基于平面坐标变换法和斜率递推法的的曲线重构模型

摘　要

随着科技的进步，光纤传感器技术作为一种先进的监测手段，已被广泛应用于工业、医疗和结构健康监测等多个领域。本文将聚焦于利用光纤传感器技术进行平面曲线的重建，期望能够提高光纤传感技术的应用效率和准确性。

针对问题一，本文首先计算出题目中给出的两种状态下六个测量点的曲率，并采用**三次样条插值**和**线性插值**两种插值算法**将离散曲率值连续化**。接着，引入 **Frenet-Cartesian 坐标变换算法**精确模拟光纤平面曲线，通过微元法和坐标变换重构受力后光纤的平面曲线，构建**基于 Frenet-Cartesian 坐标变换的曲线重构模型，**在两种插值算法下分别估算出横坐标在 0.3 米、0.4 米、0.5 米、0.6 米和 0.7 米处的曲线曲率。

针对问题二，本文首先构建**基于斜率递推法的曲线重构模型，**并利用 **Frenet-Cartesian 坐标变换法和斜率递推法**两种算法重构平面曲线。其次，本文对重建出的光纤曲线在**几何特性**、**平滑度**、**插值算法种类**方面的特征进行分析。在曲线插值算法种类的特征中，我们对线性插值和三次样条插值两种方法在曲线重构中的效果进行比较，发现两种插值方法在曲线重构上几乎没有差别，均可以较好的重构曲线。

针对问题三，本文基于前两问的建模方法对给定的平面曲线进行重构，初深入探讨重构过程中的误差来源。在模型建立阶段，首先根据曲率计算公式，通过采样点获取曲率值。随后，应用插值算法对曲率数据进行连续化处理，确保了曲率值在整个曲线上的连续性和平滑性。基于此，采用 **Frenet-Cartesian 坐标变换**和**斜率递推法**依据题目中的情景建立了曲线重构的数学模型。求解过程中，我们采用了递推思想，通过迭代计算逐步逼近曲线上每一点的位置坐标，实现曲线的重构。最后，我们对重构曲线与原曲线进行误差分析，误差分析过程采用**蒙特卡洛积分法**和**均方误差(MSE)**来评估重构曲线与原始曲线之间的差异，将产生误差的原因总结为以下三类：**模型误差，插值点个数**，**采样点个数**，并用**控制变量法**对误差进行计算和分析。

综上所述，本文通过线性插值和三次样条插值算法对曲率数据进行连续化处理，并建立了两个平面曲线重构模型，实现了从离散曲率数据到连续平面曲线的有效重构，以提高重构曲线的物理特性匹配度进而实现对平面光纤曲线的重构。同时，本文也对曲线的特点和误差进行了总结归纳和分析。

关键词：曲线重构　插值算法　Frenet-Cartesian 坐标变换法　斜率递推法　MSE　蒙特卡洛积分法

目录

一、 问题重述

1.1 问题背景

随着科技的进步，光纤传感器技术作为一种先进的监测手段，已被广泛应用于工业、医疗和结构健康监测等多个领域。这种传感器技术利用光波作为传感信号，通过光纤这一传输载体，精确地感知和测量外界环境中的变化。它的基本原理在于，当外界环境参数如温度、压力或形状发生变化时，光纤中的光波参数（波长、相位、强度等）也会相应变化。由于光纤传感器具有体积小、质量轻、抗电磁干扰能力强、灵敏度高等特点，它可以被应用在许多传统传感器难以适应的环境中。

特别地，光纤传感器在形状和位移监测方面显示出独特的优势。通过测量受力后光纤中的波长变化，我们可以计算出材料或结构的实时曲率，进而推断出其具体的形变情况。这种技术不仅可以应用于普通的工业生产线，以监测设备运行状态和预防故障，还可以应用在诸如桥梁和建筑物的结构安全监测中，帮助工程师及时发现潜在的结构问题，避免灾难性的后果。

本文将聚焦于利用光纤传感器技术进行平面曲线的重建。这一研究不仅能够深入探讨光纤传感器的物理和工程应用，还能够推动其在更多高精度监测场景下的应用，例如在微型机器人技术、精密仪器设计以及医疗设备中的实际应用，这些都要求极高的监测和操作精确性。本研究期望能够提高光纤传感技术的应用效率和准确性，为未来的技术发展提供坚实的基础。

1.2 问题提出

根据以上背景和题目所给的数据信息，需要解决以下三个问题：

问题一：根据题目中给出的波长测量数据，建模估计平面光栅 FBG1-FBG6 传感点的曲率。令原点为初始点坐标，x 轴为初始水平光纤方向，y 轴为垂直方向，当光纤在平面内受力后，其起始位置的切线与水平方向形成$45°$，建模估算横坐标中x轴对应的五个位置的曲率。

问题二：在表格数据和问题一的基础上，设计数学模型重构该平面曲线，并且对曲线的特征进行分析解释。

问题三：对平面曲线方程$y=x^3+x(0\leqslant x\leqslant 1)$进行适当等间距弧长采样以计算各采样点的曲率。利用这些曲率数据，建立数学模型以重构该平面曲线，并探讨重构曲线与原始曲线间误差的成因。

二、 问题分析

2.1 问题一的分析

根据问题一的要求，我们需要构建数学模型来估计传感器各测量点的曲率，并进一步估算出横坐标相应位置处的曲率。本问题可以视为一个典型的物理建模问题。首先利用已知的波长测量数据计算各传感点的初始和变形后的曲率值，这是问题求解的基础。接下来，为了从离散数据中重建连续曲率曲线，本文采用**线性插值法**和**三次样条插值法**，三次样条法对曲率的插值后曲线更光滑，但并不能直接得出三次样条插值法在曲线重构中具有较大优势的结论，两种插值方式优劣会在问题二中进行比对。最后，本文建立**基于 Frenet-Cartesian 坐标变换的曲线重构模型**，通过微元法和坐标变

换重构受力后光纤的平面曲线，估算出横坐标在 0.3 米、0.4 米、0.5 米、0.6 米和 0.7 米处的曲率。

2.2 问题二的分析

在问题二中，本文构建数学模型来重构光纤的平面曲线，并对重构曲线的特征进行了归纳总结，并第一问采用的两种插值方法在曲线重构中的效果进行了分析。首先，本题设计了**斜率递推算法**用于曲线重构，并利用第一问中建立的 **Frenet-Cartesian 坐标变换法曲线重构模型**和**斜率递推法的曲线重构模型**两种数学模型实现了对光纤平面曲线的精细重构。其次，我们对重建出的光纤曲线在**几何特性**、**平滑度**、**插值算法种类**方面的特征进行分析。在曲线插值算法种类的特征中，我们对线性插值和三次样条插值两种方法在曲线重构中的效果进行比较，发现两种插值方法在曲线重构上几乎没有差别，均可以较好的重构曲线。

2.3 问题三的分析

在问题三中，本文基于前两问的建模方法对给定的平面曲线进行重构，初深入探讨重构过程中的误差来源。在模型建立阶段，首先根据曲率计算公式，通过采样点获取曲率值。随后，应用插值算法对曲率数据进行连续化处理，确保了曲率值在整个曲线上的连续性和平滑性。基于此，采用 **Frenet-Cartesian 坐标变换**和**斜率递推法**依据题目中的情景建立了曲线重构的数学模型。求解过程中，我们采用了递推思想，通过迭代计算逐步逼近曲线上每一点的位置坐标，实现曲线的重构。最后，我们对重构曲线与原曲线进行误差分析，误差分析过程采用蒙特卡洛积分法和均方误差(MSE)来评估重构曲线与原始曲线之间的差异，将产生误差的原因总结为以下三类：**模型误差，插值点个数**，**采样点个数**，并用**控制变量法**对误差进行计算和分析。

三、 模型假设

假设 1：假设插值点数足够满足插值个数的需求；

假设 2：假设曲线曲率与波长变化量近似成正比。

假设 3：每个微元弧段的曲率半径都取左端点的曲率半径，每一个微元扇形都简化成一个三角形。

假设 4：计算 MSE 积分时取横坐标与蒙特卡洛得到的随机数最近的点的纵坐标作为拟合值

四、 符号说明

符号	说明	单位
κ	曲率	
λ_0	水平光纤在初始状态下测量的波长	纳米
λ	光纤在受到外力后测量的波长	纳米

s_i	点$O_i O_{i+1}$间的弧长	
ρ_i	曲线在点O_i处的曲率半径	
θ_i	圆弧$O_i O_{i+1}$对应圆心角θ_i	rad
α_i	曲线在O_i点切线与X轴正方向的夹角	rad
β_i	直线$O_i O_{i+1}$与X轴正方向的夹角	rad
k_i	点O_i处曲线切线的斜率	
MSE	均方误差	

