中国研究生创新实践系列大赛
“华为杯”第十八届中国研究生数学建模竞赛

学　　校		清华大学
参赛队号		21100030005
队员姓名	1.	杨朔
	2.	郑昕然
	3.	阮俊翔

中国研究生创新实践系列大赛
“华为杯”第十八届中国研究生
数学建模竞赛

题　　目　信号干扰下的 UWB 精确定位模型

摘　　要：

超宽带无线通信技术（Ultra-Wide band，UWB）是一种无需任何载波，通过发送纳秒级脉冲而完成数据传输的短距离范围内无线通信技术。基于 UWB 的定位技术具备实时的室内外精确跟踪能力，定位精度高，可达到厘米级甚至毫米级定位。由于室内环境复杂多变，UWB 通信信号极易受到遮挡，会导致定位精度的急剧下降。本文对 UWB 定位系统展开探索，实现了信号干扰下的精确定位。

数据预处理（问题一），根据问题的要求，首先对采集到的数据进行了整理，使用易读易操作的矩阵形式存储；然后对正常数据、异常数据以及理论数据进行了联合可视化分析，以期能够掌握数据的分布规律，便于下一步的操作；最后在数据清洗阶段，我们分别对缺失值、重复值以及异常值进行了检测，其中针对异常值的检测，我们提出了基于规则的异常值检测、单维异常值检测和多维异常值检测三种方案，从数据预处理的结果来看，有效去除了采集数据中的无用值，完成了数据清洗的工作。

定位模型的建立与求解（问题二），根据问题要求，首先利于锚点测距的测量值与真实值建立线性回归模型消除数据随机误差。对于正常数据，将坐标估计场景建立为非线性规划模型，采用 Jacobi 迭代法，牛顿法，以及基于加权最小二乘的 Chan 算法，并将其迁移至三维空间定位中用于靶点定位。对于异常数据，首先通过分析异常数据，通过锚点被遮挡的特征建立基于贪心误差搜索的异常数据补偿模型，获得使定位误差最小的锚点测距最优补偿值。其次通过将非视距误差引入卡尔曼滤波计算过程中，得到有偏的卡尔曼滤波，用于消除非视距误差。对异常数据进行补偿或消除后再使用正常数据坐标估计模型计算。分别对正常和异常模型在 1 维，2 维，3 维，利用其均方根误差，误差分布直方图及误差值波动曲线进行精度分析。

定位模型在不同场景下的应用（问题三），根据问题要求，我们修改了问题二模型中的锚点坐标。由于在实验环境 2 中未提供真实坐标用于锚点测量误差校准，因此对附件 3 前五个（标记为正常数据）采用问题二提出的三维 Chan 模型，对后五个（标记为异常数据）首先采用有偏卡尔曼滤波进行异常消除后采用三维 Chan 模型进行坐标估计。

分类模型的建立与求解（问题四），这是一个异常检测的分类问题，首先我们充分挖掘数据中存在的模式和特征，建立了关于非视距误差的数学模型，提出了一种基于室内环境误差建模的分类算法，通过误差对异常数据进行识别。此外，本文评估了 KNN、支持向量机、决策树、多层感知机、随机森林等机器学习算法以及 stacking，blending，boosting 等集成学习方法在该分类问题中的性能。实验结果显示，采用最佳的集成 stacking 的分类方法可以在问题一中的数据集达到 88% 的准确率。将该最佳结果的分类器用于问题五中异常数据的判断。

运动轨迹定位（问题五），根据问题要求，由于实验环境不变，因此我们首先采用问题二中测量误差校准方法对原始数据进行校正。之后利用问题四中的分类模型对附件 5 提供的数据进行分类，为数据添加正常/异常标签。分别针对不包含非视距误差的正常数据和非视距误差下的异常数据利用问题二中的模型进行坐标估计。为更精准的绘制运动轨迹，我们将靶点移动的时间维度引入模型当中，对靶点运动这一非线性问题，建立了扩展卡尔曼滤波模型，用于进行靶点运动轨迹坐标的校正。由实验验证可得，采用扩展卡尔曼滤波后，相比于直接采用问题二中的模型或采用线性平滑的方式，我们提出的模型能够更加精确且平滑的拟合物体运动的轨迹。

关键字：UWB　非视距误差　误差建模　卡尔曼滤波　精准定位

目录

1 问题背景与问题重述 5

1.1 问题背景 5

1.2 问题重述 5

2 模型假设与符号说明 7

2.1 模型假设 7

2.2 符号说明 7

3 问题一——数据预处理 7

3.1 问题一分析 7

3.2 数据整理 8

3.3 数据可视化 10

3.4 数据清洗 13

3.5 数据预处理结果 15

4 问题二——定位模型的建立与求解 17

4.1 问题二分析 17

4.2 测距校正 18

4.3 针对正常数据的空间定位模型 18

4.3.1 基于加权最小二乘法的三维 Chan 定位模型 19

4.3.2 基于梯度下降的定位模型 21

4.3.3 基于牛顿法的定位模型 21

4.4 针对异常数据的空间定位模型 22

4.4.1 基于贪心误差搜索的异常数据校正 22

4.4.2 基于有偏卡尔曼滤波的异常数据校正 23

4.5 基于卡尔曼滤波的 NLOS/LOS 定位模型 24

4.6 模型精度检验 24

4.6.1 正常数据下的模型精度检验结果 24

4.6.2 异常数据下的模型精度检验结果 27

5 问题三——定位模型在不同场景下的应用 31

5.1 问题三分析 31
5.2 问题三结果 31
6 问题四——分类模型的建立与求解 32
6.1 问题四分析 32
6.2 基于误差建模的分类算法 33
6.3 基于机器学习的分类算法 35
6.4 基于集成学习的分类算法 39
6.5 算法性能对比与结果分析 42
7 问题五——运动轨迹定位 45
7.1 问题五分析 45
7.2 LOS/NLOS 环境中的移动目标定位模型 46
7.3 基于扩展卡尔曼滤波的轨迹校准 46
7.3.1 原始数据测量误差校正 46
7.3.2 等采样间隔线性插值 47
7.3.3 数据分类及定位 47
7.4 结果分析 47
参考文献 50
附录 A 问题结果 51
A.1 问题一结果 51
A.2 问题二结果 58
A.3 问题三结果 59
A.4 问题四结果 59
A.5 问题五结果 60
附录 B Python 源程序 61
B.1 第 1 问程序 61
B.2 第 4 问程序 69
附录 C Matlab 源程序 73
C.1 第 2 问程序 73
C.2 第 5 问程序 75

1 问题背景与问题重述

1.1 问题背景

超宽带无线通信技术（Ultra-Wide band，UWB）是一种无需任何载波，通过发送纳秒级脉冲而完成数据传输的短距离范围内无线通信技术，又称之为脉冲无线电技术。UWB 技术最早起源于 19 世纪的电火花隙传输实验 [1]。UWB 定位系统的频谱较宽，发出的脉冲信号功率很低，传输速率通常在 GB/s 级别，具有功耗极低，定位性能优异，抗干扰能力强，信号穿透能力好以及安全等优点。

因其独有的特点，使 UWB 系统在军事、物联网等各个领域都有着广阔的应用。其中，基于 UWB 的定位技术具备实时的室内外精确跟踪能力，定位精度高，可达到厘米级甚至毫米级定位。UWB 在室内精确的定位将会对卫星导航起到一个极好的补充作用，可在军事及民用领域有广泛应用，比如：电力、医疗、化工行业、隧道施工、危险区域管控等。

常见的 UWB 定位算法有基于双向飞行时间（two way time of flight，TOF）的定位算法、基于到达时间 (time of arrival，TOA) 的定位算法、基于到达时间差 (time difference of arrival，TDOA) 的定位算法、基于到达角度 (angel of arrival，AOA) 的定位算法和基于接收信号强度 (received signal strengths，RSS) 的定位算法，AOA 和 RSS 算法使用较少，因为其很容易受到噪声和环境干扰导致定位效果较差。TOA 算法是基于同步时间戳来保证测量精度的，需要将靶点和锚点的时钟进行同步，会消耗更多的资源，也可以通过测量 TDOA 来代替 T0A 的计算，但是依然存在时间同步的误差问题。为了避免时钟同步造成的误差和节约资源，当前大部分的 UWB 定位系统采用 TOF 测距技术，它属于双向测距技术，其通过计算信号在两个模块的飞行时间，再乘以光速求出两个模块之间的距离，即使存在时钟漂移、天线、数据传输线等影响导致的误差，TOF 的精度已经达到了定位系统的容错度。

在高精度定位应用中, 需要对 UWB 设备进行校正。环境因素对 UWB 定位系统的影响很大，一种典型的干扰场景是非视距 (non-line of sight，NLOS) 环境，在 NLOS 场景下，UWB 基站发出的信号会被障碍物遮挡，即使信号穿透遮挡也会造成传播的时间差发生变化，影响测量的精度。除此以外，复杂的室内环境同样存在着多径效应，如何抑制多径效应来缓解干扰也是 UWB 系统解决方案的一个重点。因此，信号干扰下的超宽带（UWB）精确定位问题成为亟待解决的问题。

1.2 问题重述

基于上述研究研究背景，题目提供了在有干扰和无干扰条件下采集的 UWB 数据集，围绕信号干扰下的超宽带（UWB）精确定位问题，本文需要解决以下五个问题：

问题一：数据预处理（清洗）

根据在室内定位有干扰和无干扰的实际场景中，利用 UWB 的定位技术（TOF）采集到锚点（anchor）与靶点（Tag）之间的原始距离数据，进行数据清洗工作。需要从原始数

据文件中抓取靶点到四个锚点的距离数值，将其整合成二维表（矩阵）的形式。由于原始数据中包括一定程度的不良数据，例如部分时刻的测距数据异常，超出所设定的合理范围，或是由于环境的干扰导致存在数值缺失的情况，因此需要通过一定的数据筛选准则和相应的数据预处理方法，整定数据并筛选样本，为后续建模任务奠定基础。

问题二：定位模型的建立与求解

在基于 UWB 的定位场景下，需要根据靶点到锚点的距离数据，建立何时的数学模型，实现三维空间内靶点的精准定位。本问题中使用的数据来自实验场景一，靶点范围为 5000mm*5000mm*3000mm，四个锚点 A0、A1、A2、A3 的位置（单位：mm）分别为（0，0，1300）、（5000，0，1700）、（0，5000，1700）和（5000，5000，1300），结合题目要求，需要针对“正常数据”和“异常数据”在该实验场景下分别设计合适的模型，并说明所设计的定位模型的有效性。为了保证模型的可迁移性，需要在模型中体现场景信息，并且同时给出定位模型的 3 维 x, y, z 精度、2 维 x, y 精度以及 1 维的各自精度。

问题三：定位模型在不同场景下的应用

在问题二中针对特定的实验场景建立了定位模型，但是定位模型应该能够在不同的实际场景下使用，所以本题使用的数据来自在实验场景二，靶点范围为 5000mm*3000mm*3000mm，四个锚点 A0、A1、A2、A3 的位置（单位：mm）分别为（0，0，1200）、（5000，0，1600）、（0，3000，1600），（5000，3000，1200）。需要完成定位模型的迁移，实现在场景二在有干扰和无干扰条件下的精确定位，给出相应的三维坐标。

问题四：分类模型的建立与求解

上述的定位模型是在已知信号有无干扰的条件下建立的，但是在实际环境中，UWB 的数据采集过程并不知道信号是否收到干扰，所以判断信号有无干扰是 UWB 能否精准定位的关键。因为本题需要利用数据预处理后的数据，建立分类模型（算法），识别 UWB 的信号采集过程中是否受到了干扰。要求说明分类模型（算法）的有效性，并且对附件 4 中的数据进行分类识别。

问题五：运动轨迹的定位

基于 UWB 技术的室内定位系统不仅能够提供定位功能，还能进行历史轨迹回放，这大大提高了实用性。本题要求利用附件五中所提供的一段时间内连续采集的多组数据，通过静态定位模型加靶点自身运动规律的方法，计算出靶点的动态轨迹，并将靶点的运动轨迹图可视化。另外，数据在采集时会出现随机的信号干扰，因此为了实现运动轨迹的精确定位，需要识别哪些数据受到了干扰，进行相应的去扰处理。

2 模型假设与符号说明

2.1 模型假设

1. UWB 设备在无干扰条件下的测量数据中的噪声符合正态分布；
2. UWB 在无干扰条件下未受到障碍物的遮挡；

2.2 符号说明

符号	意义
i	靶点的 ID，i$\in [1,324]$
j	锚点的 ID，j$\in [0,3]$
$d_{i,j}$	靶点 i 到锚点 A_j 的测量距离
$d'_{i,j}$	靶点 i 到锚点 A_j 的实际距离
D	一段时间测量得到的数据序列
V	离散度，用于评估数据受到干扰的程度

3 问题一——数据预处理

3.1 问题一分析

在 5000mm*5000mm*3000mm 的测试场地中，UWB 锚点（anchor）放置在 4 个角落 A0，A1，A2，A3，锚点向所有方向发送信号。Tag 是 UWB 标签（靶点），即需要定位的目标（只在测试环境范围内）。Tag 接收到 4 个 UWB 锚点（anchor）的信号（无论信号是否干扰，Tag 一般都可以接收到信号），利用 TOF 技术，分别解算出对应的 4 个距离数据。本任务所用到的数据即为以上实验条件下，在 324 个不同位置，在有信号干扰和无信号干扰下的 UWB 数据。其中干扰因素为锚点与靶点之间有遮挡。

UWB 定位模型的构建需要符合实际情况且误差控制在一定范围内的、相对准确的相关距离测量数据作为支撑。但是在实际的实验环境中，在进行数据测量、数据记录、数据导出等过程中难免会存在一定问题, 使得最终得到的数据 (原始数据) 与希望得到的良好数据存在一定差距。包括连续或间断性的数值缺失、数值漂移 (偏大或偏小) 等情况。因此, 对不良数据进行科学有效地预处理，对于 UWB 定位模型模型的构建有着决定性意义。

任务一要求对无干扰状态和有干扰状态下采集到的数据进行预处理，目前已知共有 324 个靶点坐标，每个靶点在同一位置上都采集多组数据，以特定的数据格式进行存储。为了便于后续的处理，我们在数据预处理阶段，首先根据有无干扰因素对数据文件进行分

类整合，将其转变成二维矩阵的形式进行存储；然后我们对每个靶点对应的二维数据矩阵进行异常值的检测，其中异常值主要包括主要包括对缺失值、重复值、及离群值三类；最后将异常数据剔除并保存。这样就完成了任务一要求的数据预处理工作。

