数值计算方法：原理、算法和应用
Numerical Methods: Principles, Algorithms and Applications

授课教师：周铁

北京大学数学科学学院

2021 年 11 月 2 日

1. 数值积分
   - 定积分的数值计算–求积公式
   - 复合求积公式
   - 求积公式的统一形式
   - Gauss 求积公式
   - 自适应 (self-adaptive) 方法
   - 多重积分的数值计算
   - Monte Carlo 数值积分

定积分的数值计算

函数 $f(x)$ 在区间 $[a, b]$ 上连续或 Riemann 可积，如何计算

$$I(f) = \int_a^b f(x)dx$$

是本章讨论的问题.

1. 如果 $f(x)$ 的原函数 $F(x)$ 容易写出，由微积分基本定理
$$\int_a^b f(x)dx = F(b) - F(a).$$
2. $F(x)$ 不是初等函数，或者比较复杂时，上述公式不好使.
3. 很多时候被积函数表达式都不知道.

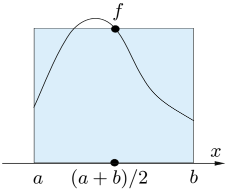
中点求积公式

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

当 $f \in C^2[a,b]$ 时，可以证明

$$\int_a^b f(x)dx - (b-a)f\left(\frac{a+b}{2}\right) = \frac{(b-a)^3}{24}f''(\xi), \quad \xi\text{为}[a,b]\text{中的某一点}.$$

这说明区间长度很 $b-a$ 很小时，中点公式很准！



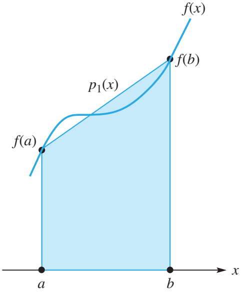
梯形求积公式

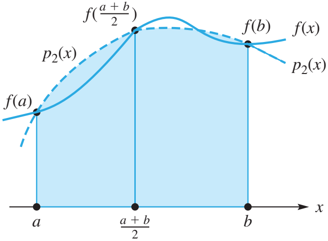
$$\int_a^b f(x)dx \approx \frac{b-a}{2}[f(a)+f(b)].$$

当 $f \in C^2[a,b]$ 时，可以证明

$$\int_a^b f(x)dx - \frac{b-a}{2}[f(a)+f(b)] = -\frac{(b-a)^3}{12}f''(\xi), \quad \xi \text{为}[a,b]\text{中的某一点}.$$

要计算 2 个端点的函数值，误差比中点公式大一倍.





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

当 $f \in C^4[a,b]$ 时，可以证明

$$\int_a^b f(x)dx - \frac{b-a}{6}\left[f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right] = -\frac{(b-a)^5}{2880}f^{(4)}(\xi).$$

要计算 3 个点的函数值，$b-a$ 很小时误差比前两个公式高两个数量级.

Simpson 公式的推导：用 $f(a), f((a+b)/2), f(b)$ 三点做二次插值多项式 $p(x)$，再精确计算积分

$$\int_a^b p(x)dx.$$

用等距节点插值多项式近似被积函数，然后精确计算插值多项式的定积分得到原定积分的近似公式，这样的方法称为 Newton-Cotes 公式.

复合中点公式、复合梯形公式

当 $[a,b]$ 区间不小时，引入 $m+1$ 个等距节点

$$x_k = a + kh, \quad h = (b-a)/m, \quad k = 0, 1, \cdots, m.$$

在每个小区间 $[x_k, x_{k+1}]$ 上用前面介绍的方法.
复合中点公式:

$$\int_a^b f(x)dx = h\sum_{k=1}^{m} f((x_{k-1}+x_k)/2) + \frac{b-a}{24}h^2 f''(\xi)$$

复合梯形公式:

