DEN: ODE - numerical approach

Algorithm (Euler (forward) formula for IVP $y' = f(x, y)$)
Given: ODE $y' = f(x, y)$ on $[x_0, x_0 + T]$, initial value y_0, and $n \in \mathbb{N}$.
0. Set $h = \frac{T}{n}$.
1. x_0 and y_0 are given.
2. For $i = 0, \ldots, n-1$ set $x_{i+1} = x_i + h$ and $y_{i+1} = y_i + f(x_i, y_i) \cdot h$.

Definition.
Consider some numerical method for solving initial value problems $y' = f(x, y(x))$, $y(x_0) = y_0$ that for given $T > 0$ and $n \in \mathbb{N}$ produces approximations $\{(x_0, y_0), \ldots, (x_n, y_n)\}$ of the solution on $[x_0, x_0 + T]$. We say that this method is **convergent** if the following is true:
For any IVP with f Lipschitz in the second variable that has a solution $y(x)$ on some $[x_0, x_0 + T]$ and every $n \in \mathbb{N}$, consider the corresponding approximations $\{(x_0, y_0), \ldots, (x_n, y_n)\}$ and define

$$E_n = \max_i |y(x_i) - y_i|.$$

We require that $E_n \to 0$.

Definition.
Consider a one-step method Φ_f for solving initial value problems that for an equation $y' = f(x, y(x))$ and a point (x^*, y^*) generates an estimate $\Phi_f(x^*, y^*, h)$ for the value of the corresponding solution at $x^* + h$.
Given an ODE $y' = f(x, y)$, we define the **local error** of the method as

$$d_f(x^*, y^*, h) = y_*(x^* + h) - \Phi_f(x^*, y^*, h)$$

for all $x^*, y^* \in \mathbb{R}$ and all step sizes $h > 0$ such that the IVP $y' = f(x, y)$, $y(x^*) = y^*$ has a solution $y_*(x)$ on $[x^*, x^* + h]$.

Definition.
Consider a one-step method Φ_f for solving initial value problems that for an equation $y' = f(x, y(x))$ and a point (x^*, y^*) generates an estimate $\Phi_f(x^*, y^*, h)$ for the value of the corresponding solution at $x^* + h$.
We say that the method is of **order** p, or that it has error of order p, if the following is true: For every differential equation $y' = f(x, y)$ and rectangle $I \times J$ such that f is Lipschitz with respect to y in $I \times J$ and sufficiently smooth there is some $C > 0$ so that

$$|d_f(x^*, y^*, h)| \leq Ch^{p+1}$$

for all $(x^*, y^*) \in I \times J$ and $h > 0$ such that $(x^* + h, \Phi_f(x^*, y^*, h)) \in I \times J$.

Theorem.
The Euler method is of order 1 with respect to differential equations $y' = f(x, y)$ such that f is differentiable on its domain and its derivatives are bounded on bounded rectangles.

Algorithm (backward Euler formula for IVP $y' = f(x, y)$)

Given: ODE $y' = f(x, y)$ on $[x_0, x_0 + T]$, initial value y_0, and $n \in \mathbb{N}$.
0. Set $h = \frac{T}{n}$.
1. x_0 and y_0 are given.
2. For $i = 0, \ldots, n-1$ set $x_{i+1} = x_i + h$ and solve $y_{i+1} = y_i + f(x_{i+1}, y_{i+1}) \cdot h$ for y_{i+1}.

Algorithm (Heun formula (improved Euler formula) for IVP $y' = f(x, y)$)

Given: ODE $y' = f(x, y)$ on $[x_0, x_0 + T]$, initial value y_0, and $n \in \mathbb{N}$.
0. Set $h = \frac{T}{n}$.
1. x_0 and y_0 are given.
2. For $i = 0, \ldots, n-1$ set $x_{i+1} = x_i + h$ and:
a) Estimate the slope $y'(x_i)$: $k_1 = f(x_i, y_i)$.
b) Estimate y_{i+1}: $y^*_{i+1} = y_i + k_1 h$, then estimate the slope $y'(x_{i+1})$: $k_2 = f(x_{i+1}, y^*_{i+1})$.
c) Set $y_{i+1} = y_i + \frac{1}{2}(k_1 + k_2) \cdot h$.

Fact.
The Heine formula method is of order 2.

Algorithm (**RK2** (midpoint, modified Euler formula, improved polygon) for IVP $y' = f(x,y)$)
Given: ODE $y' = f(x,y)$ on $[x_0, x_0+T]$, initial value y_0, and $n \in I\!N$.
0. Set $h = \frac{T}{n}$.
1. x_0 and y_0 are given.
2. For $i = 0, \dots, n-1$ set $x_{i+1} = x_i + h$ and:
a) Estimate the slope $y'(x_i)$: $k_1 = f(x_i, y_i)$.
b) Estimate $y(x_i + \frac{1}{2}h)$: $y^*_{i+1/2} = y_i + \frac{1}{2}k_1 h$, then estimate the slope $y'(x_i + \frac{1}{2}h)$: $k_2 = f(x_i + \frac{1}{2}h, y^*_{i+1/2})$.
c) Set $y_{i+1} = y_i + k_2 h$.