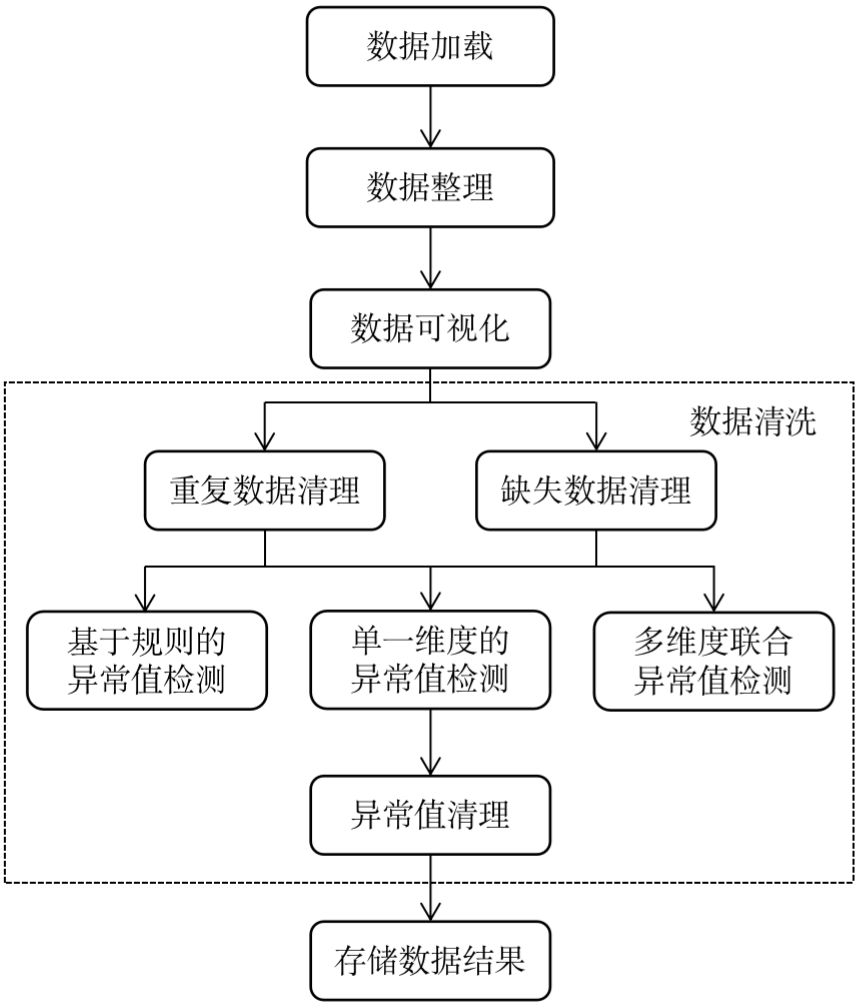


图 3.1　数据预处理流程

3.2 数据整理

靶点在每个位置都采集了无干扰和有干扰两组数据格式相同的数据，每个数据文件开头第 1 行为无实际意义的采集开始标志，因此可以直接删除，后面的数据每 4 行为一组，表示 UWB 在同一时间自动采集的一组完整数据（一组数据表示一个样品），数据之间用“:”分隔开，以无干扰数据集中的 1. 正常.txt 的前 4 条数据集为例：

T:090531088:RR:0:0:760:760:229:3301

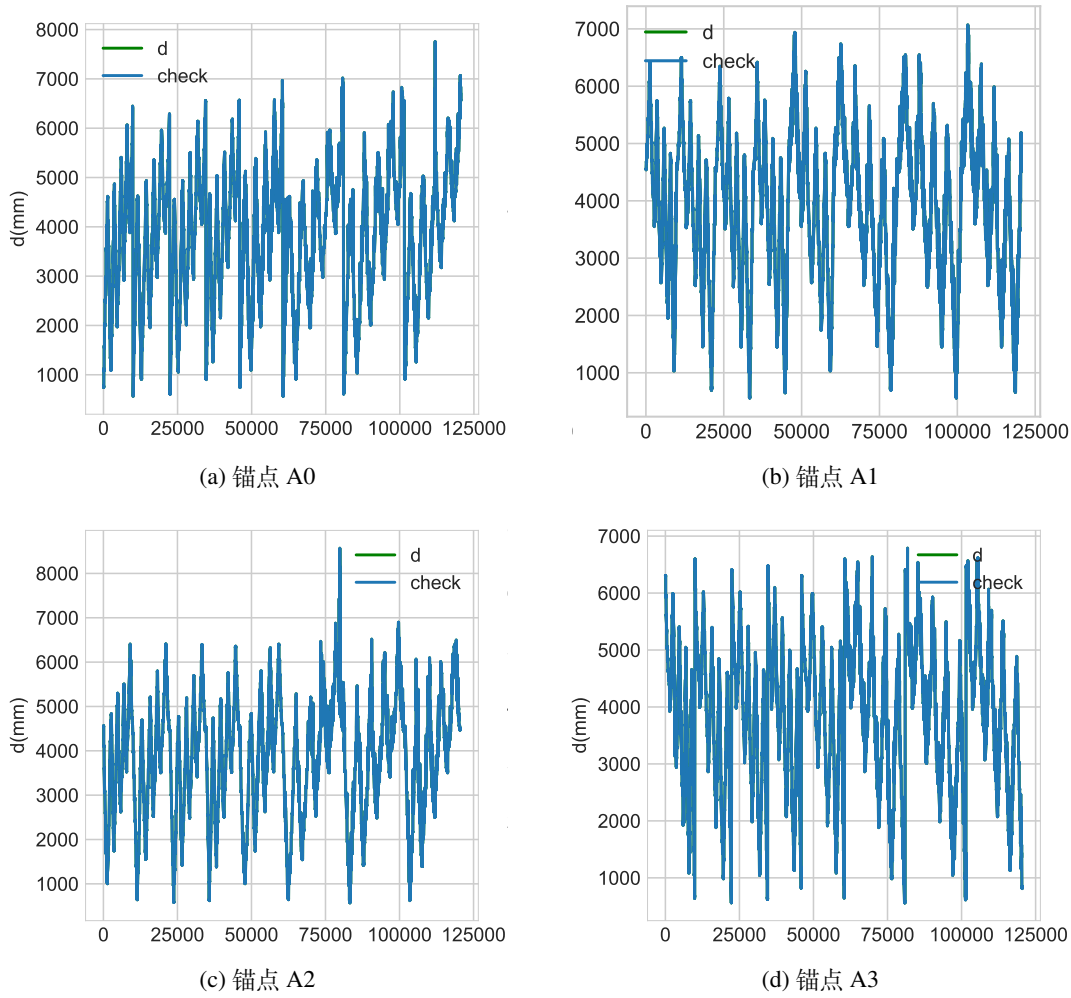
T:090531088:RR:0:1:4550:4550:229:3301

T:090531088:RR:0:2:4550:4550:229:3301

T:090531088:RR:0:3:6300:6300:229:3301

每一行数据的含义分别是 Tag 标识：时间戳：Range Report 的缩写：TagID：锚点 ID：该锚点的测距值 (mm)：测距值的校验值：数据序列号：数据编号。由上面的示例数据就可以得到 4 个锚点 A0、A1、A2、A3 到靶点的距离分别是 7600mm、4550mm、4550mm、6300mm。

在数据整理之前，我们首先对数据进行了检验，发现测距值和测距值的校验值基本上完全一致，如图 3.2 所示。另外，时间戳、数据序列号和数据编码对于本问题的解决属于无效信息，因此直接将测距值的校验值、数据序列号、数据编码三列数据做删除处理。



(a) 锚点 A0　(b) 锚点 A1

(c) 锚点 A2　(d) 锚点 A3

图 3.2　测距值检验

在数据整理阶段我们仅保留靶点到四个锚点的测距值，形成一个 n*4 的矩阵，其中 n 为数据的条数，我们将处理好的数据保存为 CSV 格式。无干扰状态下 TagID=1 的数据整合结果的前五行，如表 3.1 所示。

表 3.1　数据整合示例（单位：mm）

A0	A1	A2	A3
760	4550	4550	6300
760	4550	4550	6300
770	4550	4550	6300
780	4550	4550	6300

3.3 数据可视化

在数据清洗之前，为了对数据的分布有一个大概的印象，便于下一步的处理，我们分别对正常数据文件夹中的 24. 正常.txt 和 109. 正常.txt 以及异常数据文件夹中的 1. 异常.txt 和 100. 异常.txt 四组数据进行了可视化分析。首先我们找到了这四个文件分别对应的靶点坐标，如表 3.2 所示。

表 3.2　靶点坐标（单位：mm）

TagID	x	y	z
1	500	500	880
24	1500	3000	880
100	1500	500	1300
109	2000	4500	1300

根据三维空间座标系 o-xyz 中，可以得到两点间的距离公式为：

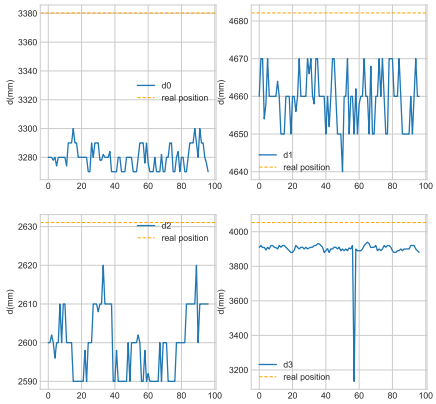
$$d_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \tag{3.1}$$

其中（x_i，y_i，z_i）为靶点的坐标，x_j 为锚点的坐标，$d_{i,j}$ 代表靶点和锚点的实际距离。已知四个锚点的坐标分别为 A0（0，0，1300）、A1（5000，0，1700）、A2（0，5000，1700）、A3（5000，5000，1300），则根据公式 3.1 就可以计算出靶点与四个锚点的实际距离，计算结果如表 3.3 所示。

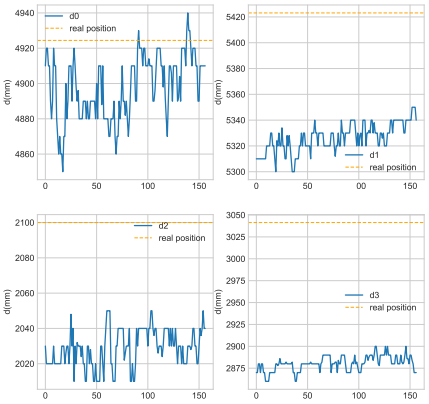
表 3.3　实际距离 (单位：mm)

	A0	A1	A2	A3
1	50	50	88	88
24	150	300	88	88
100	150	50	130	88
109	200	450	130	88

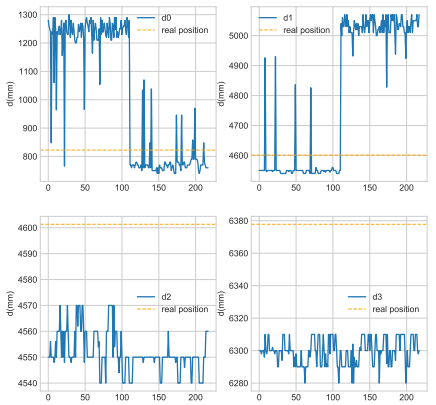
理想情况下，测量得到的距离数据应该和表中计算的实际距离数据相同，可是由于设备测量精度的限制以及外界环境干扰的问题，二者有一定的差距，如图 3.3 所示。



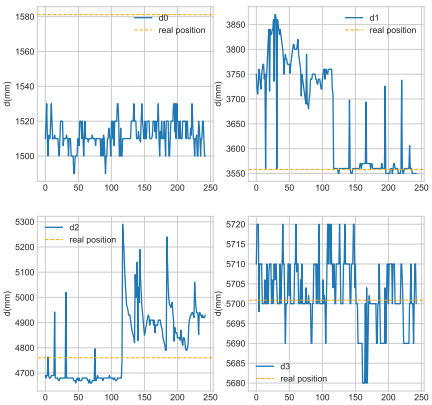
(a) 24-正常



(b) 109-正常



(c) 1-异常

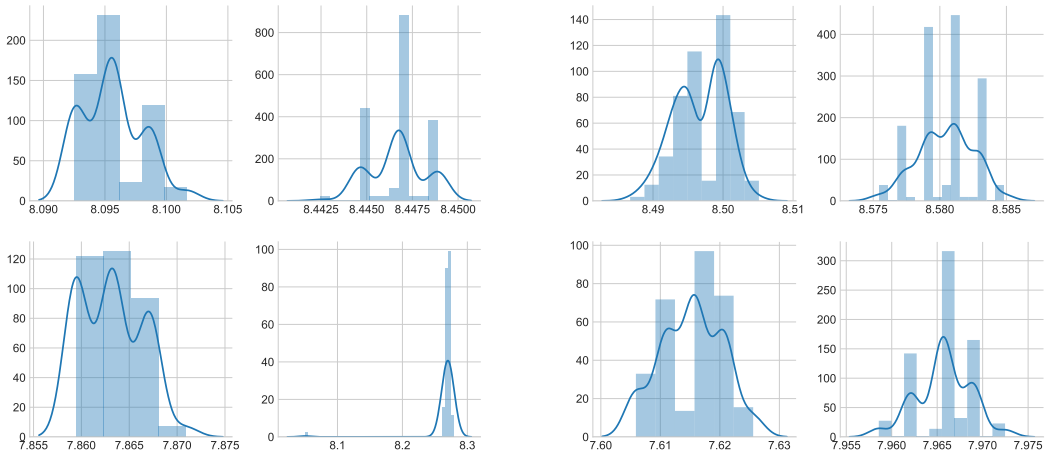


(d) 100-异常）

图 3.3　测量值与实际值对比

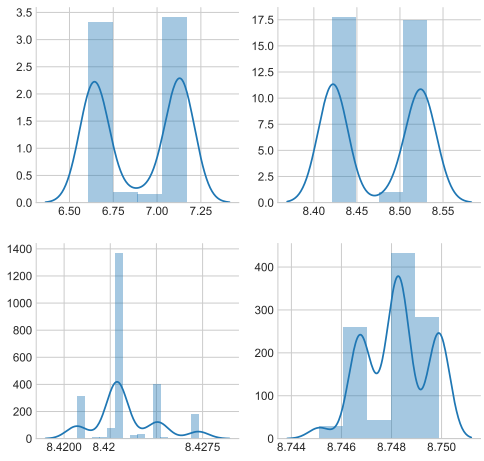
从图 3.3 可以看出，在无干扰的情况下，采集到的数值上微小的差异可能只是由于数据采集装置受到噪声所致。在受到干扰的情况下，其测试数据中中存在的变量数值突变而后稳定的情况，这是典型的非视距干扰 [2]。另外还出现了脉冲的形状，这是波动异常的情况，这些离群的点就需要在数据清洗阶段将全部剔除掉。