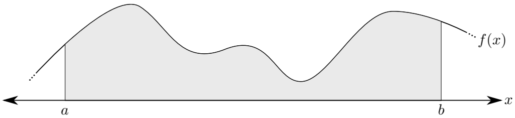
$$\begin{aligned}\int_a^b f(x)dx &= \frac{h}{2}\sum_{k=1}^{m}[f(x_{k-1}) + f(x_k)] - \frac{b-a}{12}h^2 f''(\xi) \\ &= \frac{h}{2}(f(a)+f(b)) + h\sum_{k=1}^{m-1} f(x_k) - \frac{b-a}{12}h^2 f''(\xi)\end{aligned}$$

复合 Simpson 求积公式

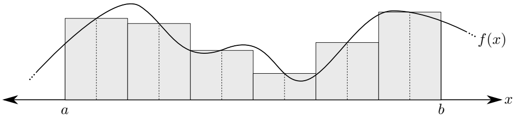
$$
\begin{aligned}
I(f) &= \frac{h}{6}\sum_{k=1}^{m}\left[f(x_{k-1}) + 4f\left(\frac{x_{k-1}+x_k}{2}\right) + f(x_k)\right] - \frac{b-a}{180}h^4 f^{(4)}(\xi) \\
&= \frac{2h}{3}\sum_{k=1}^{m} f\left(\frac{x_{k-1}+x_k}{2}\right) + \frac{h}{6}(f(a)+f(b)) + \frac{h}{3}\sum_{k=1}^{m-1} f(x_k) + \mathcal{O}(h^4)
\end{aligned}
$$

记复合中点公式为 $M(h)$，复合梯形公式为 $T(h)$，复合 Simpson 公式为 $S(h)$. 则有

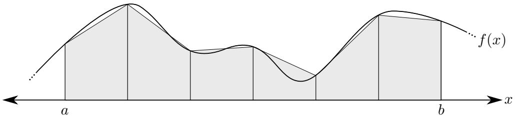
$$
S(h) = \frac{2}{3}M(h) + \frac{1}{3}T(h).
$$



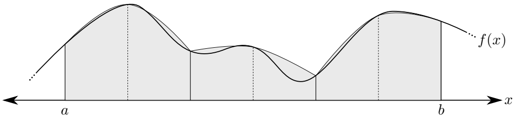
Actual integral



Composite midpoint rule (6 samples)



Composite trapezoidal rule (7 samples)



Composite Simpson's rule (7 samples)

求积公式的统一形式与代数精度

观察前面提到的数值积分方法可以发现，它们都具有

$$\int_a^b f(x)dx \approx \sum_{k=1}^{m} w_k f(x_k) \tag{3.1}$$

的形式. 其中的 x_k 称为节点，w_k 称为权重. 显然一个数值积分方法的计算工作量主要是计算函数值 $f(x_k)$. 在计算一定次数函数值的条件下，节点 x_k，权重 w_k 的选择应该使求积公式的截断误差尽量小.

定义 (求积公式的代数精度)

如果求积公式(3.1)能精确计算所有不高于 $n(n \geq 1)$ 次多项式的定积分，但不能精确计算所有 $n+1$ 次多项式的定积分，就称这个方法具有 n 阶代数精度.

在计算同样多次函数值的前提下，代数精度高的方法更准确.
复合中点公式和复合梯形公式有 1 次代数精度，前者只计算 1 次函数值.
复合 Simpson 公式有 3 次代数精度，计算 3 次函数值.

在计算固定次函数值的条件下，代数精度越高越好.
前面介绍的求积公式都采用等距求积节点 x_k，变化只在函数值前面的权重 w_k. 现在我们让节点也可以变化. 首先，令

$$x = \frac{b-a}{2}y + \frac{a+b}{2}$$

则

$$\int_a^b f(x)dx = \frac{b-a}{2}\int_{-1}^{1} f\left(\frac{b-a}{2}y + \frac{a+b}{2}\right)dy.$$

所以只要讨论 $[-1,1]$ 区间上的求积公式

$$\int_{-1}^{1} f(x)dx \approx \sum_{k=1}^{m} w_k f(x_k). \tag{4.1}$$

如何选取节点 x_k 和权重 w_k 使得其代数精度最高.

