数值分析上机实验

数值积分

1　问题描述

本实验的目的是用五点 Gauss-Legendre 求积公式和 Romberg 积分算法分别求下面积分的数值值：

$$I(f)=\int_1^3 f(x)\mathrm{d}x,\quad f(x)=\frac{1}{x^2}\sin\frac{2\pi}{x}, \tag{1}$$

参考值为：

$$I(f)=-0.238732414637843\cdots. \tag{2}$$

2　实验内容

对于 Gauss-Legendre 积分法，即选取权函数 $\rho\equiv 1$ 的正交多项式（Legendre 多项式）作为插值积分格式的节点. 对于 $[-1,1]$ 的积分格式在表格中可以查到，因此，在实际使用时，我们将 $[a,b]$ 上的积分通过积分变换站华为 $[-1,1]$ 上的积分：

$$\int_a^b f(x)\mathrm{d}x=\frac{b-a}{2}\int_{-1}^1 f\left(s\left(\frac{b-a}{2}\right)+\frac{b+a}{2}\right)\mathrm{d}s, \tag{3}$$

查表可以得到在 $[-1,1]$ 上积分的格式如下：

节点：$0,\pm\frac{1}{3}\sqrt{5-2\sqrt{\frac{10}{7}}},\pm\frac{1}{3}\sqrt{5+2\sqrt{\frac{10}{7}}}$

权重：$\frac{128}{225},\frac{332+13\sqrt{70}}{900},\frac{332-13\sqrt{70}}{900}$.

复合的 Gauss-Legendre 积分公式就是将 $[1,3]$ 分成若干个小区间，每个小区间上分别使用以上积分格式，然后将结果相加.

对于 Romberg 积分算法，按照下面几个过程进行：首先求出 $T_0^{(0)}=\frac{h}{2}[f(a)+f(b)]$；然后求梯形值 $T_0(\frac{b-a}{2^k})$，也就是按照递推公式 $T_{2n}=\frac{1}{2}T_n+\frac{h}{2}\sum_{k=0}^{n-1}f(x_{k+1/2})$ 计算每个 $T_0^{(k)}$；随后用 Richardson 外推公式：

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

逐个计算可求出 T 表的第 k 行的每个元素 $T_j^{(k-j)}$，$j=1,2,\cdots$.

最后，依据 $|T_k^{(0)}-T_{k-1}^{(0)}|<\varepsilon$，则终止计算，并 $T_k^{(0)}\approx I$，否则 $k+1\to k$ 继续计算.

3 代码实现

对于 Gauss-Legendre 积分，我们首先定义函数计算一般 $[a,b]$ 区间上按照表中查得的节点、权重的插值积分值，然后将 $[1,3]$ 划分为四个小区间，在每个小区间上分别调用函数计算插值积分，然后将结果相加，具体代码如下：

```
########5-point Gauss-Legendre quadrature########
points = [0, (1/3)*np.sqrt(5-2*np.sqrt(10/7)), -(1/3)*np.sqrt(5-2*np.sqrt(10/7)),\
              (1/3)*np.sqrt(5+2*np.sqrt(10/7)), -(1/3)*np.sqrt(5+2*np.sqrt(10/7))]
weights = [128/225, (322+13*np.sqrt(70))/900, (322+13*np.sqrt(70))/900, (322-13*np.sqrt(70))/900, (322-13*np.sqrt(70))/900]
def GL5(fun, a, b):
    fi = [((b-a)/2)*func(xi*((b-a)/2)+(b+a)/2) for xi in points]
    return np.array(fi)@np.array(weights)
complex_5GL = GL5(func, 1, 1.5)+GL5(func, 1.5, 2)+GL5(func, 2, 2.5)+GL5(func, 2.5, 3)
print("="*10+"5-point Gauss-Legendre"+"="*10)
print("quadrature = {}".format(complex_5GL))
```

对于 Romberg 积分公式，我们首先定义函数计算梯形值：

```
def Trap(fun, a, b, Iold, k):
if k == 1:
    Inew = (fun(a)+fun(b))*(b-a)/2
else:
    n = 2**(k-2)
    h = (b-a)/n
    x = a+(h/2)
    sum_k = 0
    for i in range(n):
        sum_k = sum_k + fun(x)
        x = x + h
    Inew = (Iold+h*sum_k)/2
return Inew
```

然后，根据 Richardson 加速公式：

$$T_m^{(k)} = \frac{4^m}{4^m-1} T_{m-1}^{(k+1)} - \frac{1}{4^m-1} T_{m-1}^{(k)}, \tag{5}$$

可以定义函数计算 T 表中第 k 行中每个元素的值：

```
def Richardson(R,k):
for i in range(k-1,0,-1):
    c = 2**(2*(k-i))
    R[i] = (c*R[i+1]-R[i])/(c-1)
return R
```

最后根据 Romberg 迭代算法，设置 $\varepsilon = 10^{-7}$ 为收敛精度，可计算积分值：

```python
def romberg(f, a, b, eps=1e-7):
T = {}
k = 1
T[1] = Trap(f,a,b,0.0,1)
former_R = T[1]
while True:
    k += 1
    T[k] = Trap(f,a,b,T[k-1],k)
    T = Richardson(T,k)
    if abs(T[1] - former_R) < eps:
        return T[1]
    former_R = T[1]
```

4 结果

运行代码的结果为：

```
==========5-point Gauss-Legendre==========
quadrature = -0.2387323403436461
==========Romberg==========
quadrature = -0.23873241462162365
```

与参考值进行对比，可以看出，5 点 Gauss-Legendre 求积公式已经有 8 位有效位数，而 Romberg 算法的精度则可以达到 10 位有效位数，结果都很好. 而且，进一步地，如果我们调整 Romberg 算法的收敛精度为 10^{-13}，得到的结果是 -0.23873241463784312，有效位数又增加了 5 位（达到了参考值的精度），因此 Romberg 算法是一种非常准确且灵活可调整的数值积分格式.