在数据清洗时，一个常用的异常值筛选标准就是拉依达准则（3σ 准则），该准则要求样本数据符合正态分布，因此，我们使用 K-S 方法先对样本进行正态检验，K-S 方法主要是计算出经验分布和理论分布之间的距离，并将其中最大的距离（差异）作为检验统计量。检验结果如图 3.4 所示。

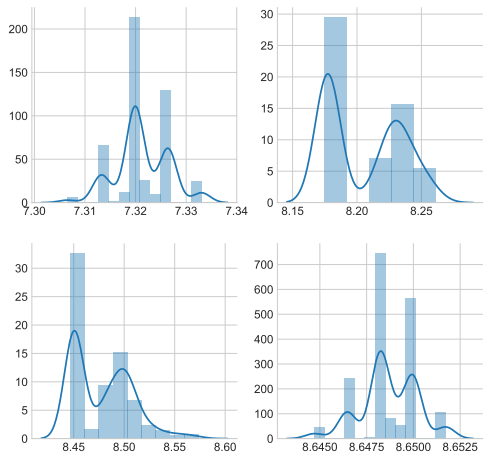


(a) 24-正常

(b) 109-正常



(c) 1-异常



(d) 100-异常

图 3.4　正态检验结果

从图 3.4 可以看出，四个靶点位置的测试数据均满足正态分布，另外，我们对其他靶点位置做了 K-S 检验（表 3.4 中展示了部分文件的 KS 值），发现其 KS 值均小于 0.05，即

以 95% 的概率接受数据服从正态分布的假设，因此我们在异常值剔除时可以使用拉依达准则（3σ 准则）。

表 3.4　正态检验 KS 值

文件	KS 值 (A0)	KS 值 (A1)	KS 值 (A2)	KS 值 (A3)
24 正常	5.00e-06	5.65e-10	9.99e-08	4.53e-30
109-正常	8.42e-14	2.81e-08	2.14e-08	8.43e-54
1-异常	1.42e-30	1.86e-32	7.93e-46	2.80e-21
100-异常	2.53e-18	2.23e-24	2.574-14	2.87e-13

3.4 数据清洗

通过数据的可视化我们把握了数据的大体分布，并且对数据清洗有了基本的解决方案。良好的数据能让更好的适应我们的数学模型，接下来我们将介绍我们对不良数据的清洗方法。

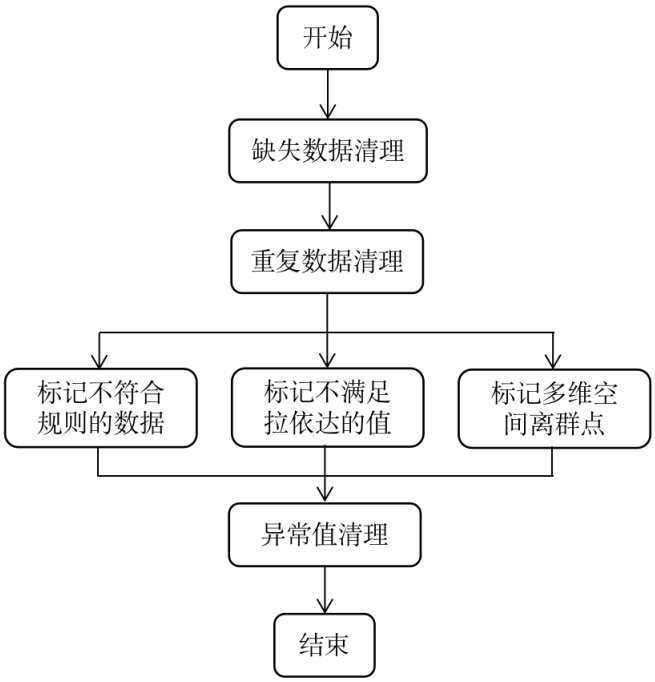


图 3.5　数据清洗流程图

从图 3.5 中可以看出，我们首先采用遍历所有文件的方法对缺失值和重复值进行检测和剔除，当一个靶点的四个测距结果完全一致的情况下才会剔除。在异常值的识别与剔除阶段我们采用了基于规则的异常值检测方法、单维异常值检测和多维异常值检测方法相结合的的形式。

（1）基于规则的异常值检测

在实验环境下，通过锚点的坐标及实验场地的大小，我们可以计算出测量得到的距离的最大范围，若超过了这个最大范围，那么我们就认为这是异常的测试数据。以无干扰数据为例，假设第 i 个靶点和第 j 个锚点测距得到的数据序列为 $D_{ij}=\{d_{ij}^1,d_{ij}^2,\cdots,d_{ij}^n\}(n=1,2,3)$，$d_{ij}$ 距离的取值范围为 $(\mathrm{d}_{min},d_{max})$ $\forall d_{ij}^n\in D_{ij}$，对于序列 D_{ij} 做如下处理：

$$D_{ij}\leftarrow\begin{cases}D_{ij},\ \text{if } d_{min}\le d_{ij}^n\le d_{max}\\ D_{ij}-\left\{d_{ij}^n\right\},\ \text{if } d_{ij}^n\le d_{min} \text{ or } d_{ij}^n\ge d_{max}\end{cases}\tag{3.2}$$

进一步地，我们去除序列 D_{ij} 中的最大值和最小值，以获取更加准确的变量值，即分别作如下处理：

$$D_{ij}\leftarrow D_{ij}-\left\{d_{ij}^n\right\}\ \text{if } d_{ij}^n=\max_i\left(D_{ij}\right)\tag{3.3}$$

$$D_{ij}\leftarrow D_{ij}-\left\{d_{ij}^n\right\}\ \text{if } d_{ij}^n=\min_i\left(D_{ij}\right)\tag{3.4}$$

（2）单维异常值检测在数据可视化阶段，我们验证了样本数据是符合正态分布的，因此我们使用拉依达准则（3σ 准则）进行异常值的去除。在正态分布中 σ 代表标准差,μ 代表均值。$x=\mu$ 即为图像的对称轴，3 准则认为当测量次数足够大时，y 的取值几乎全部集中在 $\mu-3\sigma,\mu+3\sigma)$ 区间内，超出这个范围的可能性仅占不到 0.3%。假设在靶点 i 的测量时，共采集了 N_i 组数据，则该次测量的均值和标准差为：

$$\mu_{ij}=\frac{1}{N_i}\sum_{n=1}^{N_i}d_{ij}^n\tag{3.5}$$

$$\sigma_{ij}=\sqrt{\frac{1}{N_i-1}\sum_{n=1}^{N_i}\left(d_{ij}^n-\mu_{ij}\right)^2}\tag{3.6}$$

依据拉依达准则（3σ 准则），按照如下所示的方法处理异常值：

$$D_{ij}\leftarrow\begin{cases}D_{ij},\ \text{if } \mu-3\sigma\le d_{ij}^n\le\mu+3\sigma\\ D_{ij}-\left\{d_{ij}^n\right\},\ \text{otherwise}\end{cases}\tag{3.7}$$

（3）多维异常值检测

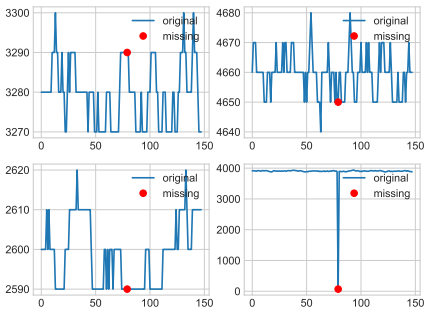
UWB 在数据采集时，会同时测量靶点到四个锚点的距离，因此多维数据之间具有一定的相关性，我们不能只考虑单维异常值的检测，还要考虑针对多维的异常值检测，因为可能会存在数据是单个维度的偏差不大，但是整体表现是离群的。

谱聚类（spectral clustering）是高维空间数据常用的聚类算法，它基于计算机中图的概念，把数据的特征成空间中的点坐标，点于点之间存在着图结构关系，点与点存在着相应的权重，权重与点与点的距离成反比、。通过对所有的数据点组成的图进行遍历切分，使

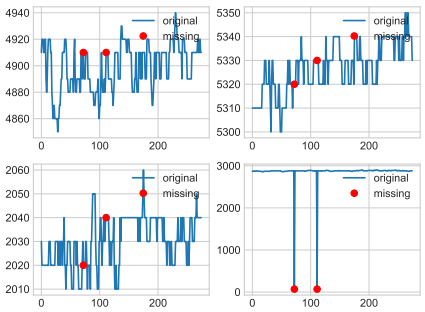
得切分后相似的子图之间的权重尽可能高，不同子图之间的权重尽可能低，实现图结构的聚类效果。

3.5 数据预处理结果

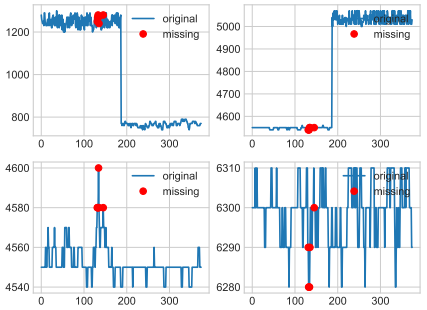
题目中给出了 324 个靶点分别在有干扰和无干扰情况下的测距结果，基于以上数据预处理的方法，我们编写 python 程序，以题目中要求的正常数据文件夹中的 24. 正常.txt 和 109. 正常.txt 以及异常数据文件夹中的 1. 异常.txt 和 100. 异常.txt 四组数据为例，展示数据预处理结果。



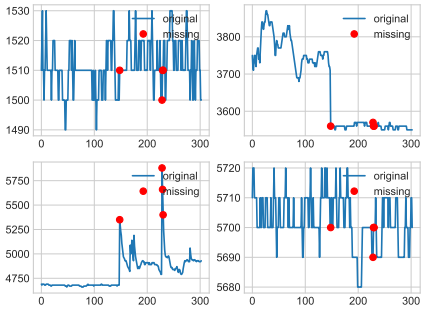
(a) 24-正常



(b) 109-正常



(c) 1-异常



(d) 100-异常

图 3.6　无用值检测结果

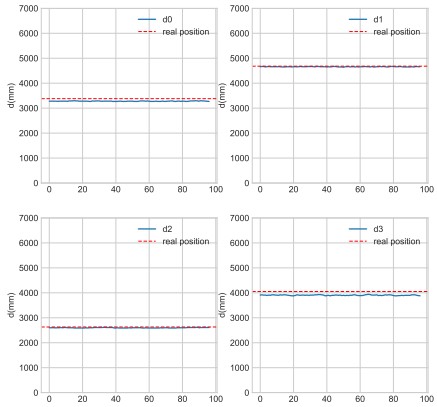
图 3.6 中展示了四个文件的无用数据检测的结果，蓝色的线代表原始的测量数据；红色的点代表我们使用异常值检测算法所检测出的异常值，当四个维度中有一个异常值时，那么我们就会判定整组数据异常。另外，由于由于重复值的数量较多，在这里不做展示，数据预处理前后的文件数量对比，如表 3.5-3.6 所示。

表 3.5　处理前后数据量

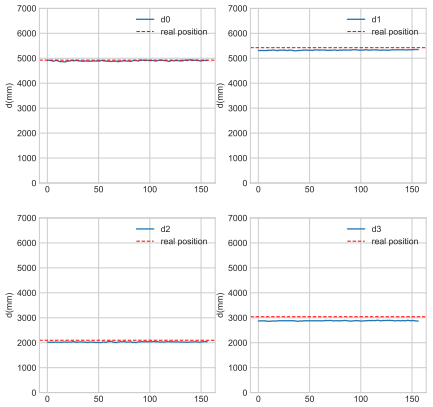
文件类型	原始数据量	处理后数据量
正常文件	77122	37028
异常文件	79176	65824

表 3.6　数据预处理结果

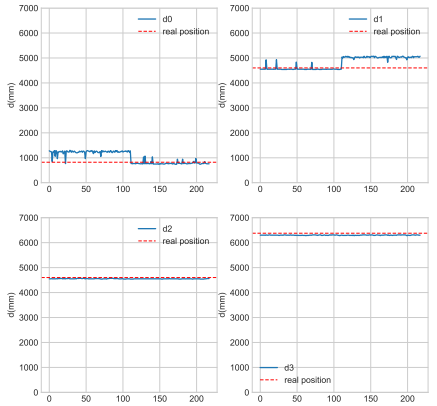
文件	原始数据量	缺失值数量	重复值数量	异常值数量	处理后数量
24-正常	148	0	65	1	82
109-正常	275	0	145	2	128
1-异常	375	156	186	6	183
100-异常	303	0	81	4	218



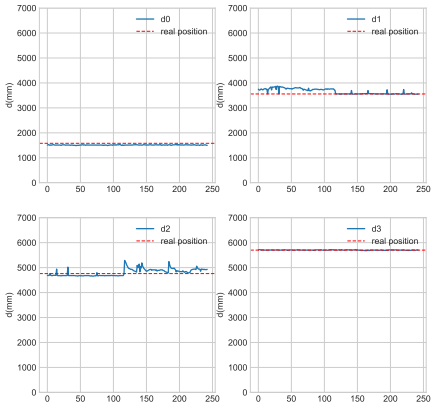
(a) 24-正常



(b) 109-正常



(c) 1-异常



(d) 100-异常）

图 3.7　数据预处理结果

从图 3.7 数据预处理结果图中可以看出，我们在数据预处理阶段成功将无用数据清洗干净，像图 3.3 中所呈现的尖峰、突刺都已经没有了，数据整体呈现较为稳定的状态。

4 问题二——定位模型的建立与求解

4.1 问题二分析

在问题一中，我们完成了对存在尖峰，缺失，重复等异常数据的清洗工作。问题二要求根据问题一处理后的各锚点测量数据建立室内定位模型，尽可能准确的估计（预测）靶点的正确位置。该问题中涉及了“正常数据”和存在遮挡环境 (NLOS) 的“异常数据”，因此需要分别建立模型实现靶点定位，主要实现方案如下：