如果只有 1 个节点，取 $f(x)=1$, x 分别带入(4.1)式两边，得到方程组

$$\begin{cases} w_1 & = 2 \\ w_1 x_1 & = 0 \end{cases} \Rightarrow \begin{cases} w_1 & = 2 \\ x_1 & = 0 \end{cases}$$

由此得到的求积公式为

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

这就是中点公式.
显然

$$\int_{-1}^{1} x^2 dx \neq 0 = w_1 x_1^2.$$

这表明 1 个节点的求积公式，其代数精度最高为 1.
所以中点公式又称为 1 个节点的 Gauss 求积公式.

如果有 2 个节点，取 $f(x)=1,\ x,\ x^2,\ x^3$ 分别带入(4.1)式两边，得到

$$\begin{cases} w_1+w_2 &= 2 \\ w_1x_1+w_2x_2 &= 0 \\ w_1x_1^2+w_2x_2^2 &= 2/3 \\ w_1x_1^3+w_2x_2^3 &= 0 \end{cases} \quad \Rightarrow \quad \begin{cases} w_1 &= 1 \\ w_2 &= 1 \\ x_1 &= -\frac{1}{\sqrt{3}} \\ x_2 &= \frac{1}{\sqrt{3}} \end{cases}$$

不难验证

$$\frac{2}{5}=\int_{-1}^{1} x^4 dx \neq \left(-\frac{1}{\sqrt{3}}\right)^4+\left(\frac{1}{\sqrt{3}}\right)^4=\frac{2}{9}$$

所以 2 点 Gauss 求积公式为

$$\int_{-1}^{1} f(x)dx \approx f\left(-1/\sqrt{3}\right)+f\left(1/\sqrt{3}\right)$$

其代数精度是 3.

利用上述方法，还可以推导出 3 点 Gauss 求积公式 (需要解一个 6×6 非线性方程组)：

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

其代数精度是 5.

定理 (Gauss 求积公式)

m 个节点的求积公式(4.1)中的节点 $x_1,\cdots,x_m$ 如果取为 m 次正交多项式的零点，权重 $w_1,\cdots,w_m$ 取为相应的 Lagrange 插值基函数的积分，这样得到的求积公式代数精度最高，为 $2m-1$.

1. Gauss 求积公式有很多种–我们只介绍了 Gauss-Legendre 公式
2. 复合 Gauss 求积公式–节点处函数值的计算

定义： $\mathbb{P}[x] \triangleq \{$所有的实系数多项式$p(x)\}$

- $\mathbb{P}[x]$ 是实数域上的一个无穷维线性空间，$\{1, x, x^2, \cdots\}$ 是一组基.
- 设 $f(x), g(x) \in \mathbb{P}[x]$，$f, g$ 的内积 (inner product) 定义为：

$$(f, g) \triangleq \int_{-1}^{1} f(x)g(x)dx.$$

- 如果 $(f, g) = 0$，我们就称 $f(x)$ 与 $g(x)$ 是正交的 (orthogonal).
- 如果 $\mathbb{P}[x]$ 的一个不含零元素子集，其中任意两个元素都正交，称这个集合为正交多项式系.

Legendre 多项式系：

$$L_n(x) = \frac{1}{2^n n!} \frac{\mathrm{d}^n}{\mathrm{d}x^n}(x^2 - 1)^n, \quad n = 0, 1, 2, \cdots,$$

n	$L_n(x)$
0	1
1	x
2	$\frac{1}{2}\left(3x^2-1\right)$
3	$\frac{1}{2}\left(5x^3-3x\right)$
4	$\frac{1}{8}\left(35x^4-30x^2+3\right)$
5	$\frac{1}{8}\left(63x^5-70x^3+15x\right)$
6	$\frac{1}{16}\left(231x^6-315x^4+105x^2-5\right)$
7	$\frac{1}{16}\left(429x^7-693x^5+315x^3-35x\right)$
8	$\frac{1}{128}\left(6435x^8-12012x^6+6930x^4-1260x^2+35\right)$
9	$\frac{1}{128}\left(12155x^9-25740x^7+18018x^5-4620x^3+315x\right)$
10	$\frac{1}{256}\left(46189x^{10}-109395x^8+90090x^6-30030x^4+3465x^2-63\right)$

- 递推公式：$(n+1)L_{n+1}(x)=(2n+1)xL_n(x)-nL_{n-1}(x)$.
- 正交多项式的所有根都是单重实根.

