



華東師範大學

EAST CHINA NORMAL UNIVERSITY

# 第二讲

---

## 常微分方程数值求解

—— Euler 法与 R-K 法

—— 常微分方程组与高阶方程

# 主要内容

---

- Euler 法与改进的 Euler 法
- 算法分析：误差与收敛性
- Runge-Kutta 法
- 一阶常微分方程组与高阶方程
- 应用举例
- Matlab 相关函数

# 初值问题

## ■ 考虑一维经典初值问题

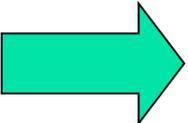
$$\begin{cases} \frac{dy}{dx} = f(x, y) \\ y(a) = y_0 \end{cases} \quad x \in [a, b]$$

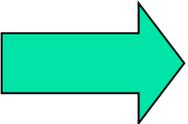
问题：如何计算  $y(b)$  的近似值？

# Euler 公式

两边求积分可得

$$\int_a^b \frac{dy}{dx} dx = \int_a^b f(x, y) dx$$


$$y(b) - y(a) = \int_a^b f(x, y) dx$$

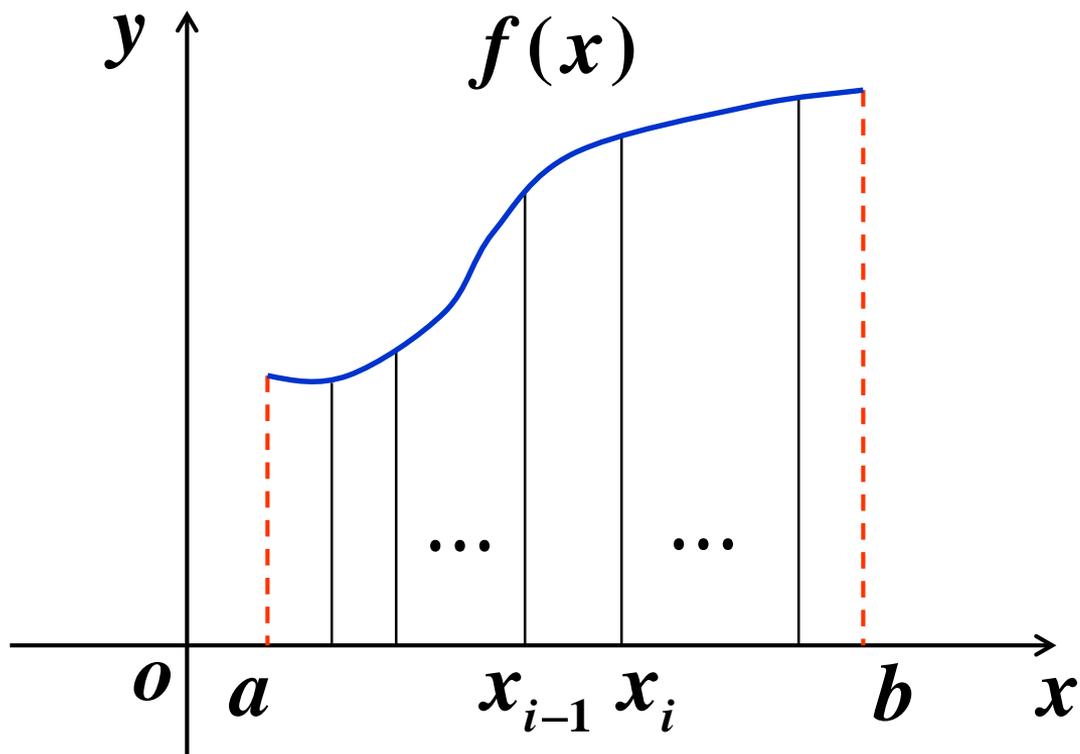
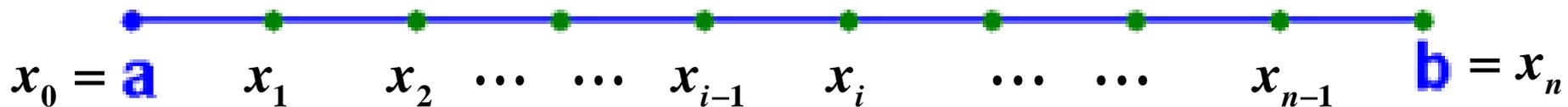

