2.6 Runge-Kutta Method

huge family of methods

we will look at Runge-Kutta order 4 method.

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)h$$
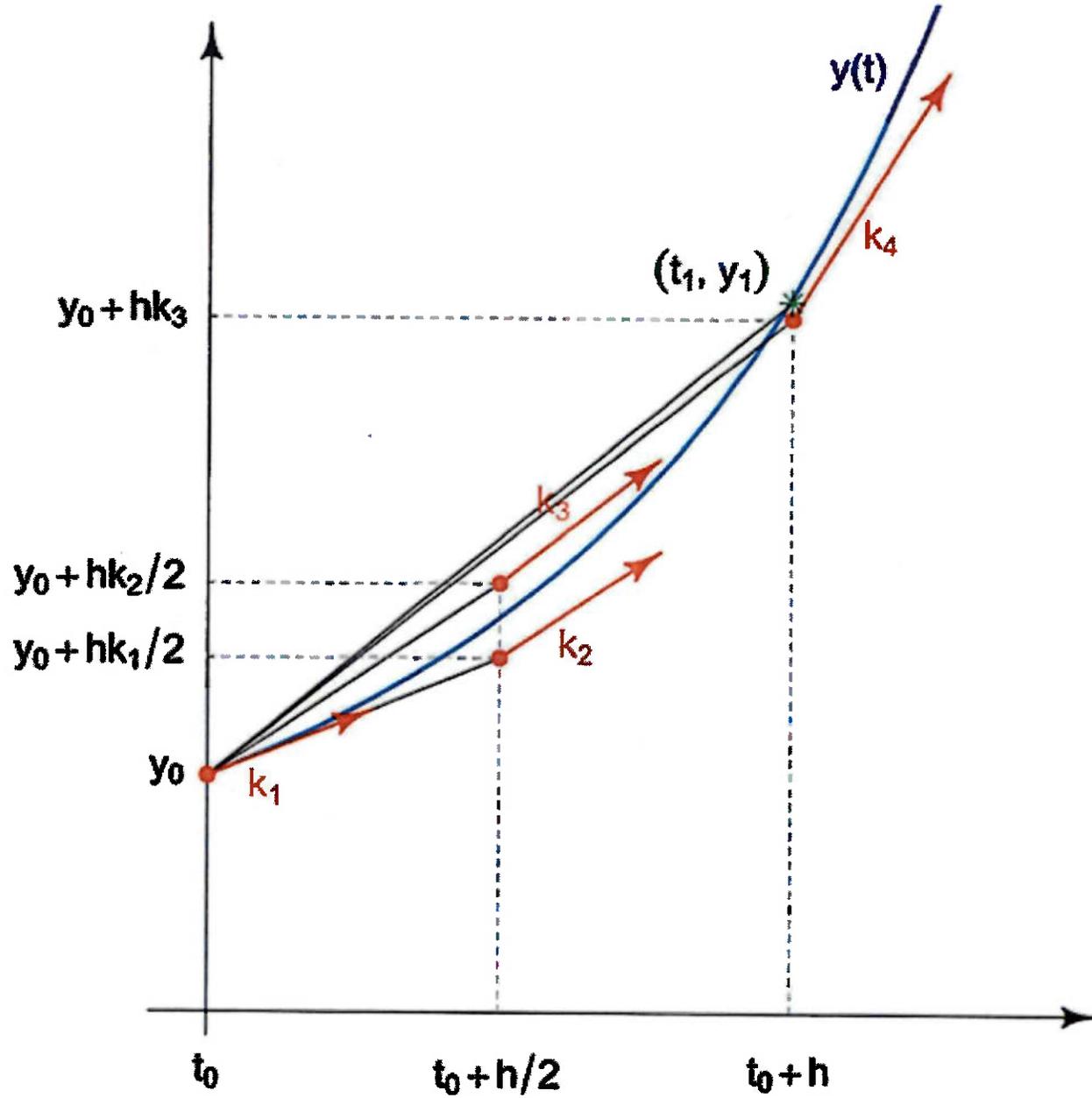
$k_1 = f(t_n, y_n)$ slope at start (Euler)

$k_2 = f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1 h)$ slope at mid pt using k_1 to get there

