Previously we saw that the Euler method had order $\mathcal{O}(h)$.

Midpoint method, Runge-Kutta2 (RK2):

Also called the midpoint method. We want to draw tangent lines, just like with the Euler method, but halfway, we want to compute a new line with half the slope of the previous one.

Slope at $t_{n+\frac{1}{2}}$ is $k_1 = f(t_n, y_n)$. While slope at t_{n+1} is
So then the RK2 method decides y_{n+1} via the formula

$$\begin{aligned} k_1 =& f(t_n, y_n) \\ k_2 =& f(t_n + \frac{h}{2},\ y_n + \frac{h}{2} f(t_n, y_n)) \\ y_{n+1} =& y_n + hk_2. \end{aligned}$$

This method is then explicit, there is no need to solve systems of equations or whatever.

Order of RK2

Taking the test-equation $y' = \lambda y$ for $t \in [0, 1]$. Inserting gives us

$$\begin{aligned} k_1 =& \lambda y_n \\ k_2 =& \lambda \left(y_n + \frac{h}{2} k_1 \right). \\ y_{n+1} =& y_n + hk_2. \end{aligned}$$

The number of FLOPs per time step here then is equal to 7. Compare this with Euler Forward that instead has 3 FLOPs per time step. The trade-off seems to be quite big then. But it actually would need to depend on your error tolerance.

Assume $\mathcal{O}(h^2)$ means that the error = h^2. If we have an error tolerance of 10^{-4}. Then that means we require our error to be $< 10^{-4}$. Then $h < 10^{-2}$ with RK2, but with Euler forward, we instead need to have $h < 10^{-4}$.

In an interval of length 1, RK2 requires $\frac{1}{h} = \frac{1}{10^{-2}} = 100$ steps.
Meanwhile with Euler Forward we would need $\frac{1}{h} = \frac{1}{10^{-4}} = 10000$ steps.

Further, the total FLOPs will be

- RK2: $7 \cdot 100 = 700$
- Euler: $3 \cdot 10000 = 30000.$

So the question becomes, at what error tolerance does Euler have less FLOPs?

Basically we want to solve the equation

$$\begin{aligned} & \text{FLOP \_ RK2} = \text{ FLOP \_ EULER} \\ \implies & 7 \cdot \text{N\_of \_steps\_RK2} = 3 \cdot \text{N\_ ofsteps\_Euler} \\ \implies & 7 \cdot \frac{1}{\sqrt{\text{tol}}} = 3 \cdot \frac{1}{\text{tol}} \\ \implies & \text{tol} = \left(\frac{3}{7}\right)^2 \approx 0.183\ldots \end{aligned}.$$

So when tolerance is pretty big (10^{-2}), Euler actually gives less FLOPs!

When do we need high accuracy?

- We are simulating a physical/real-life scenario that needs an accurate representation.
- The ODE is extremely sensitive to error propagation, such that a small error in the solution at time t_n has large consequences at $t > t_n$.

An example of the second type is given by the pitchfork bifurcation:

$$y' = ry - y^3.$$

General Runge-Kutta (explicit)

$$\begin{aligned} y_{n+1} = & y_n + h\sum_{i=1}^{S} b_i k_i \\ k_1 = & f(t_n, y_n) \\ k_2 = & f(t_n + c_2 h,\ \ y_n + h(a_{21}k_1)) \\ k_3 = & f(t_n + c_3 h,\ \ y_n + h(a_{31}k_1 + a_{32}k_2)) \\ & \vdots \\ k_s = & f(t_n + c_s h,\ \ y_n + h(a_{s1}k_1 + \cdots + a_{s,s-1}k_{s-1})) \end{aligned}.$$

A specification of s, a_{ij}, b_i, c_i gives a specific Runge-Kutta method. Often presented in a Butcher table:

Implicit Higher Order Methods

Used for high accuracy, and stability (?)

Crank-Nicholson

Given by the formula

$$y_{n+1} = y_n + h\frac{1}{2}(f(t_n, y_n) + f(t_{n+1}, y_{n+1})).$$

Multistep methods

The Euler methods are single-stepped, as well as RK. Meaning each computation only depends on the previous/next one. This leads to few saved variables, meaning less computations.

Two step method: Adams-Bashfort (AB)

Given by the formula

$$y_{n+2} = y_{n+1} + \frac{3}{2}hf(t_{n+1}, y_{n+1}) - \frac{1}{2}hf(t_n, y_n)..$$

The implicit version of this is called **Adams-Moulton**:

$$y_{n+2} = y_{n+1} + h\left(\frac{5}{12}f(t_{n+2}, y_{n+2}) + \frac{8}{12}f(t_{n+1}, y_{n+1}) - \frac{1}{12}f(t_n, y_n)\right).$$

Adaptivity

When one adapts (in our case) the step-size h depending on the expected error order.

Methods of different order can be used together for adaptivity. Meaning you don't need to have to same step-size h throughout the whole interval. This is useful when computing certain sub-intervals where the error might be larger.

So how do we determine when the error is large? By using two methods with different order. For example, RK4 and RK5 are usually used. Then one compares each solution, and then computes the difference. When the difference is high, the error in the lower order method is high(in our case RK4). Then it's advantageous to reduce h in those areas.