内容简介

利用复化梯形积分公式和复化三点 Gauss 积分公式编写计算积分的通用程序，并计算下列积分：

- $I_1(f) = \int_0^1 e^{-x^2} dx$
- $I_2(f) = \int_0^4 \frac{1}{1+x^2} dx$
- $I_3(f) = \int_0^{2\pi} \frac{1}{2+\cos(x)} dx$

取结点 x_i，$i = 0, \cdots, N$，N 为 2^k，$k = 1, \cdots, 7$ 给出误差表格，其中收敛阶为

$$\frac{\ln(Error_{old}/Error_{now})}{\ln(N_{now}/N_{old})}$$

工作环境

程序所用语言：**python**
软件：**JupyterLab, Mathematica**
使用的包：**mpmath**

输出结果

```
Integrate[E^(- x * x), {x, 0.000000, 1.000000}] = 0.7468241328124270
Composite Trapezoid method:
I(k = 1) = 0.7313702518285630   Error = 1.545388098386e-02   Order = 0.0000
I(k = 2) = 0.7429840978003812   Error = 3.840035012046e-03   Order = 2.0088
I(k = 3) = 0.7458656148456952   Error = 9.585179667318e-04   Order = 2.0022
I(k = 4) = 0.7465845967882215   Error = 2.395360242055e-04   Order = 2.0006
I(k = 5) = 0.7467642546522942   Error = 5.987816013281e-05   Order = 2.0001
I(k = 6) = 0.7468091636378279   Error = 1.496917459909e-05   Order = 2.0000
I(k = 7) = 0.7468203905416179   Error = 3.742270809142e-06   Order = 2.0000

Composite Gauss method:
I(k = 1) = 0.7468240967018682   Error = 3.611055884508e-08   Order = 0.0000
I(k = 2) = 0.7468241324102746   Error = 4.021524498050e-10   Order = 6.4885
I(k = 3) = 0.7468241328066848   Error = 5.742270248690e-12   Order = 6.1300
I(k = 4) = 0.7468241328123394   Error = 8.768565026993e-14   Order = 6.0331
I(k = 5) = 0.7468241328124257   Error = 1.362201955843e-15   Order = 6.0083
I(k = 6) = 0.7468241328124270   Error = 2.125341463362e-17   Order = 6.0021
I(k = 7) = 0.7468241328124270   Error = 3.318997105304e-19   Order = 6.0008
```

```
Integrate[1 / (1 + x * x), {x, 0.000000, 4.000000}] = 1.3258176636680326
Composite Trapezoid method:
I(k = 1) = 1.4588235294117646   Error = 1.330058657437e-01   Order = 0.0000
I(k = 2) = 1.3294117647058823   Error = 3.594101037850e-03   Order = 5.2097
I(k = 3) = 1.3252534024970781   Error = 5.642611709544e-04   Order = 2.6712
I(k = 4) = 1.3256735817329137   Error = 1.440819351187e-04   Order = 1.9695
I(k = 5) = 1.3257816256818826   Error = 3.603798614981e-05   Order = 1.9993
I(k = 6) = 1.3258086530760484   Error = 9.010591983910e-06   Order = 1.9998
I(k = 7) = 1.3258154109515374   Error = 2.252716495020e-06   Order = 2.0000

Composite Gauss method:
I(k = 1) = 1.3256909037243096   Error = 1.267599437228e-04   Order = 0.0000
I(k = 2) = 1.3256917328820794   Error = 1.259307859530e-04   Order = 0.0095
I(k = 3) = 1.3258174178690789   Error = 2.457989534737e-07   Order = 9.0009
I(k = 4) = 1.3258176636701031   Error = 2.070642775051e-12   Order = 16.8570
I(k = 5) = 1.3258176636680783   Error = 4.592596126546e-14   Order = 5.4946
I(k = 6) = 1.3258176636680332   Error = 7.182505640318e-16   Order = 5.9987
I(k = 7) = 1.3258176636680326   Error = 1.122527455403e-17   Order = 5.9997

Integrate[1 / (2 + Cos[x]), {x, 0.000000, 6.283185}] = 3.6275987284684357
Composite Trapezoid method:
I(k = 1) = 4.1887902047863914   Error = 5.611914763180e-01   Order = 0.0000
I(k = 2) = 3.6651914291880923   Error = 3.759270071966e-02   Order = 3.9000
I(k = 3) = 3.6277915166453565   Error = 1.927881769208e-04   Order = 7.6073
I(k = 4) = 3.6275987335910123   Error = 5.122576778448e-09   Order = 15.1998
I(k = 5) = 3.6275987284684357   Error = 3.616826829289e-18   Order = 30.3995
I(k = 6) = 3.6275987284684357   Error = 1.803043458253e-36   Order = 60.7990
I(k = 7) = 3.6275987284684357   Error = 4.480878338110e-73   Order = 121.5980

Composite Gauss method:
I(k = 1) = 3.6337152835897490   Error = 6.116555121314e-03   Order = 0.0000
I(k = 2) = 3.6268604008950978   Error = 7.383275733380e-04   Order = 3.0504
I(k = 3) = 3.6275944023937576   Error = 4.326074677891e-06   Order = 7.4151
I(k = 4) = 3.6275987283534121   Error = 1.150233639204e-10   Order = 15.1988
I(k = 5) = 3.6275987284684357   Error = 8.121295469872e-20   Order = 30.3995
I(k = 6) = 3.6275987284684357   Error = 4.048589927202e-38   Order = 60.7990
I(k = 7) = 3.6275987284684357   Error = 1.006145404963e-74   Order = 121.5980
```