- 步骤一：测量数据校准
  由于实验环境中的非人为干扰可能会导致锚点测量时存在一定的随机误差，因此在通过测量数据定位之前需要对测量数据与真实数据进行线性拟合，从而实现数据校准。
- 步骤二：基于正常数据的空间定位模型
  为选出更优的空间定位算法，本节将空间定位问题演化为非线性规划问题的求解。本节选取了非线性规划问题三种常规解法中的典型算法：Jacobi 迭代法，牛顿法和基于加权最小二乘的三维空间 Chan 算法对靶点坐标进行估计。
- 步骤三：基于异常数据的空间定位模型
  根据题目说明，本问题中的异常数据由非视距误差导致。通过对异常测量数据的分析发现，同一时刻对于一个靶点位置，仅会存在一个锚点的测量数据异常。因此首先根据该原理建立了基于贪心误差搜索的误差补偿算法，获取使得定位误差最小的补偿值。其次在标准卡尔曼滤波模型中引入 NLOS 估计值，通过不断迭代消除异常的数据偏移。
- 步骤四：NLOS/LOS 定位模型精确度分析
  计算估计坐标与真实坐标之间的欧氏距离作为估计误差。分别针对三种定位算法估计坐标的每个维度，从均方根误差，误差分布以及每个靶点的误差变化曲线，进行了三个方面的误差分析。

4.2 测距校正

根据题目描述，靶点与锚点之间的距离通过 TOF 测距原理得到。室内 UWB 定位的测距精度可能受到脉冲指数、大气折射、天线相位中心偏移、穿透介质等环境误差影响。因此在通过定位算法计算之前需要对锚点的测量数据进行校正。由于附件一中包含所测 324 个靶点的真实坐标，故能够确定靶点在每个位置与锚点之间的准确距离。将测得的数据与实际距离作对比。理想情况下，分别以正确的测量距离向量和理论距离向量为横纵坐标，所拟合的环境误差曲线表现为笛卡尔空间的对角线。然而在图中可以看出测量数据相比于真实值存在一定的偏移，并不是标准对角线，说明锚点的测距存在一定误差。故在进行定位之前，首先将原始数据带入拟合方程进行校正。

4.3 针对正常数据的空间定位模型

在有 4 个锚点的情况下，根据题目给出的距离能够建立以靶点坐标为未知数的 4 个“距离-坐标” 多项式方程，可以规约为非线性规划问题。对于非线性方程组求解的问题，主要分为非迭代法和迭代法。非迭代法将非线性方程组转化为线性方程组，常用加权最小二乘法求解 [3]；迭代法通过误差迭代的方式，找到方程解的最优值，常用 Jacobi 迭代法和牛顿法。本节分别实现了针对题目假设空间的非迭代和迭代常用的三种算法，通过对比三种算法的定位精度，最终选择三维 Chan 算法作为用于正常数据的空间定位。

4.3.1 基于加权最小二乘法的三维 Chan 定位模型

Chan 算法是一种非递归双曲线方程组的解法，可以通过解析方式得到方程组的解。在噪声满足正态分布的条件下，该算法计算量小，定位精度较高。该算法是将根据靶点的 TOF 测距的非线性方程组转化为线性方程组，采用加权最小二乘法 (WLS) 得到初始解，再将第一次估计的坐标位置及噪声变量等值作为第二次 WLS 计算的约束条件得到第二次估计值，从而得到估计坐标。该算法常用于二维空间定位，因此针对于本题的空间定位问题需要将其迁移至三维空间中。三维空间下的 Chan 模型推导如下：在三维场景下，靶点 T_0 在以锚点 A_i 为球心，距离 d_i 为半径的球面上，假设待定位的靶点坐标为 (x, y, z)，第 i 个锚点的位置为 (x_i, y_i, z_i)，距靶点的距离为 d_i 则可以根据空间中点与点之间的距离公式得到

$$d_i^2 = (x - x_i)^2 + (y - y_i)^2, i = 1, 2, \ldots, n \tag{4.8}$$

由 (4.8) 可得

$$d_i^2 - K_i = -2x_i x - 2y_i y - 2z_i z + R \tag{4.9}$$

其中，$K_i = x_i^2 + y_i^2 + z_i^2, R = x^2 + y^2 + z^2$。从几何角度考虑，由于题目给出的 4 个锚点不共面，因此锚点所构成的球面可以相交于唯一点，从非线性方程组求解的角度来看，也将有唯一解。令 $Z_a = [x, y, , z, R]$ 为解向量，$Z_a^0 = [x^0, y^0, z^0, R^0]$ 为真实值，考虑到可能存在的测量噪声 N，建立以 Z_a 为变量的方程组

$$\psi = H - G_a Z_a \tag{4.10}$$

其中，$H = \begin{bmatrix} d_1^2 - K_1 \\ d_2^2 - K_2 \\ \vdots \\ d_n^2 - K_n \end{bmatrix}$，$G_a = \begin{bmatrix} -2x_1 & -2y_1 & -2z_1 \\ 1 & & \\ -2x_2 & -2y_2 & -2z_2 \\ 1 & & \\ \vdots & \vdots & \vdots \\ -2x_n & -2y_n & -2z_n \\ 1 & & \end{bmatrix}$，$Z_a = \begin{bmatrix} x \\ y \\ z \\ R \end{bmatrix}$。

假设在进行基于 TOF 的测量时，假设各个锚点的距离测量误差为 δ_i，那么：

$$\psi_i = d_i^2 - \left(d_i^0\right)^2 = \left(d_i^0 + \delta_i\right)^2 - \left(d_i^0\right)^2 = 2d_i^0\delta_i + \delta_i^2 \tag{4.11}$$

其中，d_i^0 表示未知量 d_i 的真实值，由于现实环境中测量所带来的随机误差远小于距离的真实值，那么估计误差向量可以近似的表示为 $\psi_i \approx 2d_i^0\delta_i$，即 $\psi \approx 2B\delta$。其中，$B = \text{diag}\left(d_1^0, d_2^0, \ldots, d_n^0\right)$，$\delta = [\delta_1, \delta_2, \ldots, \delta_n]^T$。那么估计误差向量 psi 的协方差矩阵可以表示为：

$$\Psi = E\left(\psi\psi^T\right) = 4BE\left(\delta\delta^T\right)B = 4BQB \tag{4.12}$$

其中 Q 为测量误差向量 δ_i 的协方差矩阵。因此由加权最小二乘算法可以得到估计值。

$$\widehat{Z}_a = \left(G_a^T \Psi^{-1} G_a\right)^{-1} G_a^T \Psi^{-1} H \tag{4.13}$$

实际应用中可以使用 Q 代替 psi 计算得到 $\widehat{Z}_a = [\hat{x}, \hat{y}, \hat{z}, \hat{R}]^T$，从而利用一个估计的值代替 B 矩阵，且由于第一次估计时估计误差矩阵的协方差是未知的，因此使用单位矩阵来代替。因此第一次估计值计算由下式得到：

$$\widehat{Z}_a = \left(G_a^T G_a\right)^{-1} G_a^T H \tag{4.14}$$

由于解向量中的 R 值与靶点坐标相关，因此根据第一次估计结果以及变量 R 作为下一次 WLS 估计的约束条件。第一次估计值与真实值之间的关系为：

$$\begin{cases} \widehat{Z}_a(1) = x^0 + e_1 \\ \widehat{Z}_a(2) = y^0 + e_2 \\ \widehat{Z}_a(3) = z^0 + e_2 \\ \widehat{Z}_a(4) = R^0 + e_3 \end{cases} \tag{4.15}$$

其中 e_1,e_2,e_3 为估计的误差。由此可以得到方程：

$$\psi' = H' - G'_a Z_p \tag{4.16}$$

上式中 $H' = \begin{bmatrix} \widehat{Z}_a^2(1) \\ \widehat{Z}_a^2(2) \\ \widehat{Z}_a^2(3) \\ \widehat{Z}_a(4) \end{bmatrix}$，$A_S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$，$Z_p = \begin{bmatrix} x^2 \\ y^2 \end{bmatrix}$。第二次估计的流程为：

$$\Delta Z_a \approx \left(G_a^{\mathrm{T}} \Psi^{-1} \mathrm{G}_a\right)^{-1} G_a^T \Psi^{-1} B \delta \tag{4.17}$$

$$\mathrm{cov}\left(Z_a\right) = E\left[\Delta Z_a \Delta Z_a^T\right] = \left(G_a^T \Psi^{-1} G_a\right)^{-1} \tag{4.18}$$

其中 ψ' 的协方差矩阵为

$$\Psi' = E\left[\psi' \psi'^T\right] = 4B' \,\mathrm{cov}\left(Z_a\right) B' \tag{4.19}$$

$$B' = \mathrm{diag}(x, y, z, 0.5) \tag{4.20}$$

第二轮估计值为：

$$Z_p = \left(G_a{}'^T \Psi^{-1} G_a\right)^{-1} G_a{}'^T \Psi'^{-1} H' \tag{4.21}$$

通过两轮估计（实际上加上初始化估计，共有三轮计算），可以得到靶点的定位结果 $\hat{Z} = \pm\sqrt{Z_p}$，由于本问题中的实验环境坐标均为正值，因此最终估计结果取开方的绝对值。

4.3.2 基于梯度下降的定位模型

三维空间坐标系下的定位问题，需要求解三元二次方程，即其目标函数的非线性的，因此我们可以使用梯度下降的方法求解求解定位模型。梯度下降法基于梯度的方向是函数当前位置上升最快的方向，梯度的反方向是函数在当前位置下降最快的方法这一思想，沿着梯度的方向就可以快速获得最优解。在 UWB 定位任务重，需要最小化损失函数 $L(\theta)$，其中 θ 就是要求解的模型的参数，梯度下降法是一种迭代算法，迭代公式如下所示：

$$\theta^t = \theta^{t-1} + \Delta\theta \tag{4.22}$$

在求解问题时，对损失函数的一阶泰勒展开为：

$$\begin{aligned} L\left(\theta^t\right) &= L\left(\theta^{t-1} + \Delta\theta\right) \\ &\approx L\left(\theta^{t-1}\right) + L'\left(\theta^{t-1}\right)\Delta\theta \end{aligned} \tag{4.23}$$

为使 $L\left(\theta^t\right) < L\left(\theta^{t-1}\right)$, 可取 $\Delta\theta = -\alpha L'\left(\theta^{t-1}\right)$，则 $\theta^t = \theta^{t-1} - \alpha L'\left(\theta^{t-1}\right)$。其中 α 是步长, 一般直接赋一个较小的值。

4.3.3 基于牛顿法的定位模型

牛顿法是一种求解方程近似根的方法，实质使用泰勒级数的前几项的展开来寻找 $f\ x\ =\ 0$ 的根 [8]。牛顿法本质上也是迭代算法法，其可以求出方程的近似的一个或者多个解。在求解问题时，对损失函数的二阶泰勒展开为：

$$\begin{aligned} L\left(\theta^t\right) &= L\left(\theta^{t-1} + \Delta\theta\right) \\ &\approx L\left(\theta^{t-1}\right) + L'\left(\theta^{t-1}\right)\Delta\theta + L''\left(\theta^{t-1}\right)\frac{\Delta\theta^2}{2} \end{aligned} \tag{4.24}$$

为简化分析过程，假定 θ 只有一维，将一阶和二阶导数分别记为 g 和 h ，即:

$$L\left(\theta^t\right) \approx L\left(\theta^{t-1}\right) + g\Delta\theta + h\frac{\Delta\theta^2}{2} \tag{4.25}$$

为使 $L\left(\theta^t\right)$ 极小，即 $g\Delta\theta + h\frac{\Delta\theta^2}{2}$ 极小，令

$$\frac{\partial\left(g\Delta\theta + h\frac{\Delta\theta^2}{2}\right)}{\partial(\Delta\theta)} = 0 \Rightarrow \Delta\theta = -\frac{g}{h} \tag{4.26}$$

故

$$\theta^t = \theta^{t-1} - \frac{g}{h} \tag{4.27}$$

以上讨论的是二维的情况，在本题的背景下，需要将 θ 推广到向量形式, 高维情况的牛顿迭代公式为：

$$\theta^t = \theta^{t-1} - \left[HL\left(\theta^{t-1}\right)\right]^{-1}\nabla L\left(\theta^{t-1}\right) \tag{4.28}$$

此处 H 是海森矩阵，为目标函数的二阶导数矩阵。

4.4 针对异常数据的空间定位模型

4.4.1 基于贪心误差搜索的异常数据校正

通过对异常测量数据的分析，对比同一个位置正常和异常的测量距离均值，发现锚点 A0 和锚点 A3 的测量数据较为精准，在本题所给的异常数据中较少出现异常。常出现的异常的为锚点 A1 和锚点 A2。锚点 2 和锚点 3 的较大误差值之间交错出现，可以证明在题目所提供的环境异常数据中，对于同一个靶点位置，在同一时刻的单次测量当中，仅会出现一个锚点的测量误差。而且统计之后发现，单个锚点测量距离的误差分布在 [a,b] 的范围内。因此本问题能够转化为如何识别锚点误差并补偿每个锚点的测量数据，从而获得准确的定位值。在算法中，通过贪心的方式对每一维度找到使估计误差最小的数据偏移量，对异常数据进行修正，修正后的数据可以通过正常数据下的定位算法估计靶点坐标。贪心算法（贪婪算法）是指在求解问题时，不关注整体最优解，而是总做出在当前看来最好的选择。考虑到在实际应用过程中，采用异常数据的定位需要具备一定的时效性，故我们设计了针对当前实验环境的贪心误差搜索异常数据校正算法，快速找到锚点测量数据的局部最优补偿值，算法流程如图 4.8 所示。

