数值分析上机习题报告 (7)

张宏毅 1500017736

March 27, 2017

1 Problem

已知积分

$$\int_0^1 \frac{4}{1+x^2} = \pi$$

成立，我们可以通过对上面给定被积函数求数值积分来计算 π 的近似值。

（1）分别使用复合中点公式、复合梯形公式和复合 Simpson 公式计算 π 的近似值。选择不同的 h，对每种求积公式，试将误差刻画为 h 的函数，并比较各方法的精度。是否存在某个 h 值，当小于这个值之后再继续减小 h 的值，计算不再有所改进？为什么？

（2）实现 Romberg 求积方法，并重复上面的计算。

（3）使用自适应求积方法重复上面的计算。

2 Solution

2.1 Subproblem A

复合中点公式、复合梯形公式和复合 Simpson 公式就是在积分区间上，分段应用中点公式、梯形公式和 Simpson 公式。若要近似求函数 $f(x)$ 在区间 $[a, b]$ 上的积分，记步长 $h = (b-a)/n$，子区间端点 $x_i = a + ih$ $(0 \le i \le n)$，则有积分近似公式

$$\int_a^b f(x)\mathrm{d}x \approx h \sum_{i=0}^{n} f(x_{i+\frac{1}{2}}) \triangleq M(h), \tag{1}$$

$$\int_a^b f(x)\mathrm{d}x \approx \frac{h}{2} \sum_{i=0}^{n-1} [f(x_i) + f(x_{i+1})] \triangleq T(h), \tag{2}$$

$$\int_a^b f(x)\mathrm{d}x \approx \frac{h}{6} \sum_{i=0}^{n-1} [f(x_i) + 4f(x_{i+\frac{1}{2}}) + f(x_{i+1})] \triangleq S(h). \tag{3}$$

且有截断误差估计

$$\left| \int_a^b f(x)\mathrm{d}x - M(h) \right| \le \frac{M_2(b-a)}{24} h^2, \tag{4}$$

$$\left|\int_a^b f(x)\mathrm{d}x - T(h)\right| \le \frac{M_2(b-a)}{12}h^2, \tag{5}$$

$$\left|\int_a^b f(x)\mathrm{d}x - S(h)\right| \le \frac{M_4(b-a)}{2880}h^4. \tag{6}$$

利用三个公式，针对不同的步长 h，计算积分 $\int_0^1 \frac{4}{1+x^2}\mathrm{d}x$ 来作为 π 的近似值，结果如表 1 所示。可以看出，随着步长的减小，中点公式和梯形公式的逼近误差都在不断减小。Simpson 公式的收敛速度很快，在 $h=10^{-2}$ 时就达到了 14 位有效数字的良好精度，但当 h 进一步减小时，计算结果并没有得到改进。可以推测，中点公式和梯形公式也存在这样的精度上限，只是因为收敛速度太慢，没有体现在计算结果中。

表 1: 复合求积公式的计算结果

步长	复合中点公式	复合梯形公式	复合 Simpson 公式
10^{-1}	3.1424259850010974	3.1399259889071591	3.1415926529697851
10^{-2}	3.1416009869231227	3.1415759869231308	3.1415926535897922
10^{-3}	3.1415927369231307	3.1415924869231278	3.1415926535897913
10^{-4}	3.1415926544231363	3.1415926519231183	3.1415926535897887
10^{-5}	3.1415926535981669	3.1415926535731322	3.1415926535897722
10^{-6}	3.1415926535899756	3.1415926535897176	3.1415926535898948

下面我们进行误差分析。设积分公式的截断误差为 $e_1 = Ch^m$，舍入误差从三个积分公式的形式可以估计 $e_2 \le h\sum_{i=0}^{n-1}\epsilon = nh\epsilon = (b-a)\epsilon$, 其中 ϵ 为机器精度，因而总误差

$$E(h) = Ch^m + (b-a)\epsilon.$$

当 $Ch^m > (b-a)\epsilon$ 时，总误差主要受 h^m 项影响，会随着 h 的减小而明显减小。但当步长小到满足不等式 $Ch^m < (b-a)\epsilon$ 时，总误差主要受常数 $(b-a)\epsilon$ 的影响，此时 h 的减小不会带来明显的计算改进。但令人欣慰的是，虽然减小 h 并不总能保证精度的提高，但至少这样的计算在数值上是稳定的。

