These update formulas could come from Taylor's method, RKF etc...

Assuming $y_i = \tilde{y}_i = y(t_i)$

$$\underbrace{y_{i+1} - y(t_{i+1})}_{\substack{\text{LTE for ①} \\ = O(h^{n+1})}} = \underbrace{\tilde{y}_{i+1} - y(t_{i+1})}_{\substack{\text{LTE for ②} \\ = O(h^{n+2})}} + y_{i+1} - \tilde{y}_{i+1}$$

$$\Rightarrow \quad \underbrace{y_{i+1} - y(t_{i+1})}_{\text{LTE for ①}} = y_{i+1} - \tilde{y}_{i+1} + O(h^{n+2}) \qquad (*)$$

How do we use this estimate of LTE for ① to get

LTE for ① $\leq \delta$ = prescribed tolerance ?

We use the simplifying assumption:

$$\tau_{i+1}(h) = |y_{i+1} - y(t_{i+1})| \approx K h^{n+1} \quad (\text{LTE for ①})$$

If we were to have a new step: $h_{new} = qh \quad (q>0)$

then our estimate will give an LTE:

$$\tau_{i+1}(qh) \approx K(qh)^{n+1} = q^{n+1} K h^{n+1} \approx q^{n+1} \tau_{i+1}(h)$$

$$\underset{(*)}{\approx} q^{n+1} |y_{i+1} - \tilde{y}_{i+1}|$$

Hence requiring that LTE for ① $\leq \delta$ means (approx)

$$\boxed{\tau_{i+1}(qh) \approx q^{n+1} |y_{i+1} - \tilde{y}_{i+1}| \leq \delta}$$

Since power is monotonic:

$$q \le \left(\frac{\delta}{|y_{i+1} - \tilde{y}_{i+1}|} \right)^{\frac{1}{n+1}}$$

$\Rightarrow$ we can set:

$$\boxed{h_{new} = h \left(\frac{\delta}{|y_{i+1} - \tilde{y}_{i+1}|} \right)^{\frac{1}{n+1}}}$$

However many recommend to enforce the more strict inequality:

$$\tau_{i+1}(qh) \approx q^{n+1} |y_{i+1} - \tilde{y}_{i+1}| \le \delta \frac{qh}{2}$$

$$\Rightarrow \quad q \le \left(\frac{\delta h}{2|y_{i+1} - \tilde{y}_{i+1}|} \right)^{\frac{1}{n}} \approx 0.84 \left(\frac{\delta h}{|y_{i+1} - \tilde{y}_{i+1}|} \right)^{\frac{1}{n}}$$

for $n=4$

hence rule used in B&F:

$$\boxed{h_{new} = h(0.84) \left(\frac{\delta h}{|y_{i+1} - \tilde{y}_{i+1}|} \right)^{\frac{1}{n}}}$$

Fehlberg (1969) used order 4 RK method w/ 5 fun evals / iter
coupled w/ — 5 ——— w/ 6 ———

= Total of 11 fun evals / iter? Not very efficient...
However the coeff in Fehlberg's method are carefully chosen so that only 6 fun eval/ iter are needed.
This an example of embedded RK methods: the eval pts for the 4-th order method match those of 5-th order.

The updates take the form:

$$y_{i+1} = y_i + \sum_{j=1}^{6} a_j F_j$$

$$\tilde{y}_{i+1} = \tilde{y}_i + \sum_{j=1}^{6} b_j F_j$$

where the F_j are of the form:

$$F_i = h f\left(t_i + c_i h,\ y_i \sum_{j=1}^{i-1} d_{ij} F_j\right)$$

= depends on previous F_j.

cf. book for coeff in RKF 4-5 method (ode45 in Matlab)

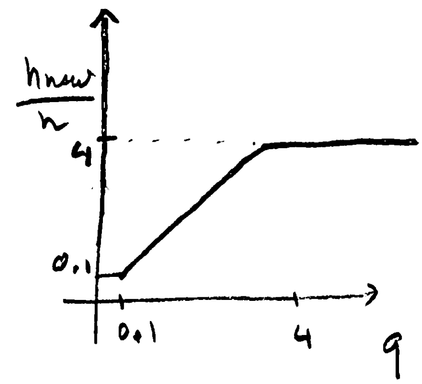
sometimes only $a_i - b_i$ are specified instead of b_i.

The complete algo is:

$y_0 = \alpha$, $h = h_{max}$, flag $= 1$

while (flag $= 1$)

- compute $F_1 \dots F_6$
- compute $e = |y_{i+1} - \tilde{y}_{i+1}|$ (* a lin. comb of F_i, w/ coeff given in a table *)
- if $e \leq \delta$
  - (* accept step *)
  - $t = t + h$
  - $y_{i+1} = y_i + \sum_{j=1}^{6} b_j F_j$ (* use higher order method *)
- set $q = 0.84\,(\delta/e)^{\frac{1}{4}}$



- set $h = \begin{cases} 0.1h & \text{if } q \leq 0.1 \\ 4h & \text{if } q \geq 4 \\ qh & \text{otherwise} \end{cases}$
- set $h = \min(h, h_{max})$
- if $t \geq b$
  - flag $= 0$ (* we are done *)

  else
  - if $t + h > b$
    - $h = b - t$ (* adjust last step so that last y_i falls at $t = b$ *)
- if $h < h_{min}$
  - flag $= 0$
  - error: step size too small - abnormal termination

(normal termination)