Numerical Solution of Single ODEs

1. Euler methods
2. Euler method examples
3. In-class exercise
4. More advanced methods

Leonhard Euler
1768

Carl Runge
1900

Martin Wilhelm Kutta
1900

Numerical Solution of Single ODEs

Euler Methods

Introduction

- Linear ODE systems can be analytically solved using eigenvalues and eigenvectors
- The qualitative properties of nonlinear ODE systems can be determined by linearization
- Very few nonlinear ODEs can be solved analytically
- Most nonlinear ODEs must be solved numerically
- Many numerical methods are available for ODE solution
- The simplest methods are based on finite difference approximations to generate difference equations

Euler Methods

- Initial value problem

$$\frac{dy}{dx} = f(x, y) \quad y(x_0) = y_0$$

  - » Often impossible to generate analytical solution $y(x)$
  - » Instead compute approximate solution at equally spaced node points

$$x_1 = x_0 + h \quad x_2 = x_0 + 2h \quad x_3 = x_0 + 3h \quad \cdots$$

$$y(x_1) \equiv y_1 \quad y(x_2) \equiv y_2 \quad y(x_3) \equiv y_3 \quad \cdots$$

  - » h = step size

- Taylor series expansion

$$y(x+h) = y(x) + h\frac{dy(x)}{dx} + \frac{h^2}{2!}\frac{d^2y(x)}{dx^2} + \cdots$$

Forward Euler Method

- First-order Taylor series expansion

$$y(x+h) \cong y(x) + h\frac{dy(x)}{dx} = y(x) + hf(x,y)$$

- Iterative calculation

$$y_1 = y_0 + hf(x_0, y_0) \qquad y_2 = y_1 + hf(x_1, y_1)$$

$$y_{n+1} = y_n + hf(x_n, y_n) \qquad n = 0,1,2,\ldots$$

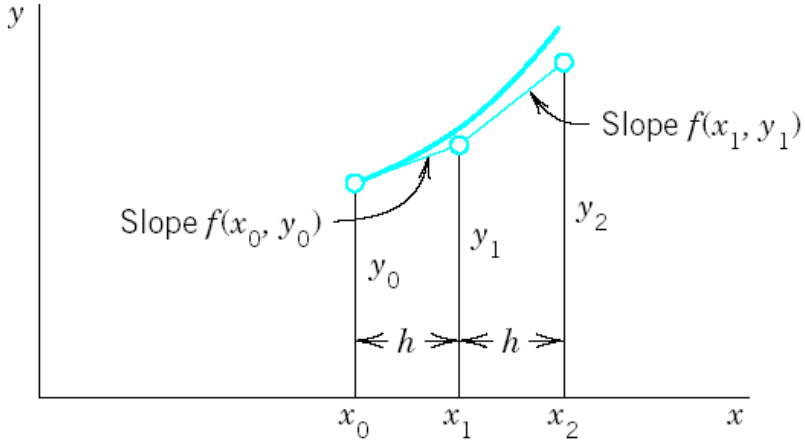


Fig. 448. Euler method

- Approximation error

$$y(x+h) = y(x) + h\frac{dy(x)}{dx} + \frac{h^2}{2!}\frac{d^2y(\xi)}{dx^2} \qquad x \le \xi \le x+h$$

  - » Local truncation error: $O(h^2)$
  - » Global truncation error: $O(h)$ → first-order method

Backward Euler Method

- Forward Euler

$$\frac{dy(x)}{dx} = f(x,y) \quad \frac{y_{n+1} - y_n}{h} = f(x_n, y_n) \quad y_{n+1} = y_n + hf(x_n, y_n)$$

  - » Explicit formula for y_{n+1} (explicit method)

- Backward Euler

$$\frac{dy(x)}{dx} = f(x,y) \quad \frac{y_{n+1} - y_n}{h} = f(x_{n+1}, y_{n+1}) \quad y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$$

  - » Implicit formula for y_{n+1} (implicit method)
  - » Requires repeated solution of a nonlinear algebraic equation
  - » Allows larger *h* values to be used with comparable errors → more stable
  - » Generally preferred to forward Euler