2.2 Subproblem B

记复合梯形公式

$$T_1(h) = \frac{h}{2}\sum_{i=0}^{n-1}[f(x_i) + f(x_{i+1})].$$

根据 Euler-Maclaurin 公式，复合梯形公式的截断误差可表示为

$$\int_a^b f(x)\mathrm{d}x - T_1(h) = c_2h^2 + c_4h^4 + \cdots \tag{7}$$

利用 Richardson 外推技术，将步长减半后有

$$\int_a^b f(x)\mathrm{d}x - T_1\left(\frac{h}{2}\right) = c_2\frac{h^2}{4} + c_4\frac{h^4}{16} + \cdots \tag{8}$$

从 (7)、(8) 两式中消去 h^2 项得

$$\int_a^b f(x)\mathrm{d}x - \frac{4T_1(h/2) - T_1(h)}{3} = O(h^4).$$

就导出了一个截断误差为四阶的近似公式。一般地，令

$$T_{k+1}(h) = \frac{4^k T_k(h/2) - T_k(h)}{4^k - 1}, \quad k = 1, 2, \cdots \tag{9}$$

则有截断误差估计

$$\int_a^b f(x)\mathrm{d}x - T_k(h) = O(h^{2k}). \tag{10}$$

这就是 Romberg 求积方法。若记 $R_{i,j} = T_i(h/2^j)$，则递推公式 (9) 可以写成

$$R_{i+1,j} = \frac{4^i R_{i,j+1} - R_{i,j}}{4^i - 1}, \tag{11}$$

其中边界 $R_{1,j} = T_1(h/2^j)$ 由复合梯形公式直接计算即得，$R_{i,0}$ 为所求积分的 $2i$ 阶近似。

表 2 给出了利用 Romberg 求积方法得到的数值积分，初始步长分别为 1, 0.1 和 0.01。可以看到，随着外推次数的增加，计算的误差也在相应减小，但当阶数高到一定程度时，计算不再改进。对于以更小初始步长的梯形公式为基础的外推而言，由于其初始的精度就相对较好，故其收敛的速度也更快。若与第一问中的复合梯形公式做对比，可以发现，我们只需取步长为 10^{-2} 并进行一次外推，便可与步长为 10^{-6} 的复合梯形公式拥有相同的精度，计算效率大大提升。

表 2: Romberg 求积方法的计算结果

	$h=1$	$h=0.1$	$h=0.01$
$T_1(h)$	3.0000000000000000	3.1399259889071591	3.1415759869231308
$T_2(h)$	3.1333333333333333	3.1415926529697864	3.1415926535897909
$T_3(h)$	3.1421176470588246	3.1415926536207928	3.1415926535897896
$T_4(h)$	3.1415857837618737	3.1415926535897940	3.1415926535897993
$T_5(h)$	3.1415926652777171	3.1415926535897953	3.1415926535897922
$T_6(h)$	3.1415926536382432	3.1415926535897913	3.1415926535897971
$T_7(h)$	3.1415926535897229	3.1415926535897927	3.1415926535897762
$T_8(h)$	3.1415926535897936	3.1415926535898056	3.1415926535897927

该方法的误差分析与第一问类似。对外推得到的 $T_k(h)$，其截断误差为 $O(h^{2k})$，而舍入误差由 (9) 式可以认为基本保持在 ϵ，即机器精度左右，因而总误差 $E(h) = Ch^{2k} + \epsilon$ 会随着 h 的减小先减小，然后保持在一个较稳定的水平上。

2.3 Subproblem C

自适应求积方法的出发点在于，对一个函数用固定步长求数值积分有一些缺陷——在函数的导数变化幅度较大的子区间上，应考虑使用更小的步长作估计，而导数变化不大的区间可以采用更大的步长来估计，这样既能提高前者区间上的精度，又能减少后者区间上不必要的计算。

因此，在一个区间 $[a, b]$ 上求定积分的时候，我们可以通过估计当前的计算误差，来决定是接受当前

的数值结果，还是将当前区间继续二分为更小的区间 $[a, \frac{a+b}{2}]$ 和 $[\frac{a+b}{2}, b]$ 进行计算。于是，整个算法的效率和精度就与人为选取的数值积分公式，以及计算误差的估计这两项密切相关。算法框架如下，其中的误差估计条件会在下面进行说明。

Algorithm : Adaptive Quadrature Method

Input: Target function f, the interval $[a, b]$ and the tolerance ϵ.
Given: A numerical integration formula $I(a, b)$ of n-th order for $\int_a^b f(x)\mathrm{d}x$.
function Integrate($f,\ a,\ b,\ \epsilon$)
  if $|I(a,b) - I(a, \frac{a+b}{2}) - I(\frac{a+b}{2}, b)| < (2^{n-1} - 1)\epsilon$ **then**
    return $I(a, \frac{a+b}{2}) + I(\frac{a+b}{2}, b)$
  else
    return Integrate($f,\ a,\ \frac{a+b}{2},\ \frac{\epsilon}{2}$) + Integrate($f,\ \frac{a+b}{2},\ b,\ \frac{\epsilon}{2}$)