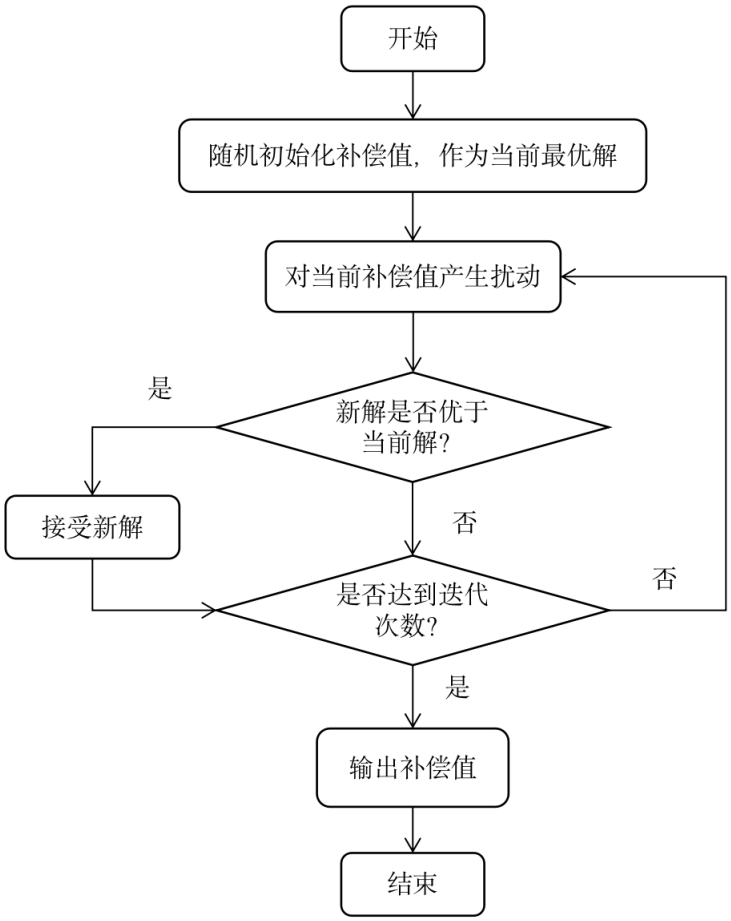


图 4.8　贪心误差搜索算法流程

4.4.2 基于有偏卡尔曼滤波的异常数据校正

在真实的测量环境中，由于障碍物的位置多样性，可能导致用于测量的多个锚点出现NLOS 误差。NLOS 会使得 UWB 锚点接收信号的时延受到影响，因此测距的数值会与没有障碍物的情况下产生较大的差异。使用误差很大的测距值进行定位分析，会造成 UWB 定位系统性能的下降，因此需要对异常的测量值进行校正。我们需要对偏差很大的测距值进行估计，需要把卡尔曼滤波可以从一连串带噪声的信号测量值中估计出系统的状态，因此，它是一种高效的可以通过迭代算法对状态向量进行动态估计的自回归滤波器 [6]。假设系统的状态方程具有线性，状态向量的噪声满足高斯分布且观测的状态方程也是线性的，就可以利用卡尔曼滤波器进行迭代求下一时刻系统的状态，不需要其他的先验信息。卡尔曼滤波器的两个基本状态方程如下：

$$x(k+1) = Ax(k) + u(k) \tag{4.29}$$

$$z(k) = Hx(k) + v(k) \tag{4.30}$$

其中，$x(k)$ 表示当前 k 时刻的状态，A 是系统的状态转移矩阵，表示没有影响时从 k 时刻到 k+1 时刻系统的状态转移形式，$u(k)$ 表示状态更新过程中产生的噪声，(k) 的协方差矩阵设为 $Q(k)$，表示系统噪声的协方差。$z(k)$ 是当前时刻 k 观测到的系统状态，H 是观测矩阵，表示状态 $x(k)$ 转移到观测状态 $z(k)$ 的方式，$v(k)$ 是测量的偏差，$v(k)$ 的协方差矩阵设为 $R(k)$，表示测量误差的协方差。可以发现，这两个状态方程构建了当前时刻 k 的状态 $x(k)$ 和下一个时刻 $k+1$ 的系统状态 $x(k+1)$ 的转移方程，也建立了当前时刻 k 的系统真实状态 $x(k)$ 和观测状态 $z(k)$ 的关系矩阵。一般而言, 误差相关矩阵 $P(k)$ 和卡尔曼增益 $K(k)$ 的计算方式如下：

$$P(k+1) = AP(k)A^T + Q(k) \tag{4.31}$$

$$K(k) = P(k)H^T(HP(k)H^T + R(k)^{-1} \tag{4.32}$$

得到上述值开始更新状态。首先更新误差相关矩阵 $P(k)$：

$$P(k) = P(k) - KHP(k) \tag{4.33}$$

之后可以求得更新后的状态变量 $x(k)$：

$$x(k) = x(k) + K(z(k) - Hx(k)) \tag{4.34}$$

在本文中，我们设基站数为 N，UWB 装置锚点发出信号，靶点接收信号，得到时间差乘上 $c = 3 * 10^8 m/s$, 就可以得到附件中各个锚点到靶点距离的测量值。我们为室内环境建立一个 NLOS 误差状态向量模型：

$$d_n(t_i) = D_m(t_i) + error_n(t_i) + \text{NLOS}_n(t_i) \quad n = 1, 2, \cdots N; \quad i = 0, 1, \cdots K-1$$

其中，$error_n$ 是测量产生的误差，D_m 是锚点到靶点的真实距离，$NLOS_n$ 是非视距传播造成的误差。有偏估计的卡尔曼滤波器是将 NLOS 的误差以及测量值的一阶导数加入到状态方程中进行估计：

$$x(k+1) = \begin{bmatrix} x(k+1) \\ x'(k+1) \\ NLOS(k+1) \end{bmatrix}$$

其中，$x'(k+1)$ 表示测量值的一阶导数，$NLOS(k+1)$ 表示非视距误差。利用卡尔曼滤波的两个状态方程，可以利用当前时刻的测量值求得一个距离的估计值，直接消除非视距造成的误差，具有较好的实时性。

4.5 基于卡尔曼滤波的 NLOS/LOS 定位模型

上文我们讨论了空间定位算法及误差校正方式，对于 NLOS 使用有偏卡尔曼滤波，对于 LOS 使用标准卡尔曼滤波，可以在非视距干扰的时候直接去除噪声，不用再修改原先的定位模型，增强了定位模型的普适性和泛化能力。

4.6 模型精度检验

本节我们针对上文提出的正常距离和异常距离两种室内定位算法，分别针对 1 维，2 维，3 维误差进行分析。模型误差定义：定义模型估计误差为估计出的坐标与已知真实坐标与锚点的欧式距离。模型精度检验指标：模型在不同维度下估计坐标与真实坐标相欧式距离的均方根误差；不同维度下模型估计误差的分布直方图；不同维度下每个靶点位置的误差值。均方根误差 (RMSE) 是观测值与真值偏差的平方和观测次数 n 比值的平方根，能够很好地反映出测量的精密度。RMSE 越小，表示测量精度越高，因此可用 RMSE 作为评定这一测量过程精度的标准。计算公式如下：

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} \left(X_{obs,i} - X_{\text{model},i}\right)^2}{n}} \tag{4.35}$$

4.6.1 正常数据下的模型精度检验结果

本节对比了提出的三种针对正常空间的定位模型。如图 4.9-4.12 所示。三种模型在不同维度的 RMSE 值如表 4.7 所示。

表 4.7　正常数据下各维度坐标估计均方根误差

维度	迭代法	Chan 算法	牛顿法
3 维	33.0438	14.4381	35.4136
2 维	4.4662	3.3840	6.7690
1 维 Z 轴	32.6971	13.9547	34.1890
1 维 Y 轴	2.6424	2.0476	4.1343
1 维 X 轴	3.5507	2.6539	4.5955

