

# 第三章

---

## 数值积分

- 复合求积公式
- Romberg 算法

# 本讲内容

---

## ■ 复合求积公式

- 复合梯形公式
- 复合 Simpson 公式

## ■ Romberg（龙贝格）算法

- 梯形法的递推化计算
- Romberg 算法基本思想: 外推技巧
- Romberg 算法: 计算过程

# 复合求积公式

## 什么是复合求积公式

- 提高积分计算精度的常用两种方法
  - 用复合公式
  - 用非等距节点
- 复合求积公式
  - 将积分区间分割成多个小区间
  - 在每个小区间上使用低次 Newton-Cotes 求积公式

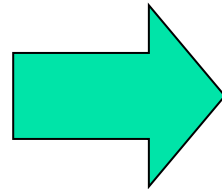
复合求积公式 (Composite Numerical Integration) 也称为复化求积公式

# 复合梯形公式

- 将  $[a, b]$  分成  $n$  个小区间  $[x_i, x_{i+1}]$ ，其中


$$a = x_0 < x_1 < \cdots < x_{n-1} < x_n = b$$

通常是  $n$  等分


$$\int_a^b f(x) \, dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x) \, dx$$

$$\approx \sum_{i=0}^{n-1} \frac{h_i}{2} [f(x_i) + f(x_{i+1})] \quad h_i = x_{i+1} - x_i$$

取等距节点


$$h = (b - a) / n$$

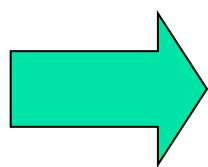
$$T_n = \frac{h}{2} \left[ f(a) + 2 \sum_{i=1}^{n-1} f(x_i) + f(b) \right]$$

# 余项公式

$$R[f] = -\sum_{i=0}^{n-1} \frac{h_i^3}{12} f''(\eta_i)$$

$$h_i = x_{i+1} - x_i, \quad \eta_i \in (x_i, x_{i+1})$$

- 当  $x_i$  其中为等距节点时, 即  $x_i = a + ih, \quad h = \frac{b-a}{n}$



$$\begin{aligned} R[f] &= -\frac{h^3}{12} \sum_{i=0}^{n-1} f''(\eta_i) \\ &= -\frac{b-a}{12} h^2 \left( \frac{1}{n} \sum_{i=0}^{n-1} f''(\eta_i) \right) = -\frac{b-a}{12} h^2 f''(\eta) \end{aligned}$$

$$\eta \in (a, b)$$

# 复合 Simpson 公式

$$\int_a^b f(x) dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x) dx$$

取等距节点


$$S_n = \sum_{i=0}^{n-1} \frac{h}{6} [f(x_i) + 4f(x_{i+\frac{1}{2}}) + f(x_{i+1})]$$
$$h = \frac{b-a}{n}$$
$$= \frac{h}{6} \left[ f(a) + 4 \sum_{i=0}^{n-1} f(x_{i+\frac{1}{2}}) + 2 \sum_{i=1}^{n-1} f(x_i) + f(b) \right]$$

注：复合 Simpson 公式实际使用了  $2n+1$  个节点

# 余项公式

$$\begin{aligned} R[f] &= -\frac{h^5}{2880} \sum_{i=0}^{n-1} f^{(4)}(\eta_i) \quad \boxed{\eta_i \in (x_i, x_{i+1})} \\ &= -\frac{(b-a)h^4}{2880} \left( \frac{1}{n} \sum_{i=0}^{n-1} f^{(4)}(\eta_i) \right) \\ &= -\frac{(b-a)h^4}{2880} f^{(4)}(\eta) \quad \boxed{\eta \in (a, b)} \end{aligned}$$

**性质：** 复合梯形公式和复合 Simpson 公式都是收敛的，也都是稳定的。

# 举例

例：设  $f(x) = \frac{\sin x}{x}$ ，利用下表中的数据分别用复合梯形公式和复合simpson公式计算定积分  $\int_0^1 f(x) dx$ 。

$x_i$	0	1/8	2/8	3/8	4/8	5/8	6/8	7/8	1.0
$f(x_i)$	1	0.997	0.990	0.977	0.954	0.936	0.909	0.877	0.841