$$y(b) = y(a) + \int_a^b f(x, y) dx$$

$$\approx y(a) + (b - a) f(a, y(a)) \leftarrow \text{左点矩形公式}$$

$$= y_0 + (b - a) f(a, y_0) \quad \text{Euler 公式}$$

# Euler 法

分割区间，在每个小区间上使用 Euler 公式，逐步递推



# Euler 法

---

## ■ Euler 法

$$y(x_1) \approx y_0 + (x_1 - x_0)f(x_0, y_0) \triangleq y_1$$

$$y(x_2) \approx y_1 + (x_2 - x_1)f(x_1, y_1) \triangleq y_2$$

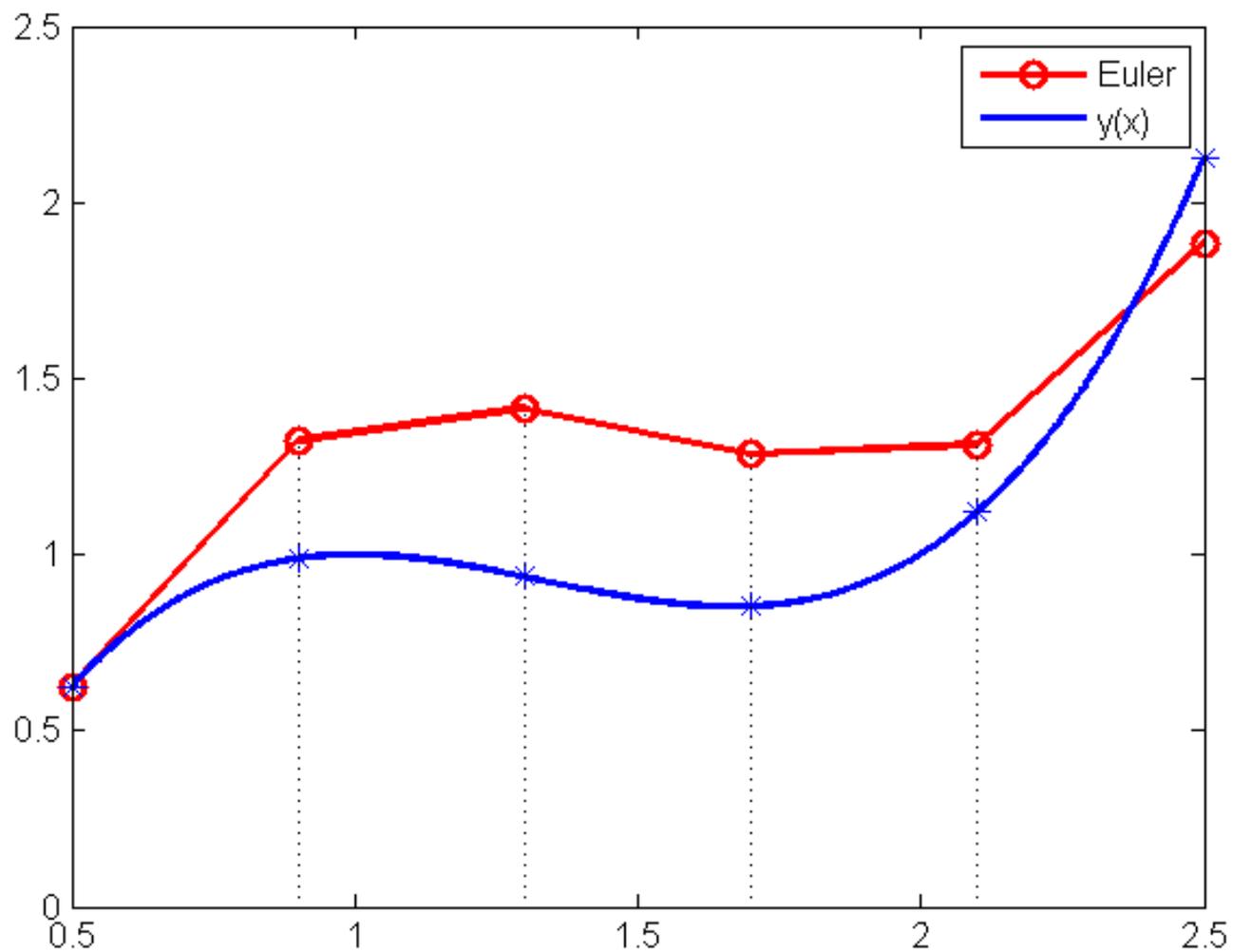
⋮

$$y(x_n) \approx y_{n-1} + (x_n - x_{n-1})f(x_{n-1}, y_{n-1}) \triangleq y_n$$

$$y_{k+1} = y_k + (x_{k+1} - x_k)f(x_k, y_k)$$

