



北京化工大学

Beijing University of Chemical Technology

计算方法讲义 (三)

数值积分

Cheng Yong

目录

第 1 章 数值积分	1
1.1 机械求积	1
1.2 Newton-Cotes 公式	5
1.3 复化求积公式	12
1.4 龙贝格算法	16
1.5 Gauss 公式	21
1.6 本章小结	25

创建日期: 2019 年 7 月 5 日
更新日期: 2020 年 2 月 11 日

第 1 章 数值积分

对于积分

$$\int_a^b f(x) dx \quad (1.1)$$

如果知道 $f(x)$ 的原函数 $F(x)$, 则由 Newton-Leibniz 公式可求得:

$$\int_a^b f(x) dx = F(x) \Big|_a^b = F(b) - F(a) \quad (1.2)$$

但事实上, 我们不难发现, 微积分方法求积分也确有其局限性:

1. $f(x)$ 根本不存在, 往往只是给出一张数据表来表示函数的关系;
2. 被积函数没有初等函数表示的原函数;
3. $f(x)$ 的表达式结构复杂, 求原函数较困难;

显然这些情况下将无法运用 Newton-Leibniz 来求积分, 而需要建立定积分的近似计算公式。这类方法很多, 但最常用的一种方法是利用插值多项式来构造数值求积公式, 具体步骤如下:

1. 构造一个多项式 $p(x)$ 逼近被积函数 $f(x)$;
2. 以 $p(x)$ 的积分近似代替 $f(x)$ 的积分; 即

$$\int_a^b f(x) dx \approx \int_a^b p(x) dx \quad (1.3)$$

多项式 $p(x)$ 的积分则是大家都会计算的, 下面就根据这一想法构造计算定积分的各种近似计算公式。

1.1 机械求积

依据积分中值定理, 对于连续函数 $f(x)$, 在 $[a, b]$ 内存在一点 ξ , 成立

$$\int_a^b f(x) dx = (b - a) f(\xi) \quad (1.4)$$

即底为 $b - a$ 而高为 $f(\xi)$ 的矩形面积恰恰等于所求曲边梯形的面积。称 $f(\xi)$ 为区间 $[a, b]$ 上的平均高度, 而这一平均高度是很难计算的, 但可以采用另一类办法来解决这一问题, 就是对平均高度 $f(\xi)$ 提供一种近似算法。

$$\int_a^b f(x) dx \approx (b - a) \sum_{i=0}^n \lambda_i f(x_i) \quad (1.5)$$

一般地，在求积分时，取 $[a, b]$ 内若干个节点 x_i 处的高度 $f(x_i)$ ，通过加权平均的方法得出平均高度 $f(\xi)$ 的近似值，这类求积方法称机械求积，式中 x_i 称为求积节点， λ_i 称为求积系数。

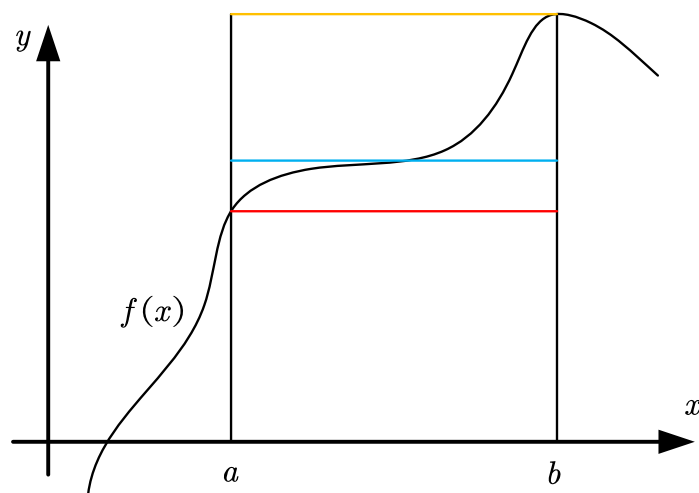


图 1.1: 机械求积公式

$$S_1 = f(a)(b - a) \quad (1.6)$$

$$S_2 = f(b)(b - a) \quad (1.7)$$

$$S_3 = f\left(\frac{a+b}{2}\right)(b - a) \quad (1.8)$$

$$S_4 = \frac{S_1 + S_2}{2} = \frac{f(a) + f(b)}{2}(b - a) \quad (1.9)$$

(1) 梯形求积公式

过两点 a, b 作直线 (两点 Lagrange 插值)

$$p_1(x) = \frac{x - b}{a - b}f(a) + \frac{x - a}{b - a}f(b) \quad (1.10)$$

用 $p_1(x)$ 代替 $f(x)$ 得:

$$\int_a^b f(x)dx \approx \int_a^b p_1(x)dx \quad (1.11)$$

$$= \int_a^b \left(\frac{x - b}{a - b}f(a) + \frac{x - a}{b - a}f(b) \right) dx \quad (1.12)$$

$$= (b - a) \left(\frac{1}{2}f(a) + \frac{1}{2}f(b) \right) \quad (1.13)$$

$$= \frac{b - a}{2} (f(a) + f(b)) \quad (1.14)$$

称为梯形求积公式 (两点公式)。

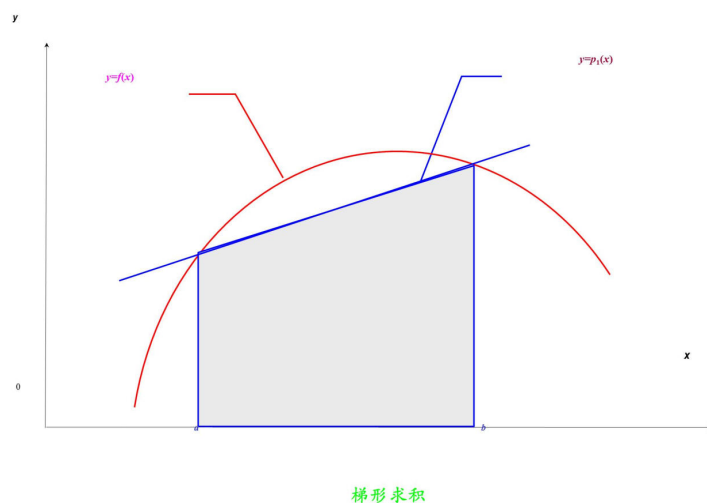


图 1.2: 梯形求积公式几何意义

例 1. 利用梯形求积公式计算定积分:

$$\int_{0.5}^1 \sqrt{x} dx \quad (1.15)$$

解: 依梯形求积公式有:

$$\int_{0.5}^1 \sqrt{x} dx \approx \frac{1-0.5}{2} (\sqrt{0.5} + \sqrt{1}) \quad (1.16)$$

$$= 0.4267767 \quad (1.17)$$

积分准确值为:

$$\int_{0.5}^1 \sqrt{x} dx = \frac{2}{3} x^{\frac{3}{2}} \Big|_{0.5}^1 \quad (1.18)$$

$$= 0.43096441 \quad (1.19)$$

定义 1. 机械求积公式

$$\int_a^b f(x) dx \approx (b-a) \sum_{i=0}^n \lambda_i f(x_i) \quad (1.20)$$

具有 m 阶 (代数) 精度, 如果它对于任意次数不超过 m 次多项式 $f(x) = p_k(x) (0 \leq k \leq m)$ 是准确的, 但对于 $m+1$ 次多项式 $f(x) = p_{m+1}(x)$ 不准确。即有:

$$\int_a^b p_k(x) dx = (b-a) \sum_{i=0}^n \lambda_i p_k(x_i) \quad k = 0, 1, 2, \dots, m \quad (1.21)$$

$$\int_a^b p_{m+1}(x) dx \neq (b-a) \sum_{i=0}^n \lambda_i p_{m+1}(x_i) \quad (1.22)$$