五、 问题一模型的建立与求解

问题一分析的流程图如下：

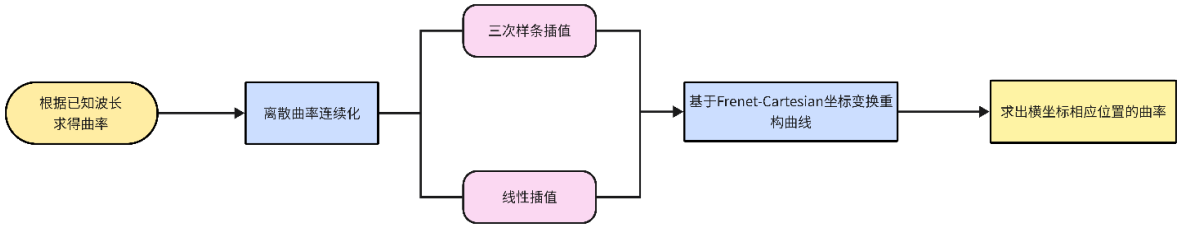

图 1：问题一分析的流程图

5.1 基于插值算法的离散曲率连续化

已知根据波长求曲率κ计算的公式为：

$$k = c\left(\frac{\lambda - \lambda_0}{\lambda_0}\right) \tag{1}$$

其中，λ_0是水平光纤在初始状态下测量的波长，λ是光纤在受到外力后测量的波长，c为某个常数（此处令c=4200）

题目中给出了实验测量的两组不同初始状态下光纤受力前后的波长值，具体数据如表 1 所示：

表 1：波长（纳米）测量数据

测量点	初始状态 1	测试 1	初始状态 2	测试 2
FBG1	1529	1529.808	1540	1541.095
FBG2	1529	1529.807	1540	1541.092
FBG3	1529	1529.813	1540	1541.090
FBG4	1529	1529.812	1540	1541.093
FBG5	1529	1529.814	1540	1541.094
FBG6	1529	1529.809	1540	1541.091

根据表中数据可以计算处两种不同受力状态下六个测量点的曲率值，如表 2 所示：

表 2：测试 1 与测试 2 的曲率值计算结果

传感器	测试 1 的曲率	测试 2 的曲率
FBG1	2.219490	2.986364
FBG2	2.216743	2.978182
FBG3	2.233224	2.972727
FBG4	2.230477	2.980909
FBG5	2.235971	2.983636
FBG6	2.222237	2.975455

对表 2 中离散曲率值需要进行连续化，需要通过插值算法完成，采用线性插值和三条样值插值法进行插值。

线性插值法是一种数学方法，用于在两个已知数据点之间估算一个新的数据点。这种方法假定两个已知点之间的变化是线性的，即形成一条直线。如果有两个已知点(x_0,y_0)和(x_1,y_1)，且需要在这两点之间插值得到x对应的y值，则线性插值y公式如下：

$$y = y_0 + \frac{(y_1 - y_0)}{(x_1 - x_0)} \times (x - x_0) \tag{2}$$

连续化过程可通过对邻近的离散点进行线性插值实现，其主要目的在于迅速生成多个微段的曲率值。这种方法的优势在于计算速度快，然而其劣势在于将曲率的划分太过简单，因为实际的空间曲线的曲率变化通常比线性模型所描述的更为复杂。

本文引入三次样条插值法对离散曲率的连续化。该插值方法的核心优势在于，在连续曲率插值曲线的连接节点上，可以保持一阶和二阶导数的连续性，从而确保了拟合出的曲率连续化的曲线是绝对光滑的，更符合实际曲率曲线的特性。在每一段$[x_i,x_{i+1}]$上，插值函数$S(x)$是一个三次多项式：

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \tag{3}$$

其中i表示区间索引，a_i,b_i,c_i,d_i是多项式的系数，需要通过条件求解得到。在每个内部节点x_i上，需确保样条函数的值等于已知数据点的y值，确保每个节点处相邻多项式的一阶和二阶导数连续，以保证曲线的平滑过渡，在边界节点，通常设定二阶导数为零（自然边界条件），或根据实际情况设定一阶导数的值。根据给定x值所在的具体区间，使用相应的三次多项式$S_i(x)$进行插值，以求得对应的y值。

根据线性插值和三次样条插值算法对曲率进行连续化，得到插值结果如图 2 所示，其中横坐标为光纤曲线弧长长度，纵坐标为插值后曲线每一点对应的曲率值。

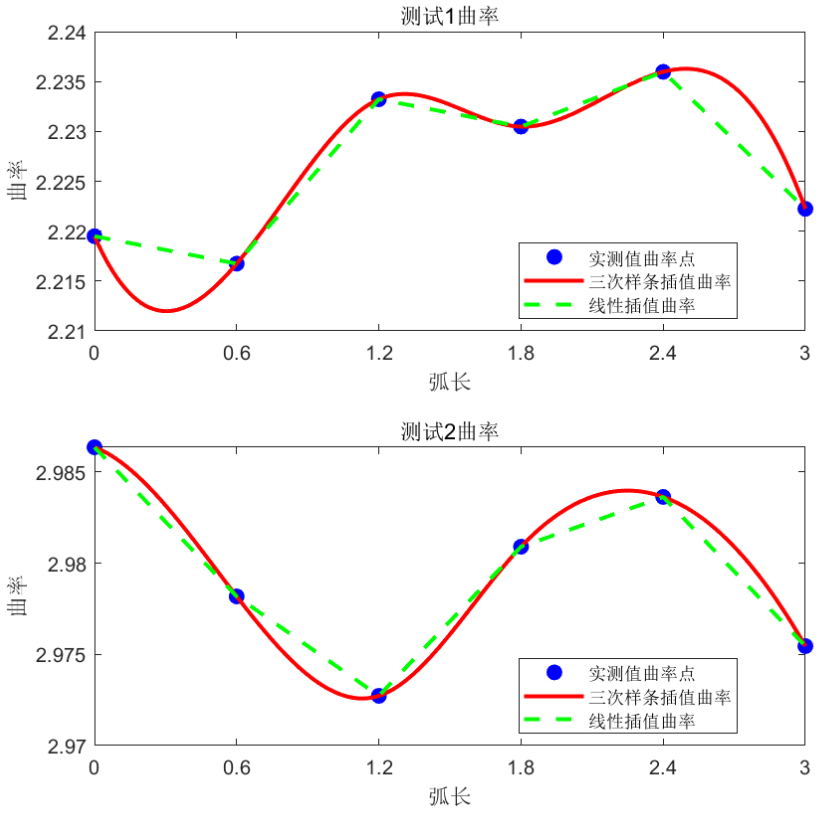

图 2：插值结果图

根据两种插值算法插值得到的曲率—弧长图像可以看出，三次样条插值法在处理连续性和平滑性要求高的弧长——曲率数据对曲线方面表现出显著的优势，而线性插值方法将曲率的划分太过简单，但并不能就此得出三次样条插值法在曲线重构中具有较大优势的结论，具体对两种插值方法的评价在问题二的 5.2.3 小节中平面曲线重建后的曲线特征处进行详细比对。

5.2 基于 Frenet-Cartesian 坐标变换的曲线重构模型

本文建立 Frenet-Cartesian 坐标变换的曲线重构模型，这种方法主要是将有限离散的曲率-弧长数据对，通过 5.1.1 节插值方法可以得到划分更细的曲率-弧长数据对，作为 Frenet-Cartesian 坐标变换递推算法的输入条件之一。

在微分几何中，曲线描述通常采用 Frenet 坐标标架。Frenet 坐标是一种用于描述曲线上点的运动和几何的坐标系统，T 为切向量方向，是指沿曲线方向的单位向量；N 为主法线方向，是垂直于切向量的单位向量，指向曲线的曲率中心；B 为副法线方向，也称为扭率向量，是切向量和法向量的向量积，垂直于前两者。三者关系为：$B=T\times N$。

本题在二维空间中使用 Frenet 坐标系$T-O_i-N$，由切向量T和法向量N构成，曲线上的点可以通过沿曲线的距离（弧长参数s）和从曲线到该点的最短距离（法向偏移d）来描述。

假设以O_i为坐标为原点建立 Cartesian 坐标系$X—O_i—Y$，初始的水平光纤方向为X轴，垂直方向为Y轴，光纤受力后发生弯曲，根据微积分的思想，将弯曲后的光纤

弧分成n个相等的微元段，每段微元的弧长分别为s_1，s_2，…，s_i，…，s_n满足$s_1=s_2=\cdots=s_n$，一共得到$n+1$个端点，记为O_0，O_1，…，O_n.

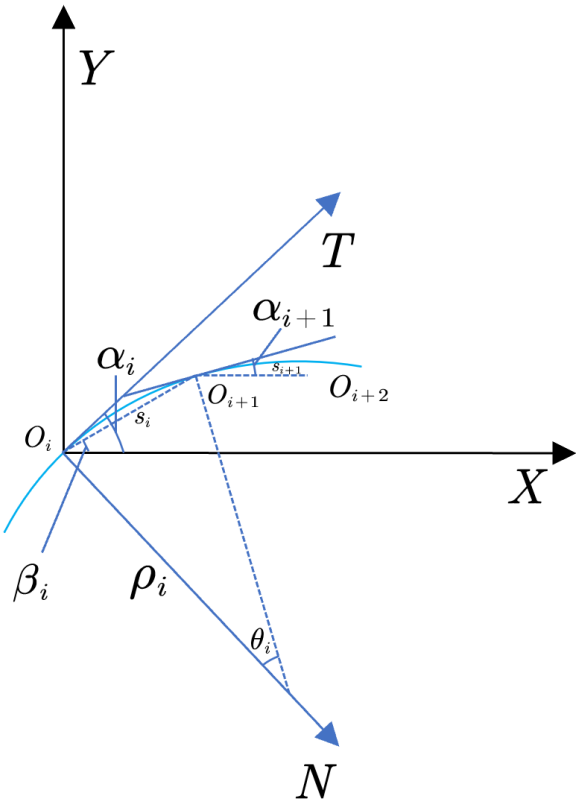

图 3：平面坐标变换模型原理图

图 3 为平面坐标变换算法原理图，图中点O_{i+1}在坐标系$X-O_i-Y$中坐标为$(\Delta x_{i+1}, \Delta y_{i+1})$，在 Frenet 坐标系$T-O_i-N$中坐标为$(t_{i+1}, n_{i+1})$$(i=0,1,2,\cdots,n)$. 点$O_{i+1}$由坐标系$T-O_i-N$向坐标系$X-O_i-Y$中变换步骤如下：