$$k = 0, 1, 2, \dots, n-1$$

# 几何含义



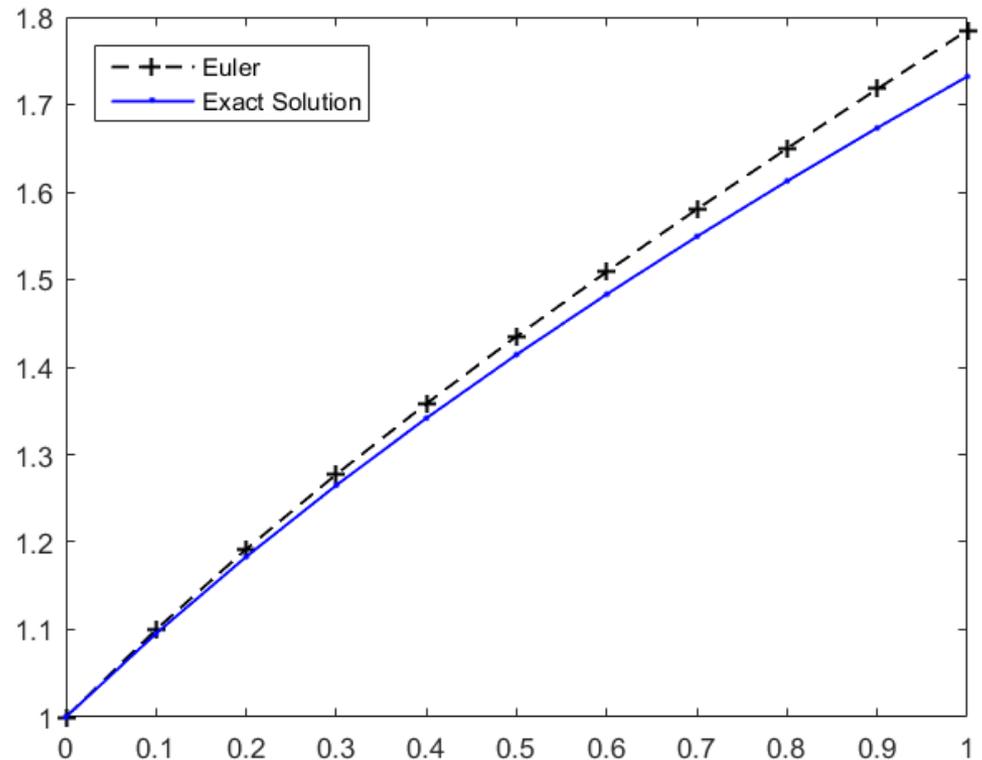
# 举例

例：用 Euler 法解初值问题

$$\begin{cases} \frac{dy}{dx} = y - \frac{2x}{y} \\ y(0) = 1 \end{cases} \quad x \in [0, 1]$$

解：Matlab 代码见  
Euler.m, Example1.m

真解  $y = \sqrt{2x + 1}$



# 梯形法

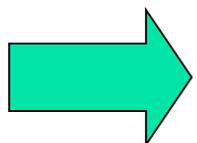
计算定积分时采用梯形公式，即

$$y(x_{k+1}) = y(x_k) + \int_{x_k}^{x_{k+1}} f(x, y) dx$$

梯形公式



$$\approx y(x_k) + \frac{x_{k+1} - x_k}{2} [f(x_k, y(x_k)) + f(x_{k+1}, y(x_{k+1}))]$$