求积公式代数精度的判断可以简化如下：如果求积公式对于任意次数不超过 m 次的幂函数 $f(x) = x^k (0 \leq k \leq m)$ 是准确的，但对于 $m+1$ 次幂函数 $f(x) = x^{m+1}$ 不准确，则可判定求积公式：

$$\int_a^b f(x) dx \approx (b-a) \sum_{i=0}^n \lambda_i f(x_i) \quad (1.23)$$

具有 m 阶 (代数) 精度。

例 2. 判断梯形求积公式的代数精度。

对梯形求积公式：

$$\int_a^b f(x) dx \approx \frac{b-a}{2} (f(a) + f(b)) \quad (1.24)$$

当 $f(x) = 1$ 时

$$\int_a^b f(x) dx = \int_a^b 1 dx = b-a \quad (1.25)$$

$$\frac{b-a}{2} (f(a) + f(b)) = \frac{b-a}{2} (1+1) = b-a \quad (1.26)$$

因此有：

$$\int_a^b f(x) dx = \frac{b-a}{2} (f(a) + f(b)) \quad (1.27)$$

当 $f(x) = x$ 时

$$\int_a^b f(x) dx = \int_a^b x dx = \frac{1}{2} (b^2 - a^2) \quad (1.28)$$

$$\frac{b-a}{2} (f(a) + f(b)) = \frac{b-a}{2} (a+b) = \frac{1}{2} (b^2 - a^2) \quad (1.29)$$

因此有：

$$\int_a^b f(x) dx = \frac{b-a}{2} (f(a) + f(b)) \quad (1.30)$$

当 $f(x) = x^2$ 时

$$\int_a^b f(x) dx = \int_a^b x^2 dx = \frac{1}{3} (b^3 - a^3) \quad (1.31)$$

$$\frac{b-a}{2} (f(a) + f(b)) = \frac{b-a}{2} (a^2 + b^2) \quad (1.32)$$

因此：

$$\int_a^b f(x) dx \neq \frac{b-a}{2} (f(a) + f(b)) \quad (1.33)$$

由定义可知，梯形求积公式具有 1 次代数精度。

1.2 Newton-Cotes 公式

Newton-Cotes 公式是指在等距节点下使用 Lagrange 插值多项式建立的数值求积公式。

设将求积区间 $[a, b]$ 划分为 n 等分, 选取等分点

$$x_i = a + ih, \quad h = \frac{b-a}{n}, \quad i = 0, 1, 2, \dots, n \quad (1.34)$$

作为求积节点构造求积公式

$$\int_a^b f(x)dx \approx (b-a) \sum_{i=0}^n \lambda_i f(x_i) \quad (1.35)$$

这种求积公式称为 Newton-Cotes 公式。

(2) Simpson 求积公式

将求积区间 $[a, b]$ 两等分, 过三点 $a, (a+b)/2, b$ 作抛物线 (三点 Lagrange 插值)

$$p_2(x) = \frac{2}{(b-a)^2} \left(x - \frac{a+b}{2} \right) (x-b) f(a) \quad (1.36)$$

$$- 2 \frac{2}{(b-a)^2} (x-a)(x-b) f\left(\frac{a+b}{2}\right) \quad (1.37)$$

$$+ \frac{2}{(b-a)^2} (x-a) \left(x - \frac{a+b}{2} \right) f(b) \quad (1.38)$$

用 $p_2(x)$ 代替 $f(x)$ 得:

$$\int_a^b f(x)dx \approx \int_a^b p_2(x)dx = \frac{b-a}{6} \left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right) \quad (1.39)$$

称为 Simpson 公式, 这是 Newton-Cotes 公式的一个特例。

例 3. 利用 Simpson 公式计算定积分:

$$\int_{0.5}^1 \sqrt{x}dx \quad (1.40)$$

解: 由 Simpson 公式:

$$\int_{0.5}^1 \sqrt{x}dx \approx \frac{1-0.5}{6} (\sqrt{0.5} + 4\sqrt{0.75} + \sqrt{1}) \quad (1.41)$$

$$= 0.43093403 \quad (1.42)$$

积分准确值为:

$$\int_{0.5}^1 \sqrt{x}dx = \frac{2}{3} x^{\frac{3}{2}} \Big|_{0.5}^1 = 0.43096441 \quad (1.43)$$

同样可以证明 Simpson 求积公式具有三次代数精度。

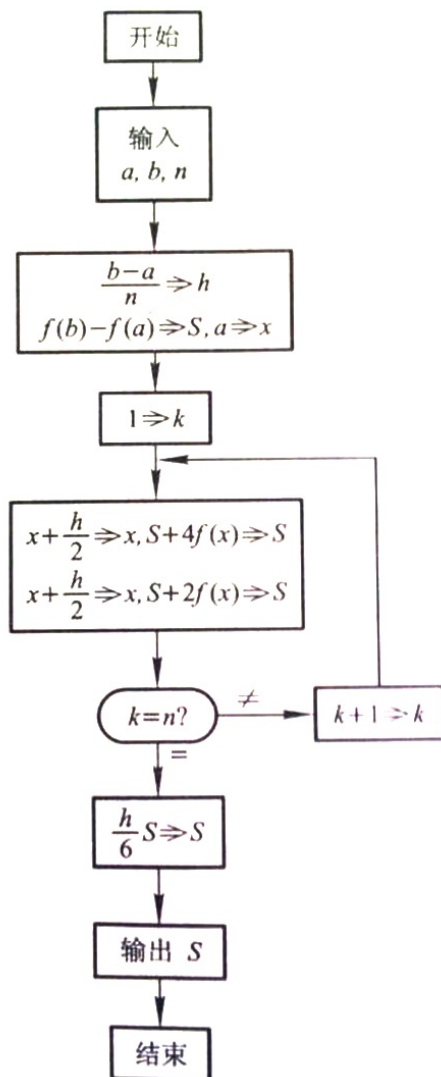


图 1.3: Simpson 求积公式计算流程图

(3) Newton-Cotes 公式

将求积区间 $[a, b]$ 划分为 n 等分, 选取等分点

$$x_i = a + ih, \quad h = \frac{b-a}{n}, \quad i = 0, 1, 2, \dots, n \quad (1.44)$$