精度检验

n	Trapezoid Method Error	Order	Gauss Method Error	Order
1	1.5454E-02	–	3.6111E-08	–
2	3.8400E-03	2.0088	4.0215E-10	6.4885
3	9.5852E-04	2.0022	5.7423E-12	6.1300
4	2.3954E-04	2.0006	8.7686E-14	6.0331
5	5.9878E-05	2.0001	1.3622E-15	6.0083
6	1.4969E-05	2.0000	2.1253E-17	6.0021
7	3.7423E-06	2.0000	3.3190E-19	6.0008

表 1: L_∞ 范数意义下 $I_1(f)$ 的精度检验

n	Trapezoid Method Error	Order	Gauss Method Error	Order
1	1.3301E-01	–	1.2676E-04	–
2	3.5941E-03	5.2097	1.2593E-04	0.0095
3	5.6426E-04	2.6712	2.4580E-07	9.0009
4	1.4408E-04	1.9695	2.0706E-12	16.8570
5	3.6038E-05	1.9993	4.5926E-14	5.4946
6	9.0106E-06	1.9998	7.1825E-16	5.9987
7	2.2527E-06	2.0000	1.1225E-17	5.9997

表 2: L_∞ 范数意义下 $I_2(f)$ 的精度检验

n	Trapezoid Method Error	Order	Gauss Method Error	Order
1	5.6119E-01	–	6.1166E-03	–
2	3.7593E-02	3.9000	7.3833E-04	3.0504
3	1.9279E-04	7.6073	4.3261E-06	7.4151
4	5.1226E-09	15.1998	1.1502E-10	15.1988
5	3.6168E-18	30.3995	8.1213E-20	30.3995
6	1.8030E-36	60.7990	4.0486E-38	60.7990
7	4.4809E-73	121.5980	1.0061E-74	121.5980

表 3: L_∞ 范数意义下 $I_3(f)$ 的精度检验

误差分析

一、复化 3 点 Gauss 积分公式的推导及误差

本实验中使用的复化 3 点 Gauss 积分公式为：

$$\int_a^b f(x)dx = \frac{h}{18}\left[5\sum_{i=0}^{2^k-1} f\left(\frac{1-\sqrt{\frac{3}{5}}}{2}h + x_i\right) + 8\sum_{i=0}^{2^k-1} f\left(\frac{1}{2}h + x_i\right) + 5\sum_{i=0}^{2^k-1} f\left(\frac{1+\sqrt{\frac{3}{5}}}{2}h + x_i\right)\right]$$