解：  $T_8 = \frac{h_T}{2} \left[ f(x_0) + 2 \sum_{i=1}^7 f(x_i) + f(x_8) \right] = 0.9456909$

$$S_4 = \frac{h_S}{6} \left[ f(x_0) + 4(f(x_1) + f(x_3) + f(x_5) + f(x_7)) + 2(f(x_2) + f(x_4) + f(x_6)) + f(x_8) \right] = 0.9460832$$

example\_3\_2.m



# 举例

例：计算定积分

$$\int_0^1 e^x dx$$

用复合梯形公式和复合simpson公式时， $n$  分别取多大时才能使得误差不超过  $0.5 \times 10^{-5}$

解：  $f(x) = e^x \quad \longrightarrow \quad \max_{0 \leq x \leq 1} |f^{(k)}(x)| = \max_{0 \leq x \leq 1} |e^x| = e$

复合梯形公式

$$|R_T[f]| = \left| -\frac{b-a}{12} h_T^2 f''(\eta) \right| \leq \frac{e}{12} \left( \frac{1}{n} \right)^2$$

要使误差不超过  $0.5 \times 10^{-5}$ ，需要

**213 等分**

$$\frac{e}{12} \left( \frac{1}{n} \right)^2 \leq \frac{1}{2} \times 10^{-5} \quad \longrightarrow \quad n \geq 212.85 \quad \text{取 } n=213$$

# 举例

## 复合 simpson 公式

$$|R_s[f]| = \left| -\frac{b-a}{2880} h_s^4 f^{(4)}(\eta) \right| \leq \frac{e}{2880} \left( \frac{1}{n} \right)^4$$

要使误差不超过  $0.5 \times 10^{-5}$ ，需要

$$\frac{e}{2880} \left( \frac{1}{n} \right)^4 \leq \frac{1}{2} \times 10^{-5} \quad \Rightarrow \quad n \geq 3.71 \quad \text{故取 } n=4$$

8 等分

# 本讲内容

---

## ■ 复合求积公式

- 复合梯形公式
- 复合 Simpson 公式

## ■ Romberg（龙贝格）算法

- 梯形法的递推化计算
- Romberg 算法基本思想: 外推技巧
- Romberg 算法: 计算过程

# Romberg 算法

利用复合梯形公式、复合simpson公式、复合Cotes公式等计算定积分时，**如何选取步长  $h$  ?**

太 **大**  计算精度难以保证

太 **小**  增加额外的计算量

解决办法：采用 **变步长算法**

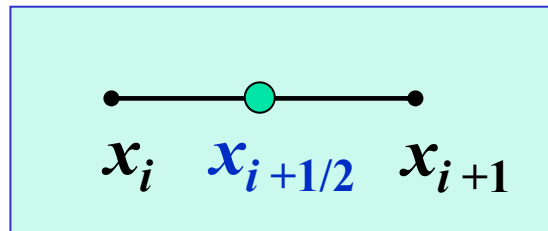
通常采取将区间**不断对分**的方法，即取  $n = 2^k$ ，**反复使用复合求积公式**，直到所得到的计算结果满足指定的精度为止。

# 梯形法递推公式

- 将  $[a, b]$  分成  $n$  等分  $[x_i, x_{i+1}]$  ,  $x_i = a + i \cdot h$ ,  $h = \frac{b-a}{n}$

→ 
$$T_n = \sum_{i=0}^{n-1} \frac{h}{2} [f(x_i) + f(x_{i+1})] = \frac{h}{2} \left[ f(a) + 2 \sum_{i=1}^{n-1} f(x_i) + f(b) \right]$$

- 步长折半:  $[x_i, x_{i+1/2}]$  ,  $[x_{i+1/2}, x_{i+1}]$



→ 
$$\begin{aligned} T_{2n} &= \sum_{i=0}^{n-1} \frac{h}{4} \left[ \left( f(x_i) + f(x_{i+1/2}) \right) + \left( f(x_{i+1/2}) + f(x_{i+1}) \right) \right] \\ &= \sum_{i=0}^{n-1} \frac{h}{4} \left[ f(x_i) + 2f(x_{i+1/2}) + f(x_{i+1}) \right] \\ &= \frac{h}{4} \sum_{i=0}^{n-1} [f(x_i) + f(x_{i+1})] + \frac{h}{2} \sum_{i=0}^{n-1} f(x_{i+1/2}) = \frac{1}{2} T_n + \frac{h}{2} \sum_{i=0}^{n-1} f(x_{i+1/2}) \end{aligned}$$