假设每个微元弧段的曲率半径都取左端点的曲率半径。作曲线在点O_i处的曲率半径ρ_i，点O_i处的曲率半径ρ_i与曲率κ_i的关系如下：

$$\rho_i=\frac{1}{\kappa_i} \tag{4}$$

由于S_i近似为一段微小圆弧，由圆的弧长公式得：

$$s_i=\rho_i\theta_i \tag{5}$$

则可推导出圆弧O_iO_{i+1}对应圆心角θ_i的计算公式如下：

$$\theta_i=s_i\kappa_i \tag{6}$$

分析图 1 中的几何关系，可以得出点O_{i+1}在坐标系$T-O_i-N$中的横纵坐标分别可以表示为：

$$\begin{cases} t_{i+1}=(1-\cos\theta_i)\rho_i \\ n_{i+1}=\sin\theta_i\rho_i \end{cases} \tag{7}$$

即：

$$\begin{cases} t_{i+1} = \dfrac{(1-\cos\theta_i)}{\kappa_i} \\ n_{i+1} = \dfrac{\sin\theta_i}{\kappa_i} \end{cases} \tag{8}$$

由公式(3)计算出的θ_i的值，进而计算出点O_{i+1}在坐标系$T-O_i-N$下的坐标$(t_{i+1}\ ,\ n_{i+1})$

令β_i为直线O_iO_{i+1}与X轴正方向的夹角，α_i为曲线在O_i点处切线与X轴正方向的夹角。根据图 1 中的几何关系进行角度变换，可以得出β_i的计算公式如下：

$$\beta_i = \alpha_i - \left[\frac{\pi}{2} - \left(\frac{\pi-\theta_i}{2}\right)\right] = \alpha_i - \frac{\theta_i}{2} \tag{9}$$

其中，α_i的初始值α_0为光纤在平面内受力后在初始位置的切线与水平方向的夹角，即 $\alpha_0 = 45°$

根据点O_{i+1}在坐标系$T-O_i-N$下的坐标$(t_{i+1}\ ,\ n_{i+1})$可计算得向量$\overrightarrow{O_iO_{i+1}}$的模值$l_i$，计算公式如下：

$$l_i = |\overrightarrow{O_iO_{i+1}}| = \sqrt{{t_{i+1}}^2 + {n_{i+1}}^2} \tag{10}$$

点O_{i+1}在坐标系$X-O_i-Y$中的横纵坐标分别可以表示为：

$$\begin{cases} \Delta x_{i+1} = l_i\cos\beta_i \\ \Delta y_{i+1} = l_i\sin\beta_i \end{cases} \tag{11}$$

由公式(6)(7)可计算出β_i和l_i的值，进而根据公式(8)计算出点O_{i+1}在坐标系$X-O_i-Y$下的坐标值$(\Delta x_{i+1}\ ,\ \Delta y_{i+1})$

过点O_{i+1}作切线，令该切线与水平X轴方向夹角为α_{i+1}，由几何关系可得出α_{i+1}的计算公式如下：

$$\alpha_{i+1} = \alpha_i - \theta_i \tag{12}$$

从公式(9)可以看出α_{i+1}与α_i存在递推关系，又因为已知$\alpha_1 = 45°$，可以根据该递推式计算出α_1，α_2，…，α_n的值。

由公式(6)至公式(12)可求出$\Delta x_1, \Delta x_2, ..., \Delta x_n$和$\Delta y_1, \Delta y_2, ..., \Delta y_n$的值，通过累加，可求出$O_0$，$O_1$，…，$O_n$在$X-O_0-Y$坐标系下的坐标$(x_0, y_0)$，$(x_1, y_1)$，$(x_2, y_2)$，…，$(x_n y_n)$，具体公式如下：

$$\begin{cases} x_i = \sum_{j=1}^{i} \Delta x_j \\ y_i = \sum_{j=1}^{i} \Delta x_j \end{cases} ,\ i = 1, 2, \cdots, n \tag{13}$$

已知初始点在$X-O_0-Y$坐标系下的坐标(x_0, y_0)为$(0, 0)$，根据坐标计算公式(13)可计算出O_0，O_1，…，O_n的坐标，进而可以对光纤曲线进行重构。

5.3 模型的求解

将测试 1 下的光纤曲线等分为 3000 等份，每份近似为一个无限小弧微元，弧端点为O_0，O_1，…O_i，O_{i+1}…，O_n（n=3000），根据 Frenet-Cartesian 坐标变换模型计算出点O_{i+1}在$X-O_i-Y$坐标系下横坐标Δx_i，$i=0,1,2,\cdots,n$，通过逐步累加的方式得到点O_i和点O_{i+1}在$X-O_0-Y$坐标系下横坐标的值x_i和x_{i+1}，利用判断条件搜索i的值，判断条件如下：

$$\begin{cases} x_i=\sum_{j=1}^{i}\Delta x_i \leqslant 0.3 \\ x_{i+1}=\sum_{j=1}^{i+1}\Delta x_i > 0.3 \end{cases} \tag{14}$$

满足公式(14)的判断条件可通寻找到点O_i和点O_{i+1}，即可确定横坐标$x=0.3$位于x_i和x_{i+1}之间。点O_i和点O_{i+1}对应的曲率为κ_i和κ_{i+1}，取$\frac{\kappa_i+\kappa_{i+1}}{2}$为横坐标$x=0.3$对应曲线上的点的曲率值。同理，计算 0.4 米，0.5 米 0.6 米，0.7 米处的曲率k值，并根据测试 2 下的光纤曲线计算测试 2 中横坐标为 0.3 米，0.4 米，0.5 米 0.6 米，0.7 米处的曲率k值。

根据线性插值法和三次样条插值法分别计算横坐标为 0.3 米，0.4 米，0.5 米 0.6 米，0.7 米处的曲率k值如下：

表 3：线性插值法曲率值计算结果

横坐标x（米）	0.3	0.4	0.5	0.6	0.7
测试 1 曲率κ	2.217958	2.217501	2.217020	2.218295	2.222443
测试 2 曲率κ	2.981925	2.980466	2.978666	—	—

表 4：三次样条插值法曲率值计算结果

横坐标x（米）	0.3	0.4	0.5	0.6	0.7
测试 1 曲率κ	2.212046	2.213045	2.215155	2.218404	2.223264
测试 2 曲率κ	2.982781	2.981042	2.978785	—	—

由表 3 和表 4 可以看出，测试 2 在横坐标x为 0.6 米和 0.7 米没有曲率，推测原因是测试二重构出的曲线没有横坐标x为 0.6 米和 0.7 米的部分，具体重构曲线验证这个推测在问题二 5.2.2 小节中。

六、 问题二模型的建立与求解

问题二的分析流程图如下：

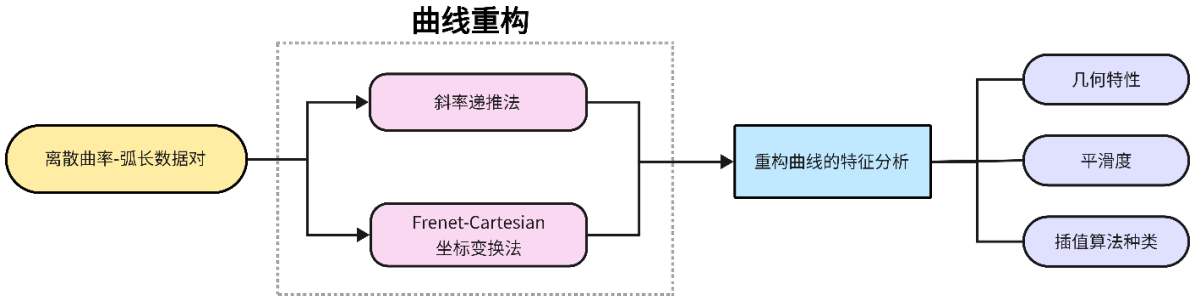

图 4：问题二的分析流程图

6.1 基于斜率递推法的曲线重构模型

斜率递推法根据有限的离散曲率-弧长数据对，采用 5.1.1 节合适的插值方法插值后得到划分更细的离散曲率-弧长数据对，将初始斜率倾角作为输入条件之一。

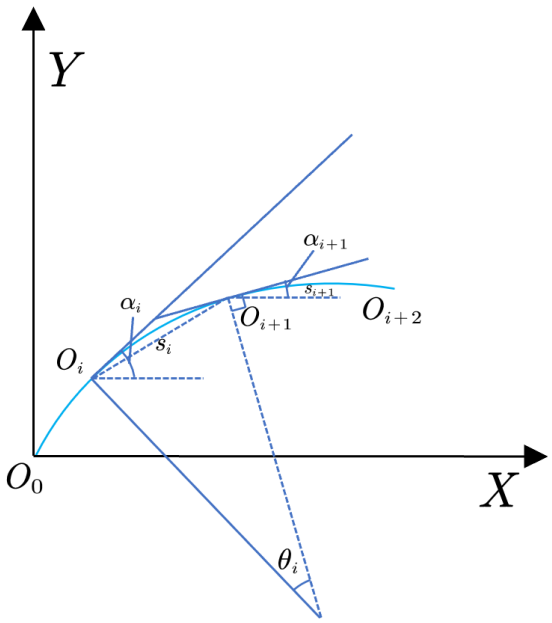

图 5：斜率递推法原理图

如图 5 斜率递推法原理图所示，以光纤初始点为原点建立 Cartesian 坐标系 $X-O_0-Y$，初始点关于曲线的切线相对于 X 轴正方向的倾角为 $45°$，将弯曲后的光纤弧分成 n 个相等的微元段，每段微元的弧长分别为 s_1，s_2，…，s_i，…，s_n 满足 $s_1=s_2=\cdots=s_n$，一共得到 $n+1$ 个端点，记为 O_0，O_1，…，O_n．设 O_i 和 O_{i+1} 点处的斜率分别为 k_i 和 k_{i+1} $(i=0,1,2,\cdots,n)$，曲率分别为 κ_i 和 κ_{i+1}．O_i 和 O_{i+1} 的坐标分别为 $(x_i,\ y_i)$ 和 $(x_{i+1},\ y_{i+1})$，曲线在 O_i 和 O_{i+1} 两点处的切线对 X 轴正方向的倾角分别为 α_i 和 α_{i+1}，θ_i 为两点切向角的变化值，s_i 为点 O_i 和点 O_{i+1} 之间的弧长。