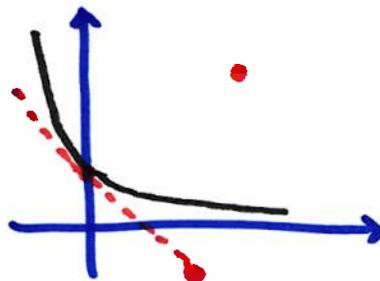$k_3 = f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}k_2 h)$ slope at midpt using k_2 to get there

$k_4 = f(t_n + h, y_n + k_3 h)$ slope at end using k_3 to get there

$\frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$ is weighted avg. of the slopes

(more weight to middle)

advantages: stability

$$y' = -10y$$

could blow up if
h is too big

w/ RK4 that is less likely to happen even w/ big h

errors: Euler has ~~bad~~ global error $\sim h$ (on the order of
(decrease error by factor of 10 →
10 times as many steps)

Imp. Euler has global error $\sim h^2$
(doubling steps → decrease error
by factor of 4)

RK4: $\sim h^4$
(double step count → decrease error
by factor of 16)

example

$y' = 1 - t + 4y \quad y(0) = 1$

$h = 0.1$, estimate $y(0.1)$

$\left. \begin{array}{l} t_0 = 0 \\ y_0 = 1 \end{array} \right\}$ given

$t_1 = t_0 + h = 0.1$ (target)

$k_1 = f(t_0, y_0) = 1 - (0) + 4(1) = 5$

$k_2 = f(t_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_1 h)$

$\quad = 1 - (0 + 0.05) + 4(1 + \frac{1}{2} \cdot 5 \cdot 0.1) = 5.95$

$k_3 = f(t_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2 h)$

$\quad = 1 - (0 + 0.05) + 4(1 + \frac{1}{2} \cdot 5.95 \cdot 0.1) = 6.14$

$k_4 = f(t_0 + h, y_0 + k_3 h)$

$\quad = 1 - (0 + 0.1) + 4(1 + 6.14 \cdot 0.1) = 7.356$

$y_1^* = y_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)h$

$\quad = 1.60893$

How good? RK4 w/ h=0.1 → 1.60893 ⟩ error 1.085×10^{-4}

true value → 1.60904

RK4 w/ h=0.05 → 1.609034   error 8×10^{-6}

(decrease by factor of 14)

Euler w/ h=0.1 → 1.5 (1 step) error 1×10^{-1}

w/ h=0.01 → 1.59529 (10 steps) error 1×10^{-2}

w/ h=0.005 → 1.60206

Imp. Euler w/ h=0.1 → 1.595
(RK 2)   w/ h=0.01 → 1.60886

given h, RK4 will always better (at least no worse)
than Euler

beyond order 4, math gets messy

Runge-Kutta-Fehlberg (RKF45)

The RKF45 method uses six stages to calculate both a 4th-order and a 5th-order approximation, allowing for adaptive step-size control.

$$k_1 = f(t_n, y_n)$$

$$k_2 = f(t_n + \frac{1}{4}h, y_n + \frac{1}{4}hk_1)$$

$$k_3 = f(t_n + \frac{3}{8}h, y_n + \frac{3}{32}hk_1 + \frac{9}{32}hk_2)$$

$$k_4 = f(t_n + \frac{12}{13}h, y_n + \frac{1932}{2197}hk_1 - \frac{7200}{2197}hk_2 + \frac{7296}{2197}hk_3)$$

$$k_5 = f(t_n + h, y_n + \frac{439}{216}hk_1 - 8hk_2 + \frac{3680}{513}hk_3 - \frac{845}{4104}hk_4)$$

$$k_6 = f(t_n + \frac{1}{2}h, y_n - \frac{8}{27}hk_1 + 2hk_2 - \frac{3544}{2565}hk_3 + \frac{1859}{4104}hk_4 - \frac{11}{40}hk_5)$$