$$y_{k+1} = y_k + \frac{x_{k+1} - x_k}{2} [f(x_k, y_k) + f(x_{k+1}, y_{k+1})]$$

$$k = 0, 1, 2, \dots, n-1$$

梯形法

缺点：公式右端含有  $y_{k+1}$ ，求解较困难

# 改进的Euler法

- 先用 Euler 法计算出  $y_{k+1}$  的近似值，然后代入后端项中

$$\tilde{y}_{k+1} = y_k + (x_{k+1} - x_k) f(x_k, y_k)$$

$$y_{k+1} = y_k + \frac{x_{k+1} - x_k}{2} [f(x_k, y_k) + f(x_{k+1}, \tilde{y}_{k+1})]$$

$$k = 0, 1, 2, \dots, n-1$$

## 改进的 Euler 法

- 第一步称为**预估**，第二步称为**校正**
- 通常也写为：

$$y_{k+1} = y_k + \frac{1}{2} h_k (K_1 + K_2)$$

$$h_k = x_{k+1} - x_k$$

$$\text{其中： } K_1 = f(x_k, y_k), K_2 = f(x_k + h_k, y_k + h_k K_1)$$

# 举例

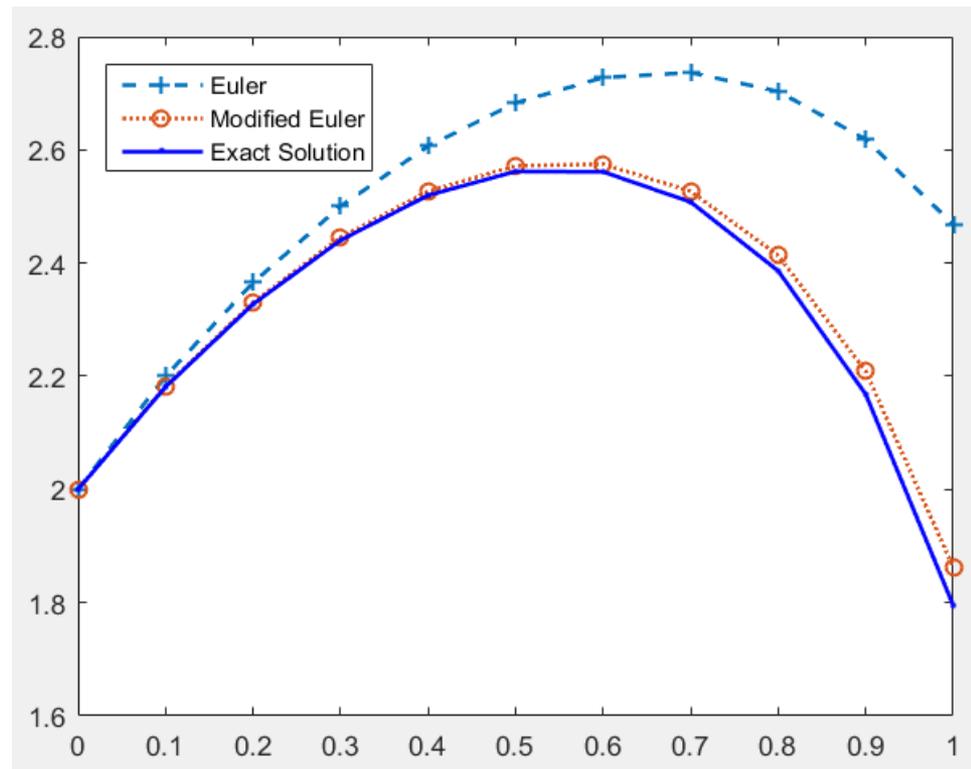
例：用 Euler 法和改进的 Euler 法解初值问题

$$\begin{cases} \frac{dy}{dx} = y - \frac{12x}{y} \\ y(0) = 2 \end{cases} \quad x \in [0, 1]$$

解：Matlab 代码见

EulerM.m, Example2.m

真解  $y = \sqrt{12x - 2e^{2x}} + 6$



# 主要内容

---

- Euler 法与改进的 Euler 法
- 算法分析：误差与收敛性
- Runge-Kutta 法
- 一阶常微分方程组与高阶方程
- 应用举例
- Matlab 相关函数

# 单步法

## ■ 单步法与多步法

- 如果在计算  $y_{k+1}$  时，只需用到  $y_k$  的值，则称为**单步法**
- 如果在计算  $y_{k+1}$  时，需用到  $y_k, y_{k-1}, \dots, y_{k-r+1}$  的值（即需要使用前  $r$  的点的值），则称为**多步法**