带权 (weight) 积分：

$$I(f) \triangleq \int_a^b \omega(x) f(x) dx$$

其中 $\omega(x) > 0,\ x \in (a,b)$ 是权函数.
带权内积：

$$(f,g) \triangleq \int_a^b \omega(x) f(x) g(x) dx$$

$\omega(x)$	区间 $[a,b]$	正交多项式名称
$\frac{1}{\sqrt{1-x^2}}$	$[-1,1]$	Chebyshev 多项式
e^{-x}	$[0,+\infty)$	Laguerre 多项式
e^{-x^2}	$(-\infty,+\infty)$	Hermite 多项式
1	[-1,1]	Legendre 多项式
$1-x^2$	[-1,1]	Lobatto 多项式

自适应 (self-adaptive) 方法

采用复合梯形公式计算 $f(x)$ 在 $[a,b]$ 区间上的定积分. 需要一些等距节点 $x_k = a + kh, h = (b-a)/m, k = 0, 1, \cdots, m$. 节点数 m 的选取应该满足

$$|I(f,a,b) - T(h,a,b)| = \frac{(b-a)h^2}{12}|f''(\xi)| \le \epsilon$$

其中 ϵ 为要达到的误差精度 (tolerance). 由于 ξ 未知，只能取

$$\frac{(b-a)h^2}{12} \max_{x\in[a,b]} |f''(x)| \le \epsilon.$$

这往往得到一个非常小的 h(非常大的 m)，要计算很多次函数值. 由于定积分具有可加性，可以把整个区间上的定积分分成若干子区间的定积分之和. 如果在 $|f''(x)|$ 大的子区间取比较多的节点，在 $|f''(x)|$ 小的子区间取比较少的节点，就可以提高计算效率，这就是设计自适应方法的原因.

需要设计一个逐渐增加节点的策略.

(i) 先从 2 个节点开始，取 $h=b-a$，计算 $T(h,a,b)$，如果不等式

$$|I(f,a,b)-T(h,a,b)|\le\epsilon$$

成立，计算就停止，$T(h,a,b)$ 就是积分的值；如果这个不等式不成立，就执行下一步.

(ii) 增加一个节点，把 $[a,b]$ 二分为

$$[a_{11},b_{11}]\cup[a_{12},b_{12}],\ a_{11}=a,b_{11}=(a+b)/2=a_{12},b_{12}=b,$$

计算 $T(h/2,a_{11},b_{11})$ 和 $T(h/2,a_{12},b_{12})$，如果不等式

$$|I(f,a_{11},b_{11})-T(h/2,a_{11},b_{11})|\le\epsilon/2, \tag{5.1}$$
$$|I(f,a_{12},b_{12})-T(h/2,a_{12},b_{12})|\le\epsilon/2, \tag{5.2}$$

都成立，计算就停止；否则就执行下一步.

(iii) 如果不等式(5.1)不成立，就对区间 $[a_{11},b_{11}]$ 执行第(ii)步，误差取为 $\epsilon/4$. 如果不等式(5.2)不成立，就对区间 $[a_{12},b_{12}]$ 执行第(ii)步，误差取为 $\epsilon/4$.

上述方法中有个问题：$I(f,\cdot,\cdot)$ 都未知!

