1 Solving ODE with four methods

A decaying radioactive element changes its concentration according to the following ODE:

$$\frac{dy}{dt} = f(t, y) = -\lambda y \tag{1}$$

The analytical solution reads

$$y(t) = y_0 exp(-\lambda t) \tag{2}$$

We have already implement the forward and backward Euler methods :

- The forward Euler method reads :

$$y_{n+1} = y_n + h \cdot f(t_n, y_n)$$

- The backward Euler algorithm uses the following step formula :

$$y_{n+1} = y_n + h \cdot f(t_{n+1}, y_{n+1})$$

Butcher tableaus for explicit Runge-Kutta methods can be read as :

$$\begin{array}{c|ccccc} 0 & & & & & \\ c_2 & a_{21} & & & & \\ c_3 & a_{31} & a_{32} & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ c_m & a_{m1} & a_{m2} & \cdots & a_{mm-1} & \\ \hline & b_1 & b_2 & \cdots & b_{m-1} & b_m \end{array} \Leftrightarrow \begin{array}{l} k_1 = f(t_n, y_n) \\ k_2 = f(t_n + hc_2, y_n + ha_{21}k_1) \\ k_3 = f(t_n + hc_3, y_n + h(a_{31}k_1 + a_{32}k_2)) \\ \vdots \\ k_m = f\left(t_n + hc_m, y_n + h\left(\sum_{i=1}^{m-1} a_{mi}k_i\right)\right) \\ y_{n+1} = y_n + h\left(\sum_{j=1}^{m} b_j k_j\right) \end{array}$$

Solve the radioactive decay problem (1) using the 2nd order Heun method and «the» 4th order RK method using the conditions $y_0 = 1$, $\lambda = 1$ and $h = 0.1$ from $t_0 = 0$ to $t_{End} = 10$.

- Define four new functions such as

```
function [t,y] = eulerForward(f,t0,tend,y0,h)
function [t,y] = eulerBackward(f,t0,tend,y0,h)
function [t,y] = Heun2(f,t0,tend,y0,h)
function [t,y] = RK4(f,t0,tend,y0,h)
```

  to be called in your main code.

- Note that you cannot just put the backward Euler formula into Matlab! Use `fsolve` to solve for y_{n+1} with the same conditions as for the forward Euler.

- You might want to use these to avoid spamming you command line

```
options = optimset('display','off');
y(i+1)= fsolve( ........, options);
```

- The Butcher tableaus for the methods are :

Heun2:
$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & 1/2 & 1/2 \end{array}$$

RK4:
$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array}$$

- As shown in the slides, discretize your integration steps until $t_{End} + h$ and interpolate the final value between t_{End} and $t_{End} + h$.
- Compare the orders of accuracy for the four methods (Forward and Backward Euler, 2nd order and 4th order RK) plotting the global truncation errors of the last element defined as $e_n = y(t_n) - y_n$ against h with `h=logspace(-4,0,8)` in a double logarithmic plot `plot(loglog)`. Note that a method has order of accuracy p if $e_n = C \cdot h^p$.

2 Van der Waals equation

The van der Waals equation for some non-ideal gas reads :

$$P = \frac{2.4}{V - \frac{1}{3}} - \frac{3}{V^2} \tag{3}$$

The analytical derivative reads

$$\frac{dP}{dV} = \frac{6}{V^3} - \frac{2.4}{(V - \frac{1}{3})^2} \tag{4}$$

- Plot the van der Waals equation for 1000 points between V = 0.34 and V = 4
- Try to approximate the equation by solving the ODE

  - Use `ode45, tSpan = [0.34, 4];` and `y0 = P(0.34);`

  - Note that the ODE does not depend on the solution, but only the independent variable (i.e. the first input into your odefun!)

  - Plot the solution of the ODE together with the analytical solution and zoom in to `ylim([0, 2])`. What do you observe?

  - Try the same with `ode15s`. What do you observe when you zoom in?
- Plot dP/dV against V in the range we considered. What might be the problem with the solvers, considering what they have to do in the slope field?
- Tighten the tolerances using

```
options = odeset(AbsTol, newAbsTol, RelTol, newRelTol);
[t,y] = ode45( ......, options);
```

  The defaults are 1e-3 (relative) and 1e-6 (absolute).
  Plot the solutions again and zoom in to `ylim([0, 2])`.

3 Runge Method

Compute the stability function of the Runge method (also known as the explicit midpoint method (EM)).