## ■ 解初值问题的等步长单步法的一般形式

$$y_{k+1} = y_k + h \cdot \Phi(x_k, y_k, y_{k+1}, h_k)$$

- 如果  $\Phi$  中含有  $y_{k+1}$ ，则称算法为**隐式的**，否则为**显式的**
- 显式单步法的一般形式

$$y_{k+1} = y_k + h \cdot \Phi(x_k, y_k, h_k)$$

称  $\Phi$  为**增量函数**

# 局部截断误差

**定义：** 设  $y(x)$  是初值问题的精确解，则称

$$R_{k+1} = y(x_{k+1}) - y(x_k) - h_k \Phi(x_k, y(x_k), h_k)$$

为显式单步法的**局部截断误差**。

●  $R_{k+1}$  是局部的，因为这里假定  $y_k$  是精确的，即  $y_k = y(x_k)$

**例：** 采用**等步长**的 Euler 法和梯形公式 Euler 法的局部截断误差

Euler 法的局部截断误差：
$$R_{k+1} = \frac{h^2}{2} y''(x_k) + O(h^3)$$

梯形公式 Euler 法的局部截断误差：

$$R_{k+1} = -\frac{h^3}{12} y'''(x_k) + O(h^4)$$

# 相容性与阶

- 等步长显式单步法的相容性与阶

**定义：** 若增量函数  $\Phi(x, y, h)$  在  $h=0$  处连续，且满足

$$\Phi(x, y, 0) = f(x, y)$$

则称显式单步法是**相容的**。

**定义：** 设  $p$  是使得下式成立的最大正整数

$$R_k = O(h^{p+1})$$

则称显式单步法是  $p$  **阶的**。

**例：** 等步长的 Euler 法是 1 阶的，  
等步长的梯形公式 Euler 法是 2 阶的。

# 收敛性

**定义：** 若设  $y(x)$  是解析解， $y_k$  是  $x_k$  处的数值解，若

$$\lim_{h \rightarrow 0} |y_k - y(x_k)| = 0 \quad (k = 1, 2, \dots, n)$$

则称算法是 **收敛的**。

- 如果不是等步长算法，则  $h$  是指小区间长度的最大值

**定理：** 若增量函数  $\Phi(x, y, h)$  关于  $x, y, h$  均满足 Lipschitz 条件，则显式单步法的收敛性与相容性等价。

# 收敛性

**定理：** 设显式单步法

$$y_{k+1} = y_k + h_k \Phi(x_k, y_k, h_k)$$

是  $p$  阶的，且增量函数  $\Phi$  关于  $y$  满足 Lipschitz 条件，又设初值是精确的，即  $y_0 = y(x_0)$ ，则算法整体误差为

$$y_k - y(x_k) = O(h^p)$$

即算法是收敛的，且收敛阶是  $p$ 。

**推论：** 设  $f(x, y)$  关于  $y$  满足 Lipschitz 条件，则

- (1) Euler 法收敛
- (2) 改进的 Euler 法收敛

# 主要内容

---

- Euler 法与改进的 Euler 法
- 算法分析：误差与收敛性
- Runge-Kutta 法
- 一阶常微分方程组与高阶方程
- 应用举例
- Matlab 相关函数

# Runge-Kutta法

$$y(x_{k+1}) = y(x_k) + \int_{x_k}^{x_{k+1}} f(x, y) dx$$

$$k = 1, 2, \dots, n$$

- 算法的精度取决于数值积分的精度

$$\int_{x_k}^{x_{k+1}} f(x, y) dx \approx h \sum_{i=1}^r \omega_i f(x_k + \lambda_i h, y(x_k + \lambda_i h))$$

- 采用高精度的数值积分方法就可能构造出高精度的算法
- 一般来说，积分点越多，精度可能越高（但不宜太多）
- 由于  $y(x_k + \lambda_i h)$  无法获得，因此需要用近似方法计算