利用截断误差公式

$$I(f,u,v) - T(h,u,v) = -\frac{v-u}{12}h^2 f''(\xi_1)$$

$$I(f,u,v) - T(h/2,u,v) = -\frac{v-u}{12}\left(\frac{h}{2}\right)^2 f''(\xi_2)$$

这两个等式相减，并假设$f''(\xi_1) = f''(\xi_2)$，可得

$$\frac{1}{3}(T(h/2,u,v) - T(h,u,v)) = -\frac{v-u}{12}\left(\frac{h}{2}\right)^2 f''(\xi)$$

于是有

$$I(f,u,v) - T(h/2,u,v) = \frac{1}{3}(T(h/2,u,v) - T(h,u,v))$$

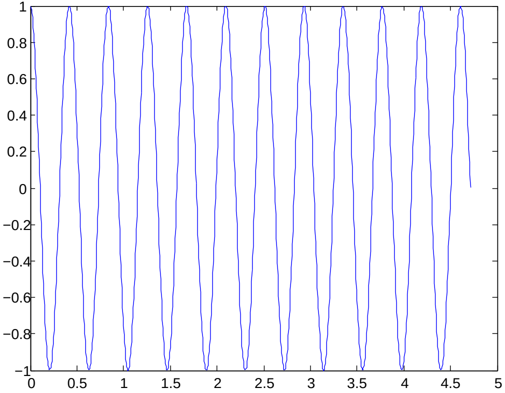
于是可以用不等式

$$|T(h/2,u,v) - T(h,u,v)| \le \epsilon \ (3\epsilon?)$$

来代替上述算法中的不等式

$$|I(f,u,v) - T(h/2,u,v)| \le \epsilon$$

$$I = \int_0^{3\pi/2} \cos(15x)dx$$



trapz 是 Matlab 的复合梯形方法数值积分命令，integral 是 Matlab 的自适应数值积分命令，下面是它们的比较.

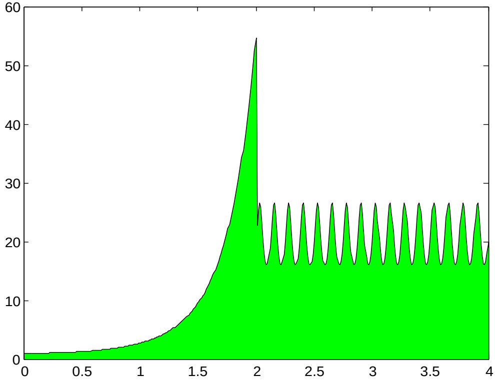
```
>> syms x;
>> A = int(cos(15*x),0,3*pi/2)

A = 1/15

>> f=@(x)cos(15*x)
>> tic,I2 = integral(f,0,3*pi/2,'RelTol',1e-20),toc
I2 = 0.066666666666667
Elapsed time is 0.002035 seconds.

>> h=1e-7;
>> x = [0:h:3*pi/2];
>> y = cos(15*x);
>> tic, I=trapz(x,y),toc
I = 0.066666666666602
Elapsed time is 3.318782 seconds.
```

$$I = \int_0^4 f(x)dx, \quad f(x) = \begin{cases} e^{x^2}, & 0 \leq x \leq 2 \\ \frac{80}{4-\sin(16\pi x)}, & 2 < x \leq 4 \end{cases}$$



```
>> x = [0:0.0001:4];
>> f=@(x)exp(x.^2).*(x<=2)+80./(4-sin(16*pi*x)).*(x>2);
>> y = f(x);
>> plot(x,y)

>> f=@(x)exp(x.^2).*(x<=2)+80./(4-sin(16*pi*x)).*(x>2);

>> I = integral(f,0,4) %(comments: RelTol = 1e-6)
 I = 57.764450125048512

>> syms x;
>> I1 =int(exp(x^2),0,2) + int(80/(4-sin(16*pi*x)),2,4)
 I1 = (32*15^(1/2))/3 + (pi^(1/2)*erfi(2))/2

>> vpa(I1)
 ans = 57.764450125053010333315235385 18

>> I2 = integral(f,0,4,'RelTol',1e-20)  %(comments: RelTol=1e-
 I2 = 57.764450125053017
```

二重积分的数值计算

考虑二重积分

$$I=\iint_{\Omega} f(x,y)dxdy, \quad \Omega=\{a\le x\le b, \varphi_1(x)\le y\le \varphi_2(x)\}$$

其中 $f(x,y)\in C(\Omega)$. 可以写成累次积分

$$I=\iint_{\Omega} f(x,y)dxdy=\int_a^b dx\int_{\varphi_1(x)}^{\varphi_2(x)} f(x,y)dy.$$