其中 $h=\dfrac{b-a}{2^k}$，记结点为 $x_0,\cdots,x_{2^k}$。下面对其作简单推导并分析该公式的误差。对子区间 $[x_i,\,x_{i+1}]$ 即 $[a+ih,\,a+(i+1)h]$，作变量替换

$$x=\frac{(t+1)h}{2}+x_i \tag{1}$$

并应用 3 点 Gauss 公式，得

$$\begin{aligned}\int_{x_i}^{x_{i+1}} f(x)dx &= \int_{-1}^{1} f(\frac{t+1}{2}h+x_i)\frac{h}{2}dt \\ &= \frac{h}{18}\left[5f\left(\frac{1-\sqrt{\frac{3}{5}}}{2}h+x_i\right)+8f\left(\frac{1}{2}h+x_i\right)+5f\left(\frac{1+\sqrt{\frac{3}{5}}}{2}h+x_i\right)\right]\end{aligned} \tag{2}$$

记 $g(t)=f(\dfrac{t+1}{2}h+x_i)$，$t_{i0}=-\sqrt{\dfrac{3}{5}}$，$t_{i1}=0$，$t_{i2}=\sqrt{\dfrac{3}{5}}$，根据 Hermite 插值，存在一个次数最多为 5 次的多项式 p，使得

$$p(t_{ij})=g(t_{ij}) \qquad p'(t_{ij})=g'(t_{ij}) \qquad 0\leqslant j\leqslant 2 \tag{3}$$

误差公式为

$$g(t)-p(t)=\frac{1}{6!}g^{(6)}(\xi_t)\omega^2(t) \tag{4}$$

其中 $\omega(x)=(t-t_{i0})(t-t_{i1})(t-t_{i2})$，$\xi_t\in(t_{i0},\,t_{i2})$。对等号两端积分并应用积分中值定理可得

$$\int_{-1}^{1}g(t)dt-\int_{-1}^{1}p(t)dt=\frac{g^{(6)}(\xi)}{6!}\int_{-1}^{1}\omega^2(t)dt=\frac{8}{175}\frac{g^{(6)}(\xi_i)}{6!} \tag{5}$$

因 Gauss 公式对 p 是准确成立的，有

$$\int_{-1}^{1}p(t)dt=A_0p(t_{i0})+A_1p(t_{i1})+A_2p(t_{i2})=A_0f(t_{i0})+A_1f(t_{i1})+A_2f(t_{i2}) \tag{6}$$

那么

$$\int_{-1}^{1}g(t)dt=\sum_{j=0}^{2}A_jg(t_{ij})+\frac{8}{175}\frac{g^{(6)}(\xi)}{6!} \tag{7}$$

作积分换元 $t=2\dfrac{x-x_i}{h}-1$，于是又有

$$\begin{aligned}\int_{x_i}^{x_{i+1}}f(x)dx &= \frac{h}{2}\int_{-1}^{1}g(t)dt \\ &= \frac{h}{2}\sum_{j=0}^{2}A_jg(t_{ij})+\frac{h}{2}\frac{8}{175}\frac{h^6}{2^6}\frac{f^{(6)}(\zeta_i)}{6!} \\ &= \frac{h}{2}\sum_{j=0}^{2}A_jg(t_{ij})+\frac{h^7}{175}\frac{f^{(6)}(\zeta_i)}{6!\,2^4}\end{aligned} \tag{8}$$

对 (2) 式与 (8) 式的等号两端求和，于是分别得到复化 3 点 Gauss 公式 (9) 和其误差项估计

式 (10)：

$$\begin{aligned}\int_a^b f(x)dx &= \sum_{i=0}^{2^k-1}\int_{x_i}^{x_{i+1}} f(x)dx \\ &= \frac{h}{18}\sum_{i=0}^{2^k-1}\left[5f\left(\frac{1-\sqrt{\frac{3}{5}}}{2}h+x_i\right)+8f\left(\frac{1}{2}h+x_i\right)+5f\left(\frac{1+\sqrt{\frac{3}{5}}}{2}h+x_i\right)\right] \\ &= \frac{h}{18}\left[5\sum_{i=0}^{2^k-1}f\left(\frac{1-\sqrt{\frac{3}{5}}}{2}h+x_i\right)+8\sum_{i=0}^{2^k-1}f\left(\frac{1}{2}h+x_i\right)+5\sum_{i=0}^{2^k-1}f\left(\frac{1+\sqrt{\frac{3}{5}}}{2}h+x_i\right)\right]\end{aligned} \tag{9}$$