# R-K法

## ■ 显式 R-K 法的一般格式

$$y_{k+1} = y_k + h \sum_{i=1}^r \alpha_i K_i \quad k = 1, 2, \dots, n$$

$$\begin{cases} K_1 = f(x_k, y_k) \\ K_i = f\left(x_k + \lambda_i h, y_k + h \sum_{j=1}^{i-1} \mu_{ij} K_j\right) \end{cases} \quad i = 2, \dots, r$$

- 这里  $\alpha_i, \lambda_i, \mu_{ij}$  是参数
- 参数选取准则：使得公式具有尽可能高的精度
- 工具：Taylor 展开

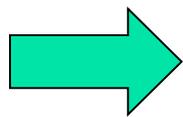
# $r = 1$ 时的 R-K 法

## 课堂板书

$$y_{k+1} = y_k + h\alpha_1 K_1 = y_k + \alpha_1 h f(x_k, y_k)$$

利用 Taylor 展开, 可得局部截断误差:

$$\begin{aligned} R_{k+1} &= y(x_{k+1}) - y(x_k) - \alpha_1 h f(x_k, y_k) \\ &= y'(x_k)h + \frac{1}{2}y''(x_k)h^2 + O(h^3) - \alpha_1 y'(x_k)h \\ &= (1 - \alpha_1)y'(x_k)h + \frac{1}{2}y''(x_k)h^2 + O(h^3) \end{aligned}$$



利当  $\alpha_1 = 1$  时, 达到最高阶数=1

这事实上就是 Euler 法

# $r = 2$ 时的 R-K 法

## 课堂板书

$$y_{k+1} = y_k + h(\alpha_1 K_1 + \alpha_2 K_2)$$

$$K_1 = f(x_k, y_k), \quad K_2 = f(x_k + \lambda_2 h, y_k + \mu_{21} h K_1)$$

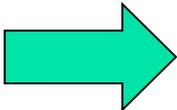
利用 Taylor 展开, 可得局部截断误差:

$$\begin{aligned} R_{k+1} &= (1 - \alpha_1 - \alpha_2) y'(x_k) h \\ &\quad + \left( \frac{1}{2} - \alpha_2 \lambda_2 \right) f'_x(x_k, y(x_k)) h^2 \\ &\quad + \left( \frac{1}{2} - \alpha_2 \mu_{21} \right) f'_y(x_k, y(x_k)) f(x_k, y(x_k)) h^2 + O(h^3) \end{aligned}$$

$$\text{令 } 1 - \alpha_1 - \alpha_2 = 0, \quad \frac{1}{2} - \alpha_2 \lambda_2 = 0, \quad \frac{1}{2} - \alpha_2 \mu_{21} = 0$$

# $r = 2$ 时的 R-K 法

解不惟一，设  $\alpha_2 = a \neq 0$

  $\alpha_1 = 1 - a, \quad \lambda_2 = \frac{1}{2a}, \quad \mu_{21} = \frac{1}{2a}$

公式一：令  $a = 1/2$

$$y_{k+1} = y_k + \frac{h}{2}(K_1 + K_2)$$

$$K_1 = f(x_k, y_k), \quad K_2 = f(x_k + h, y_k + hK_1)$$

这事实上就是改进的 Euler 法

# $r = 2$ 时的 R-K 法

公式二：令  $a = 1$

$$y_{k+1} = y_k + hK_2$$

$$K_1 = f(x_k, y_k), \quad K_2 = f\left(x_k + \frac{1}{2}h, y_k + \frac{1}{2}hK_1\right)$$

中点公式 Euler 法

注： $r = 2$  时，显式 R-K 法的阶至多只能达到二阶

# $r = 3$ 时的 R-K 法

---

## 课堂板书

### 一个常用的三阶R-K法

$$y_{k+1} = y_k + \frac{h}{6}(K_1 + 4K_2 + K_3)$$

$$K_1 = f(x_k, y_k),$$

$$K_2 = f\left(x_k + \frac{1}{2}h, y_k + \frac{1}{2}hK_1\right)$$

$$K_3 = f(x_k + h, y_k - hK_1 + 2hK_2)$$

# 四阶经典R-K法

$$y_{k+1} = y_k + \frac{h}{6} (K_1 + 2K_2 + 2K_3 + K_4)$$

$$k = 1, 2, \dots, n$$

$$\begin{cases} K_1 = f(x_k, y_k) \\ K_2 = f\left(x_k + \frac{1}{2}h, y_k + \frac{h}{2}K_1\right) \\ K_3 = f\left(x_k + \frac{1}{2}h, y_k + \frac{h}{2}K_2\right) \\ K_4 = f(x_k + h, y_k + hK_3) \end{cases}$$