令

$$F(x)=\int_{\varphi_1(x)}^{\varphi_2(x)} f(x,y)dy,$$

利用一元求积公式

$$I=\int_a^b F(x)dx\approx\sum_{i=1}^{n} w_i^{(1)}F(x_i).$$

为了计算 $F(x_i)$，在区间 $[\varphi_1(x_i),\varphi_2(x_i)]$ 上取 m 个节点，利用一元求积公式得

$$F(x_i)=\int_{\varphi_1(x_i)}^{\varphi_2(x_i)} f(x_i,y)dy \approx \sum_{j=1}^{m} w_j^{(2)} f(x_i,y_j).$$

若记 $w_{ij}=w_i^{(1)}w_j^{(2)}$，则得到二重求积公式

$$I=\iint_{\Omega} f(x,y)dxdy \approx \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij} f(x_i,y_j).$$

以 $\Omega=\{a\le x\le b, c\le y\le d\}$ 为例，在 $[a,b]$ 和 $[c,d]$ 分别引入 n 个 x 节点和 m 个 y 节点 (包含端点)，记

$$x_i=a+ih_x,\ i=0,1,\cdots,n-1,\ h_x=\frac{b-a}{n-1},$$
$$y_j=c+jh_y, j=0,1,\cdots,m-1,\ h_y=\frac{d-c}{m-1},$$

并在两个区间都采用复合梯形公式，可得，

$$
\begin{aligned}
I &= \iint_{\Omega} f(x,y)dxdy \\
&\approx \frac{h_x h_y}{4}\Bigg[f(x_1,y_1)+f(x_n,y_1)+f(x_1,y_m)+f(x_n,y_m)+2\sum_{i=2}^{n-1} f(x_i,y_1)+ \\
&\qquad +2\sum_{i=2}^{n-1} f(x_i,y_m)+2\sum_{i=2}^{m-1} f(x_1,y_j)+2\sum_{j=2}^{m-1} f(x_n,y_j)+ \\
&\qquad +4\sum_{i=2}^{n-1}\sum_{j=2}^{m-1} f(x_i,y_j)\Bigg].
\end{aligned}
$$

当 $f(x,y) \in C^2(\Omega)$ 时，截断误差为

$$
R = \mathcal{O}(h_x^2 + h_y^2) = \mathcal{O}(n^{-2} + m^{-2})
$$

以上二重积分的计算方法很容易推广到三重，乃至 d 重积分的计算. 记

$$\Omega = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_d, b_d],$$

在每个方向都采用 n 点复合梯形公式计算 d 重积分

$$I = \int_\Omega f(x_1, x_2, \cdots, x_d) dx_1 dx_2 \cdots dx_d,$$

截断误差为

$$R = \mathcal{O}(n^{-2}), \text{注意 } n \text{ 只是一个方向上的节点个数!}$$

显然这时整个积分区域 Ω 上的总积分节点数为 $N = n^d$ 个，所以

$$R = \mathcal{O}(N^{-2/d}).$$

此式表明，

$$N \sim R^{-d/2}$$

要算到比较小的误差，比如 $R = 10^{-6}$, $d = 4$，则需要 $N \sim 10^{12}$!
高维数值积分一般采用 Monte Carlo 方法.

Monte Carlo 数值积分

先看一元定积分：

$$I(f) = \int_a^b f(x)dx, \quad \text{其中} \ -\infty < a < b < +\infty$$

前面已经介绍了经典的数值求积方法. 那里的所有求积公式都表示为被积函数在某些点的函数值的加权平均. 例如，复合梯形求积公式为

$$I(f) \approx h\left(\frac{1}{2}f(x_0) + \sum_{i=1}^{n-1} f(x_i) + \frac{1}{2}f(x_n)\right),$$

其中 $h = (b-a)/n$, $x_i = a + ih$ $(i = 0, 1, 2, \dots, n)$.