构造 n 次 Lagrange 插值

$$L_n(x) = \sum_{i=0}^n \varphi_i(x) f(x_i) = \sum_{i=0}^n \left(\prod_{j=0, j \neq i}^n \frac{x - x_j}{x_i - x_j} \right) f(x_i) \quad (1.45)$$

用 $L_n(x)$ 代替 $f(x)$ 得:

$$\int_a^b f(x)dx \approx \int_a^b L_n(x)dx \quad (1.46)$$

$$= \int_a^b \sum_{i=0}^n \left(\prod_{j=0, j \neq i}^n \frac{x - x_j}{x_i - x_j} \right) f(x_i) dx \quad (1.47)$$

$$= \sum_{i=0}^n A_i f(x_i) = (b-a) \sum_{i=0}^n \lambda_i f(x_i) \quad (1.48)$$

其中

$$A_i = \int_a^b \left(\prod_{j=0, j \neq i}^n \frac{x - x_j}{x_i - x_j} \right) dx \quad (1.49)$$

称为 Newton-Cotes 公式。

Newton-Cotes 公式是指在等距节点下使用 Lagrange 插值多项式建立的数值求积公式。

设将求积区间 $[a, b]$ 划分为 n 等分, 选取等分点

$$x_i = a + ih, \quad h = \frac{b-a}{n}, \quad i = 0, 1, 2, \dots, n \quad (1.50)$$

作为求积节点构造求积公式

$$\int_a^b f(x)dx \approx (b-a) \sum_{i=0}^n \lambda_i f(x_i) \quad (1.51)$$

这种求积公式称为 Newton-Cotes 公式。

在插值求积公式

$$\int_a^b f(x)dx \approx \int_a^b P(x)dx = \sum_{k=0}^n A_k f(x_k) \quad (1.52)$$

当所取结点是等距时称为牛顿-科茨公式。其中插值多项式 $P(x)$ 为:

$$P(x) = \sum_{k=0}^n l_k(x) f(x_k) \quad (1.53)$$

求积系数为:

$$A_k = \int_a^b l_k(x)dx \quad (1.54)$$

设将积分区间为 $[a, b]$ 分为 n 等分, 求积结点为 $\{x_k\}_{k=0}^n$, 那么有 $h = \frac{b-a}{n}$:

$$x_0 = a, x_n = b, x_j = a + jh, \quad j = 0, 1, \dots, n \quad (1.55)$$

令 $x = a + th$, 则 $t = (x - a)/h$, 且由 $x \in [a, b]$ 可知: $t \in [0, n]$ 。

我们得到:

$$A_k = \int_a^b l_k(x) dx \quad (1.56)$$

$$= \int_a^b \prod_{\substack{i=0 \\ i \neq k}}^n \frac{x - x_i}{x_k - x_i} dx \quad (1.57)$$

$$= \int_0^n l_k(a + th) h dt \quad (1.58)$$

$$= \int_0^n \prod_{\substack{i=0 \\ i \neq k}}^n \frac{t - i}{k - i} h dt \quad (1.59)$$

$$= (b - a) \cdot \frac{1}{n} \frac{(-1)^{n-k}}{k!(n-k)!} \int_0^n \prod_{\substack{i=0 \\ i \neq k}}^n (t - i) dt \quad (1.60)$$

$$= (b - a) \cdot C_k^{(n)} \quad (1.61)$$

这里 $C_k^{(n)}$ 就是 Cotes 公式的系数, 即:

$$C_k^{(n)} = \frac{1}{n} \cdot \frac{(-1)^{(n-k)}}{k!(n-k)!} \int_0^n \prod_{\substack{i=0 \\ i \neq k}}^n (t - i) dt \quad k = 0, 1, 2, \dots, n \quad (1.62)$$

因此, Newton-Cotes 公式又可以表示为:

$$\int_a^b f(x) dx \approx \int_a^b L_n(x) dx = (b - a) \sum_{k=0}^n C_k^{(n)} f(x_k) \quad (1.63)$$

Newton-Cotes 系数具有如下性质:

性质 1 (对称性).

$$C_k^{(n)} = C_{n-k}^{(n)} \quad (1.64)$$

性质 2 (归一性).

$$\sum_{k=0}^n C_k^{(n)} = 1 \quad (1.65)$$

$n = 1$ 时的梯形求积公式

求得相应的系数如下:

$$C_0^{(1)} = \frac{-1}{1 \cdot 0! \cdot 1!} \int_0^1 (t - 1) dt = \frac{1}{2} \quad (1.66)$$

$$C_1^{(1)} = \int_0^1 t dt = \frac{1}{2} \quad (1.67)$$

因此得到如下的求积公式：

$$\int_a^b f(x)dx \approx \frac{(b-a)}{2}(f(a) + f(b)) \quad (1.68)$$

梯形求积公式的代数精度为 1。

$n = 2$ 时的辛普生求积公式

$n = 2$ 的抛物线求积公式相应的系数如下：

$$C_0^{(2)} = \frac{(-1)^2}{2 \cdot 0! \cdot 2!} \int_0^2 (t-1)(t-2)dt = \frac{1}{6} \quad (1.69)$$

$$C_1^{(2)} = \frac{(-1)^1}{2 \cdot 1! \cdot 1!} \int_0^2 t(t-2)dt = \frac{2}{3} \quad (1.70)$$

$$C_2^{(2)} = \frac{(-1)^0}{2 \cdot 2! \cdot 0!} \int_0^2 t(t-1)dt = \frac{1}{6} \quad (1.71)$$

所以有：

$$\int_a^b f(x)dx \approx \frac{(b-a)}{6} \left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right) \quad (1.72)$$

辛普生求积公式的代数精度为 3。

$n = 4$ 时的科茨求积公式

同理，我们可以求得科茨公式的系数为：

$$C_0^{(4)} = \frac{7}{90}, C_1^{(4)} = \frac{16}{45}, C_2^{(4)} = \frac{2}{15}, C_3^{(4)} = \frac{16}{45}, C_4^{(4)} = \frac{7}{90} \quad (1.73)$$

得到 Cotes 求积公式如下：

$$\int_a^b f(x)dx \approx \frac{b-a}{90} (7f(x_0) + 32f(x_1) + 12f(x_2) + 32f(x_3) + 7f(x_4)) \quad (1.74)$$