由图 3 中几何关系得 θ_i 的计算公式如下：

$$\theta_i=\alpha_i-\alpha_{i+1} \tag{15}$$

已知初始斜率倾角为 $45°$，由公式(6)得 $\theta_i=s_i\kappa_i$，计算 θ_{i+1} 的递推公式如下：

$$\alpha_{i+1}=\alpha_i-s_i\kappa_i \tag{16}$$

根据初始条件$\alpha_0=45°$，可计算出各个端点切线的倾角值。根据点O_i处切线倾角值可计算出点O_i处的斜率k_i，计算公式如下：

$$k_i=\tan(\alpha_i) \tag{17}$$

线段O_iO_{i+1}长度可由点O_i和点O_{i+1}间的弧长长度近似代替，O_iO_{i+1}连线的斜率近似用O_i处对曲线的切线斜率近似代替，由几何关系可得出点O_{i+1}的递推计算公式如下：

$$\begin{cases} x_{i+1}=x_i+\dfrac{s_i}{\sqrt{1+k_i^2}} \\ y_{i+1}=y_i+\dfrac{s_ik_i}{\sqrt{1+k_i^2}} \end{cases} \tag{18}$$

已知初始点坐标(x_0,y_0)为$(0,0)$，根据坐标递推公式(17)可计算出O_0，O_1，…，O_n的坐标，进而可以对光纤曲线进行重构。

6.2 平面曲线重构结果

本题采用**基于 Frenet-Cartesian 坐标变换的曲线重构算法**和**基于斜率递推法的曲线重构算法**对平面曲线进行重构。

6.2.1 基于 Frenet-Cartesian 坐标变换的曲线重构算法

根据问题一中建立的 Frenet-Cartesian 坐标变换算法，在给定初始角度为45° 的情形下，将测试 1 和测试 2 下的光纤曲线等分为 300 等份，每份近似为一个无限小弧微元，弧端点为O_0，O_1，…，O_n（n=300），根据问题一中的 Frenet-Cartesian 坐标变换算法，通过累加，计算出点O_0，O_1，…，O_n在$X-O_0-Y$坐标系下坐标，将 300 个点同光滑的曲线相连即可实现堆原始曲线的重构，用线性插值和三次样条插值分别重构，重构后的曲线如图 6 所示：

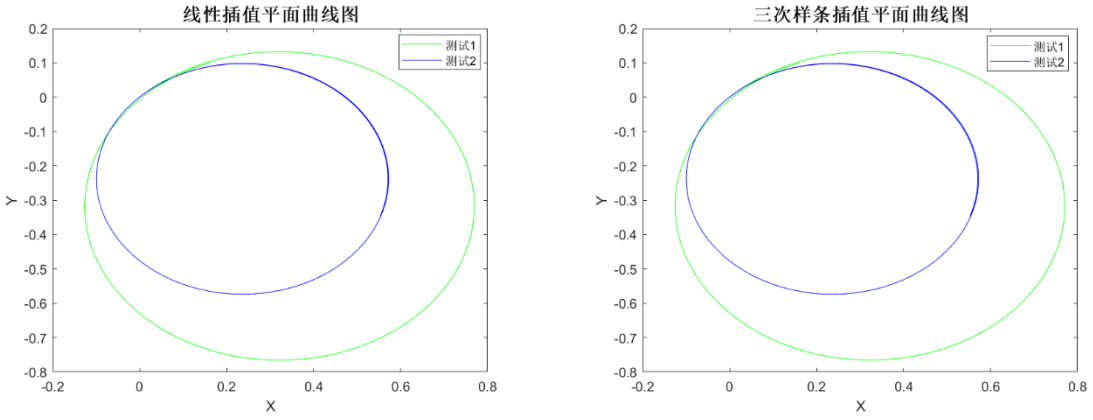

图 6：坐标变换算法重构曲线示意图

6.2.2 基于斜率递推法的曲线重构模型

根据 5.2.1 中建立的斜率递推法，在给定初始角度为45° 的情形下，将测试 1 和测试 2 下的光纤曲线等分为 300 等份，每份近似为一个无限小弧微元，弧端点为O_0，O_1，…，O_n（n=300），根据坐标递推公式(17)可计算出O_0，O_1，…，O_n的在$X-O_0-Y$坐标系下的坐标，进而可以对光纤曲线进行重构，用线性插值和三次样条插值分别重构，重构后的曲线如图 7 所示：

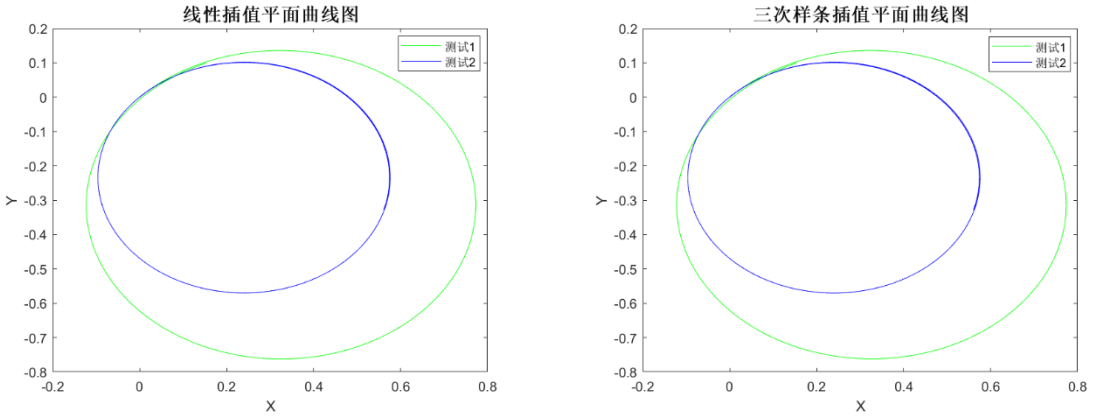

图 7：斜率递推法重构曲线示意图

由图 6 图 7 可以看出，测试二的重构曲线的最右端点的横坐标小于 0.6 米，故问题一种测试二在横坐标x为 0.6 米和 0.7 米处没有曲率。

6.3 重构曲线的特征分析

对于重构曲线的分析，本文选择**几何特性**、**平滑度**、**插值算法种类**几个方面来分析曲线的特点。

·**几何特性：**由图 6 图 7 可以看出重构出的光纤曲线形状与圆形非常近似，可以推测出光纤曲线在三维空间中的形状可能为螺旋线形状，二维重构曲线中的圆形线是三维螺旋形光纤在二维平面的一个投影。

·**平滑度：**根据图 6 图 7 可以看出在两种插值方法下得到的光纤重构曲线比较平滑的，重构曲线通过了给定的数据点，展示出较好的模拟效果，且在这些数据点处的一阶和二阶导数连续，符合理想的光纤模型下光纤曲线是绝对光滑的弧线的特点。

·**插值算法种类：**线性插值算法较为简单，具有易于理解和实现和计算量小的特点，但当数据点之间的变化较大时，线性插值无法很好地拟合数据，可能造成较大的偏差。三次样条插值能够提供非常平滑的曲线，适合用于需要平滑曲线的场合。相比于线性插值，三次样条插值在大多数情况下能够提供更小的误差，并且保证了插值曲线在数据点上的连续性，还保证了一阶和二阶导数的连续性。

我们以斜率递推法模拟测试 1 的曲线为例，将两种插值算法下的曲线展示在一张图中，如图 7 所示，我们可以对比发现，两种插值算法拟合情况均比较理想。分析原因，可能是因为问题一中给出的六个点波长数据求解出的曲率差距值并不大，导致两种方法在拟合曲率的数值上差别不大。

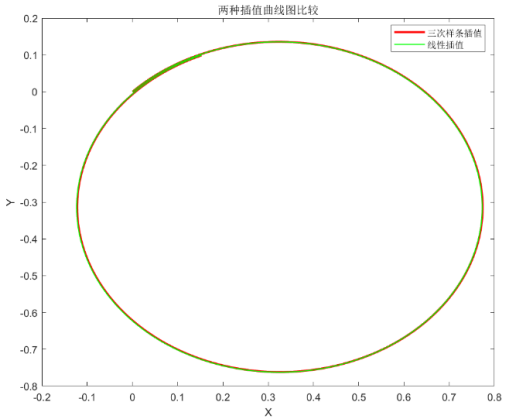

图 8：两种插值算法重构曲线比较图

图 8 是两种插值算法重构曲线比较图，从中可以看出两种插值方法重构出的曲线几乎重合。计算出图 7 中两种插值算法重构出的两条曲线的最左端点和最右端点的分别映射到X轴上的横坐标，以及两条曲线最高点最低点分别映射到Y轴上的纵坐标，分别计算两条曲线最左端点映射到X轴上横坐标的差值和最右端点映射到X轴上横坐标的差值，以及分别计算两条曲线最高点最低点映射到Y轴上纵坐标差值，如表 5 所示。

