is repeated to obtain y_{n+2}.

2.5 STABILITY ANALYSIS

While numerically solving an initial value problem for ordinary differential equations, an error is introduced at each integration step due to the inaccuracy of the formula. The magnitude of this so called local truncation error is a measure of the accuracy of the integration formula. The magnitude of the total error depends on the magnitude of the local truncation errors and their propagation. Even when the local error at each step is small, the total error may become large due to accumulation and amplification of these local errors. This growth phenomenon is called *numerical instability*. To understand this, consider the simple linear first order differential equation

$$y' = \lambda y, \; y(t_0) = y_0 \tag{2.45}$$

where λ is a constant. It can be seen that, to a first order approximation, the results obtained from a stability analysis on the above linear equation can be extended to a nonlinear case

$$y' = f(t, y), \; y(t_0) = y_0 \tag{2.46}$$

where $\partial f/\partial y$ from Equation (2.46) plays role similar to that of the constant λ in Equation (2.45).

The nonlinear function $f(t, y)$ can be linearized by expansion of the function about the point (t_n, y_n) in the Taylor series truncated after first order terms. The resulting linearized form for Equation (2.46) is given by

$$y' = \lambda y + Bt + C \tag{2.47}$$

where

$$\lambda = \left(\frac{\partial f}{\partial y}\right)_n,$$

$$B = \left(\frac{\partial f}{\partial t}\right)_n,$$

$$C = \left[f_n - y_n\left(\frac{\partial f}{\partial y}\right)_n - t_n\left(\frac{\partial f}{\partial t}\right)_n\right]$$