Cotes 求积公式的代数精度为 5。

其他求积公式系数依上述式子可以得到，不再详细描述。

梯形求积公式和 Simpson 公式是 Newton-Cotes 公式当 $n = 1$ 和 $n = 2$ 时的两个特例。

$$\int_a^b f(x)dx = \frac{b-a}{2} (f(a) + f(b)) \quad \text{梯形公式} \quad (1.75)$$

$$\int_a^b f(x)dx = \frac{b-a}{6} \left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right) \quad \text{Simpson 公式} \quad (1.76)$$

n	求积系数					
1	1/2	1/2				
2	1/6	4/6	1/6			
3	1/8	3/8	3/8	1/8		
4	7/90	16/45	2/15	16/45	7/90	
5	19/288	25/96	25/144	25/144	25/96	19/288

图 1.4: Newton-Cotes 公式求积系数

例 4. 利用 *Newton-Cotes* 公式 (取 $n = 4$, 此时称为 *Cotes* 公式) 计算定积分:

$$\int_{0.5}^1 \sqrt{x} dx \quad (1.77)$$

解: 根据 Newton-Cotes 公式:

$$\int_{0.5}^1 \sqrt{x} dx \approx \frac{1 - 0.5}{90} (7\sqrt{0.5} + 32\sqrt{0.625} \quad (1.78)$$

$$+ 12\sqrt{0.75} + 32\sqrt{0.875} + 7) \quad (1.79)$$

$$= 0.43096407 \quad (1.80)$$

积分准确值为:

$$\int_{0.5}^1 \sqrt{x} dx = \frac{2}{3} x^{\frac{3}{2}} \Big|_{0.5}^1 \quad (1.81)$$

$$= 0.43096441 \quad (1.82)$$

```

1  #include <stdio.h>
2  #define MAXSIZE 7
3  void input(double f[MAXSIZE+ 1], double a, double b, long n);
4  void main(void) {
5      long c[MAXSIZE][MAXSIZE/2+2] = {{2, 1}, {6, 1, 4}, {8, 1, 3}, {90, 7,
6          32, 12},
7          {288, 19, 75, 50}, {840, 41, 216, 27, 272}, {17280, 751, 3577, 1323,
8          2989}
9      };
10     double a, b, f[MAXSIZE + 1], integral;
11     long n, i;
12     printf("\n请输入积分区间边界a, b:");
13     scanf("%lf, %lf", &a, &b);
14     printf("\n请输入积分节点的个数(2~8):");
15     scanf("%ld", &n);

```

```

14     input(f, a, b, n);
15     integral = 0;
16     for(i = 0; i < n/2; i++)
17         integral += (f[i]+f[n-i-1])*c[n-2][i+1]/c[n-2][0];
18     if(n%2)
19         integral += f[n/2]*c[n-2][n/2+1]/c[n-2][0];
20     integral *= b-a;
21     printf("\n积分值为=%lf", integral);
22 }
23 void input(double f[MAXSIZE+ 1], double a, double b, long n) {
24     long i ;
25     double h;
26     h = (b - a)/ (n - 1);
27     printf("\n请输入求积节点纵坐标: ");
28     for(i = 0; i <= n-1; i++) {
29         printf("nx[%ld] = %lf, f[%ld] =", i, a+i*h, i);
30         scanf("%lf", &f[i]);
31     }
32 }

```

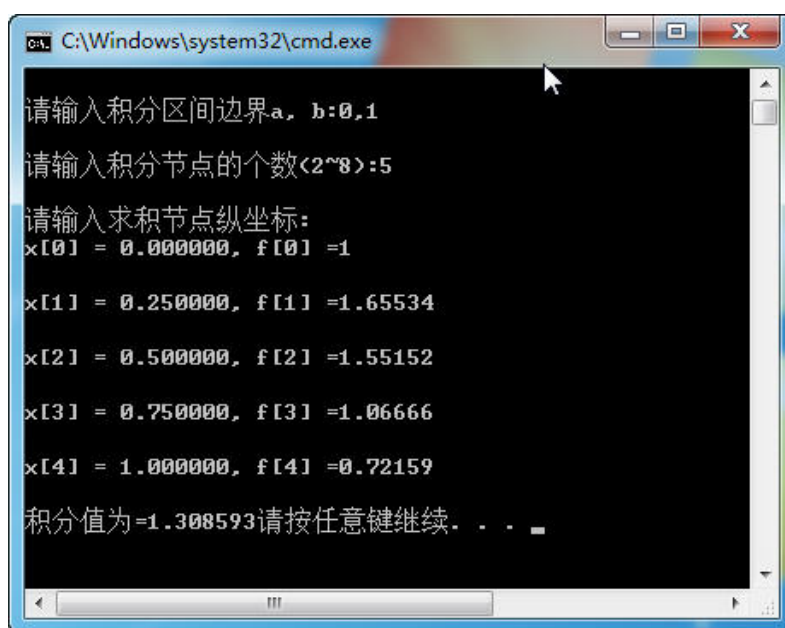


图 1.5: Newton-Cotes 公式运行结果

Newton-Cotes 公式代数精度

可以证明 Newton-Cotes 公式的代数精度至少为 n 。且当 n 为偶数时，代数精度可达 $n+1$ 。如：

$n=1$ 时 (梯形公式)，代数精度为 1；

$n=2$ 时 (Simpson 公式)，代数精度为 $2+1=3$ ；

$n=4$ 时 (Cotes 公式)，代数精度为 $4+1=5$ ，以此类推。

1.3 复化求积公式

为了提高求积公式的精度，又使算法简单易行，往往使用复合方法，即将积分区间 $[a, b]$ 分成若干个子区间，然后在每个小区间上使用低阶 Newton-Cotes 公式，最后将每个小区间上的积分的近似值相加作为积分的近似值，这种方法称为复化求积法。

复化梯形公式

将求积区间 n 等分，步长 $(b-a)/n$ ，分点为 $x_i = a + ih (i = 0, 1, 2, \dots, n)$ ，对每个小区间 $[x_i, x_{i+1}]$ 使用梯形求积公式，则

$$I = \int_a^b f(x)dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x)dx \approx \sum_{i=0}^{n-1} \frac{x_{i+1} - x_i}{2} (f(x_i) + f(x_{i+1})) \quad (1.83)$$

$$= \frac{b-a}{2n} (f(a) + 2 \sum_{i=1}^{n-1} f(x_i) + f(b)) = T_n \quad (1.84)$$

例 5. 已知 $f(x) = x^3 - 2x + 1$ 由表给出，用复化梯形公式近似计算定积分 $\int_1^3 f(x)dx$ 。

x	1	3/2	2	5/2	3
$f(x)$	0	11/8	5	93/8	22

表 1.1: 中文表

解：此时 $n = 4$

$$T_4 = \frac{b-a}{2n} (f(a) + 2 \sum_{i=1}^{n-1} f(x_i) + f(b)) \quad (1.85)$$

$$= \frac{3-1}{2 \cdot 4} (f(1) + 2(f(3/2) + f(2) + f(5/2)) + f(3)) \quad (1.86)$$

$$= \frac{1}{4} (0 + 2(11/8 + 5 + 93/8) + 22) = 14.5 \quad (1.87)$$

实际上 $\int_1^3 (x^3 - 2x + 1)dx = 14$

复化 Simpson 公式

将求积区间 n 等分的基础上，再将每个小区间 $[x_i, x_{i+1}]$ 2 等份，记内分点为 $x_{i+1/2}$ $x_i = a + ih, x_{i+1/2} = a + (i+1/2)h, (i = 0, 1, 2, 3, \dots, n)$ 。对小区间 $[x_i, x_{i+1}]$ 使用 Simpson