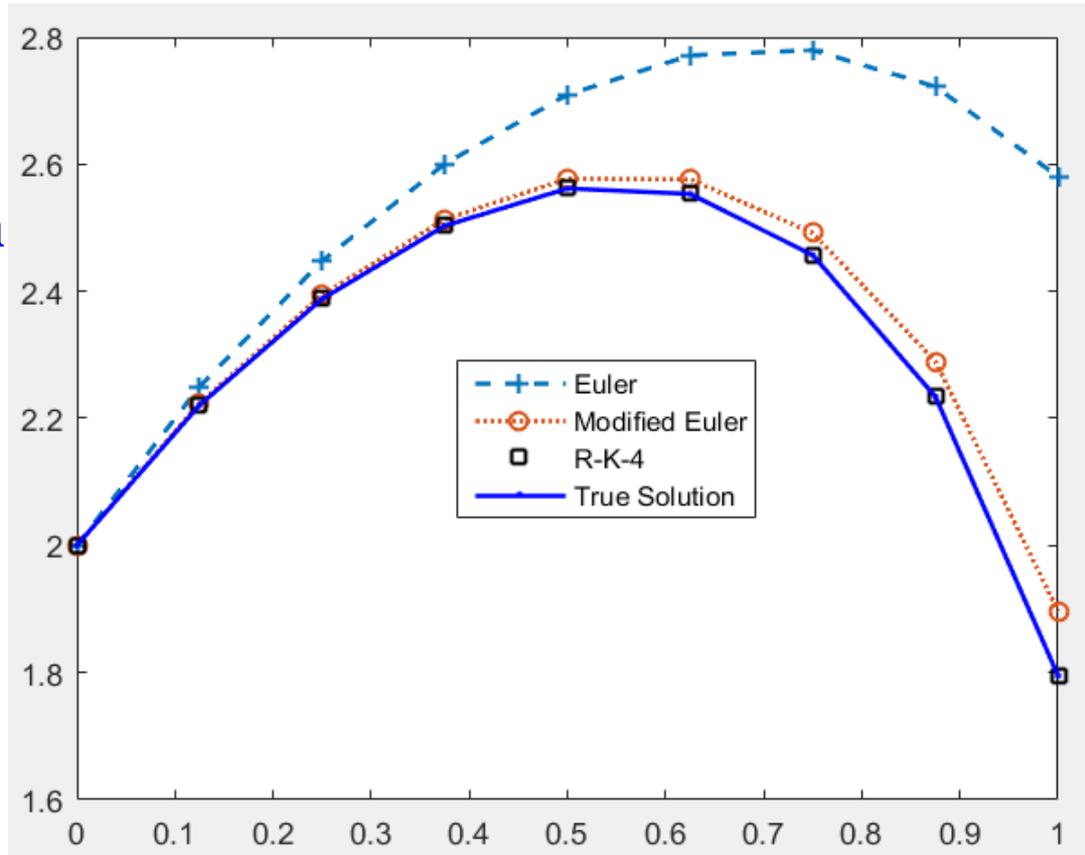
设  $f(x, y)$  关于  $y$  满足 Lipschitz 条件, 则该算法是收敛的, 且具有 4 阶收敛阶。

# 举例

例：用经典 R-K 方法解初值问题

$$\begin{cases} \frac{dy}{dx} = y - \frac{12x}{y} \\ y(0) = 2 \end{cases} \quad x \in [0, 1]$$

解：Example3.m, RK4.m



# 主要内容

---

- Euler 法与改进的 Euler 法
- 算法分析：误差与收敛性
- Runge-Kutta 法
- 一阶常微分方程组与高阶方程
- 应用举例
- Matlab 相关函数

# 一阶常微分方程组

$$\begin{cases} \frac{dY}{dx} = F(x, Y) \\ Y(x_0) = Y_0 \end{cases} \quad x_0 = a \leq x \leq b$$

其中

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \quad F = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix} \quad Y_0 = \begin{bmatrix} y_1^0 \\ y_2^0 \\ \vdots \\ y_m^0 \end{bmatrix}$$