Numerical Solution of Single ODEs

Euler Method Examples

Forward Euler Method Example

- Binary mixing tank

$$\frac{dx_3}{dt} = \frac{w_1(x_1 - x_3) + w_2(x_2 - x_3)}{\rho A h} \qquad x_3(0) = x_{30}$$

- Derivation of difference equation

$$\frac{x_{3,n+1} - x_{3,n}}{\Delta t} = \frac{w_1(x_1 - x_{3,n}) + w_2(x_2 - x_{3,n})}{\rho A h}$$

$$x_{3,n+1} = x_{3,n} + \Delta t \frac{w_1(x_1 - x_{3,n}) + w_2(x_2 - x_{3,n})}{\rho A h}$$

- Parameters: $w_1 = 10$, $x_1 = 1$, $w_2 = 2$, $x_2 = 4$, $\rho A h = 2$

Forward Euler Method Example

- Substitute parameter values

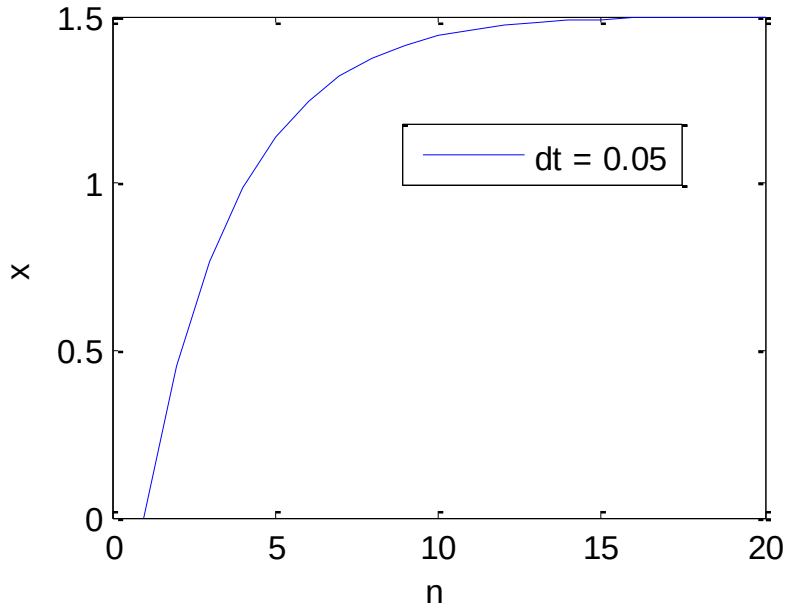
$$x_{3,n+1} = x_{3,n} + \Delta t \frac{10(1 - x_{3,n}) + 2(4 - x_{3,n})}{2}$$

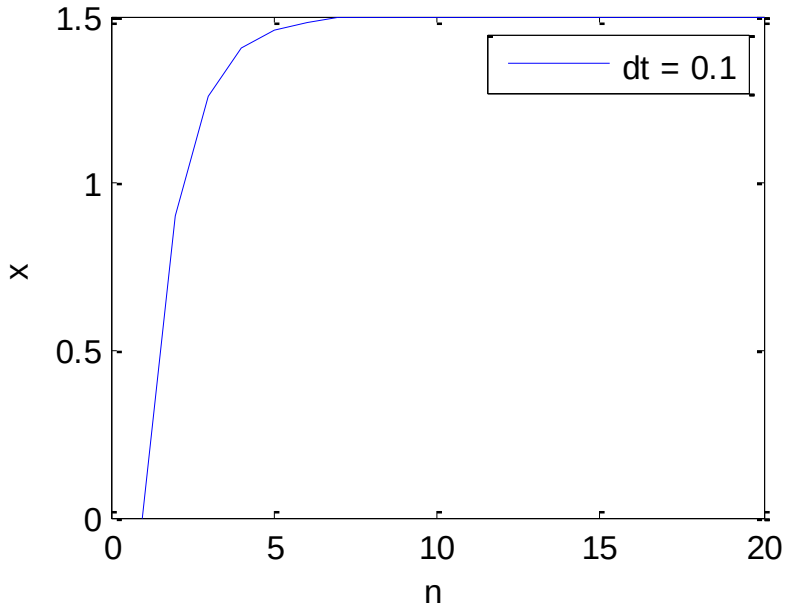
- Iterative solutions for $x_{3,0} = x_3(0) = 0$