求积公式，得到：

$$I = \int_a^b f(x)dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x)dx \quad (1.88)$$

$$\approx \sum_{i=0}^{n-1} \frac{x_{i+1} - x_i}{6} \left(f(x_i) + 4f(x_{i+\frac{1}{2}}) + f(x_{i+1}) \right) \quad (1.89)$$

$$= \frac{b-a}{6n} \left(f(a) + 4 \sum_{i=0}^{n-1} f(x_{i+\frac{1}{2}}) + 2 \sum_{i=1}^{n-1} f(x_i) + f(b) \right) \quad (1.90)$$

$$= S_n \quad (1.91)$$

复化 Cotes 公式

将求积区间 n 等分的基础上，再将每个小区间 $[x_i, x_{i+1}]$ 分为 4 等分，记内分点为 $x_{i+1/4}, x_{i+1/2}, x_{i+3/4}$ ，即：

$$x_i = a + ih \quad (1.92)$$

$$x_{i+\frac{1}{4}} = a + \left(i + \frac{1}{4}\right)h \quad (1.93)$$

$$x_{i+\frac{1}{2}} = a + \left(i + \frac{1}{2}\right)h \quad (1.94)$$

$$x_{i+\frac{3}{4}} = a + \left(i + \frac{3}{4}\right)h \quad (1.95)$$

$$x_{i+1} = a + (i+1)h \quad (i = 0, 1, 2, \dots, n) \quad (1.96)$$

对小区间 $[x_i, x_{i+1}]$ 使用 Cotes 求积公式，则

$$I = \int_a^b f(x)dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x)dx \quad (1.97)$$

$$\approx \sum_{i=0}^{n-1} \frac{x_{i+1} - x_i}{90} (7f(x_i) + 32f(x_{i+\frac{1}{4}}) \quad (1.98)$$

$$+ 12f(x_{i+\frac{1}{2}}) + 32f(x_{i+\frac{3}{4}}) + 7f(x_{i+1})) \quad (1.99)$$

$$= \frac{b-a}{90n} (7f(a) + 32 \sum_{i=0}^{n-1} f(x_{i+\frac{1}{4}}) + 12 \sum_{i=0}^{n-1} f(x_{i+\frac{1}{2}}) \quad (1.100)$$

$$+ 32 \sum_{i=0}^{n-1} f(x_{i+\frac{3}{4}}) + 14 \sum_{i=1}^{n-1} f(x_i) + 7f(b)) = C_n \quad (1.101)$$

例 6. 用函数 $f(x) = \frac{\sin x}{x}$ 的数据表计算积分 $I = \int_0^1 \frac{\sin x}{x} dx$ 。

0	1.0000000
1/8	0.9973978
2/8	0.9896158
3/8	0.9767267
4/8	0.9588510
5/8	0.9361556
6/8	0.9088516
7/8	0.8771925
1	0.8414709

解：此时 $n = 8$

$$T_8 = \frac{b-a}{2n} (f(a) + 2 \sum_{i=1}^{n-1} f(x_i) + f(b)) \quad (1.102)$$

$$= \frac{1-0}{2 \cdot 8} (f(a) + 2 \sum_{i=1}^{8-1} f(x_i) + f(b)) \quad (1.103)$$

$$= 0.945690806 \quad (1.104)$$

$$S_4 = \frac{b-a}{6n} \left(f(a) + 4 \sum_{i=0}^{n-1} f(x_{i+\frac{1}{2}}) + 2 \sum_{i=1}^{n-1} f(x_i) + f(b) \right) \quad (1.105)$$

$$= \frac{1-0}{6 \cdot 4} (f(a) + 4 \sum_{i=0}^{4-1} f(x_{i+\frac{1}{2}}) + 2 \sum_{i=1}^{4-1} f(x_i) + f(b)) \quad (1.106)$$

$$= 0.946083254 \quad (1.107)$$

同理，应用复合 Cotes 公式可求得 C_2 值如下：

$$C_2 = 0.94608307 \quad (1.108)$$

该积分的“精确值”为

$$I = \int_0^1 \frac{\sin x}{x} dx = 0.9460831 \quad (1.109)$$

对上述三个结果进行比较可以看出， T_8 精度最低、 S_4 精度次高， C_2 精度最高。上述积分都需要提供 9 个节点上的函数值，工作量基本相同，然而精度却相差很大，同积分的精确值比较，复化梯形法的结果有 3 位有效数值，而复化 Simpson 法和 Cotes 的结果有 8 位有效数值。

```
1 #include "stdio.h"
2
3 double compositeSimpson(double(*f)(double), double a, double b, int n);
```

```
4 double myfun(double x);
5
6 void main() {
7     double(*fun)(double);
8     double a,b,S4;
9     int n;
10
11     a=0;
12     b=1;
13     n=4;
14     fun=myfun;
15     S4=compositeSimpson(fun,a,b,n);
16     printf("/n积分值为:%f/n",S4);
17
18 }
19
20
21 /*
22  * 用复合辛普生法计算函数f(x)从a到b的积分值
23  * 输入: f--函数f(x)的指针
24  * a--积分下限
25  * b--积分上限
26  * n--分段数
27  * 输出: 返回值为f(x)从a点到b点的积分值
28  */
29 double compositeSimpson(double(*f)(double),double a,double b,int n) {
30     double s,h,x;
31     int i;
32     printf("x/t/tf(x)/t/ts/n");
33     s=(f)(a)-(f)(b);
34     h=(b-a)/(2*n);
35     x=a;
36     for(i=1; i<2*n; i+=2) {
37         x+=h;
38         s+=4*(f)(x);
39         printf("%f/t%f/t%f/n",x,(f)(x),s*h/3);
40         x+=h;
41         s+=2*(f)(x);
42         printf("%f/t%f/t%f/n",x,(f)(x),s*h/3);
43     }
44     return(s*h/3);
45 }
46
47
48
49 double myfun(double x) {
50     double y;
51     y=4.0/(1+x*x);
```

```

52     return(y);
53 }

