数值实验报告

实验名称	计算方法上机实践				实验时间
姓名	秦浩政 郭凯平 刘桂凡 刘佳鑫	班级	数据 2301	学号	2306030214 2306020510 2309050116 2309050117

一、实验目的，内容

实验 4.1：Lagrange 插值和 Newton 插值

编程实现 Lagrange 插值法和 Newton 插值法。

实验内容：利用插值多项式计算函数值的近似值，并用数学软件绘制函数曲线，观察插值函数与被插值函

测试用例参考：

设 f(x)=1/(2*pai)$^{1/2}$ $\int_{-\infty}^{x} e^{-t^2/2}\,dt$,已知 f(x)的函数值表，用插值法求 f(0.13)和 f(0.36)的近似值。

x	0.0	0.1	0.2	0.3	0.4
f(x)	0.500	0.5398	0.5793	0.6179	0.7554

二、算法描述

实验 4.1:给定一组离散数据点 (x0,y0),(x1,y1),⋯,(xn,yn)（其中 xi 互不相同），构造一个通过所有这些点的

1，输入据点 (x0,y0),(x1,y1),⋯,(xn,yn)。

2，构造基函数：

对于每个数据点 xi，构造一个 Lagrange 多项式 Li(x)，定义为：

$$\mathrm{Li}(x)=\prod_{\substack{j=0\\ j\neq i}}^{n}\frac{x-x_j}{x_i-x_j}$$

基函数满足：

Li(xk)=1 如果 k=i,Li(xk)=0 如果 k≠i.

3.构造插值多项式：

将插值多项式 Ln(x) 表示为所有基函数的线性组合：

$$\mathrm{Ln}(x)=\sum_{i=0}^{n} Li(x)f(x_i)$$

由于每个基函数 Li(x)只在 xi 处取值为 1，在其他数据点 x_j(j≠i) 处取值为 0，因此 Ln(x)必然通过所有给

Newton 插值法：基于差商，逐步构建多项式

输出:

插值多项式 P(x)$P(x)$ 的表达式。

分为两部分：计算差商系数和求值。

1.对于 k 从 1 到 n，对于 i 从 n 到 k(倒序)，

计算差商(y[i]-y[i-1])/(x[i]-x[i-k])，将结果存入 y[i]

2.输入：差商系数 y，数据点 x，要插值的点 g

输出：在点 g 处的插值结果

步骤：

先初始化结果 s=0

对于 i 从 0 到 n：计算连乘积项∏(k=0 到 i-1)(g-x_k)

