Numerical Analysis Programming Assignment #3 复化积分

陈宇轩 PB16060738

问题描述

1.分别编写用复化Simpson积分公式和复化梯形积分公式计算积分的通用程序。

2. 用如上程序计算积分 $I(f) = \int_1^6 \sin(x)dx$

取等距节点，记节点$\{x_i, i=0,...N\}$，其中N为 $\{2^k, k=0,1,...,12\}$，并计算误差(用科学计数形式)，同时给出**误差阶**（用浮点形式，比如 1.8789）。

3. 简要分析两种方法的优劣。

计算结果

复化梯形积分

k	误差 **e**	误差阶 **d**
0	-1.825006697305e+00	
1	-2.454792698194e-01	2.8942
2	-5.614913519149e-02	2.5112
3	-1.375739486821e-02	2.3505
4	-3.422468688994e-03	2.2647

k	误差 e	误差阶 d
5	-8.545713803539e-04	2.2121
6	-2.135776256161e-04	2.1768
7	-5.339033240903e-05	2.1516
8	-1.334732851249e-05	2.1326
9	-3.336816218225e-06	2.1179
10	-8.342030577979e-07	2.1061
11	-2.085507060379e-07	2.0965
12	-5.213767340084e-08	2.0884

复化Simpson积分

k	误差 e	误差阶 d
0	-1.356627125131e+00	
1	2.810298726757e-01	2.2712
2	6.960909684480e-03	3.8033
3	3.731852395462e-04	3.9426
4	2.250670407788e-05	3.9698
5	1.394389193443e-06	3.9784
6	8.695929759606e-08	3.9825
7	5.431993632676e-09	3.9851
8	3.394532432921e-10	3.9870
9	2.121502973296e-11	3.9885
10	1.325550780251e-12	3.9897
11	8.260059303211e-14	3.9910
12	4.607425552194e-15	4.0054

结果分析

1. 观察误差可以发现，复化梯形积分和复化Simpson积分的误差随着选取等距结点数量的增加，均逐渐减少。
2. 对于取相同数量的结点时，若结点较少，则复化梯形积分的误差略小，而当结点数较多时，复化Simpson积分的误差会比复化梯形积分小很多。
3. 对于误差阶，复化梯形积分的误差阶随结点数增加即步长的减小而变小，然而相反的是复化Simpson积分的误差阶随着结点数增加即步长减小而增大。说明Simpson的误差变化比步长的阶数要更高。

总结

本次实验我主要使用了python中的numpy库实现，本次实验编程简单，但是可以得到很有用的结果。通过这次实验，我更深刻的理解了复化梯形积分和复化Simpson积分的优劣和使用。

代码实现(Python)

```python
# 复化积分
import numpy as np


def f(x):
    return np.sin(x)


def compound_simpson(a, b, n):
    tmp1 = 0
    tmp2 = 0
    for i in range(1, n, 2):
        tmp1 += 4 * f(a + i * (b - a) / n)
    for i in range(2, n - 1, 2):
        tmp2 += 2 * f(a + i * (b - a) / n)
    result = (b - a) / (3 * n) * (f(a) + tmp1 + tmp2 + f(b))
    return result


def compound_trapezoid(a, b, n):
    tmp = 0
    for i in range(1, n):
        tmp = tmp + f(a + i * (b - a) / n)
    result = (b - a) / n * (1 / 2 * f(a) + tmp + 1 / 2 * f(b))
    return result


def main():
    I = -np.cos(6) + np.cos(1)  # 精准值
    # print("I:", I)
    print("复化梯形积分 误差和误差阶为")
    for k in range(0, 13):
```

```
        print("k={} ,".format(k), end=' ')
        N = 2 ** k
        T = compound_trapezoid(1, 6, N)
        print("e%d=" % k, end='')
        print("%.12e" % (I - T), end=' ')
        if k == 0:
            e = np.fabs(I - T)
            print('')
        if k != 0:
            print("d%d=" % k, end='')
            print("%.4f" % (-(np.log(np.fabs((I - T)) / e) / np.log(N))))
    print("复化Simpson积分 误差和误差阶为")
    for k in range(0, 13):
        print("k={} ,".format(k), end=' ')
        N = 2 ** k
        T = compound_simpson(1, 6, N)
        print("e%d=" % k, end='')
        print("%.12e" % (I - T), end=' ')
        if k == 0:
            e = np.fabs(I - T)
            print('')
        if k != 0:
            print("d%d=" % k, end='')
            print("%.4f" % (-(np.log(np.fabs((I - T)) / e) / np.log(N))))


if __name__ == "__main__":
    main()
```