```

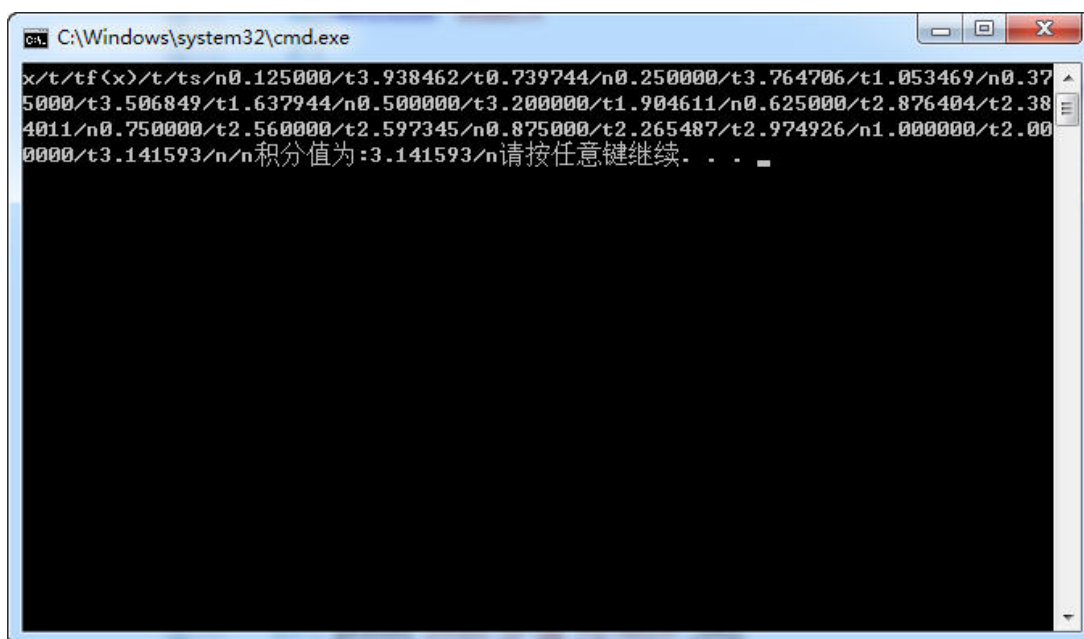


图 1.6: 复合辛普生公式结果

1.4 龙贝格算法

梯形法加速

复化梯形法的算法简单但精度低。能否加工梯形值来提高精度呢？考查二分前后的梯形公式：

$$T_1 = \frac{b-a}{2}(f(a) + f(b)) \quad (1.110)$$

$$T_2 = \frac{1}{2}T_1 + \frac{b-a}{2}f\left(\frac{a+b}{2}\right) \quad (1.111)$$

$$= \frac{b-a}{4}(f(a) + 2f\left(\frac{a+b}{2}\right) + f(b)) \quad (1.112)$$

$$S_1 = \frac{b-a}{6}(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)) \quad (1.113)$$

因此，我们得到

$$S_1 = \frac{4}{3}T_2 - \frac{1}{3}T_1 \quad (1.114)$$

Romberg 算法

推广到复合求积公式，有：

$$S_n = \frac{4}{3}T_{2n} - \frac{1}{3}T_n \quad (1.115)$$

同理，我们可以得到：

$$C_n = \frac{16}{15}S_{2n} - \frac{1}{15}S_n \quad (1.116)$$

$$R_n = \frac{64}{63}C_{2n} - \frac{1}{63}C_n \quad (1.117)$$

此方法称为 Romberg 公式，它是由复化梯形公式、复化 Simpson 公式依次递推求得的，松弛因子是 $\omega = 1/63$ 。它能将粗糙的梯形公式逐步加工成高精度的 Romberg 公式，称这种算法为 Romberg 算法。综合表示如下：

$$\begin{cases} T_1 = \frac{b-a}{2}[f(a) + f(b)] \\ T_{2n} = \frac{1}{2}T_n + \frac{h}{2} \sum_{i=0}^{n-1} f(x_{i+\frac{1}{2}}) \\ S_n = \frac{4}{3}T_{2n} - \frac{1}{3}T_n \\ C_n = \frac{16}{15}S_{2n} - \frac{1}{15}S_n \\ R_n = \frac{64}{63}C_{2n} - \frac{1}{63}C_n \end{cases} \quad (1.118)$$

$n = 2^k, k = 0, 1, 2, 3, \dots$ ，以上整个过程称为 Romberg 算法。

k	T_{2^k}	$S_{2^{k-1}}$	$C_{2^{k-2}}$	$R_{2^{k-3}}$
0	T_1			
1	T_2	S_1		
2	T_4	S_2	C_1	
3	T_8	S_4	C_2	R_1
4	T_{16}	S_8	C_4	R_2

表 1.2: 龙贝格算法

例 7. 利用 Romberg 算法计算积分

$$I = \int_0^1 \frac{\sin x}{x} dx \quad (1.119)$$

解：计算求得

k	T_{2^k}	$S_{2^{k-1}}$	$C_{2^{k-2}}$	$R_{2^{k-3}}$
0	0.9207355			
1	0.9397933	0.9461459		
2	0.9445135	0.9460869	0.9460830	
3	0.9456909	0.9460834	0.9460831	0.9460831