# 梯形法递推公式

$$T_{2n} = \frac{1}{2}T_n + \frac{h}{2} \sum_{i=0}^{n-1} f(x_{i+1/2}) = \frac{1}{2}T_n + \frac{h}{2} \sum_{i=0}^{n-1} f(a + ih + 0.5h)$$

$$h = \frac{b-a}{n}$$

$$T_1 = \frac{b-a}{2} (f(a) + f(b))$$

$$T_2 = \frac{1}{2}T_1 + \frac{h_0}{2} \sum_{i=0}^0 f(a + ih_0 + 0.5h_0)$$

$$h_0 = b - a$$

$$T_4 = \frac{1}{2}T_2 + \frac{h_1}{2} \sum_{i=0}^1 f(a + ih_1 + 0.5h_1)$$

$$h_1 = \frac{b-a}{2}$$

$$T_8 = \frac{1}{2}T_4 + \frac{h_2}{2} \sum_{i=0}^3 f(a + ih_2 + 0.5h_2)$$

$$h_2 = \frac{b-a}{4}$$

# 梯形法递推公式

$$T_{2^k} = \frac{1}{2} T_{2^{k-1}} + \frac{h_{k-1}}{2} \sum_{i=0}^{2^{k-1}-1} f(a + ih_{k-1} + 0.5h_{k-1})$$

$$h_{k-1} = \frac{b-a}{2^{k-1}}$$

记

$$T^{(k)} \equiv T_{2^k}$$

$$T^{(k)} = \frac{1}{2} T^{(k-1)} + \frac{h_{k-1}}{2} \sum_{i=0}^{2^{k-1}-1} f(a + ih_{k-1} + 0.5h_{k-1})$$

$$h_{k-1} = \frac{b-a}{2^{k-1}}$$

# 举例

$$I[f]=0.946083070367\dots$$

**例：**用梯形法的递推公式计算定积分  $\int_0^1 \frac{\sin(x)}{x} dx$  , 要求  
计算精度满足  $|T_{2n} - T_n| < \varepsilon = 10^{-7}$

**解：**

$$T^{(0)} = \frac{b-a}{2} (f(a) + f(b)) = 0.920735492$$

$$T^{(1)} = \frac{1}{2} T^{(0)} + \frac{h_0}{2} \sum_{i=0}^0 f(a + ih_0 + 0.5h_0) = 0.939793285$$

$$T^{(2)} = \frac{1}{2} T^{(1)} + \frac{h_1}{2} \sum_{i=0}^1 f(a + ih_1 + 0.5h_1) = 0.944513522$$

$$T^{(3)} = \frac{1}{2} T^{(2)} + \frac{h_2}{2} \sum_{i=0}^1 f(a + ih_2 + 0.5h_2) = 0.945690864$$

•  
•  
•

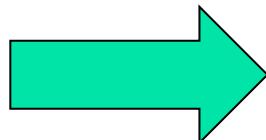
example\_3\_3.m

$k$	$T^{(k)}$
0	0.920735492
1	0.939793285
2	0.944513522
3	0.945690864
4	0.945985030
5	0.946058561
6	0.946076943
7	0.946081539
8	0.946082687
9	0.946082975
10	0.946083046



# 梯形法的加速

- 梯形法递推公式算法简单，编程方便  
但收敛速度较 慢

 梯形法的加速——龙贝格 (Romberg) 算法

定理：设  $f(x) \in C^\infty[a, b]$ , 记  $T_n = T(h)$ , 则有

$$T(h) = I[f] + \alpha_1 h^2 + \alpha_2 h^4 + \cdots + \alpha_i h^{2i} + \cdots$$