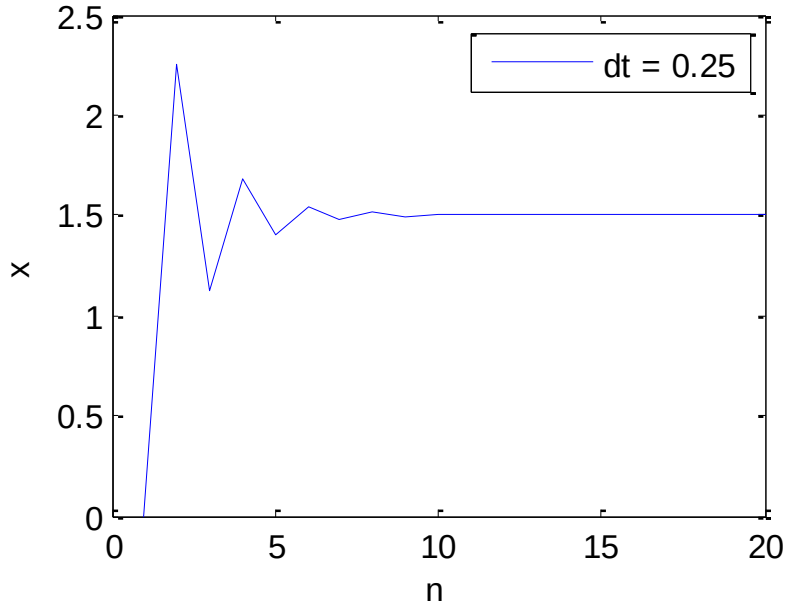
n		Δt		
	0.05	0.1	0.25	0.5
1	0.450	0.900	2.250	4.5
2	0.750	1.260	1.125	-4.5
3	0.986	1.404	1.688	13.5
4	1.140	1.462	1.406	-22.5
5	1.248	1.485	1.547	49.5
10	1.458	1.500	1.499	-1535

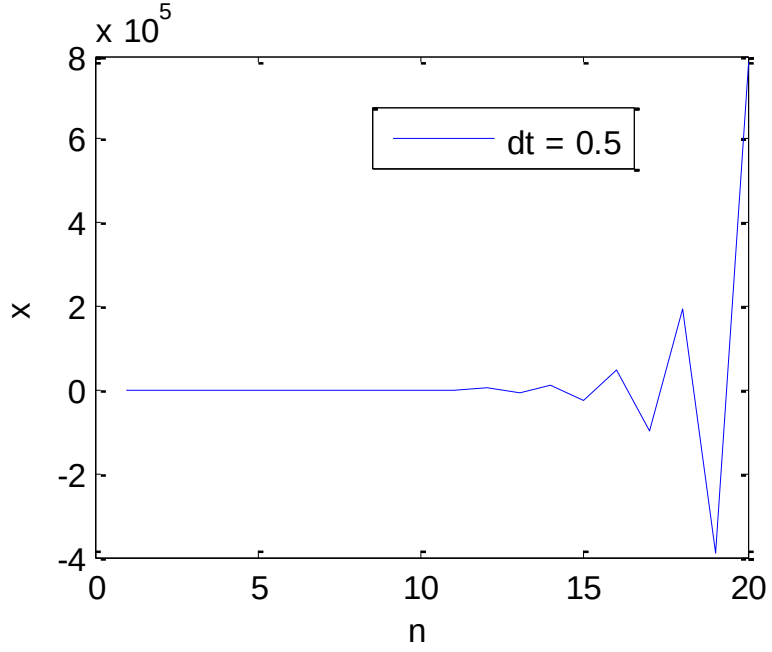
UMASS.