表 5：四组取点处坐标差值

取点的位置	坐标间差值
最左点	0.001026
最右点	0.000996
最高点	0.000219
最低点	0.000480

由表可知，两条曲线在四组取点处的差值都非常小，可以看出两种插值方法在曲线重构上几乎没有差别，均可以较好的重构曲线。

七、 问题三模型的建立与求解

问题三的分析流程图如下：

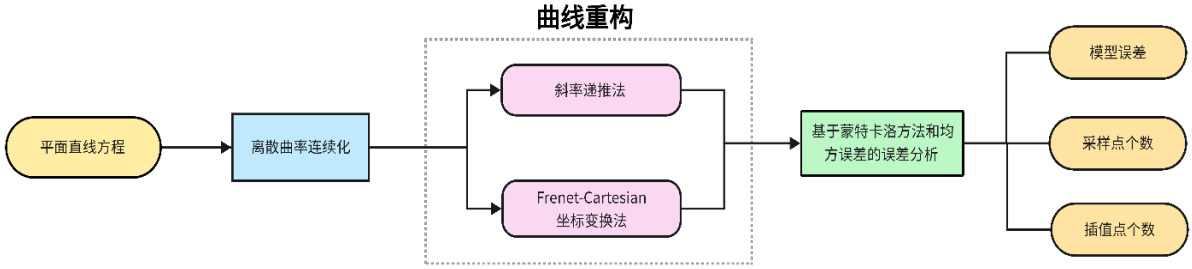

图 9：问题三分析流程图

7.1 基于采样点和插值算法的离散曲率连续化

在前两问的基础，我们仍然使用 Frenet-Cartesian 坐标变换算法来重构平面曲线。题中平面曲线函数解析式为$y=x^3+x$，我们将$0\leqslant x\leqslant 1$上函数的弧等等分成 10 份，共 11 个采样点，从曲线图像中从左至右分别为O_0，O_1，…，O_{10}，其中第i个点对应的横坐标为x_i（$i=0,1,2,\cdots,10$）

对于函数$y=f(x)$，其曲率κ在点x的定义是曲线在该点的切线方向改变速率相对于走过的弧长的比率，可以通过以下公式计算：

$$\kappa=\frac{|f''(x)|}{(1+f'(x)^2)^{3/2}} \tag{19}$$

其中，$f'(x)$是函数在x处的一阶导数，而$f''(x)$是函数在x处的二阶导数。

根据曲率公式，计算 11 个点每个点对应的曲率，得到等间距的弧长——曲率数据对，如下表所示：

表 6：等间距的弧长——曲率数据对记录表

采样点	采样点距离 O_0 的弧长	曲率值	采样点	采样点距离 O_0 的弧长	曲率值
O_0	0.000000	0.000000	O_6	1.361591	0.199761
O_1	0.226932	0.301661	O_7	1.588522	0.157386
O_2	0.453864	0.428077	O_8	1.815454	0.126229
O_3	0.680795	0.404655	O_9	2.042386	0.103084
O_4	0.907727	0.329740	O_{10}	2.269318	0.085601
O_5	1.134659	0.256942			

根据表 6 中采样点的弧长——曲率数据对，我们选择三次样条法进行插值，并以曲线弧长长度为横坐标，插值后曲线每一点对应的曲率值为纵坐标作出以下图像：

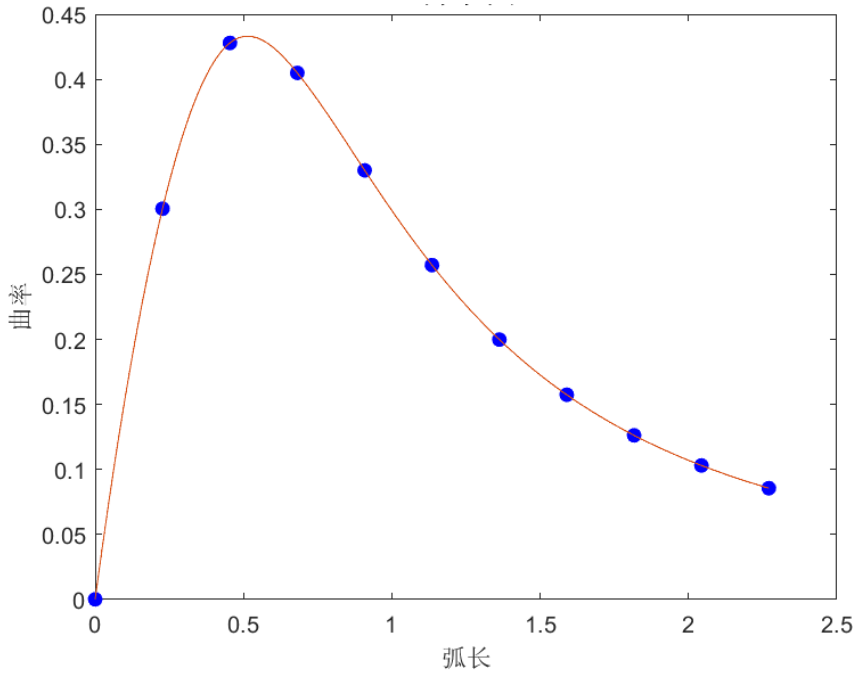

图 10：弧长——曲率数据对插值结果图

7. 2 曲线重构模型的建立和结果

7. 2. 1 基于坐标变换算法的曲线重构模型

本题可以采用 Frenet-Cartesian 坐标变换算法的思想，以曲线 $y=x^3+x$ 的采样点为基础对曲线进行重构。

由计算得曲线 $y=x^3+x$ 在 $X-O_0-Y$ 坐标系原点处初始的 α_i 角为 $45°$，由于问题一中建模中与 O_0 的相切处曲线是沿第一象限内与 X 轴成 $45°$ 的直线向下弯曲的，本题用该方式建模后拟合效果并不理想，因此采用与 O_0 的相切处曲线沿第一象限内与 X 轴成 $45°$ 的直线向上弯曲的方式进行建模。

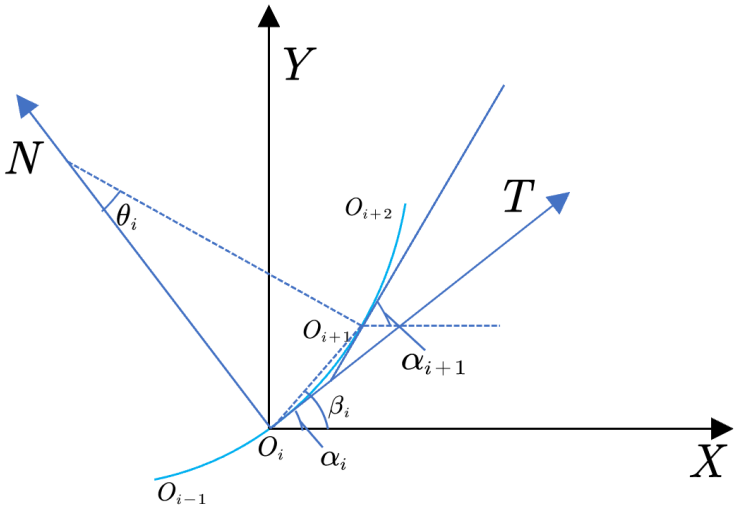

图 11：平面坐标变换法原理图

图 11 为平面坐标变换算法原理图，图中点O_{i+1}在坐标系$X-O_i-Y$中坐标为$(\Delta x_{i+1}\ ,\ \Delta y_{i+1})$，在 Frenet 坐标系$T-O_i-N$中坐标为$(t_{i+1}\ ,\ n_{i+1})$$(i=0,1,2,\cdots,n)$. 点$O_{i+1}$由坐标系$T-O_i-N$向坐标系$X-O_i-Y$中变换步骤如下：

由公式(8)可以得出$(t_{i+1}\ ,\ n_{i+1})$即为$\left(\dfrac{(1-\cos\theta_i)}{\kappa_i}\ ,\ \dfrac{\sin\theta_i}{\kappa_i}\right)$，由公式(3)计算出的$\theta_i$的值，进而计算出点$O_{i+1}$在坐标系$T-O_i-N$下的坐标$(t_{i+1}\ ,\ n_{i+1})$

根据图 11 中的几何关系进行角度变换，可以得出β_i的计算公式如下：

$$\beta_i=\alpha_i+\frac{\pi}{2}-\left(\frac{\pi-\theta_i}{2}\right)=\alpha_i+\frac{\theta_i}{2} \tag{20}$$

其中，α_i为光纤在平面内受力后在初始位置的切线与水平方向的夹角，α_i的初始值$\alpha_1=45°$

由公式(11)点O_{i+1}在坐标系$X-O_i-Y$中的坐标$(\Delta x_{i+1}\ ,\ \Delta y_{i+1})$可以表示为$(l_i\cos\beta_i\ ,\ l_i\sin\beta_i)$，由公式(10)(15)可求得向量$\overrightarrow{O_iO_{i+1}}$的模长为$l_i$和$\beta_i$的值，

过点O_{i+1}作切线，令该切线与水平X轴方向夹角为α_{i+1}，由几何关系可得出α_{i+1}的计算公式如下：

$$\alpha_{i+1}=\alpha_i+\theta_i \tag{21}$$

从公式(16)可以看出α_{i+1}与α_i存在递推关系，又因为已知$\alpha_1=45°$，可以根据该递推式计算出α_1，α_2，$\cdots$，α_n的值。

由上述公式可求出$\Delta x_1,\Delta x_2,...,\Delta x_n$和$\Delta y_1,\Delta y_2,...,\Delta y_n$的值，由公式(13)得$O_i$在$X-O_0-Y$坐标系下的坐标$(x_i\ ,\ y_i)$，$i=1,2,\cdots,n$，即$\left(\sum_{j=1}^{i}\Delta x_j\ ,\ \sum_{j=1}^{i}\Delta x_j\right)$，进而可以对光纤曲线进行重构。