下面推导误差估计的条件。设数值积分公式 $I(a, b)$ 满足截断误差

$$\int_a^b f(x)\mathrm{d}x - I(a,b) = \frac{f^{(n-1)}(\xi)}{N} h^n. \tag{12}$$

那么对两个子区间 $[a, \frac{a+b}{2}]$ 和 $[\frac{a+b}{2}, b]$ 上的积分，有

$$\int_a^b f(x)\mathrm{d}x - I\left(a, \frac{a+b}{2}\right) - I\left(\frac{a+b}{2}, b\right) = \frac{2f^{(n-1)}(\xi')}{N} \left(\frac{h}{2}\right)^n. \tag{13}$$

用 (13) 式减去 (12) 式可以推出

$$I(a,b) - I\left(a, \frac{a+b}{2}\right) - I\left(\frac{a+b}{2}, b\right) \approx \frac{f^{(n-1)}(\xi)}{N} \left(\frac{1}{2^{n-1}} - 1\right) h^n. \tag{14}$$

这里我们假定 $f^{(n-1)}(\xi) \approx f^{(n-1)}(\xi')$。结合 (13)、(14) 两式可得误差估计

$$\left|\int_a^b f(x)\mathrm{d}x - I\left(a, \frac{a+b}{2}\right) - I\left(\frac{a+b}{2}, b\right)\right| \approx \frac{1}{2^{n-1}-1} \left|I(a,b) - I\left(a, \frac{a+b}{2}\right) - I\left(\frac{a+b}{2}, b\right)\right|. \tag{15}$$

因而当

$$\left|I(a,b) - I\left(a, \frac{a+b}{2}\right) - I\left(\frac{a+b}{2}, b\right)\right| < (2^{n-1} - 1)\epsilon$$

时，我们可以认为

$$\left|\int_a^b f(x)\mathrm{d}x - I\left(a, \frac{a+b}{2}\right) - I\left(\frac{a+b}{2}, b\right)\right| < \epsilon.$$

即此时利用 $I(a, \frac{a+b}{2}) + I(\frac{a+b}{2}, b)$ 作为积分 $\int_a^b f(x)\mathrm{d}x$ 的估计，其误差大致能满足容差要求。在实际的计算中，容差前的系数 $2^{n-1} - 1$ 可以取得更小一点，使得算法更加保守，弥补一些 $f^{(n-1)}(\xi) \approx f^{(n-1)}(\xi')$ 这一假设带来的误差。

这里的算法采用了递归的方法，若要节省系统栈的内存空间，可以用一个队列来保存待求积分的区间。每次从队头取出一个子区间进行计算，若结果满足该子区间的容差要求，则将其加入到最终的数值结果当中，否则向队列中再添加它的两个子区间等待计算，如此循环，直到队列为空停止。

表 3 列出了利用自适应积分求题给积分的计算结果。一般地，容差越小，所得结果的精度就越高。

但当容差过小（比如小于机器精度）时，算法中的误差估计条件可能永远都不能被满足，导致自适应算法无限递归下去。因而，为了避免这种情况发生，我在代码中人为设定了一个递归次数上限，使得代码在容差过小时也能正常运行。当然这也就导致了当容差小于一个临界值（大约 10^{-15} 左右）时，得到的计算结果都完全相同的现象。

表 3: 自适应求积方法的计算结果

容差 ϵ	积分数值	容差 ϵ	积分数值
10^{-1}	3.14156862745098042211	10^{-13}	3.14159265358979222782
10^{-4}	3.14159250245870680374	10^{-16}	3.14159265358979311600
10^{-7}	3.14159265660246589391	10^{-19}	3.14159265358979311600
10^{-10}	3.14159265357082162495	10^{-22}	3.14159265358979311600

3 Conclusion

我们在三个小题中分别利用了不同的数值分析手段，不断地提高数值积分与精确积分之间的截断误差，效果非常显著。不同的求积方法改进的方面不同，计算效率也不同，而实际上我们可以将这些方法混合使用，在特定的情形下选取一个最好的求积方法来作为积分的近似。

最后，无论利用什么方法来进行数值积分的求解，尽管截断误差在理论上都可以被无限地减小，但是舍入误差到最终都是不可避免要面对的问题，机器精度很大程度地限制了数值解的精确上限。