Forward Euler Method Example

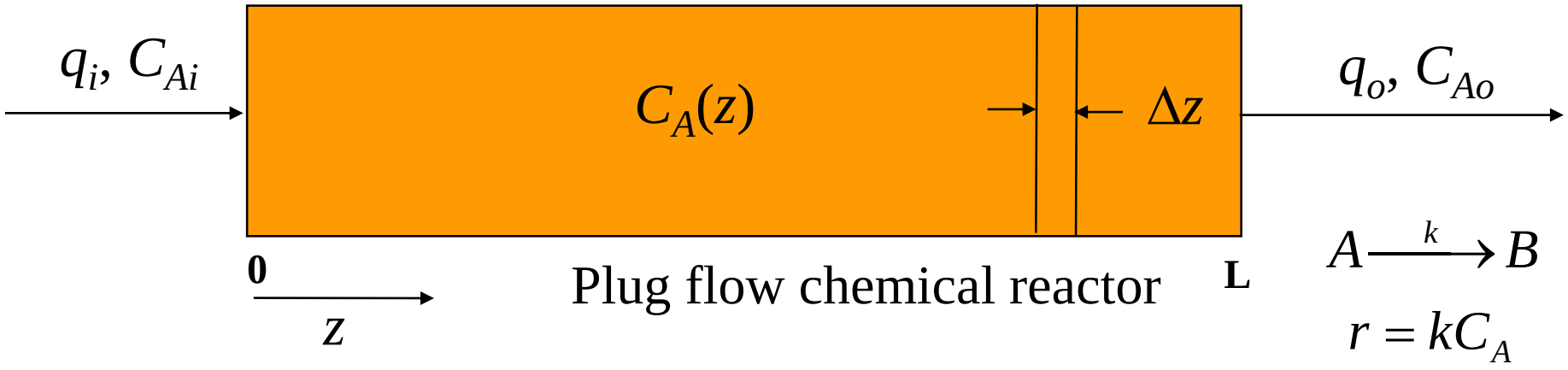








Backward Euler Method Example



$$\frac{q}{A}\frac{dC_A}{dz} + kC_A = 0 \quad C_A(0) = C_{Ai}$$

$$\frac{q}{A}\frac{C_A(z_{n+1}) - C_A(z_n)}{\Delta z} + kC_A(z_{n+1}) \approx 0$$

$$C_A(z_{n+1}) = \frac{1}{1 + kA\Delta z/q} C_A(z_n) \quad C_A(z_0) = C_{Ai}$$

$$C_A(L) \approx C_A(z_N) = \left(\frac{1}{1 + kA\Delta z/q}\right)^N C_{Ai}$$

Backward Euler Method Example

- Analytical solution

$$C_A(L) = C_{Ai} \exp\left(-\frac{kA}{q}L\right)$$

- Numerical solution

$$C_A(L) \approx \left(\frac{1}{1+kA\Delta z/q}\right)^N C_{Ai} = \left(\frac{1}{1+kAL/Nq}\right)^N C_{Ai}$$

- Convergence formula

$$\lim_{N\to\infty}\left(\frac{1}{1+a/N}\right)^N = e^{-a}$$

- Convergence of numerical solution

$$\lim_{N\to\infty}\left(\frac{1}{1+kAL/Nq}\right)^N C_{Ai} = C_{Ai}\exp\left(-\frac{kA}{q}L\right)$$

UMASS.

Numerical Solution of Single ODEs

In-class Exercise

Numerical Solution of Single ODEs

More Advanced Methods

Improved Euler Method

- Euler methods are generally too inaccurate due to the use of first-order finite difference approximations
- The use of a very small step size *h* needed for sufficient accuracy can be computationally prohibitive
- Predictor-corrector formulation of improved Euler method

Prediction $$y^*_{n+1} = y_n + hf(x_n, y_n)$$

Correction $$y_{n+1} = y_n + \tfrac{1}{2}h[f(x_n, y_n) + f(x_{n+1}, y^*_{n+1})]$$

- Approximation error
  - » Local truncation error: $O(h^3)$
  - » Global truncation error: $O(h^2)$ → second-order method