这里尽可能地减少了乘法地运算次数。

结合 $f^{(6)}$ 的连续性，即存在 ζ，使得 $f^{(6)}(\zeta)=\frac{1}{n}\sum_{i=0}^{n-1}f^{(6)}(\zeta_i)$，$n=2^k=\frac{b-a}{h}$

$$\begin{aligned}\int_a^b f(x)dx-\sum_{i=0}^{2^k-1}\frac{h}{2}\sum_{j=0}^{2}A_j g(t_{ij}) &= \sum_{i=0}^{2^k-1}\int_{x_i}^{x_{i+1}}f(x)dx-\sum_{i=0}^{2^k-1}\frac{h}{2}\sum_{j=0}^{2}A_j g(t_{ij}) \\ &= \sum_{i=0}^{2^k-1}\frac{h^7}{175}\frac{f^{(6)}(\zeta_i)}{6!\,2^4} \\ &= \frac{h^7}{175\times 6!\,2^4}\sum_{i=0}^{2^k-1}f^{(6)}(\zeta_i) \\ &= \frac{h^6}{175\times 6!\,2^4}(b-a)f^{(6)}(\zeta)\end{aligned} \tag{10}$$

可知误差项为 $O(h^6)$。实验中对于第一、二个积分最终确实得到了 6 阶代数收敛速度，而第三个积分却不符合此规律。

二、复化梯形法则的推导及误差

这部分推导可参考 Report_06.pdf **误差分析**。复化梯形法则的收敛项是 $O(h^2)$。同上，对于第一、二个积分实际误差达到了预期的 2 阶代数收敛速度。

三、精度调整后的额外测试

由于复化 Gauss 的 6 阶收敛速度非常快，numpy 包的 float64 结构不适应于本次实验，于是采用 mpmath 包，预先设置浮点数精度为 512，以保证实验不会受浮点数精度限制。但即便如此，对于第三个积分，使用复化梯形公式和复化 Gauss 积分公式所得结果的误差均远超出预期的收敛速度。另使用 Mathematica 编程计算，得到了与 python 完全相同的结果。Mathematica 代码见 mma_test.nb

使用本实验中的两个算法对一些简单函数，将精度 mpmath.mp.prec 设置为 1024，迭代的 k 上限增加为 18，进行测试

$\int_0^1 x^8dx$、$\int_0^1 e^x dx$、$\int_0^1 \sin x dx$、$\int_0^{0.9}\arcsin x dx$、低精度 π 下 $\int_0^{2\pi}\frac{1}{2+\cos(x)}dx$

测试结果详见 prec1024.log。通过直接观察，发现了以下事实：

- 使用低精度 π 作积分限。此处使用的是 numpy 包内的 numpy.pi，划分次数较少时两方法的计算结果收敛速度均不稳定，且总是高于预期。随着划分次数的增加，最后复化梯形法则误差符合预计的 2 阶代数精度收敛，复化 Gauss 积分公式误差符合预计的 6 阶代数精度收敛。但原来使用 mpmath.pi 计算时收敛阶过大的现象不再复现。原因尚不明确。

- 猜想对于精度较高的 mpmath.pi，当划分次数足够高时或许能够达到预计的收敛阶。但最终未能得到预计的收敛阶。另一方面，受主机性能限制，程序运行较慢，无法继续增加划分次数和精度来继续测试。

- 对于积分 $\int_0^1 x^8 dx$ 和 $\int_0^1 e^x dx$，在划分次数足够多后，继续增加划分次数会使复化 Gauss 积分公式误差收敛阶降至 2 阶。这里原因同样尚不明确。

  以上问题均未能在此次实验中解决。

总结

本次实验测试了复化梯形公式和复化三点 Gauss 积分公式通过个人编写代码实现，在计算机上的表现。增进了对数值积分方法的理解。

参考资料

[1] David R. Kincaid & E. Ward Cheney. *Numerical Analysis: Mathematics of Scientific of Computing Third Edition*, Brooks/Cole, 2002.