将 y[i]乘以连乘积项加到 s 中

最后返回 s

三程序代码

实验 4.1

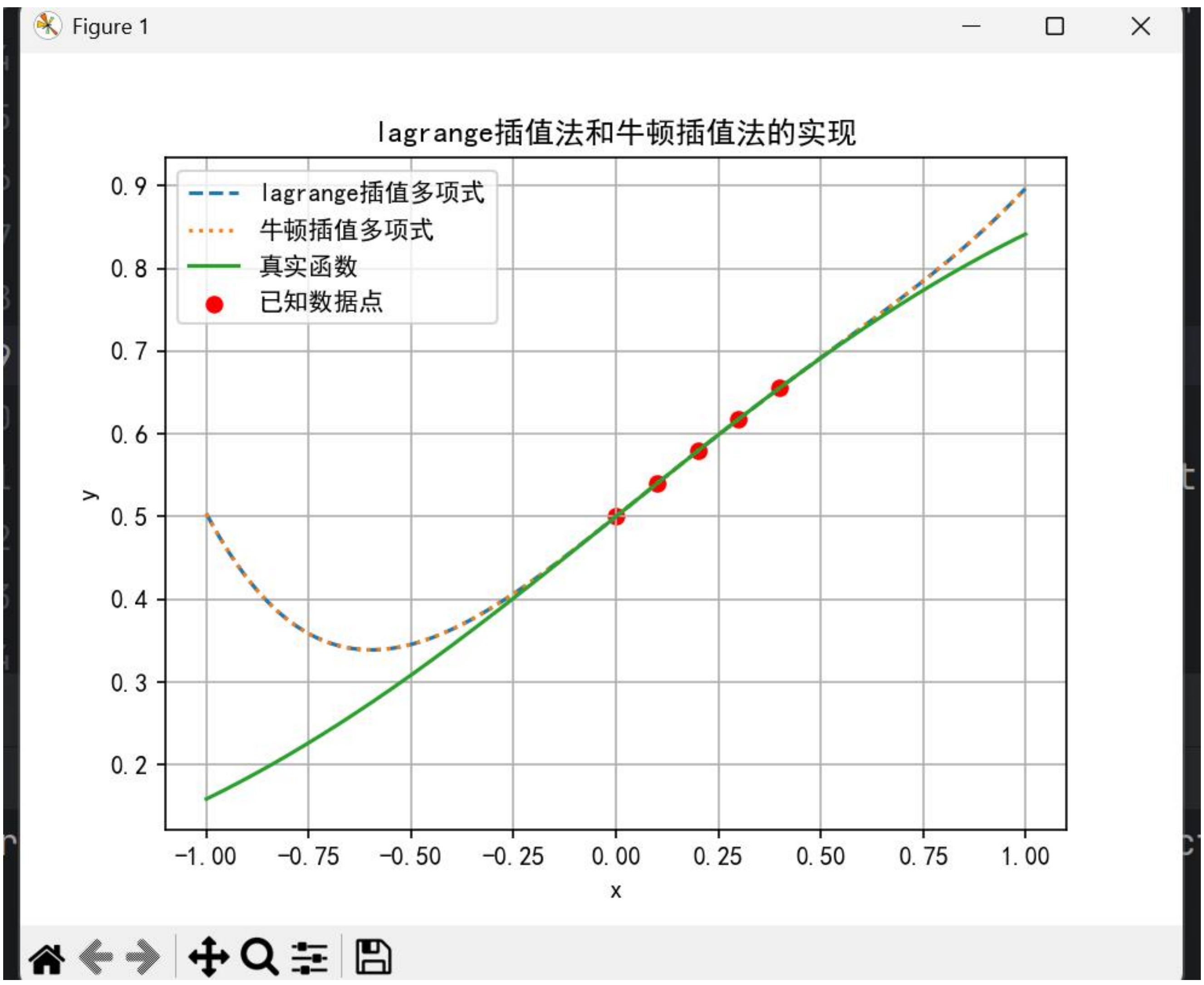
```python
import ...

# 设置字体支持中文
plt.rcParams['font.sans-serif'] = ['SimHei']  # 或使用其他支持
plt.rcParams['axes.unicode_minus'] = False  # 解决负号显示问题


def lagrange(x, y, n, g):
    s = 0
    for i in range(n + 1):
        u = y[i]
        for j in range(n + 1):
            if j != i:
                u = u * (g - x[j]) / (x[i] - x[j])
        s = s + u
    return s
#牛顿插值法的实现
#计算牛顿插值法的系数
def coefficient(x,y,n):
    for k in range(1,n+1):
        for i in range(n,-1,-1):
            if i-k>=0:
                y[i] =(y[i]-y[i-1])/(x[i]-x[i-k])
    return y
def newton(y,n,g):
    s = 0
    for i in range(n + 1):
        u = 1
        for k in range(i):
            u *= (g - x[k])
        s += y[i] * u
    return s
```

```python
def lagrange_newton_interpolation_plot(x, y, x_true, y_true):
    n = len(x) - 1
    # 定义一个较密的x值范围用于绘制插值多项式
    x_dense = np.linspace(min(x_true), max(x_true), num: 400)
    y_lag_dense = [lagrange(x, y, n, xi) for xi in x_dense]
    y = coefficient(x,y,n)
    y_new_dense = [newton(y,n,xi)for xi in x_dense]
    # 绘制lagrange插值多项式函数
    plt.plot( *args: x_dense, y_lag_dense, label='lagrange插值多项式',li
    #绘制牛顿插值多项式函数
    plt.plot( *args: x_dense, y_new_dense, label='牛顿插值多项式',linesty
    # 绘制真实函数
    plt.plot( *args: x_true, y_true, label='真实函数')
    # 绘制原始数据点
    plt.scatter(x, y1, color='red', label='已知数据点')
    plt.xlabel('x')
    plt.ylabel('y')
    plt.title(f'lagrange插值法和牛顿插值法的实现')
    plt.legend()
    plt.grid(True)
    plt.show()
```

```python
# 原始数据点
x = [0, 0.1, 0.2, 0.3, 0.4]
y=[0.5,0.5398,0.5793,0.6179,0.6554]
y1=[0.5,0.5398,0.5793,0.6179,0.6554]#由于y会更新，我们重新设置一个列表y1
# 定义一个较密的x值范围用于绘制真实函数
x_true = np.linspace(-1, stop: 1, num: 400)  # 选择一个更大的区间来展示
y_true = norm.cdf(x_true)

n = len(x) - 1
g1 = 0.13
g2 = 0.36

# 绘制插值多项式和真实函数
lagrange_newton_interpolation_plot(x, y, x_true, y_true)

# 计算并打印插值
lag_interpolated_value_g1 = lagrange(x, y1, n, g1)
lag_interpolated_value_g2 = lagrange(x, y1, n, g2)
#计算牛顿插值法的近似值
y = coefficient(x,y1,n)
new_interpolated_value_g1 = newton(y, n, g1)
new_interpolated_value_g2 = newton(y, n, g2)
print(f"这是{n}次lagrange插值法的实现")
print(f"0.13对应的近似值：是{lag_interpolated_value_g1}")
print(f"0.36对应的近似值：是 {lag_interpolated_value_g2}")
print(f"这是{n}次牛顿插值法的实现")
print(f"0.13对应的近似值：是{new_interpolated_value_g1}")
print(f"0.36对应的近似值：是 {new_interpolated_value_g2}")
```

四．数值结果

实验 4.1



五．计算结果分析

实验 4.1

理论上，拉格朗日插值和牛顿插值在给定相同数据点时，应该产生完全相同的插值多项式，只是表现形式不同。

虽然两种方法理论上等价，但在实际计算中可能因舍入误差而出现微小差异。本例中未观察到差异，说明在低阶插值(n

种方法都具有良好的数值稳定性。

六. 计算中出现的问题，解决方法及体会

实验 4.1

问题：拉格朗日插值中，当节点间距很小时，分母(x_i-x_j)可能导致数值不稳定。

解决办法：应尽量避免这种情况（节点间距关小）发生。不然会导致数值不稳定。

体会：算法做好后可以多尝试一些案例，可能会别有收获。

教 师 评 语	指导教师：