**北京信息科技大学实验报告**

**课程名称**  数值分析

**实验项目**  题目一

**实验仪器**  计算机

**学 院** 理学院

**专 业** 信息与计算科学

**小组成员(姓名+学号)** 李子睿+2023012581

贺鸣阳+2023012588

苗含稷+2023012617

**实验日期**  2025.5.23

**成 绩**

**指导教师**  路康亚

**实验报告内容**

注：实验报告要写有实验题目、实验原理、代码、结果和分析。所有数学符号或公式都用公式编辑器或者Mathtype编辑(不能截图)。

**一、实验目的与要求（实验题目）**

(1)求如下数据的7次最小二乘拟合多项式：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|  | 1.0 | 1.2 | 1.4 | 2.0 | 2.4 | 2.6 | 2.8 | 3.0 | 3.2 |
|  | 2.8 | 5.4 | 1.9 | 0 | 0.6 | 1.8 | 3.4 | 5.0 | 5.8 |

(2) 用三点和五点高斯-勒让德求积公式计算积分的近似值．

**二、实验原理（主要计算方法及公式）**

（1）通过构造设计矩阵并计算系数向量，最终得到最小二乘求解。

（2）通过定义函数gauss\_legendre\_3和gauss\_legendre\_5，分别实现三点和五点公式，输入被积函数、积分区间，返回近似积分值。之后通过变换将区间映射到标准区间后，代入节点和权重计算求和。

三、**实验过程、步骤（程序代码）**

|  |
| --- |
| **1.** |
| import numpy as np  import matplotlib.pyplot as plt  # 设置中文字体  plt.rcParams['font.sans-serif'] = ['SimHei']  # 使用黑体  plt.rcParams['axes.unicode\_minus'] = False  # 解决负号显示问题  # 输入数据  x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8])  y = np.array([2.8, 5.4, 1.9, 0, 0.6, 1.8, 3.4, 5.0, 5.8])  # 构造设计矩阵（7次多项式有8个系数）  X = np.column\_stack([x\*\*i for i in range(8)])  # 最小二乘法求解系数  coefficients = np.linalg.lstsq(X, y, rcond=None)[0]  # 输出结果  print("7次多项式系数（从a0到a7）：")  for i, coef in enumerate(coefficients):      print(f"a\_{i} = {coef:.8f}")  # 构造多项式函数  def poly(x\_val, coeffs):      return sum(coeff \* x\_val\*\*i for i, coeff in enumerate(coeffs))  # 验证拟合效果  print("\n拟合值与实际值对比：")  for xi, yi in zip(x, y):      print(f"x={xi}: 实际y={yi:.1f}, 拟合y={poly(xi, coefficients):.2f}")  # 绘制拟合曲线  plt.scatter(x, y, color='red', label='实际数据')  x\_fit = np.linspace(0, 8, 100)  y\_fit = poly(x\_fit, coefficients)  plt.plot(x\_fit, y\_fit, label='7次多项式拟合')  plt.legend()  plt.xlabel('x')  plt.ylabel('y')  plt.title('7次最小二乘多项式拟合')  plt.grid(True)  plt.show() |
| **2.** |
| import numpy as np  def gauss\_legendre\_3(f, a, b):      # 三点高斯-勒让德求积      t = np.array([-np.sqrt(3/5), 0, np.sqrt(3/5)])      w = np.array([5/9, 8/9, 5/9])      x = (b - a)/2 \* t + (a + b)/2      return (b - a)/2 \* np.sum(w \* f(x))  def gauss\_legendre\_5(f, a, b):      # 五点高斯-勒让德求积      t = np.array([          -np.sqrt((5 + 2\*np.sqrt(10/7))/3),          -np.sqrt((5 - 2\*np.sqrt(10/7))/3),          0,          np.sqrt((5 - 2\*np.sqrt(10/7))/3),          np.sqrt((5 + 2\*np.sqrt(10/7))/3)      ])      w = np.array([          (322 - 13\*np.sqrt(70))/900,          (322 + 13\*np.sqrt(70))/900,          128/225,          (322 + 13\*np.sqrt(70))/900,          (322 - 13\*np.sqrt(70))/900      ])      x = (b - a)/2 \* t + (a + b)/2      return (b - a)/2 \* np.sum(w \* f(x))  # 示例：计算 ∫e^{-x^2} dx 在 [0, 1] 上的积分  f = lambda x: 1/1+x\*\*4  a, b = 0, 1  I3 = gauss\_legendre\_3(f, a, b)  I5 = gauss\_legendre\_5(f, a, b)  print("三点高斯-勒让德近似值:", I3)  print("五点高斯-勒让德近似值:", I5) |

**四、实验结果**

|  |
| --- |
| **1.** |
| **fa98fc2ff0659dfdb8e361a965940447210dbd9fdf8d2937bcf4b0a91c5c52** |
| **2.** |
| **843726312dc8736f258cdec454c6c83** |

**五、结果分析**

最小二乘拟合通过高次多项式可精确拟合离散数据，但7次会让精确度降低。高斯-勒让德求积公式在积分计算中高效且精度可控，节点数越多，结果越精确。

**六、小组成员分工**

（写清楚每位成员负责哪一部分内容，并给出工作比例，例如三人小组给定比例为1：0.9：0.95，若教师给定得分为98，则最终成员的成绩为98，98\*0.9，98\*0.95）

注：实验报告电子版提交至学习通