Monte Carlo 数值求积方法的基本思想是：把被积函数的自变量 x 看成 $[a, b]$ 区间上的均匀分布随机变量 X，即

$$X \sim U[a,b], \qquad p(x) = \begin{cases} 1/(b-a), & x \in [a,b] \\ 0, & x \in (-\infty, a) \cup (b, +\infty) \end{cases}$$

于是 $f(X)$ 也是随机变量，而且**上述定积分就是** $f(X)$ **的数学期望**

$$I(f)/(b-a)=\int_{-\infty}^{\infty} f(x)p(x)dx=E[f(X)]$$

所以积分的计算就是期望的计算. 为了近似计算 $E(f(X))$，设

$$X_1, X_2, \ldots, X_n$$

为相互独立且都服从 $U[a,b]$ 的随机变量，则

$$f(X_1), f(X_2), \ldots, f(X_n).$$

也是相互独立且同分布的随机变量. 记

$$I_n(f) \triangleq \frac{b-a}{n}\sum_{i=1}^{n} f(X_i),$$

它也是一个随机变量，并且满足

$$E[I_n(f)]=\frac{b-a}{n}\sum_{i=1}^{n} E[f(X_i)]=\frac{1}{n}\sum_{i=1}^{n}\int_a^b f(x)dx=I(f).$$

根据大数定律，

$$\lim_{n\to+\infty}\frac{b-a}{n}\sum_{i=1}^{n}f(X_i)=I(f)$$

成立的概率为 1.

下面估计误差 $e_n=I_n(f)-I(f)$，由于 e_n 也是随机变量，且 $E(e_n)=0$.
下面计算 e_n 平方的期望，也就是 e_n 的方差.
为了简化记号，以 $a=0$, $b=1$ 为例，注意 $f(X_i)$, $i=1,2,\ldots,n$ 为 i.i.d. 的随机变量，所以有

$$\begin{aligned}E(|e_n|^2)&=E\left[(I_n(f)-I(f))^2\right]\\&=E\left[\frac{1}{n^2}\left(\sum_{i=1}^{n}(f(X_i)-I(f))\right)^2\right]\\&=\frac{1}{n^2}E\left[\sum_{i=1}^{n}(f(X_i)-I(f))\sum_{j=1}^{n}(f(X_j)-I(f))\right]\end{aligned}$$

由于 $f(X_i), f(X_j)\ (i \neq j)$ 相互独立，所以只要 $i \neq j$ 就有

$$E\left[(f(X_i) - I(f))(f(X_j) - I(f))\right] = E(f(X_i) - I(f))\, E(f(X_j) - I(f)) = 0.$$

所以

$$\begin{aligned} E(|e_n|^2) &= \frac{1}{n^2} \sum_{i=1}^{n} E\left[(f(X_i) - I(f))^2\right] \\ &= \frac{1}{n^2} \sum_{i=1}^{n} E\left[(f(X) - I(f))^2\right] \\ &= \frac{1}{n} \mathrm{Var}\,(f(X)), \end{aligned}$$

其中 $\mathrm{Var}\,(f(X))$ 为 $f(X)$ 的方差. 利用 Cauchy-Schwartz 不等式，可得

$$E(|e_n|) \leq \sqrt{E|e_n|^2} = \sqrt{\frac{\mathrm{Var}\,(f(X))}{n}} = \mathcal{O}(n^{-1/2}).$$

这样，我们就得到了 Monte Carlo 方法的**半阶收敛性**.

设 d 为正整数,

$$D = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_d, b_d] \subset \mathbb{R}^d,$$

$f(x)$ 为 D 上可积函数,

$$I(f) = \int_{a_1}^{b_1} \int_{a_2}^{b_2} \cdots \int_{a_d}^{b_d} f(x_1, x_2, \ldots, x_d) dx_1 dx_2 \cdots dx_d = \int_D f(x) dx.$$

引进 d 个 i.i.d. 的随机变量 $X_j \sim U[a_j, b_j], j = 1, 2, \ldots, d$，则

$$I(f)/|D| = \int_{\mathbb{R}^d} f(x) p_1(x_1) \cdots p_d(x_d) dx = E(f(X_1, \ldots, X_d))$$