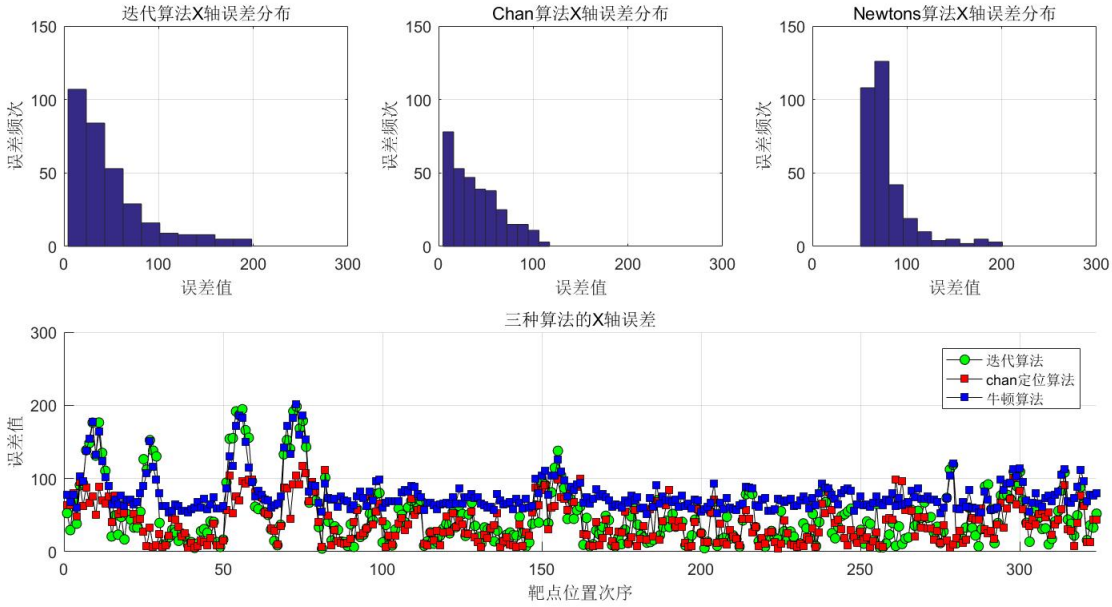


图 4.9　正常数据下 3 种算法的 X 轴误差分布

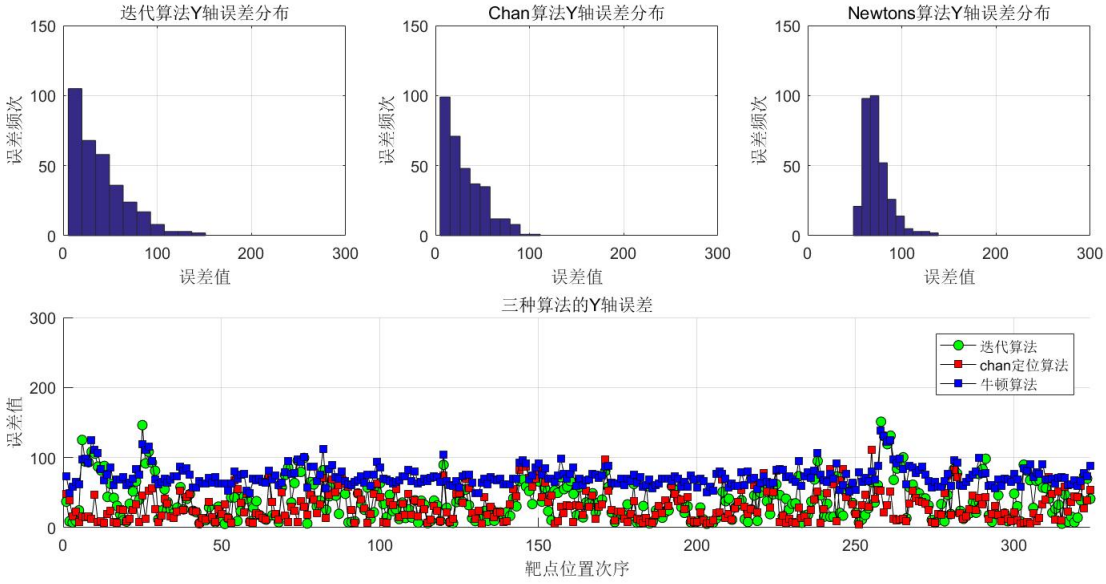


图 4.10　正常数据下 3 种算法的 Y 轴误差分布

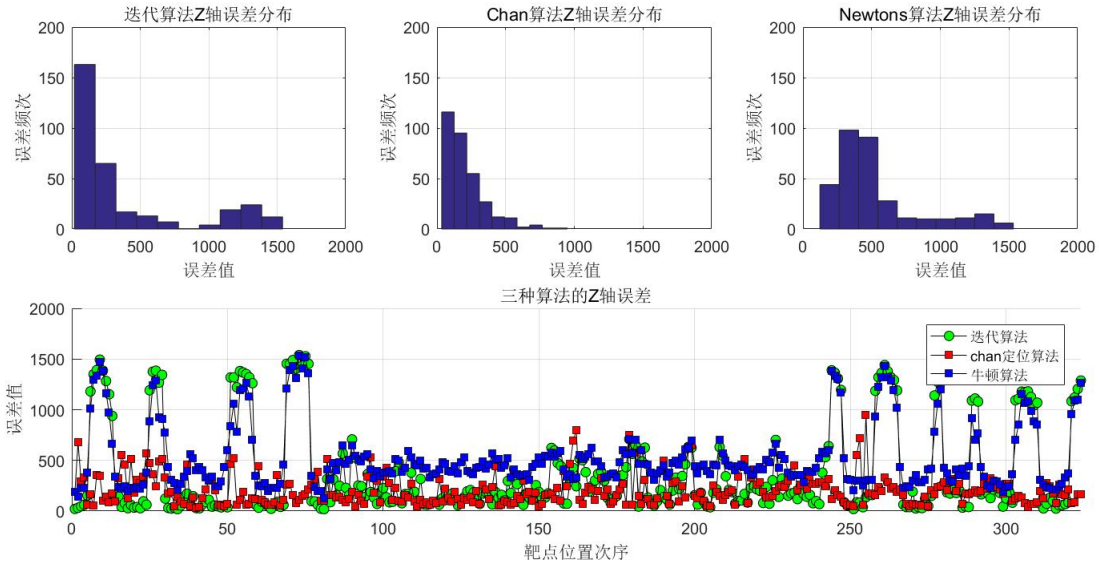


图 4.11　正常数据下 3 种算法的 Z 轴误差分布

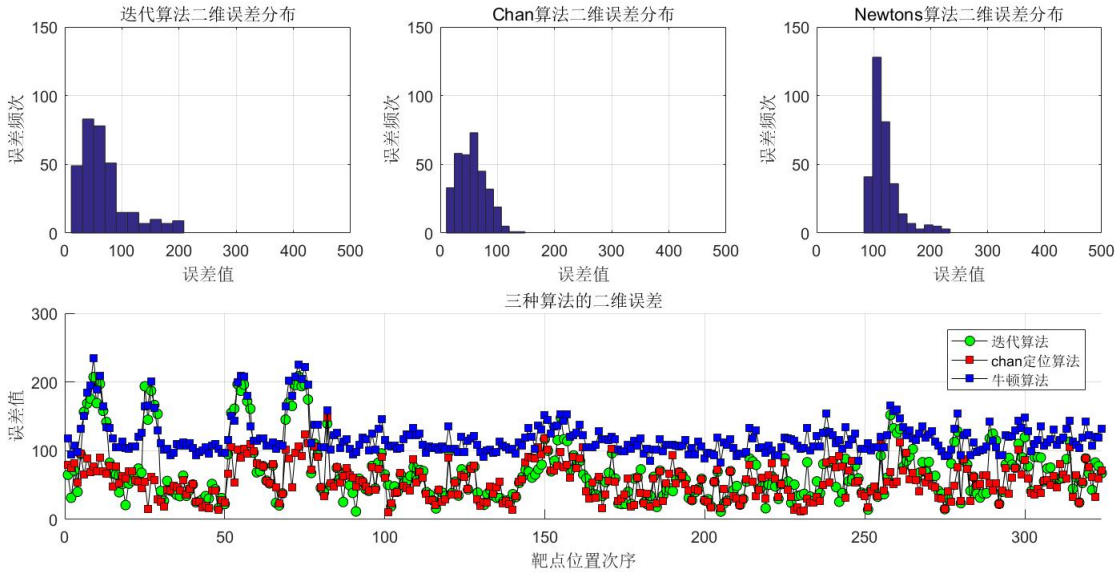


图 4.12　正常数据下 3 种算法的 2 维误差分布

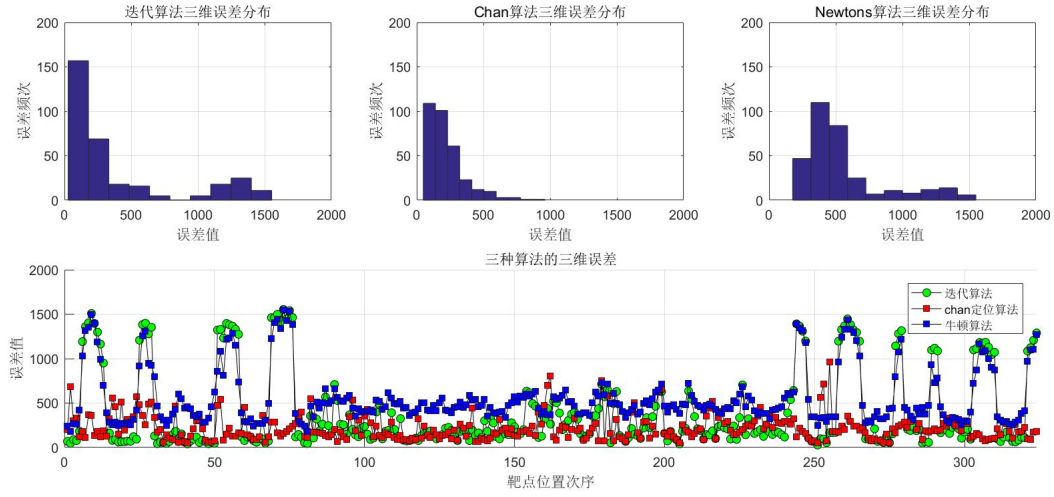


图 4.13　正常数据下 3 种算法的 3 维误差分布

根据不同维度的误差分布图可以看出 Chan 算法相比于迭代法及牛顿法在三个维度上均具有更小的估计误差，且大多数估计误差分布在 $(0, 10)cm$ 之间。

4.6.2 异常数据下的模型精度检验结果

本节对于上文提出的三种异常数据下三维定位模型分别进行了误差分布直方图及误差值可视化，如图 4.14-4.16 所示。

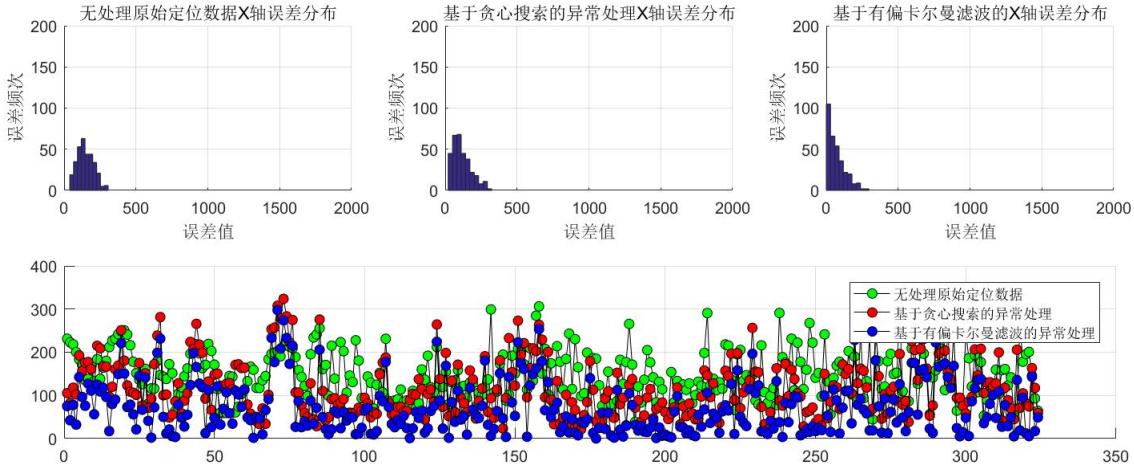


图 4.14　NLOS 环境下的 X 轴误差分布

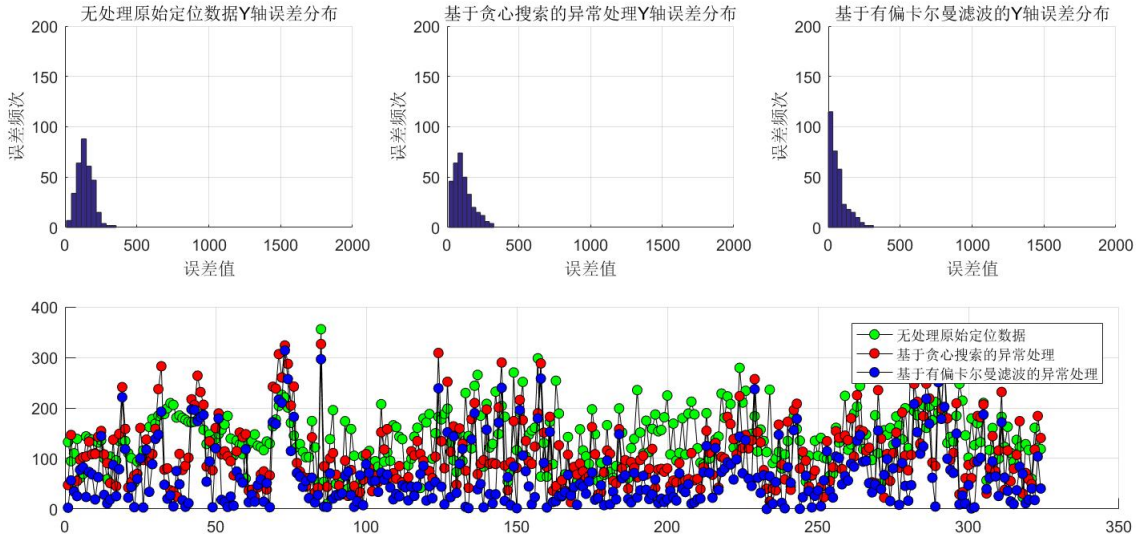


图 4.15　NLOS 环境下的 Y 轴误差分布

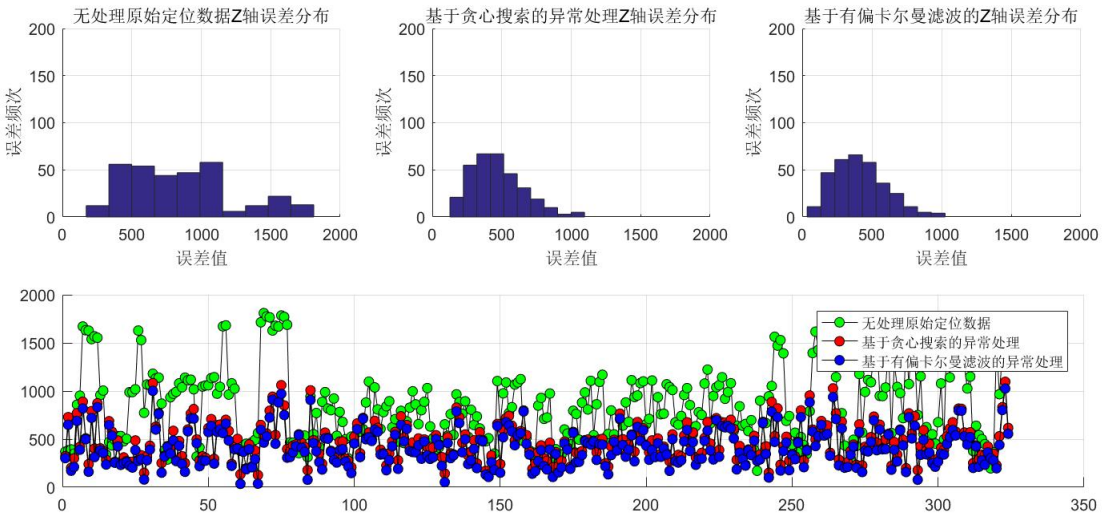


图 4.16　NLOS 环境下的 Z 轴误差分布

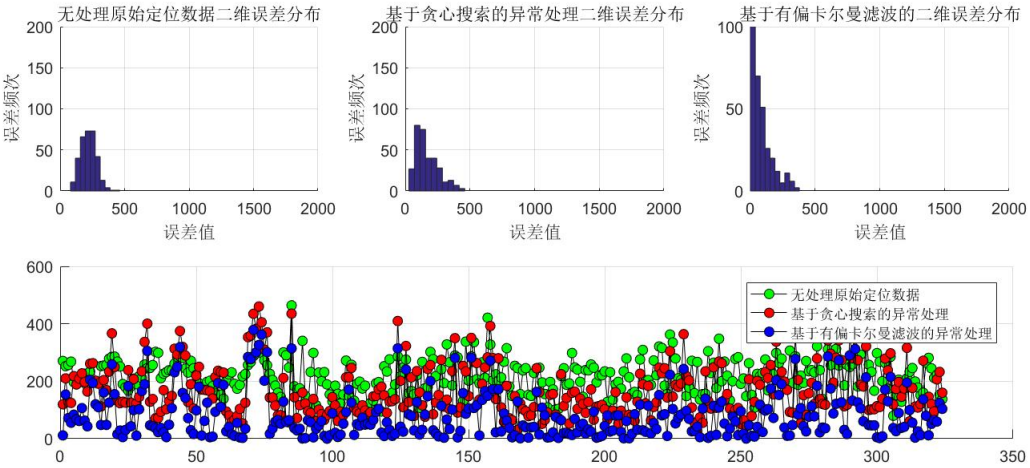


图 4.17　NLOS 环境下的 2 维定位误差分布

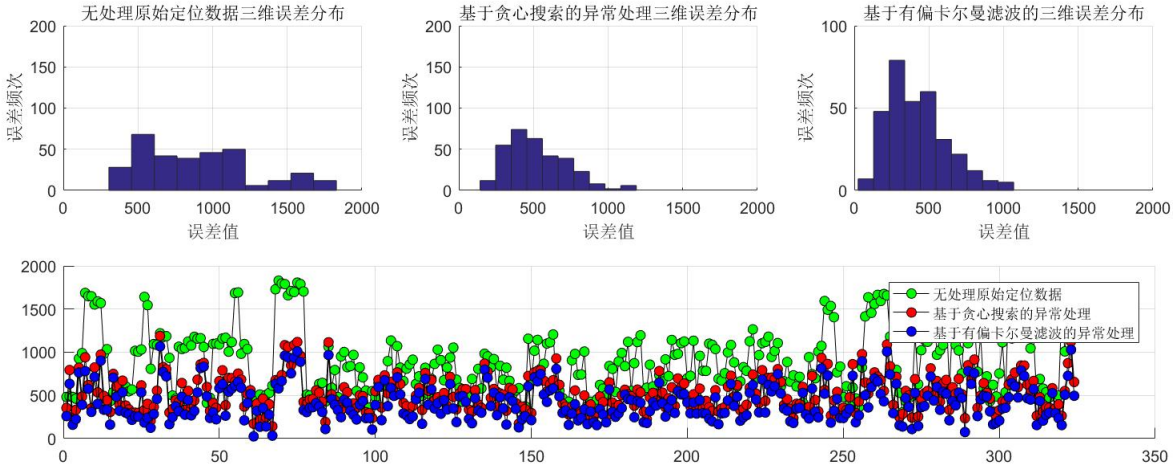


图 4.18　NLOS 环境下的 3 维定位误差分布

可以看出未对异常数据进行处理时，最终的定位结果的误差及靶点的坐标位置波动都较大。以异常数据直接通过正常数据模型进行坐标估计的结果作为对比，增加两种不同的异常数据校正方式后，对于异常数据的坐标估计误差在 1/2/3 维上均有所降低，误差分布的峰值也更加靠近零值。对于包含异常校正的模型，基于有偏卡尔曼滤波的方式相比基于贪心误差搜索的方式在性能上有所提升，而且由于贪心搜索算法是针对于本题目对应的异常情况进行设计，因此可能在实际应用中，需要通过优化改进来适配更加复杂多变的场景。基于有偏卡尔曼滤波的算法普适性更强，且能够提供较好的数据校正功能，将异常数据点定位误差下降到 10cm 以内。

表 4.8　实验环境 1 附件 2 靶点位置坐标

序号	X 轴	Y 轴	Z 轴	是否为正常测距
1	1165	668	967	是
2	3168	1721	1025	是
3	2726	1170	1099	是
4	2460	1009	1977	是
5	1480	2552	1759	是
6	1165	668	967	否
7	3168	1721	1025	否
8	2726	1170	1099	否
9	2460	1009	1977	否
10	1480	2552	1759	否

5 问题三——定位模型在不同场景下的应用

5.1 问题三分析

在问题二中分别针对特定的环境在信号有干扰和无干扰的情况下建立了定位模型，题目要求将定位模型应用到新的实验场景下，由于我们在定位模型的设计时，保留了实验环境的空间位置等信息，因此只需要进行相应的参数替换，便可以完成模型的迁移，流程如下所示。

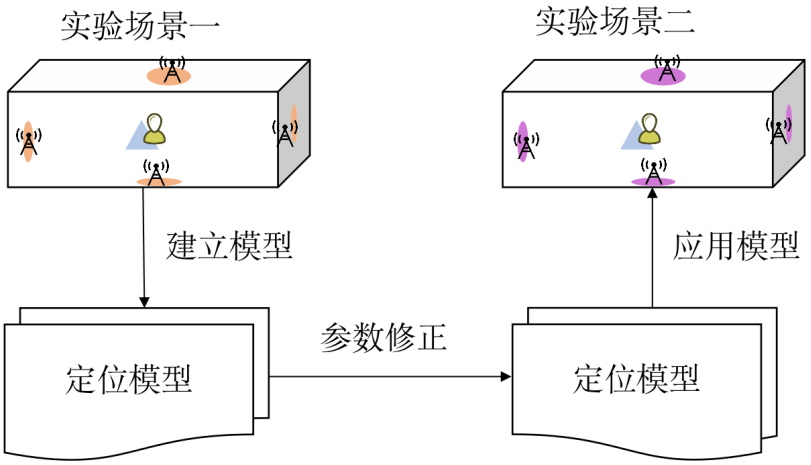


图 5.19　定位模型迁移流程图

5.2 问题三结果

修改锚点数据后借助问题二中的空间定位算法可以得到问题三实验环境中的 10 个靶点位置坐标：

表 5.9　实验环境 2 中的靶点位置坐标（单位：mm）

序号	X 轴	Y 轴	Z 轴	是否为正常测距
1	3660	2237	1396	是
2	4196	1726	1093	是
3	3172	1745	1311	是
4	2591	1884	2151	是
5	586	67	1489	是
6	3649	2222	1116	否
7	4241	1747	1214	否
8	3243	1793	1445	否
9	2600	1882	2105	否
10	539	119	738	否

