Runge-Kutta (RK) methods

characterized by number of stages and order, very well studied

1^{st} order RK is Euler.

Most famous and popular: classical 4^{th} order RK, has 4 stages K_i

rk4

$$Y_0 = y_0$$
$$Y_{n+1} = Y_n + \frac{\Delta t}{6}\left[K_1 + 2K_2 + 2K_3 + K_4\right]$$

with stages
$$K_1 = F(t_n, Y_n)$$
$$K_2 = F\left(t_n + \frac{\Delta t}{2},\ Y_n + \frac{\Delta t}{2}\cdot K_1\right)$$
$$K_3 = F\left(t_n + \frac{\Delta t}{2},\ Y_n + \frac{\Delta t}{2}\cdot K_2\right)$$
$$K_4 = F\left(t_n + \Delta t,\ Y_n + \Delta t\cdot K_3\right)$$

It is 4^{th} order accurate, i.e. discretization error $= O(\Delta t^4)$

so $\frac{\Delta t}{2}$ reduces error by $\frac{1}{2^4} = \frac{1}{16}$

It has 4 stages, so requires 4 function evaluations, so 4 times costlier than Euler,

so about 4 times more expensive per step than Euler,

but achieves much higher accuracy, so perhaps can use larger Δt.

Classical RK4 is optimal: uses 4 evaluations for 4^{th} order

but any 5^{th} order RK needs 6 stages

6^{th} 7 or 8 stages

RKF (rk45): Fehlberg (1969) devised the famous RKF method;

adaptive

uses a 4^{th} order and a 5^{th} order, with total 6 evaluations per step

to estimate local error and adapt (select) next Δt.

If error is low, increase Δt

if error is high, decrease Δt

hoping to get to t_{end} in fewer steps overall.

works well on many ODEs, excellent implementations exist (rk45 in Matlab)

(rksuite package)