Runge-Kutta Methods

- Basic Runge-Kutta (RK) method

$$k_1 = hf(x_n, y_n) \quad k_2 = hf(x_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}k_1)$$
$$k_3 = hf(x_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}k_2) \quad k_4 = hf(x_n + h, y_n + k_3)$$
$$y_{n+1} = y_n + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

  - » Global truncation error: $O(h^4)$ → fourth-order method
- Runge-Kutta-Fehlberg (RKF) method involves a combination of two Runge-Kutta formulas (see text)
- Comparison of methods

Method	Function Evaluations	Global Error	Local Error
Euler	1	$O(h)$	$O(h^2)$
Improved Euler	3	$O(h^2)$	$O(h^3)$
RK	4	$O(h^4)$	$O(h^5)$
RKF	6	$O(h^5)$	$O(h^6)$

Multistep Methods

- Definition
  - » Single-step: calculation at current step only uses value at last step ($y_n \rightarrow y_{n+1}$)
  - » Multi-step: calculation at current step uses values at several previous steps
- General development
  - » Integrate ODE

$$\frac{dy(x)}{dx} = f(x,y) \quad \int_{x_n}^{x_{n+1}} dy(x) = y(x_{n+1}) - y(x_n) = \int_{x_n}^{x_{n+1}} f[x, y(x)]dx$$

  - » Approximate $f(x,y)$ with an interpolation polynomial

$$y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} p(x)dx$$

  - » Different polynomials produce different methods

Adams-Bashforth Methods

- Utilize cubic polynomial that interpolates

$$f_n = f(x_n, y_n) \qquad f_{n-1} = f(x_{n-1}, y_{n-1})$$
$$f_{n-2} = f(x_{n-2}, y_{n-2}) \qquad f_{n-3} = f(x_{n-3}, y_{n-3})$$

- Iterative calculation

$$y_{n+1} = y_n + \tfrac{1}{24} h(55 f_n - 59 f_{n-1} + 37 f_{n-2} - 9 f_{n-3})$$

  - » Explicit method
  - » Global truncation error: $O(h^4)$ → fourth-order method

- Initialization

$$y_4 = y_0 + \tfrac{1}{24} h(55 f_3 - 59 f_2 + 37 f_1 - 9 f_0)$$

  - » Must compute y_1, y_2 and y_3 by another method of comparable accuracy (e.g. Runge-Kutta)

Adams-Moulton Methods

- Utilize cubic polynomial that interpolates

$$f_{n+1} = f(x_{n+1}, y_{n+1}) \qquad f_n = f(x_n, y_n)$$
$$f_{n-1} = f(x_{n-1}, y_{n-1}) \qquad f_{n-2} = f(x_{n-2}, y_{n-2})$$

- Iterative calculation

$$y_{n+1} = y_n + \tfrac{1}{24} h(9 f_{n+1} + 19 f_n - 5 f_{n-1} + f_{n-2})$$

  - » Implicit method
  - » Global truncation error: $O(h^4)$ → fourth-order method

- Predictor-corrector formulation

$$y^*_{n+1} = y_n + \tfrac{1}{24} h(55 f_n - 59 f_{n-1} + 37 f_{n-2} - 9 f_{n-3})$$
$$y_{n+1} = y_n + \tfrac{1}{24} h(9 f^*_{n+1} + 19 f_n - 5 f_{n-1} + f_{n-2})$$

Variable Step Size

- All the methods discussed use a fixed step size h
- Adaptive step size h_n
  - » Large h_n desirable if dy/dx changing slowly (speed)
  - » Small h_n necessary if dy/dx changing rapidly (accuracy)
  - » Adjust h_n to maintain the local error below a prespecified tolerance
- Local error control for Euler methods

$$h_n = \varphi(x_n)H \quad \varphi(x_n) = \sqrt{\frac{K}{\left|d^2y(\xi_n)/dx^2\right|}} \quad H = \sqrt{\frac{2T}{K}}$$

  - » T = specified tolerance for local error
  - » ξ_n = point in the interval $[x_n, x_{n+1}]$
  - » K = minimum value of second derivative in the interval $[x_n, x_{n+1}]$