可视化结果如图所示：

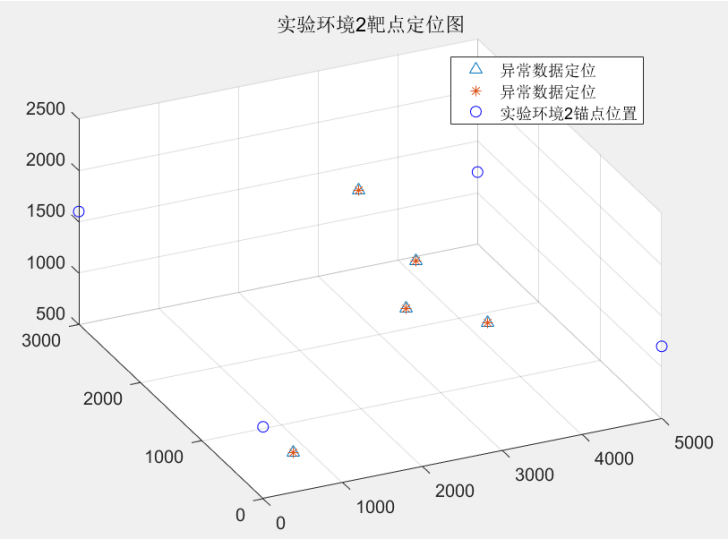


图 5.20　实验环境 2 附件 3 坐标估计

6 问题四——分类模型的建立与求解

6.1 问题四分析

在问题二中分别针对信号有干扰和无干扰的情况下建立了定位模型，但是 UWB 在采集数据时并不知道信号有无干扰，因此需要建立分类模型识别信号有无干扰。在实际定位场景中，影响 UWB 系统的测距精度是多方面的，包括自身系统硬件的性能，人为因素和

环境因素等，其中主要有非视距误差 [4]、多径效应误差和系统硬件带来的时钟漂移误差，如图 6.21 所示。

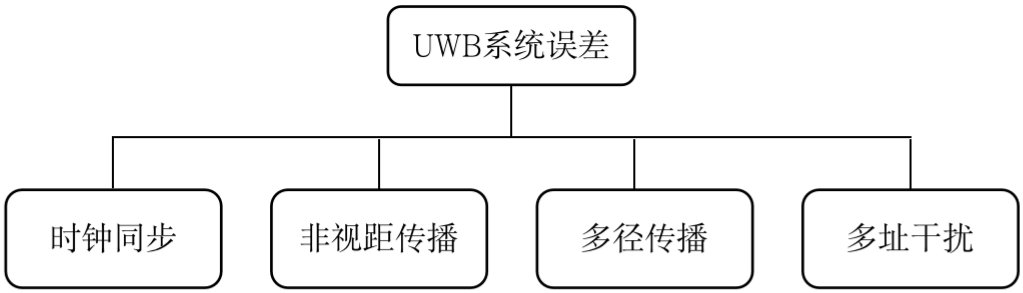


图 6.21　UWB 系统误差

在该实验环境下，结合题目要求我们主要考虑非视距误差，所谓的非视距就是指在有障碍物遮挡的情况下，信号的传播并不是按照理想的状态下的视距传播（Line of Sight，LOS），如图 6.22-6.23 所示，而是信号经过了反射、衍射或者折射，将这种传播形式称为非视距传播 NLOS，在 NLOS 的条件下，障碍物材质的种类及尺寸的不同都会对信号的传播带来不同的印象，最终导致测距结果的波动。

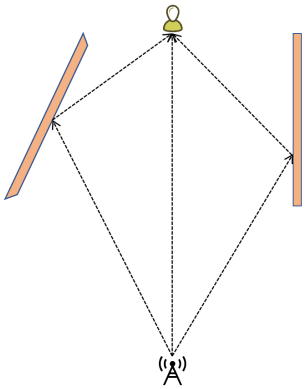


图 6.22　视距传播示意图

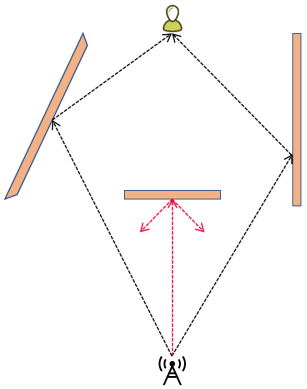


图 6.23　非视距传播示意图

针对 NLOS 造成的干扰，一个很直观的解决方案就是采用误差建模的方法，通过建立数学模型的方法识别是否在测量过程中存在遮挡物，当测量中出现了符合误差模型的数据，我们则认为该组数据受到了干扰，另外我们还可以使用分类和聚类的方法，通过机器学习的方法自动学习受到干扰的异常数据的分布，进而实现对异常数据的分类和识别。

6.2 基于误差建模的分类算法

在基于 UWB 的定位技术中，定位误差主要来源于靶点到锚点的测距误差，测距误差主要分为随机误差和系统误差两类，随机误差是指在实际测距的过程中，测量值围绕着真实值波动而产生的误差，主要由 UWB 设备的不稳定，时钟漂移等因素导致的，在第二问

中，我们使用了加权移动平均滤波算法和卡尔曼滤波算法，有效地减小了随机误差；在问题分析中我们提到系统误差主要是指由于障碍物遮挡导致的非视距误差，传统滤波算法对此毫无帮助，因此我们不需要对随机误差进行建模，只需识别测量数据中是否包含有系统误差。

在第二问中，我们对同一个靶点在正常（无干扰）和异常（有干扰）情况下的测量结果进行了探索，发现靶点在锚点 A0，A4 的测距值基本上一致，也就是说即使是在有干扰情况下采集的数据，其表现也是无干扰的；然而在锚点 A1，A2 的测距值，在不同的时间出现了明显的差距，这说明靶点在实验过程中，只在某一个方向上出现了障碍物的遮挡，而在其他方向并没有出现障碍物遮挡的情况。经过我们的分析，题目中所提供的数据基本上都符合这样的规律。

事实上，该实验环境中采集到的数据是与我们的实际生活场景一致的，为了尽量避免由锚点放置位置导致的系统误差，会将锚点放置的尽可能分散，这就意味着在同一时间障碍物能够同时遮挡多个锚点的可能性是很低的。另外，障碍物在移动的过程中也会导致测距结果的动态变化。在定位模型的建立部分，我们分析了二维和三维场景下的多边定位的求解方法，其中对于三维场景而言，当靶点 T_i 在以锚点 A_j 为球心，以距离 d_{ij} 为半径的球面上，则满足下列方程：

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \tag{6.36}$$

(x_i, y_i, z_i) 为靶点 T_i 的坐标，(x_j, y_j, z_j) 为锚点 A_j 的坐标，公式 4-1 中共有三个未知量，从代数的角度考虑，最少需要三个方程就可以进行求解，即最少需要 3 个基站。从几何的角度考虑，两个球面相交得到一个圆环，圆环与第三个球面相交就可以获得两个确定的解，这两个解是关于三个基站所在平面对称，再结合实际情况即可得到多边定位的唯一解。通过以上分析，我们设计了基于误差建模的分类算法，其流程如图 6.24 所示。

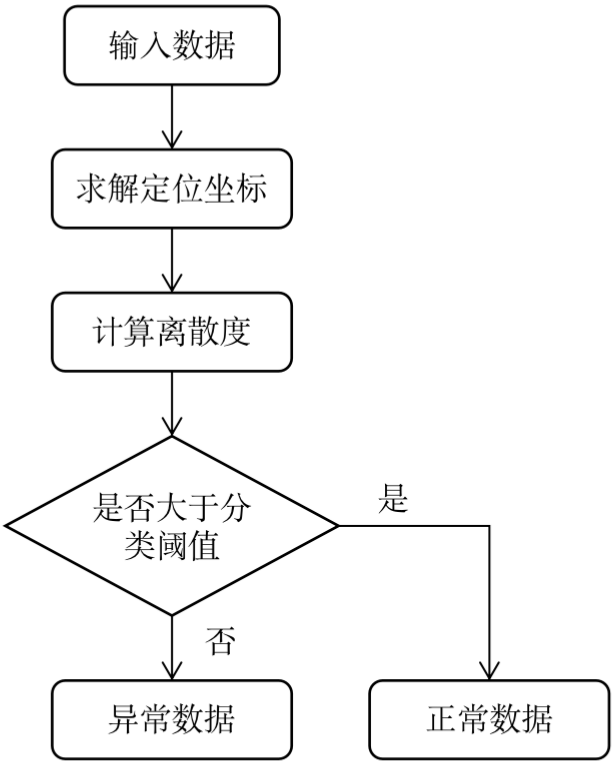


图 6.24　基于误差建模的分类算法

基于误差建模的分类算法的基本思想是无干扰情况下由任意三个锚点计算得到的靶点坐标和由四个锚点计算得到的靶点坐标理论上是一致的，由于随机误差的存在会产生些许差异，但是在有干扰情况下，同样的计算方法，由于系统误差的存在得到的靶点坐标与准确值相比会出现较大的偏差，我们定义离散度 V 作为分辨正常值和异常值的标准，靶点 T_i 的离散度的计算方法如下：

$$V_i = \frac{1}{M-1}\sum_{m=1}^{M}\sqrt{(x_m - x_i)^2 + (y_m - y_i)^2 + (z_m - z_i)^2} \tag{6.37}$$

其中 (x_i, y_i, z_i) 为靶点 T_i 根据四个锚点的测距值计算的坐标，其中 (x_m, y_m, z_m) 为靶点 T_i 根据三个锚点的测距值计算的坐标，M 代表三个锚点的组合数 $M = C_4^3$。在计算训练集中各个靶点的离散度后，就可以得到正常和异常数据分类的阈值 V_t，定义变量 $label = 1$ 代表数据是否是受到干扰的，那么对于靶点 T_i 的测距结果就可以按照以下公式进行评估和分类：

$$\text{label}_\text{i} = \begin{cases} 0, & \text{若} V_i < V_t \\ 1, & \text{若} V_i > V_t \end{cases} \tag{6.38}$$

6.3 基于机器学习的分类算法

机器学习分类通过训练集进行学习，建立一个从输入空间 X 到输出空间 Y（离散值）的映射 [7]。按输出类别（标签）不同，可以分为二分类（Binary Classification）、多分类

（Multi-Class Classification）、多标签分类（Multi-Label Classification），本题中的正常和异常数据的识别就是一个典型的二分类任务。常用的分类算法有逻辑回归、KNN、决策树、随机森林、朴素贝叶斯等，本文将几种典型的机器学习算法应用到了 UWB 测距数据异常识别中，并验证在该场景下的有效性。

1、逻辑回归

逻辑回归是在线性回归基础上衍生出来的的用于分类的模型，由于线性回归的结果输出是一个连续值，而值的范围是无法限定的，直接使用线性回归是无法作为分类的判别依据的。需要使用一个逻辑斯特函数（如 sigmoid 函数）将连续的输出值映射到 (0,1) 中，这个概率值就可以作为模型判断分类结果的依据。

$$S(x) = \frac{1}{1+e^{-x}} \tag{6.39}$$

2、KNN

KNN 是最近邻分类，是一种常用的数据聚类方法。其核心思想是在特征空间中寻找匹配的 k 个最近点，根据 k 个最近点的投票来判断当前样本点应该属于哪一个类别。KNN 算法中，所选择的邻居都是已经正确分类的对象，常用的距离函数由曼哈顿距离、欧氏距离和闵可夫斯基距离。

设特征空间 $\mathcal{X}$ 是 n 维实数向量空间 $R_n, x_i, x_j \in \mathcal{X}$，$x_i = \left(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)}\right)^T, x_j = \left(x_j^{(1)}, x_j^{(2)}, \ldots, x_j^{(n)}\right)^T, x_i, x_j$ 的 L_p 距离定义为:

$$L_p\left(x_i, x_j\right) = \left(\sum_{l=1}^{n}\left|x_i^{(l)} - x_j^{(l)}\right|^p\right)^{\frac{1}{p}} \tag{6.40}$$

这里 $p \geq 1$，当 $p = 1$ 时, 称为曼哈顿距离 (Manhattan distance), 公式为:

$$L_1\left(x_i, x_j\right) = \sum_{l=1}^{n}\left|x_i^{(l)} - x_j^{(l)}\right| \tag{6.41}$$

当 $p = 2$ 时, 称为欧式距离 (Euclidean distance), 即

$$L_2\left(x_i, x_j\right) = \left(\sum_{l=1}^{n}\left|x_i^{(l)} - x_j^{(l)}\right|^2\right)^{\frac{1}{2}} \tag{6.42}$$

当 $p = \infty$ 时, 它是各个坐标距离的最大值, 计算公式为:

$$L_\infty\left(x_i, x_j\right) = \max_l\left|x_i^{(l)} - x_j^{(l)}\right| \tag{6.43}$$

3、朴素贝叶斯

朴素贝叶斯分类器建立在贝叶斯定律和特征之间条件独立假设的基础上，贝叶斯定律如下所示：

$$P\left(B_i \mid A\right)=\frac{P\left(B_i\right) P\left(A \mid B_i\right)}{\sum_{j=1}^{n} P\left(B_j\right) P\left(A \mid B_j\right)} \tag{6.44}$$

其中 $P(\cdot)$ 为时间发生的概率，$P(A|B)$ 则表示在 B 发生的情况下 A 发生的概率。

特征之间互相独立的假设认为即使这些特征相互依赖，或者依赖于其他特征的存在，朴素贝叶斯算法都认为这些特征都是独立的。朴素贝叶斯分类器通过已给定的训练集，学习从输入到输出的联合概率分布，再基于学习到的模型，输入求出使得后验概率最大的输出。