其中 $p_j(x_j)$ 为 X_j 的概率密度函数 (pdf)

$$p_j(x_j) = \begin{cases} \frac{1}{b_j - a_j}, & x_j \in [a_j, b_j] \\ 0, & x_j \in \mathbb{R} \setminus [a_j, b_j] \end{cases}$$

对随机向量 $X=(X_1,X_2,\ldots,X_d)$ 进行 n 次独立抽样，得到

$$X^{(i)}=(X_1^{(i)},X_2^{(i)},\ldots,X_d^{(i)}),\ i=1,2,\ldots,n.$$

根据大数定律，当采样次数 n 充分大时，

$$I_n(f)\triangleq\frac{|D|}{n}\sum_{i=1}^{n}f(X^{(i)}),\quad \text{其中 } |D|=\prod_{j=1}^{d}(b_j-a_j).$$

就是 d 重积分 $I(f)$ 的近似.

与 1 维情形类似地可以证明误差估计：

$$E(|I_n(f)-I(f)|)=\mathcal{O}(n^{-1/2}).$$

注意这里的收敛速度不依赖于问题的维数，这正是 Monte Carlo 方法在计算高维积分时的优越性.

上述 Monte Carlo 方法可以推广.
设要计算的积分为

$$I(f) = \int_a^b f(x)p(x)\,dx$$

其中 $p(x) \geq 0$ 且 $\int_a^b p(x)\,dx = 1$.

把 $p(x)$ 看成某个随机变量 X 的概率密度函数 (pdf)，则 $f(X)$ 的数学期望就是要计算的积分

$$E[f(X)] = \int_a^b f(x)p(x)\,dx$$

例如

$$a = -\infty,\ b = +\infty, \quad p(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

要计算无穷积分

$$I(f) = \int_{-\infty}^{+\infty} f(x)p(x)\,dx$$

现在 X 服从标准正态分布，$X \sim N(0,1)$，对其做 n 次独立抽样

$$X_1,\, X_2,\, \ldots,\, X_n$$

则独立同分布随机变量 $f(X_i),\, i=1,2,\ldots,n$ 的算数平均值

$$I_n(f) = \frac{1}{n}\sum_{i=1}^{n} f(X_i)$$

也满足

$$E[I_n(f)] = \frac{1}{n}\sum_{i=1}^{n} E[f(X_i)] = I(f)$$

以及

$$E[|I_n(f) - I(f)|] = \mathcal{O}(n^{-1/2}).$$

- Monte Carlo 方法的 $\mathcal{O}(n^{-1/2})$ 收敛速度看起来很慢.
- 维数比较高的时候，我们做一个比较.
  如果用复合梯形公式计算一个

$$\Omega = [0,1] \times [0,1] \times \cdots \times [0,1] \subset \mathbb{R}^d$$

  区域内的 d 维积分，在每个坐标轴上取 k 个等距节点，则整个积分区域 Ω 中总共有 $n = k^d$ 个节点，于是 $h = \mathcal{O}(n^{-1/d})$，每个坐标方向的计算精度为 $\mathcal{O}(1/k^2)$，总计算精度为 $\mathcal{O}(n^{-2/d})$.
  如果用 Monte Carlo 方法计算，精度为 $\mathcal{O}(n^{-1/2})$. 当维数 $d > 4$ 时，Monte Carlo 方法就比复合梯形公式优越.
- 在维数较低 (小于 3) 的时候，Monte Carlo 方法没有实际意义.

需要强调的是，Monte Carlo 方法往往是在没有更好算法时才采用的策略，实际问题应该尽可能采取适合其自身的高效率、高精度的确定型算法.

参考书

1. 薛定宇, 陈阳泉
   高等应用数学问题的 Matlab 求解 (第三版)
   清华大学出版社, 2013.
2. Alfio Quarteroni, Riccardo Sacco, Fausto Saleri,
   Numerical Mathematics(2nd Edition),
   Springer, 2007.
3. Walter Gander, Martin J. Gander, Felix Kwok,
   Scientific Computing: an introduction using Maple and MATLAB,
   Springer 2007.