7.2.2 基于斜率递推法的曲线重构模型

本题也可以沿用问题二中的斜率递推法以曲线$y=x^3+x$的采样点为基础对曲线进行重构，采用与O_0的相切处曲线沿第一象限内与X轴成45°的直线向上弯曲的方式进行建模。

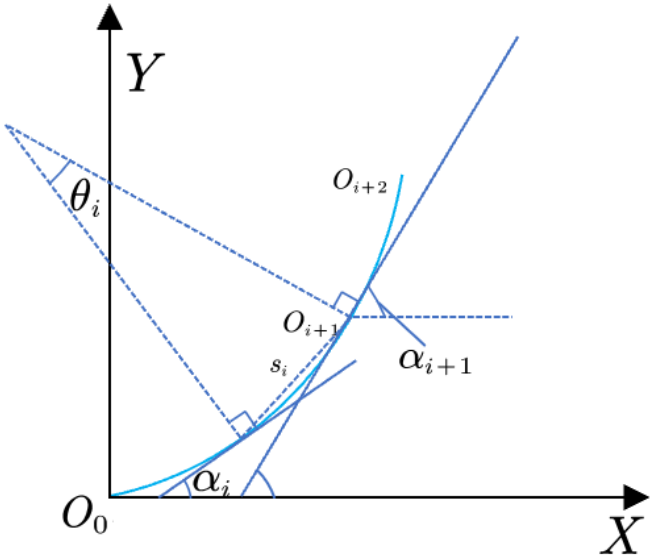

图 12：斜率递推法原理图

如图 12 斜率递推法原理图所示，以光纤初始点为原点建立 Cartesian 坐标系$X-O_0-Y$，初始点关于曲线的切线相对于X轴正方向的倾角为45°，将弯曲后的光纤弧分成n个相等的微元段，每段微元的弧长分别为s_1，s_2，…，s_i，…，s_n满足$s_1=s_2=\cdots=s_n$（$i=0,1,2,\cdots,n$），一共得到$n+1$个端点，记为O_0，O_1，…，O_n. 变量定义均与问题二的 5.2.1 小节中与O_0的相切处曲线沿第一象限内与X轴成45°的直线向下弯曲的方式的斜率递推法的变量定义相同。

由图 8 中几何关系得θ_i的计算公式如下：

$$\theta_i=\alpha_{i+1}-\alpha_i \tag{22}$$

已知初始斜率倾角为45°，由公式(6)得$\theta_i=s_i\kappa_i$，计算α_{i+1}的递推公式如下：

$$\alpha_{i+1}=\alpha_i+s_i\kappa_i \tag{23}$$

根据初始条件$\theta_0=45°$，可计算出各个端点切线的倾角值。由公式(17)可计算出点O_i处的斜率k_i. 已知初始点坐标(x_0,y_0)为$(0,0)$，根据坐标递推公式(17)可计算出O_0，O_1，…，O_n的坐标，进而可以对光纤曲线进行重构。

7.2.3 平面曲线重构结果

取插值点个数为 5000，采样点个数为 11，根据 Frenet-Cartesian 坐标变换算法和斜率递推法进行曲线重构，得到的结果如图 13 所示。

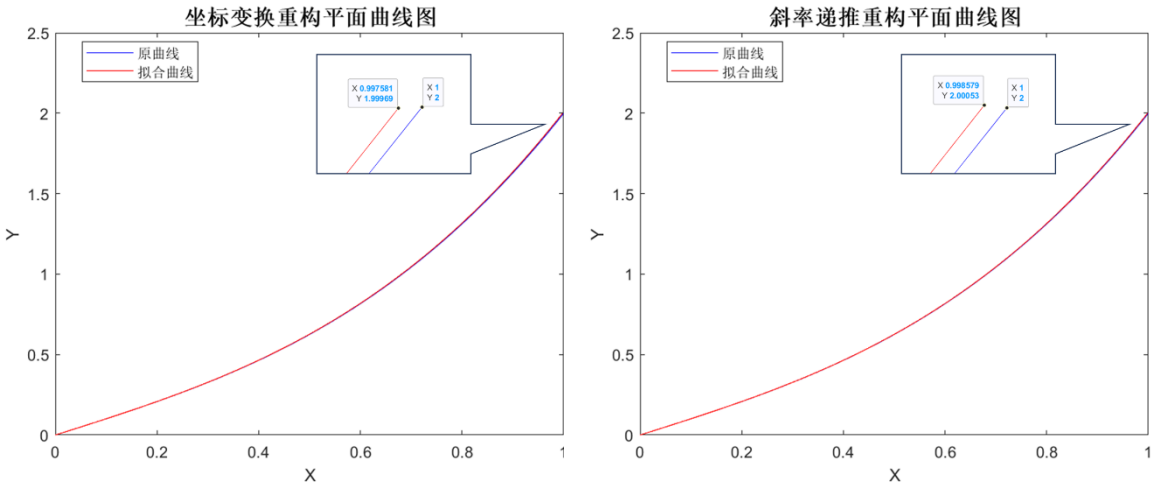

图 13：$y=x^3+x$ 曲线重构示意图

7.3 误差分析

7.3.1 误差分析方法

- **蒙特卡罗(Monte Carlo)积分**

蒙特卡罗(Monte Carlo)积分是一种基于随机抽样的统计方法，即在解决问题时利用产生的大量随机样本，并对这些样本结果进行概率分析，从而来预测结果的方法。

设X是一随机变量，有概率密度函数$f(x)$，对于任意可积函数$g(x)$，随机变量X的函数$g(X)$的数学期望定义公式为：

$$E[g(X)]=\int_{-\infty}^{\infty}g(x)f(x)dx \tag{24}$$

若从X的分布中生成m个独立同分布的随机样本$x_1,x_2,.....,x_m$,那么他们对应的函数值为$g(x_1),g(x_2),.....,g(x_m)$,由强大数定律可知：

$$\frac{1}{m}\sum_{i=1}^{m}g(x_i)\xrightarrow{a.s.}E(g(X))\quad(m\to\infty) \tag{25}$$

因此，我们将$E(g(X))$的无偏估计$\frac{1}{m}\sum_{i=1}^{m}g(x_i)$作为$E(g(X))$的估计量。

对于积分$\int_a^b g(u)du$，需要生成足够多的服从区间(a,b)上的均匀分布随机数$u_1,u_2,...,u_m$，再代入$g(x)$中，求出其均值$\hat{\theta}=\frac{1}{m}\sum_{i=1}^{m}g(u_i)$，就可以得到$\int_a^b g(u)du$的Monte Carlo 积分估计值为$(b-a)\cdot\hat{\theta}$.

- **误差函数与均方误差（*MSE*）**

均方误差（*MSE*）可以用于比较两种算法重构曲线后的误差大小。均方误差积分是误差函数$e(x)$的平方在整个区间上的积分，通常用于计算均方误差的积分形式，公式如下：

$$e(x)=f(x)-g(x) \tag{26}$$

$$MSE = \int_a^b e(x)^2 dx = \int_a^b [f(x) - g(x)]^2 dx \tag{27}$$

7.3.2 误差分析结果

我们根据图 12 两张图象中的函数曲线，利用蒙特卡洛积分法生成 1000 个服从 $U(0,1)$ 的随机数，得到对应的 MSE 分别为：

表 7：两种算法 *MSE* 计算值

算法名称	MSE
斜率递推法	4.1812×10^{-6}
坐标转换法	1.3728×10^{-5}

根据表 7 可知，两种算法均存在误差，但是在插值点个数为 5000，采样点个数为 11 的情况下斜率递推法的 MSE 更小。下文针对斜率递推法重构出的曲线进行误差分析。

根据斜率递推法重构出的曲线图，我们将重构曲线和原曲线出现误差的原因总结为以下三类：**模型误差，插值点个数，采样点个数**。

- **模型误差**

模型误差即模型自身存在的误差，将插值点为 5000，采样点为 11 的情况下画出误差绝对值的变化情况，如图 14 所示。由于斜率递推法在建模时将直线 O_iO_{i+1} 的斜率近似视为点 O_i 的斜率，故系统模型自身存在误差，且在迭代的过程中误差会逐渐积累。模型自身存在的误差是系统误差的，不可避免。

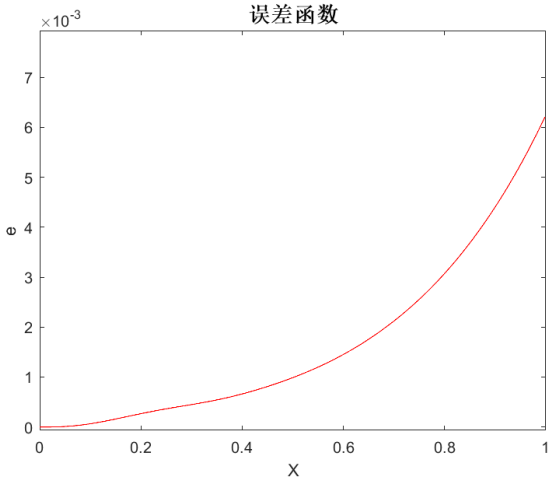

图 14：误差绝对值的变化情况图

- **采样点个数**

采用控制变量的思想，固定插值点个数为 2000 个不变，改变采样个数，观察在采样点个数变化的情况下模型 MSE 的变化，得到的表格如下：

表 8：不同采样点对应 *MSE* 值

采样点个数	MSE	采样点个数	MSE
4	7.192×10^{-3}	10	1.086×10^{-5}
6	3.175×10^{-4}	12	6.089×10^{-7}
8	9.935×10^{-5}	14	1.496×10^{-7}