得到 $R_1 = 0.9460831$ 。

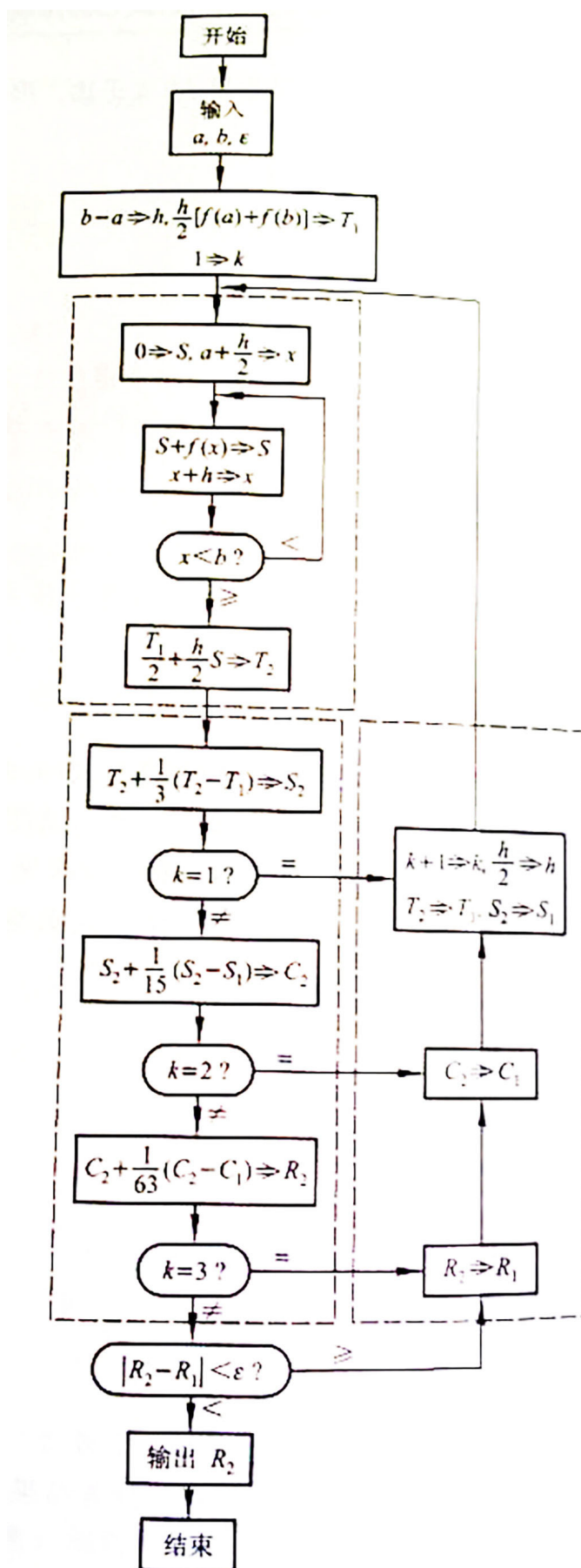


图 1.7: 龙贝格公式流程图

```
1  #include<iostream>
2  #include<math.h>
3
4  using namespace std;
5
6  float f(float x) {
7      return 1/(1+x*x);
8  }
9  float romberg(float a,float b,float (*f)(float),float epsilon) {
10     int n=1,k;
11     float h=b-a,x,temp;
12     float T1,T2,S1,S2,C1,C2,R1,R2;
13     T1=(b-a)/2*((*f)(a)+(*f)(b));
14     while(1) {
15         temp=0;
16         for(k=0; k<=n-1; k++) {
17             x=a+k*h+h/2;
18             temp+=(*f)(x);
19         }
20         T2=(T1+temp*h)/2;
21         if(fabs(T2-T1)<epsilon)
22             return T2;
23         S2=T2+(T2-T1)/3.0;
24         if(n==1) {
25             T1=T2;
26             S1=S2;
27             h/=2;
28             n*=2;
29             continue;
30         }
31         C2=S2+(S2-S1)/15;
32         if(n==2) {
33             C1=C2;
34             T1=T2;
35             S1=S2;
36             h/=2;
37             n*=2;
38             continue;
39         }
40         R2=C2+(C2-C1)/63;
41         if(n==4) {
42             R1=R2;
43             C1=C2;
44             T1=T2;
45             S1=S2;
46             h/=2;
47             n*=2;
48             continue;
```

```

49     }
50     if(fabs(R2-R1)<epsilon)
51         return R2;
52     R1=R2;
53     C1=C2;
54     T1=T2;
55     S1=S2;
56     h/=2;
57     n*=2;
58 }
59 }
60 void main() {
61     float epsilon=5e-6;
62     cout<<"R="<<romberg(0,1,f,epsilon)<<endl;
63 }

```

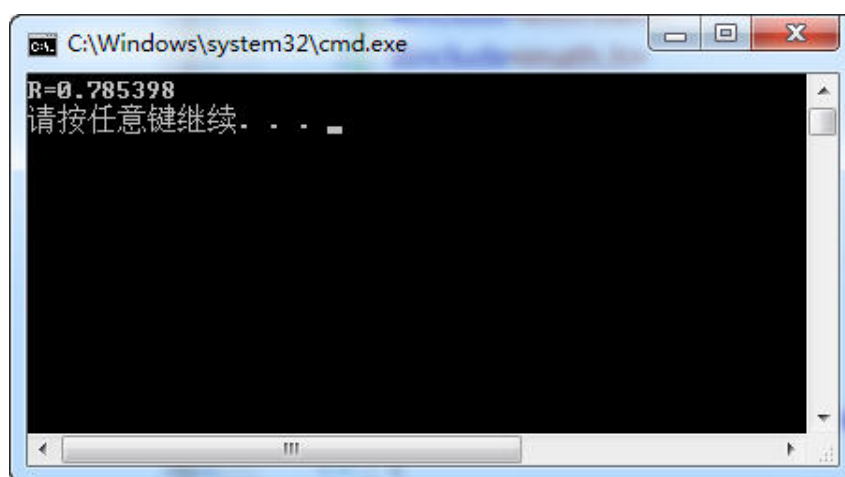


图 1.8: 龙贝格公式运行结果

1.5 Gauss 公式

Newton-Cotes 公式在构造时，限定积分区间的等分点作为求积节点，这样做在简化处理的同时也限制了代数精度。如果求积节点也可自由选择，即机械求积公式：

$$\int_a^b f(x)dx \approx (b-a) \sum_{i=0}^n \lambda_i f(x_i) \quad (1.120)$$

中的 λ_i , x_i 共 $2n+2$ 个均为待定参数，适当选取这些参数可以使公式具有 $2n+1$ 次代数精度。这种高精度的求积公式称为 Gauss 公式。

积分区间变换

不失一般性，把积分区间取成 $[-1, 1]$ ，这是因为利用变换：

$$x = \frac{a+b}{2} + \frac{b-a}{2}t \quad (1.121)$$

总可将区间 $[a, b]$ 变换成 $[-1, 1]$, 而积分变为:

$$\int_a^b f(x)dx = \frac{b-a}{2} \int_{-1}^1 g(t)dt \quad (1.122)$$

其中

$$g(t) = f\left(\frac{a+b}{2} + \frac{b-a}{2}t\right) \quad (1.123)$$

一点 Gauss 公式

对于一点 Gauss 公式 ($n=0$, 具有一次代数精度)

$$\int_{-1}^1 f(x)dx \approx 2 \sum_{i=0}^0 \lambda_i f(x_i) = 2\lambda_0 f(x_0) \quad (1.124)$$

因此当 $f(x) = 1, f(x) = x$ 时上式精确成立, 有:

$$\begin{cases} 2\lambda_0 = 2 \\ 2\lambda_0 x_0 = 0 \end{cases} \quad (1.125)$$

解得: $\lambda_0 = 1, x_0 = 0$, 因此得到:

$$\int_{-1}^1 f(x)dx \approx G_1 = 2f(0) \quad (1.126)$$

两点 Gauss 公式

对于两点 Gauss 公式 ($n=1$, 具有三次代数精度)

$$\int_{-1}^1 f(x)dx \approx 2 \sum_{i=0}^1 \lambda_i f(x_i) = 2(\lambda_0 f(x_0) + \lambda_1 f(x_1)) \quad (1.127)$$

因此当 $f(x) = 1, x, x^2, x^3$ 时上式精确成立, 有:

$$\begin{cases} \lambda_0 + \lambda_1 = 1 \\ \lambda_0 x_0 + \lambda_1 x_1 = 0 \\ \lambda_0 x_0^2 + \lambda_1 x_1^2 = \frac{1}{3} \\ \lambda_0 x_0^3 + \lambda_1 x_1^3 = 0 \end{cases} \quad (1.128)$$