Final Estimates

The 4th-order estimate and the 5th-order estimate are calculated as follows:

$$y_{n+1} = y_n + h\left(\frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5\right)$$

$$\hat{y}_{n+1} = y_n + h\left(\frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6\right)$$

compare 4th-order to 5th-order

if |difference| is small → h is ok, move on

if |difference| > tolerance → h is too big. refine, repeat
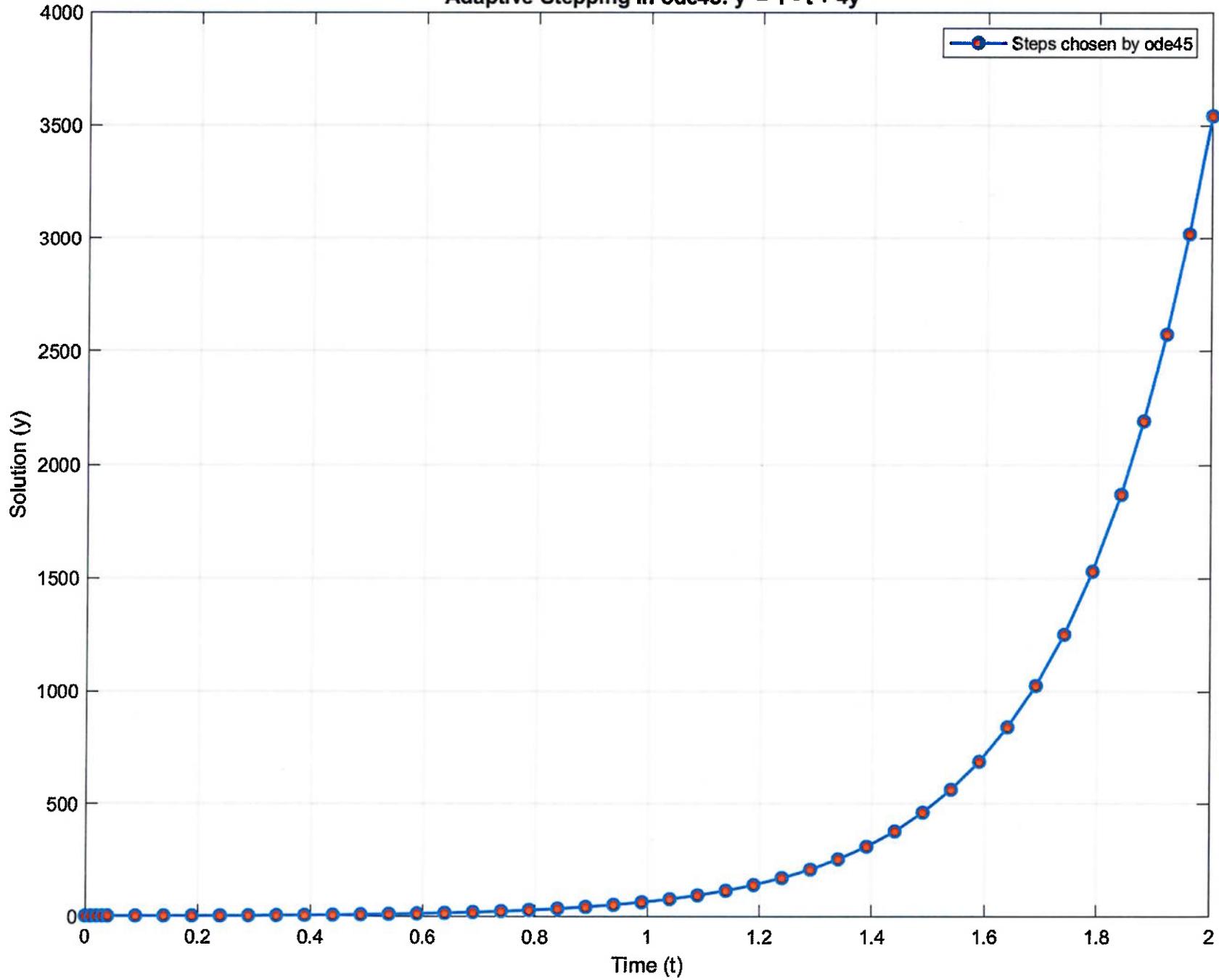
ode45 in Matlab uses a different RK45 (Dormand-Prince)

→ change h dynamically and refine as need

Straight portion: big h ok

Curvy portion: small step size needed

Adaptive Stepping in ode45: y' = 1 - t + 4y

ode45 reacting to a 'Sharp Event'