证明：略（利用 Taylor 展开即可）

$$h = \frac{b-a}{n}$$

# 梯形法的加速

$$T(h) = I[f] + \alpha_1 h^2 + \alpha_2 h^4 + \cdots + \alpha_i h^{2i} + \cdots = I[f] + O(h^2)$$

$$T\left(\frac{h}{2}\right) = I[f] + \alpha_1 \left(\frac{h}{2}\right)^2 + \alpha_2 \left(\frac{h}{2}\right)^4 + \cdots + \alpha_i \left(\frac{h}{2}\right)^{2i} + \cdots$$

$$4T(h/2) - T(h) = 3I[f] + (-3/4)\alpha_2 h^4 + (-15/16)\alpha_3 h^6 + \cdots$$

$$S(h) \equiv \frac{1}{3} \left( 4T\left(\frac{h}{2}\right) - T(h) \right) = I[f] + \beta_1 h^4 + \beta_2 h^6 + \cdots = I[f] + O(h^4)$$

$$C(h) \equiv \frac{1}{15} \left( 16S\left(\frac{h}{2}\right) - S(h) \right) = I[f] + \gamma_1 h^6 + \gamma_2 h^8 + \cdots = I[f] + O(h^6)$$

$$R(h) \equiv \frac{1}{63} \left( 64C\left(\frac{h}{2}\right) - C(h) \right) = I[f] + O(h^8)$$

⋮

**Richardson 外推算法**

# 举例

$$I[f]=0.946083070367\dots$$

例：计算定积分  $\int_0^1 \frac{\sin(x)}{x} dx$

$$T_1 = 0.920735492$$

$$T_2 = 0.939793285$$

$$T_4 = 0.944513522$$

$$S_1 = \frac{1}{3}(4T_2 - T_1) = 0.946145882$$

$$S_2 = \frac{1}{3}(4T_4 - T_2) = 0.946086934$$

$$C_1 = \frac{1}{15}(16S_2 - S_1) = 0.946083004$$

example\_3\_4.m

$k$	$T_0^{(k)}$
0	0.920735492
1	0.939793285
2	0.944513522
3	0.945690864
4	0.945985030
5	0.946058561
6	0.946076943
7	0.946081539
8	0.946082687
9	0.946082975
10	0.946083046

# Romberg 算法

记:  $T_0^{(k)} = T_{2^k}$ ,  $T_1^{(k)} = S_{2^k}$ ,  $T_2^{(k)} = C_{2^k}$ ,  $T_3^{(k)} = R_{2^k}$

$T_0^{(k)}$  :  $k$  次等分后梯形公式计算所得的近似值

$T_m^{(k)}$  :  $m$  次加速后所得的近似值

$$T_m^{(k)} = \frac{4^m T_{m-1}^{(k+1)} - T_{m-1}^{(k)}}{4^m - 1}$$

①  $T_1 = T_0^{(0)}$

②  $T_2 = T_0^{(1)}$

③  $S_1 = T_1^{(0)}$

④  $T_4 = T_0^{(2)}$

⑤  $S_2 = T_1^{(1)}$

⑥  $C_1 = T_2^{(0)}$

⑦  $T_8 = T_0^{(3)}$

⑧  $S_4 = T_1^{(2)}$

⑨  $C_2 = T_2^{(1)}$

⑩  $R_1 = T_3^{(0)}$

⋮

⋮

⋮

⋮

⋮

Romberg 算法是收敛的

# 举例

$$I[f]=0.4$$

**例：**用 Romberg 算法计算定积分  $\int_0^1 \sqrt{x^3} dx$ ，要求计算精度满足  $|T_m^{(k)} - T_{m-1}^{(k)}| < \varepsilon = 10^{-7}$

example\_3\_5.m

**解：**逐步计算可得

$k$	$T_0^{(k)}$	$T_1^{(k)}$	$T_2^{(k)}$	$T_3^{(k)}$	$T_4^{(k)}$	$T_5^{(k)}$
0	0.50000000					
1	0.42677670	0.40236893				
2	0.40701811	0.40043192	0.40030278			
3	0.40181246	0.40007725	0.40005361	0.40004965		
4	0.40046340	0.40001371	0.40000948	0.40000878	0.40000862	
5	0.40011767	0.40000243	0.40000168	0.40000155	0.40000152	0.40000152