观察可知，随着采样个数的增加，MSE 呈现逐渐减小的趋势，不难看出，增加采样点，就会增加原曲线的信息，重构的曲线就会更接近于原曲线。

• 插值点个数

采用控制变量的思想，固定采样点个数为 10 个不变，观察在插值点个数变化的情况下模型MSE的变化，得到表格如下：

表 9：不同插值点对应 *MSE* 值

插值点个数	MSE	插值点个数	MSE
100	3.298×10^{-4}	1500	8.588×10^{-6}
300	1.197×10^{-5}	3000	1.315×10^{-5}
500	1.499×10^{-6}	4000	1.444×10^{-5}
1000	5.229×10^{-6}		

观察可知，随着插值点数的增加，MSE 呈现先减小再增大的趋势，且在插值点为 500 的附近 MSE 达到了最小，分析原因如下：

在插值点较少时，模型可能无法捕捉到数据的所有特征，无法精确匹配原函数的所有局部变化，插值误差较大，此时 MSE 相对较大。

随着插值点的增加，模型开始更好地捕捉数据的特征，一个合理的复杂度范围内，插值函数能够更精确地逼近原函数，拟合度提高，因此 MSE 减小。

当插值点继续增加到一定程度后，模型可能会开始捕捉到数据中的噪声，导致过拟合，此时 MSE 会增大。

八、 模型的评价与总结

8.1 模型的优点

1 使用 Frenet-Cartesian 坐标变换算法和斜率递推法进行曲线重构，这些算法在数学上很先进严谨，并且能够处理复杂的几何问题。

2.通过插值算法对离散曲率数据进行连续化处理，使得模型能够生成更加平滑和连续的曲线。

3. 本文可视化展示重构曲线和原曲线，直观地展示了模型的有效性。

4. 采用蒙特卡洛方法和均方误差(MSE)进行误差分析，为模型提供了一种量化误差的手段。

5. 模型不仅适用于工业和医疗领域，还能扩展到微型机器人技术、精密仪器设计等高精度监测场景。

8.2 模型的缺点

1.Frenet-Cartesian 坐标变换和斜率递推法可能涉及复杂的数学运算，这可能导致计算成本较高。

2. 模型可能存在由于系统本身的误差，如将连续曲率近似为离散点的曲率，这可能导致误差逐渐累积。

8.3 模型的总结

在本研究中，本文成功建立了基于光纤传感器技术的平面曲线重构模型。通过斜率递推法和 Frenet-Cartesian 坐标变换，可以有效地从离散的曲率数据重构出连续的平面曲线。此外，采用了线性插值和三次样条插值算法，对曲率数据进行连续化处理，使得重构的曲线更加平滑和符合实际物理特性。

本文还对重构曲线进行了特征分析，包括几何特性、平滑度和插值算法种类的分析，以及通过蒙特卡洛方法和均方误差(MSE)对模型的误差进行了量化分析。这些分析不仅验证了模型的有效性，还为模型的改进提供了方向。

尽管模型在处理复杂几何问题上表现出色，但也存在计算成本和系统误差等潜在问题。未来的工作将集中在优化算法以降低计算成本，以及改进模型以减少系统误差，从而提高模型的实用性和准确性。

九、 参考文献

[1] 程文胜.基于超弱光纤光栅的曲线重构方法研究[D].三峡大学, 2021.
[2] 章亚男,肖海,沈林勇. 用于光纤光栅曲线重建算法的坐标点拟合[J]. 光学精密工程, 2016, 24(09):2149-2157.
[3] 肖海,章亚男,沈林勇,等.光纤光栅曲线重建算法中的曲率连续化研究[J].仪器仪表学报,2016,37(05):993-999.
[4]Rizzo, M.L. (2019). Statistical Computing with R, Second Edition (2nd ed.). Chapman and Hall/CRC.

附录

附录 1
介绍：支撑材料的文件列表
1. Matlab 程序源代码及变量.mat 文件 **2. 参考文献** **3. 论文 Word 版本**

附录 2：q1.m	
介绍：问题一代码	编程语言：MATLAB

```
% 光纤传感器的原始数据和常数
Init_WL1 = [1529, 1529, 1529, 1529, 1529, 1529];
Test_WL1 = [1529.808, 1529.807, 1529.813, 1529.812, 1529.814, 1529.809];
Init_WL2 = [1540, 1540, 1540, 1540, 1540, 1540];
Test_WL2 = [1541.095, 1541.092, 1541.090, 1541.093, 1541.094, 1541.091];

% 常数 c
c = 4200;

% 计算曲率
k1 = c * ((Test_WL1 - Init_WL1) ./ Init_WL1);
k2 = c * ((Test_WL2 - Init_WL2) ./ Init_WL2);

% 传感器位置（米）
sensor_positions = [0, 0.6, 1.2, 1.8, 2.4, 3.0];

% 指定的横坐标点和更细的插值点
x_points = [0.3, 0.4, 0.5, 0.6, 0.7];
x_fine = linspace(0, 3.0, 301);
x_fine_sec=linspace(0, 3.0, 3001);

% 使用三次样条插值

K1_spline = spline(sensor_positions, k1, x_fine);
K1_spline_sec = spline(sensor_positions, k1, x_fine_sec);
K2_spline = spline(sensor_positions, k2, x_fine);
K2_spline_sec = spline(sensor_positions, k2, x_fine_sec);

% 使用线性插值
K1_linear = interp1(sensor_positions, k1, x_fine, 'linear');
K1_linear_sec=interp1(sensor_positions, k1, x_fine_sec, 'linear');
K2_linear = interp1(sensor_positions, k2, x_fine, 'linear');
```

```
K2_linear_sec=interp1(sensor_positions, k2, x_fine_sec, 'linear');

% 绘制曲率折线图和曲率插值函数图
figure;
subplot(2,1,1); % 第一个子图
plot(sensor_positions, k1, 'bo', 'LineWidth', 2, 'MarkerFaceColor', 'b'); % 只绘制点
hold on;
plot(x_fine, K1_spline, 'r-', 'LineWidth', 2);
plot(x_fine, K1_linear, 'g--', 'LineWidth', 2);
title('测试 1 曲率');
% xlabel('横坐标 x');
ylabel('曲率');
legend('实测值曲率点', '三次样条插值曲率', '线性插值曲率', 'Location', 'best');
% grid on;
xticks(sensor_positions); % 设置 X 轴的刻度
% 设置 X 轴的标签
xticklabels(arrayfun(@num2str, sensor_positions, 'UniformOutput', false));

subplot(2,1,2); % 第二个子图
plot(sensor_positions, k2, 'bo', 'LineWidth', 2, 'MarkerFaceColor', 'b'); % 只绘制点
hold on;
plot(x_fine, K2_spline, 'r-', 'LineWidth', 2);
plot(x_fine, K2_linear, 'g--', 'LineWidth', 2);
title('测试 2 曲率');
% xlabel('横坐标 x');
ylabel('曲率');
legend('实测值曲率点', '三次样条插值曲率', '线性插值曲率', 'Location', 'best');
% grid on;
xticks(sensor_positions); % 设置 X 轴的刻度
% 设置 X 轴的标签
xticklabels(arrayfun(@num2str, sensor_positions, 'UniformOutput', false));


[x1_sum, y1_sum] = Coordinate_Converter(K1_spline_sec);
[x2_sum, y2_sum] = Coordinate_Converter(K2_spline_sec);
K1_s_values=findKValuesFromXSum(x1_sum, K1_spline_sec);
K2_s_values=findKValuesFromXSum(x2_sum, K2_spline_sec);

[x1_L_sum, y1_L_sum] = Coordinate_Converter(K1_linear_sec);
[x2_L_sum, y2_L_sum] = Coordinate_Converter(K2_linear_sec);
K1_L_values=findKValuesFromXSum(x1_L_sum, K1_linear_sec);
K2_L_values=findKValuesFromXSum(x2_L_sum, K2_linear_sec);

%计算曲率[0.3, 0.4, 0.5, 0.6, 0.7]
```

```
function K_values = findKValuesFromXSum(x_sum, K)
    % 需要查找的累加和目标值
    targets = [0.3, 0.4, 0.5, 0.6, 0.7];
    K_values = zeros(size(targets));  % 存储对应的 K 值

    for j = 1:length(targets)
        idx = find(x_sum > targets(j), 1);  % 找到第一个大于目标值的索引
        if ~isempty(idx) && idx > 1
            % 返回 K 值，MATLAB 索引从 1 开始，返回 i-1 的 K 值
            K_values(j) = (K(idx - 1)+K(idx))/2;
        end
    end
end
```