Fact.
The midpoint method is of order 2.

Definition.
An explicit Runge-Kutta method for solving IVP $y' = f(x, y)$ is given by fixing parameters

$$\begin{array}{c|ccccc} 0 & & & & & \\ c_2 & a_{21} & & & & \\ c_3 & a_{31} & a_{32} & & & \\ \vdots & \vdots & & \ddots & & \\ c_N & a_{N1} & a_{N2} & \cdots & a_{N,N-1} & \\ \hline & w_1 & w_2 & \cdots & w_{N-1} & w_N \end{array}$$

Here N is the number of steps, c_j defines nodes, a_{jl} forms the matrix of the method and w_j are weights. When determining y_{i+1} using y_i we first estimate slopes at various points,

$$\begin{aligned} k_1 &= f(x_i, y_i), \\ k_2 &= f(x_i + c_2 h, y_i + a_{21} k_1 h), \\ k_3 &= f(x_i + c_3 h, y_i + (a_{31} k_1 + a_{32} k_2) h), \\ &\vdots \\ k_N &= f(x_i + c_N h, y_i + (a_{N1} k_1 + a_{N2} k_2 + \dots + a_{N,N-1} k_{N-1}) h), \end{aligned}$$

these estimates are averaged to get the best slope $k = \sum\limits_{j=1}^{N} w_j k_j$ and then we set $y_{i+1} = y_i + k \cdot h$.

Algorithm (RK4 for IVP $y' = f(x, y)$)
Given: ODE $y' = f(x, y)$ on $[x_0, x_0 + T]$, initial value y_0, and $n \in I\!N$.
0. Set $h = \frac{T}{n}$.
1. x_0 and y_0 are given.
2. For $i = 0, \dots, n-1$ set $x_{i+1} = x_i + h$ and:
a) Estimate the slope $y'(x_i)$: $k_1 = f(x_i, y_i)$.
b) Estimate $y(x_i + \frac{1}{2}h)$: $y^*_{i+1/2} = y_i + \frac{1}{2}k_1 h$ and then estimate the slope $y'(x_i + \frac{1}{2}h)$: $k_2 = f(x_i + \frac{1}{2}h, {y_{i+1/2}}^*)$.
c) Again (to improve?) estimate $y(x_i + \frac{1}{2}h)$: $y^{**}_{i+1/2} = y_i + \frac{1}{2}k_2 h$ and then estimate the slope $y'(x_i + \frac{1}{2}h)$: $k_3 = f(x_i + \frac{1}{2}h, y^{**}_{i+1/2})$.
d) Estimate $y(x_i + h)$: $y^*_{i+1} = y_i + k_3 h$ and then estimate the slope $y'(x_{i+1})$: $k_4 = f(x_{i+1}, y^*_{i+1})$.
e) Set $y_{i+1} = y_i + \frac{1}{6}[k_1 + 2k_2 + 2k_3 + k_4] \cdot h$.

Fact.
The RK4 method is of order 4.

Algorithm (RKF45, adaptive Runge-Kutta-Fehlberg method for IVP $y' = f(x,y)$)
Given: ODE $y' = f(x,y)$ on $[x_0, x_0+T]$, initial value y_0, initial step $h_0 > 0$ and desired precision $\varepsilon > 0$.
1. x_0 and y_0 are given.
2. For $i = 0, \dots, n-1$:
a) Evaluate $k_1 = f(x_i, y_i)$.
b) Evaluate $k_2 = f\big(x_i + \frac{1}{4}h_i, y_i + \frac{1}{4}k_1 h_i\big)$.
c) Evaluate $k_3 = f\big(x_i + \frac{3}{8}h_i, y_i + \big(\frac{3}{32}k_1 + \frac{9}{32}k_2\big)h_i\big)$.
d) Evaluate $k_4 = f\big(x_i + \frac{12}{13}h_i, y_i + \big(\frac{1932}{2197}k_1 - \frac{7200}{2197}k_2 + \frac{7296}{2197}k_3\big)h_i\big)$.
e) Evaluate $k_5 = f\big(x_i + h_i, y_i + \big(\frac{439}{216}k_1 - 8k_2 + \frac{3680}{513}k_3 - \frac{845}{4104}k_4\big)h_i\big)$.
f) Evaluate $k_6 = f\big(x_i + \frac{1}{2}h_i, y_i + \big(-\frac{8}{27}k_1 + 2k_2 - \frac{3544}{2565}k_3 + \frac{1859}{4104}k_4 - \frac{11}{40}k_5\big)h_i\big)$.
g) Estimate $y_{i+1} = y_i + \big(\frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5\big)h_i$
and $z_{i+1} = y_i + \big(\frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6\big)h_i$.

If $\frac{1}{h_i}|z_{i+1} - y_{i+1}| > \varepsilon$, set $s = \big(\frac{h\varepsilon}{|z_{i+1}-y_{i+1}|}\big)^{1/4}$ and redo the calculations starting from a) on with the step $h_i = sh_i$.
Otherwise set $x_{i+1} = x_i + h_i$, $h_{i+1} = h_i$ and go to the next cycle, that is, increase i by one etc.