解得: $\lambda_0 = \lambda_1 = \frac{1}{2}, x_1 = -x_0 = \frac{1}{\sqrt{3}}$, 因此得到:

$$\int_{-1}^1 f(x)dx \approx G_2 = f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right) \quad (1.129)$$

三点 Gauss 公式

对于三点 Gauss 公式 ($n=2$, 具有五次代数精度)

$$\int_{-1}^1 f(x)dx \approx 2 \sum_{i=0}^2 \lambda_i f(x_i) = 2(\lambda_0 f(x_0) + \lambda_1 f(x_1) + \lambda_2 f(x_2)) \quad (1.130)$$

因此当 $f(x) = 1, x, x^2, x^3, x^4, x^5$ 时上式精确成立，有：

$$\begin{cases} \lambda_0 + \lambda_1 + \lambda_2 = 1 \\ \lambda_0 x_0 + \lambda_1 x_1 + \lambda_2 x_2 = 0 \\ \lambda_0 x_0^2 + \lambda_1 x_1^2 + \lambda_2 x_2^2 = \frac{1}{3} \\ \lambda_0 x_0^3 + \lambda_1 x_1^3 + \lambda_2 x_2^3 = 0 \\ \lambda_0 x_0^4 + \lambda_1 x_1^4 + \lambda_2 x_2^4 = \frac{1}{5} \\ \lambda_0 x_0^5 + \lambda_1 x_1^5 + \lambda_2 x_2^5 = 0 \end{cases} \quad (1.131)$$

解得：

$$\begin{cases} \lambda_0 = \frac{5}{18} \\ \lambda_1 = \frac{4}{9} \\ \lambda_2 = \frac{5}{18} \\ x_0 = -\sqrt{\frac{3}{5}} \\ x_1 = 0 \\ x_2 = \sqrt{\frac{3}{5}} \end{cases} \quad (1.132)$$

因此得到：

$$\int_{-1}^1 f(x)dx \approx G_3 = \frac{5}{9}f(-\sqrt{\frac{3}{5}}) + \frac{8}{9}f(0) + \frac{5}{9}f(\sqrt{\frac{3}{5}}) \quad (1.133)$$

一般高斯求积公式

取积分区间为 $[-1, 1]$ ，考察如下形式的求积公式：

$$\int_{-1}^1 f(x)dx \approx 2 \sum_{i=0}^n \lambda_i f(x_i) \quad (1.134)$$

使其成为 Gauss 型的，具有 $2n+1$ 次代数精度。即满足：

$$\begin{cases} \int_{-1}^1 1dx = 2 \sum_{i=0}^n \lambda_i \cdot 1 \\ \int_{-1}^1 xdx = 2 \sum_{i=0}^n \lambda_i \cdot x_i \\ \int_{-1}^1 x^2dx = 2 \sum_{i=0}^n \lambda_i \cdot x_i^2 \\ \dots \\ \int_{-1}^1 x^{2n+1}dx = 2 \sum_{i=0}^n \lambda_i \cdot x_i^{2n+1} \end{cases} \quad (1.135)$$

对上述积分计算可得如下的 $2n+2$ 个方程组：

$$\begin{cases} \sum_{i=0}^n \lambda_i = 1 \\ \sum_{i=0}^n \lambda_i \cdot x_i = 0 \\ \sum_{i=0}^n \lambda_i \cdot x_i^2 = \frac{1}{3} \\ \dots \\ \sum_{i=0}^n \lambda_i \cdot x_i^{2n+1} = 0 \end{cases} \quad (1.136)$$

其中，

$$\int_{-1}^1 x^k dx = 0, \quad k = 1, 3, \dots, 2n+1 \quad (1.137)$$

因此，可解出 λ_i 和 x_i 共 $2n+2$ 个未知数。

一般积分区间的高斯公式

一般情况下，对于积分区间 $[a, b]$ ，则可由

$$\int_a^b f(x) dx = \frac{b-a}{2} \int_{-1}^1 f\left(\frac{b+a}{2} + \frac{b-a}{2}t\right) dt \quad (1.138)$$

得：

$$\int_{-1}^1 f(x) dx \approx G_1 = 2f(0) \quad (1.139)$$

$$\int_a^b f(x) dx \approx G_1 = (b-a)f\left(\frac{a+b}{2}\right) \quad (1.140)$$

$$\int_{-1}^1 f(x) dx \approx G_2 = f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right) \quad (1.141)$$

$$\int_a^b f(x) dx \approx G_2 = \frac{b-a}{2} \left(f\left(\frac{a+b}{2} - \frac{b-a}{2\sqrt{3}}\right) + f\left(\frac{a+b}{2} + \frac{b-a}{2\sqrt{3}}\right) \right) \quad (1.142)$$

$$\int_{-1}^1 f(x) dx \approx G_3 = \frac{5}{9}f\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{9}f(0) + \frac{5}{9}f\left(\sqrt{\frac{3}{5}}\right) \quad (1.143)$$

$$\int_a^b f(x) dx \approx G_3 = \frac{b-a}{2} \left(\frac{5}{9}f\left(\frac{a+b}{2} - \sqrt{\frac{3}{5}}\frac{b-a}{2}\right) \right. \quad (1.144)$$

$$\left. + \frac{8}{9}f\left(\frac{a+b}{2}\right) + \frac{5}{9}f\left(\frac{a+b}{2} + \sqrt{\frac{3}{5}}\frac{b-a}{2}\right) \right) \quad (1.145)$$

例 8. 试构造如下形式的 Gauss 型求积公式：

$$\int_0^1 \sqrt{x} f(x) dx \approx A_0 f(x_0) + A_1 f(x_1) \quad (1.146)$$

解：此时 $n = 1$ ，另它对于 $f(x) = 1, x, x^2, x^3$ 精确成立，得到如下方程组：

$$\begin{cases} A_0 + A_1 = \frac{2}{3} \\ A_0 x_0 + A_1 x_1 = \frac{2}{5} \\ A_0 x_0^2 + A_1 x_1^2 = \frac{2}{7} \\ A_0 x_0^3 + A_1 x_1^3 = \frac{2}{9} \end{cases} \quad (1.147)$$

求得：

$$x_0 = 0.821163, x_1 = 0.289949, \quad (1.148)$$

$$A_0 = 0.389111, A_1 = 0.277556 \quad (1.149)$$

因此，求得的高斯积分公式为：

$$\int_0^1 \sqrt{x} f(x) dx \approx 0.389111 \cdot f(0.821163) + 0.277556 \cdot f(0.289949) \quad (1.150)$$

1.6 本章小结

- 机械求积；
- Newton-Cotes 公式；
- 复化求积公式；
- 龙贝格算法；
- Gauss 公式；

本章习题

1. 证明 Simpson 公式具有 3 次代数精度？
2. 用 Newton-Cotes 公式计算定积分 $\int_1^2 \frac{1}{x} dx$ ，并指出具有几阶代数精度？

参考文献

- [1] Author. Title. <http://www.baidu.com>, 2019.