# 四阶 R-K 法

$$Y_{k+1} = Y_k + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

$$k = 1, 2, \dots, n$$

$$\left\{ \begin{array}{l} K_1 = F(x_k, Y_k) \\ K_2 = F\left(x_k + \frac{1}{2}h, Y_k + \frac{h}{2}K_1\right) \\ K_3 = F\left(x_k + \frac{1}{2}h, Y_k + \frac{h}{2}K_2\right) \\ K_4 = F(x_k + h, Y_k + hK_3) \end{array} \right.$$

# 举例

例： $m=2$  时的四阶 R-K 方法

$$\begin{cases} y' = f(x, y, z) \\ z' = g(x, y, z) \\ y(x_0) = y_0, z(x_0) = z_0 \end{cases}$$

四阶 R-K 方法

$$y_{k+1} = y_k + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

$$z_{k+1} = z_k + \frac{h}{6}(L_1 + 2L_2 + 2L_3 + L_4)$$

$$k = 1, 2, \dots, n$$

# 举例

---

$$K_1 = f(x_k, y_k, z_k)$$

$$L_1 = g(x_k, y_k, z_k)$$

$$K_2 = f\left(x_k + \frac{1}{2}h, y_k + \frac{h}{2}K_1, z_k + \frac{h}{2}L_1\right)$$

$$L_2 = g\left(x_k + \frac{1}{2}h, y_k + \frac{h}{2}K_1, z_k + \frac{h}{2}L_1\right)$$

$$K_3 = f\left(x_k + \frac{1}{2}h, y_k + \frac{h}{2}K_2, z_k + \frac{h}{2}L_2\right)$$

$$L_3 = g\left(x_k + \frac{1}{2}h, y_k + \frac{h}{2}K_2, z_k + \frac{h}{2}L_2\right)$$

$$K_4 = f(x_k + h, y_k + hK_3, z_k + hL_3)$$

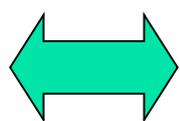
$$L_4 = g(x_k + h, y_k + hK_3, z_k + hL_3)$$

# 高阶方程

## ■ 处理方法：化为一阶方程组

$$\begin{cases} \frac{d^m y}{dx^m} = f(x, y, y', \dots, y^{(m-1)}) \\ y(x_0) = y_0, y'(x_0) = y_0', \dots, y^{(m-1)}(x_0) = y_0^{(m-1)} \end{cases}$$

引入变量： $z_1 = y, z_2 = y', \dots, z_m = y^{(m-1)}$


$$\begin{cases} z_1' = z_2 \\ z_2' = z_3 \\ \vdots \\ z_m' = f(x, z_1, z_2, \dots, z_m) \\ z_1(x_0) = y_0, z_2(x_0) = y_0', \dots, z_m(x_0) = y_0^{(m-1)} \end{cases}$$

# 举例：Van der Pol

## 例：求解 Van der Pol 方程



Balthasar van der Pol  
1889 -- 1959

Balthasar van der Pol 是荷兰的一位著名电子工程师。在20世纪20年代至30年代之间，他开创了现代实验动力学。1927年，为了描述在电子电路中三极管的震荡效应，第一次推导出了著名的van der Pol方程

$$\ddot{y} - \mu(1 - y^2)\dot{y} + y = 0$$

van der Pol 震荡系统作为一种经典的自激震荡系统，已经作为一种重要的数学模型并且在更加复杂的动力系统的建模中广泛使用。

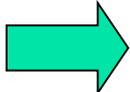
# 举例：Van der Pol

---

**例：求解 Van der Pol 方程**

$$\frac{d^2 y}{dt^2} + \mu \left( \frac{1}{3} \left( \frac{dy}{dt} \right)^3 - \frac{dy}{dt} \right) + y = 0$$

初值： $y(0) = 1$ ， $y'(0) = 1$ ，求解区间  $[0, 100]$

 转化为一阶方程组：

$$\begin{cases} z_1' = z_2 \\ z_2' = -z_1 - \mu \left( \frac{1}{3} z_2^3 - z_2 \right) \end{cases}$$

以  $\mu=5$  为例，编程求解：`vandelpol.m`, `vandelpol_main.m`