POLITECNICO MILANO 1863

DEPARTMENT OF CHEMISTRY MATERIALS AND CHEMICAL ENGINEERING

Ordinary Differential Equations Part 1

Calcoli di Processo dell' Ingegneria Chimica

Timoteo Dinelli

27th of November 2025

Department of Chemistry, Materials and Chemical Engineering, "Giulio Natta", Politecnico di Milano.

email: timoteo.dinelli@polimi.it

Ordinary Differential Equations

We will discuss different methods to approximate numerically the solution of the following Ordinary Differential Equation:

$$\frac{dy}{dt} = f(t, y, \Theta)$$

Generally speaking we are going to solve what it is usually called an **IVP** (Initial Value Problems), a differential equation associated with a set of initial conditions.

$$\begin{cases} \frac{dC_A}{dt} = -C_A \\ C_A(t = t^*) = C_A^* \end{cases}$$

Forward Euler Method - Implementation

Explicit scheme: $y_{n+1} = y_n + hf(t_n, y_n)$

Implementation steps:

1. Start with initial condition: $y_0 = y(t_0)$
2. Evaluate the derivative at current point: $f(t_n, y_n)$
3. Update directly: $y_{n+1} = y_n + h \cdot f(t_n, y_n)$
4. Advance time: $t_{n+1} = t_n + h$

Practical example: $\frac{dC_A}{dt} = -kC_A$, with $C_A(0) = 1\ \text{mol/L}$, $k = 0.5\ \text{s}^{-1}$, $h = 0.1\ \text{s}$

$$C_A(t_{n+1}) = C_A(t_n) + h \cdot (-k \cdot C_A(t_n))$$

Key point: Easy to implement, but watch out for stability with large h!

Backward Euler Method - Implementation

Implicit scheme: $y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$

Implementation steps:

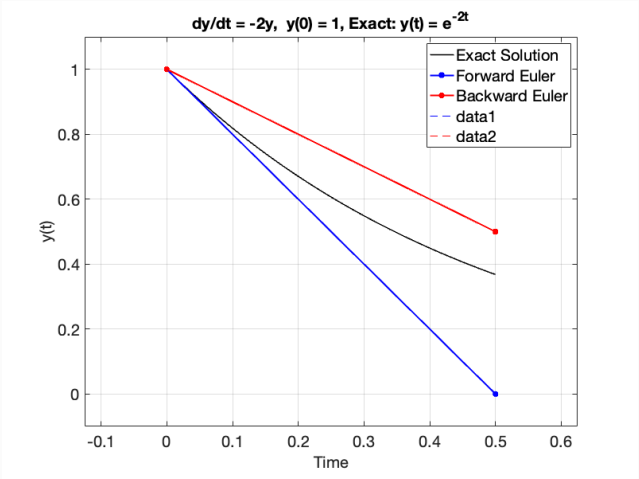
1. Start with initial condition: $y_0 = y(t_0)$
2. Set up implicit equation: $y_{n+1} = y_n + h \cdot f(t_{n+1}, y_{n+1})$
3. **Solve** for y_{n+1} (requires root-finding or algebraic manipulation)
4. Advance time: $t_{n+1} = t_n + h$

Practical example: $\frac{dC_A}{dt} = -kC_A$, with $C_A(0) = 1\ \text{mol/L}$, $k = 0.5\ \text{s}^{-1}$, $h = 0.1\ \text{s}$

$$C_A(t_{n+1}) = C_A(t_n) - h \cdot k \cdot C_A(t_{n+1}) \quad \Rightarrow \quad C_A(t_{n+1}) = \frac{C_A(t_n)}{1 + hk}$$

Key point: More stable, but requires solving (non)linear equations at each step!

Methods Comparison



Forward Euler (**explicit**):

$$y_{n+1} = y_n + hy'_n$$

Backward Euler (**implicit**):

$$y_{n+1} = y_n + hy'_{n+1}$$

Exercises

Exercise 1: ODE Integration Methods

Problem: Solve $\frac{dx}{dt} = -kx$, with $x(0) = 1.0$

Analytical solution: $x(t) = x_0 e^{-kt}$

Tasks:

1. Implement **Forward Euler**: $x_{n+1} = x_n + hf(t_n, x_n)$
2. Implement **Backward Euler**: $x_{n+1} = x_n + hf(t_{n+1}, x_{n+1})$
   - For this linear ODE: $x_{n+1} = \frac{x_n}{1+hk}$
3. Implement **Heun method with adaptive stepping**
   - Use error estimation (1 full step vs 2 half steps)
   - Adjust h based on tolerance: $|error| < tol$

Exercise 1: Testing & Analysis

Test parameters:

- $k = 1.0$, $x_0 = 1.0$, $t \in [0, 5]$
- Fixed step: $h = 0.1$ (Forward/Backward Euler)
- Adaptive: $h_{init} = 0.1$, $tol = 10^{-4}$ (Heun)

Stability study: Test with $k = [0.1, 0.5, 1.0, 2.0, 5.0]$ and $h = 0.5$

- When does Forward Euler become unstable? (Hint: $hk > 2$)
- How does Backward Euler handle large k?

Exercise 2: Isothermal Batch Reactor

Problem: Design an isothermal batch reactor for the reaction:

$$A \to B \quad k = 0.01\,\mathrm{s}^{-1}$$

Governing equations:

Species balances:

$$\begin{cases} \frac{dC_A}{dt} = -kC_A \\ \frac{dC_B}{dt} = kC_A \end{cases}$$

Conversion (X_A):

$$X_A = \frac{C_A^0 - C_A}{C_A^0}$$

$$\frac{dX_A}{dt} = k(1 - X_A)$$

Analytical solution: $X_A(t) = 1 - e^{-kt}, \quad C_A(t) = C_A^0 e^{-kt}$

Exercise 2: Implementation Tasks

Initial conditions: $C_A^0 = 1.0$ mol/L, $C_B^0 = 0$ mol/L, $X_A(0) = 0$

Tasks:

1. Solve the system using `ode45` (MATLAB built-in solver)
2. Compare with your implemented methods (Forward/Backward Euler, Heun)
3. Determine time to reach 90% conversion ($X_A = 0.9$)

Bonus: Extend to a system of ODEs - solve for both C_A and C_B simultaneously and verify mass balance: $C_A(t) + C_B(t) = C_A^0$

Thank you for your attention!