附录 3：q2.m	
介绍：问题 2 代码	编程语言：MATLAB

```
load K1_spline;
load K2_spline;
load K1_linear;
load K2_linear;

%第一种坐标方法
[x1_spline_sum, y1_spline_sum]=Coordinate_Converter(K1_spline);
[x2_spline_sum, y2_spline_sum]=Coordinate_Converter(K2_spline);

[x1_linear_sum, y1_linear_sum]=Coordinate_Converter(K1_linear);
[x2_linear_sum, y2_linear_sum]=Coordinate_Converter(K2_linear);

%第二种斜率递推法
[x1_s_recur, y1_s_recur]=Slope_Recur(K1_spline);
[x2_s_recur, y2_s_recur]=Slope_Recur(K2_spline);

[x1_l_recur, y1_l_recur]=Slope_Recur(K1_linear);
[x2_l_recur, y2_l_recur]=Slope_Recur(K2_linear);

%坐标变换图
figure(1);
plot(x1_spline_sum,y1_spline_sum,'g-');
hold on;
plot(x2_spline_sum,y2_spline_sum,'b-');
[t,s]=title('三次样条插值平面曲线图','FontWeight','bold');
t.FontSize = 14;xlabel('X');ylabel('Y');
```

```
legend('测试 1', '测试 2', 'Location', 'best');
figure(2);
plot(x1_linear_sum,y1_linear_sum,'g-');
hold on;
plot(x2_linear_sum,y2_linear_sum,'b-');
[t,s]=title('线性插值平面曲线图','FontWeight','bold');
t.FontSize = 14;xlabel('X');ylabel('Y');
legend('测试 1', '测试 2', 'Location', 'best');

%斜率递推法图
figure(3);
plot(x1_s_recur,y1_s_recur);
hold on;
plot(x2_s_recur,y2_s_recur,'b-');
[t,s]=title('三次样条插值平面曲线图','FontWeight','bold');
t.FontSize = 14;xlabel('X');ylabel('Y');
legend('测试 1', '测试 2', 'Location', 'best');
figure(4);
plot(x1_l_recur,y1_l_recur,'g-');
hold on;
plot(x2_l_recur,y2_l_recur,'b-');
[t,s]=title('线性插值平面曲线图','FontWeight','bold');
t.FontSize = 14;xlabel('X');ylabel('Y');
legend('测试 1', '测试 2', 'Location', 'best');


%误差分析
[maxx_s, maxIndex] = max(x1_s_recur);maxy_s=y1_s_recur(maxIndex);
[minx_s, minIndex] = min(x1_s_recur);miny_s=y1_s_recur(minIndex);
[maxx_l, maxIndex] = max(x1_l_recur);maxy_s=y1_l_recur(maxIndex);
[minx_l, minIndex] = min(x1_l_recur);miny_s=y1_l_recur(minIndex);
e_L=abs(minx_s-minx_l);
e_R=abs(maxx_s-maxx_l);

% %  比较插值曲线
% figure(5);
% plot(x1_s_recur,y1_s_recur,'r-', 'LineWidth', 2);
% hold on;
% plot(x1_l_recur,y1_l_recur,'g-', 'LineWidth', 1);
% title('两种插值曲线图比较');
% xlabel('X');
% ylabel('Y');
% legend('三次样条插值', '线性插值', 'Location', 'best');
```

附录 4： Coordinate_Converter.m

介绍：坐标转换函数 | 编程语言：MATLAB

```
function [x_sum, y_sum] = Coordinate_Converter(K)
    format long;
    s = 3/(length(K)-1);
    theta = s .* K;

    zx = (1 - cos(theta)) ./ K;
    zy = sin(theta) ./ K;
    L1 = sqrt(zx .* zx + zy .* zy);

    alpha = zeros(1, length(theta));
    beta = zeros(1, length(theta));
    x1 = zeros(1, length(theta));
    y1 = zeros(1, length(theta));
    alpha(1) = pi / 4;

    for i = 1:length(theta)
        if i < length(theta)    % To avoid index out of bounds error
            alpha(i + 1) = alpha(i) - theta(i);
        end
        beta(i) = alpha(i) - theta(i) / 2;
        x1(i) = L1(i) * cos(beta(i));
        y1(i) = L1(i) * sin(beta(i));
    end

    x_sum = cumsum(x1);
    y_sum = cumsum(y1);
end
```

附录 5：Slope_Recur.m

介绍：斜率递推函数 | 编程语言：MATLAB

```
function [x1, y1] = Slope_Recur(K)

    k_fine = zeros(0,301);
    k_fine(1) = 1;

    x1=zeros(1,length(k_fine));
    y1=zeros(1,length(k_fine));
    x1(1)=0;
    y1(1)=0;
```

```
        theta = zeros(0,length(k_fine));

        theta(1) = pi / 4;
        s = 0.01;
        for i=1:300
            theta(i+1)=theta(i)-K(i)*s;
            k_fine(i+1) = tan(theta(i+1));

            x1(i+1)=x1(i) + cos(theta(i))*s;
            y1(i+1)=y1(i) + sin(theta(i))*s;
        end

    end
```

附录 6：q3_cc.m

介绍：问题 3 坐标转换代码	编程语言：MATLAB

```
% 定义函数
f = @(x) x.^3 + x;
df = @(x) 3.*x.^2 + 1;
ddf = @(x) 6.*x;

% 计算弧长的积分函数
s = @(x) sqrt(1 + df(x).^2);

% 计算 0 到 1 区间的弧长
arc_length = integral(s, 0.001, 1);

% 设定每段弧长
segment_length = arc_length / 10;
x_values = zeros(1, 11);
x_values(1) = 0.001; % 起始点

% 计算每个分段的 x 值
options = optimset('TolX',1e-12); % 设置解算器的精度
arc_lengths=zeros(1,11);
arc_lengths(1)=0;
for i = 1:10
    func = @(x) integral(s, 0.001, x) - i * segment_length;
    x_values(i+1) = fzero(func, [x_values(i), 1], options);
    arc_lengths(i+1) = arc_lengths(i)+ arc_length / 10;
end
```

```
% 计算曲率
curvatures = abs(ddf(x_values)) ./ (1 + df(x_values).^2).^(3/2);

x_fine = linspace(0,arc_length, 5001);
s_position=linspace(0,arc_length, 11);
K_10 = spline(s_position, curvatures, x_fine);

%重构平面曲线
s=arc_length / 5000;
format long;
%K1_spline=curvatures;
K1_spline=K_10;
theta1=s.*K1_spline;

zx1=(1-cos(theta1))./K1_spline;
zy1=sin(theta1)./K1_spline;
L1=sqrt(zx1.*zx1+zy1.*zy1);

aplha=zeros(1,length(theta1));
beta=zeros(1,length(theta1));
x1=zeros(1,length(theta1));
y1=zeros(1,length(theta1));
aplha(1)= pi / 4;
for i=1:length(theta1)
    aplha(i+1)=theta1(i)+aplha(i);
    beta(i)=aplha(i)+theta1(i)/2;
    x1(i)=L1(i)*cos(beta(i));
    y1(i)=L1(i)*sin(beta(i));
end
x1_sum=cumsum(x1);
y1_sum=cumsum(y1);

figure;
init_x=linspace(0,1,100);
init_y=init_x.*init_x.*init_x+init_x;
plot(init_x,init_y);
hold on;
plot(x1_sum,y1_sum);
[t,s]=title('重构平面曲线图','FontWeight','bold');t.FontSize =
14;xlabel('X');ylabel('Y');
legend('原曲线', '拟合曲线', 'Location', 'best');

% 随机生成 1000 个点
```

```
x_random = rand(1000, 1);
y_true = f(x_random);   % 计算原函数的值

% 找到每个随机点在 x1_sum 中最近的索引
[idx, indices] = knnsearch(x1_sum', x_random);

% 使用这些索引从 y1_sum 中获取重构曲线的 y 值
y_reconstructed = y1_sum(idx);

% 计算误差
errors = y_true - y_reconstructed';
errors=errors.*errors;

% 计算 MSE
mean_error = mean(errors);
```

附录 7：q3_recur.m

介绍：问题 3 斜率递推代码	编程语言：MATLAB

```
% 定义函数
f = @(x) x.^3 + x;
df = @(x) 3.*x.^2 + 1;
ddf = @(x) 6.*x;

% 计算弧长的积分函数
s = @(x) sqrt(1 + df(x).^2);

% 计算 0 到 1 区间的弧长
arc_length = integral(s, 0.0001, 1);

% 设定每段弧长
segment_length = arc_length / 10;
x_values = zeros(1, 11);
x_values(1) = 0.0001; % 起始点

% 计算每个分段的 x 值
options = optimset('TolX',1e-12); % 设置解算器的精度
arc_lengths=zeros(1,11);
arc_lengths(1)=0;
for i = 1:10
    func = @(x) integral(s, 0.0001, x) - i * segment_length;
    x_values(i+1) = fzero(func, [x_values(i), 1], options);
    arc_lengths(i+1) = arc_lengths(i)+ arc_length / 10;
```

```
    end

    % 计算曲率
    curvatures = abs(ddf(x_values)) ./ (1 + df(x_values).^2).^(3/2);

    x_fine = linspace(0,arc_length,5001);
    s_position=linspace(0,arc_length, 11);
    K_10 = spline(s_position, curvatures, x_fine);

    %重构平面曲线
    s=arc_length / 5000;
    format long;
    %K1_spline=curvatures;
    K1_spline=K_10;
    dtheta=s.*K1_spline;

    k_fine = zeros(0,5001);
    k_fine(1) = 1;

    x1=zeros(1,length(K1_spline));
    y1=zeros(1,length(K1_spline));
    x1(1)=0;
    y1(1)=0;
    theta = zeros(0,length(K1_spline));
    theta(1) = pi / 4;


    for i=1:5000
        theta(i+1)=theta(i)+ dtheta(i);
        k_fine(i+1) = tan(theta(i+1));

        x1(i+1)=x1(i) + cos(theta(i))*s;
        y1(i+1)=y1(i) + sin(theta(i))*s;
    end

    figure;
    init_x=linspace(0,1,100);
    init_y=init_x.*init_x.*init_x+init_x;
    plot(init_x,init_y);
    hold on;
    plot(x1,y1);
    [t,s]=title('重构平面曲线图','FontWeight','bold');t.FontSize =
14;xlabel('X');ylabel('Y');
    legend('原曲线', '拟合曲线', 'Location', 'best');
```

```
% 随机生成 1000 个点
x_random = rand(1000, 1);
y_true = f(x_random);   % 计算原函数的值

% 找到每个随机点在 x1 中最近的索引
[idx, indices] = knnsearch(x1', x_random);

% 使用这些索引从 y1 中获取重构曲线的 y 值
y_reconstructed = y1(idx);

% 计算误差
errors = y_true - y_reconstructed';
errors=errors.*errors;

% 计算 MSE
mean_error = mean(errors);
```