5、支持向量机

支持向量机（SVM）把分类问题转化为寻找分类平面的问题，把样本空间映射到一个高维乃至无穷维的特征空间中，并通过最大化分类边界点距离分类平面的距离来实现分类。SVM 学习的基本想法是求解能够正确划分训练数据集并且几何间隔最大的分离超平面。如下图所示，$\omega x+b=0$ 即为分离超平面，对于线性可分的数据集来说，这样的超平面有无穷多个，但是几何间隔最大的分离超平面却是唯一的。

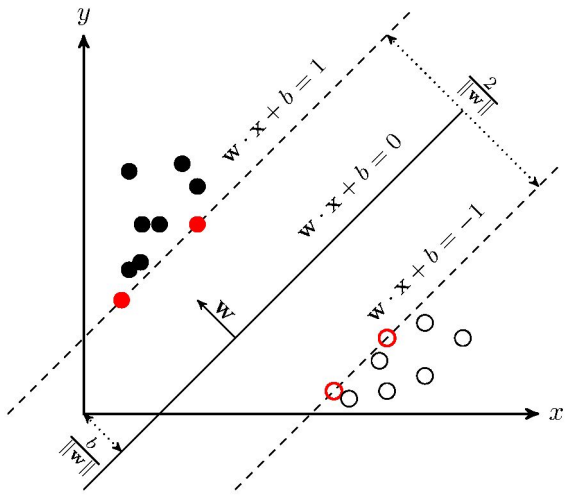


图 6.25　支持向量机示意图

将数据从低维空间映射到高维空间是 SVM 核函数的主要思想,SVM 利用内积核函数代替向高维空间的非线性映射，可以很好地减少计算量，并且将数据投影到高维空间后，数据就可能变得更可分，在某种程度上避免了“维数灾难”。

6、决策树

决策树是用树状结构构建的一类分类或回归模型。从根结点开始，算法不断通过一定的条件来将数据集拆分成更小的子集分支来划分数据集。最终长成具有决策节点（包括根节点和内部节点）和叶节点的树。如图所示。

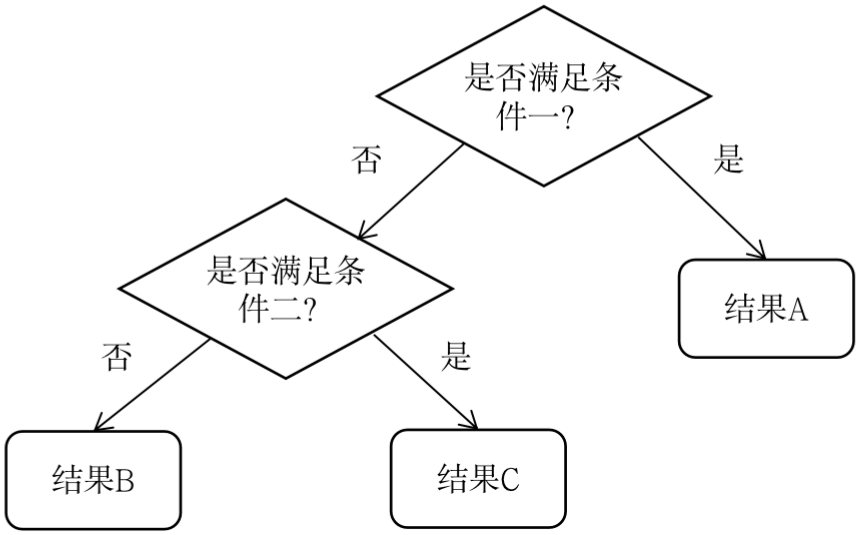


图 6.26　决策树示意图

随着树的深度不断增加，分支节点的子集越来越小，所需要提的问题数也逐渐简化。当分支节点的深度或者问题的简单程度满足一定的停止规则时, 该分支节点会停止分裂，此为自上而下的停止阈值（Cutoff Threshold）法；此外还有自下而上的剪枝（Pruning）法。在分类预测时，输入数据经过决策树内部的各个决策节点，按照不同的属性值进入不同的分支，直到到达叶子节点完成分类。

7、随机森林

随机森林指通过多颗决策树联合组成的模型随机森林是由很多决策树构成的，不同决策树之间没有关联。

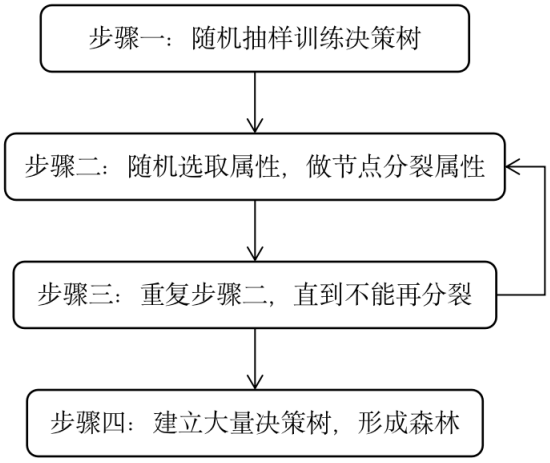


图 6.27　随机森林生成步骤

当执行分类任务时，新的输入样本进入，会输入到森林中不同到决策树内，按照决策树内的属性判断得到多个分类结果，最后采用少数服从多数的投票方式决定最终的分类结果。

8、多层感知机

多层感知器（Multilayer Perception,MLP）是一种基于前向传播的人工神经网络，模仿人类的感受器神经元将信号逐层乡下传播。多层感知机的基本结构一般由三层组成：第一层为输入层，中间为隐藏层，隐藏层节点可以自行设计，最后一层为输出层。如果有 mlp 前后连接就是神经网络的结构。其基本结构如图所示。

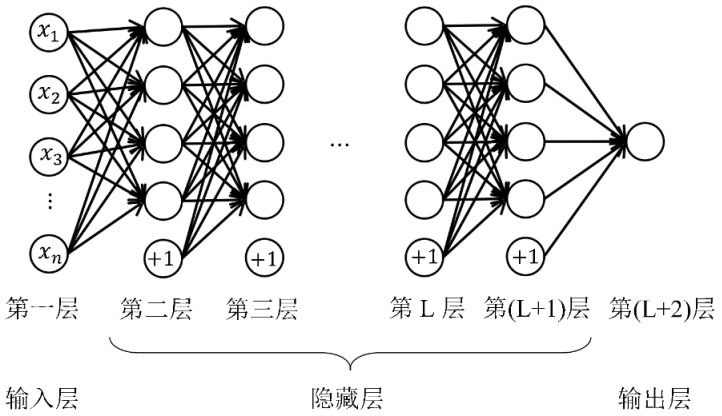


图 6.28　多层感知机结构图

除了输入节点，每个节点都是一个带有非线性激活函数的神经元。多层感知机首先学习数据的模式，然后使用权重存储数据，并使用算法来调整权重并减少训练过程中的偏差，即实际值和预测值之间的误差，通常使用反向传播算法来训练多层感知机，其主要优势在于其快速解决复杂问题的能力。

6.4 基于集成学习的分类算法

集成学习（Ensemble Learning）是一种能在各种的机器学习任务上提高准确率的强有力技术，其通过组合多基本分类器来完成学习任务。对于干扰检测和分类的场景，我们基于数据驱动的方法建立了基于误差模型的分类算法和基于机器学习的分类算法，单个模型很容易出现过拟合或者欠拟合的情况，并且各个模型在设计的时候就有自己本身的优点和缺点，因此我们可以通过基于集成学习的模型融合的方法达到取长补短的效果，提高模型的分类能力。对于二分类模型而言，常用的融合方案有 Voting、Stacking、Blending 和 Booster。

1、Voting

Voting 也就是投票法，采取少数服从多数的原则，对多个学习器的预测结果进行投票，可分为普通投票法和加权投票法。其中加权的权重可以人工主观设置或者根据模型评估分数来设置权重，投票法通常需要 3 个及以上模型，并且为了避免投票结果的偏差，需要保证模型的多样性。对于本任务而言，我们采用准确率赋权的方法进行投票，计算方法如下

所示：

$$H(x) = \arg\max_{x} \sum_{i=1}^{T} a_i h_i^j(x) \tag{6.45}$$

其中 $h_i^j(x)$ 为各个分类器输出的预测类别，满足 $h_i^j(x) = \{0, 1\}$；a_i 为投票的权重，满足 $a_i \geq 0, \sum_{i=1}^{T} a_i = 1$。

2、Bagging

在 Voting 方法中，采用相同的全部样本训练每一个基分类器的方案，而 Bagging 方法则是使用全部样本的一个随机抽样，每个分类器都是使用不同的样本进行训练，其他的地方二者完全一致。这样就避免了模型训练结果的同质化问题，提高了不稳定模型准确率的同时，降低了过拟合的程度。其基本流程如图所示。

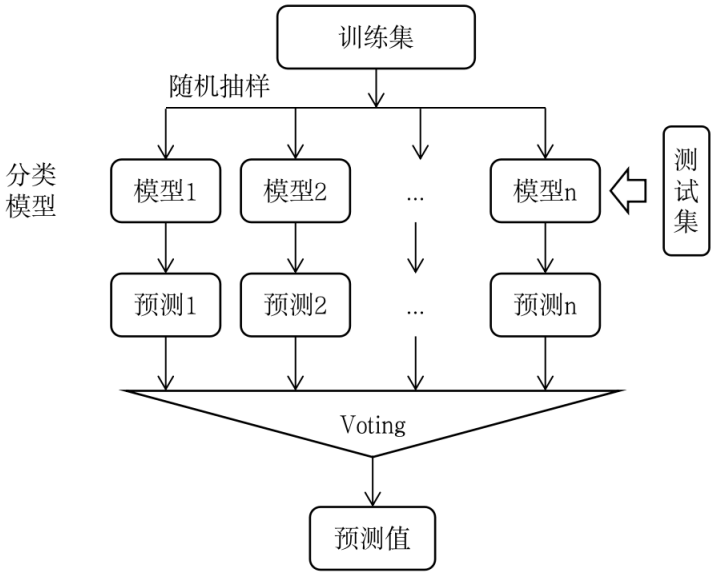


图 6.29　Bagging 框架图

3、Stacking

stacking 是一种分层模型集成框架。将若干基学习器获得的预测结果，将预测结果作为新的训练集来训练一个学习器。以一个两层的 Stacking 集成框架为例，第一层由多个基学习器组成，输入为原始训练集，第二层的模型则是以第一层基学习器的输出作为训练集进行再训练，从而得到完整的 Stacking 模型。为了防止过拟合，我们采用如图所示的双层 Stacking 集成框架，在第二层模型中使用简单的线性模型。

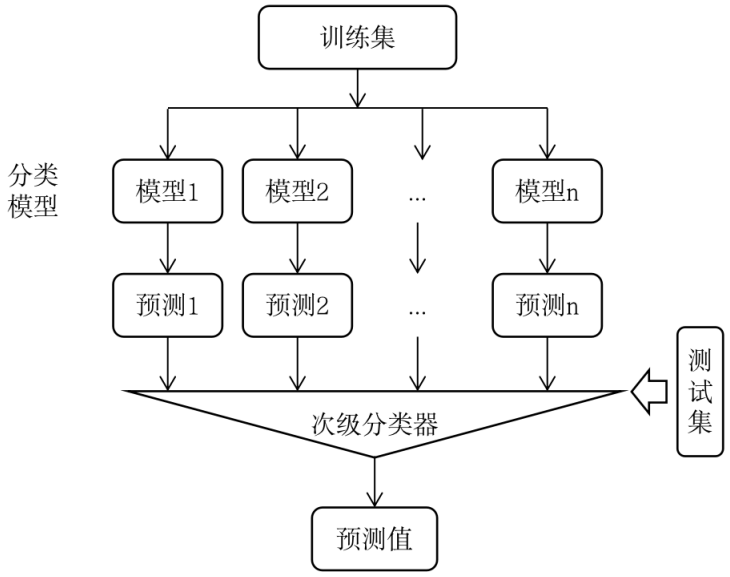


图 6.30　Stacking 框架图

4、Blending

Blending 方法和 Stacking 在模型结构上是基本一致的，其主要不同在于，Stacking 是使用全量的数据去训练第一层模型，然后将第一层模型的输出作为第二层模型的输入进行模型之间的融合；而 Blending 是在训练第一层模型时采用部分数据集，将第一层模型在另一部分数据集上的预测结果输入到下一层模型中。这样避免了基学习器和元学习器使用相同的训练数据而导致的信息泄漏的问题，Blending 方法的基本流程如图所示。

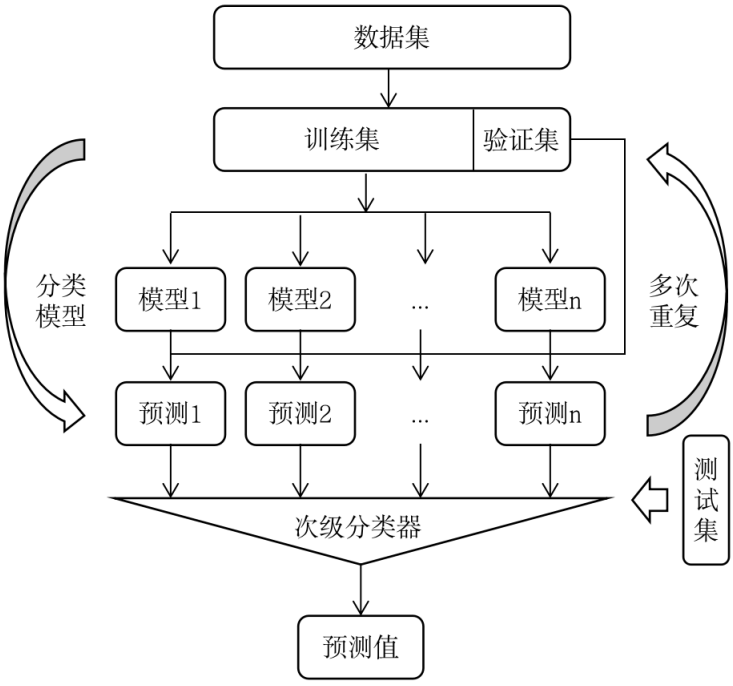


图 6.31　Blending 框架图

5、Boosting

在 Bagging 方法中，各分类器之间没有依赖，彼此是并行关系，而在 Boosting 方法中，各分类器之间有依赖关系，模型是串行的。Boosting 方法中最常用就是 AdaBoost(Adaptive Boosting), 即自适应增强算法，它重点关注那些被弱分类器分错的样本，前一个基学习器分错的样本会得到加强，加权后的全体样本再次被用来训练下一个基学习器。同时，在每一轮中加入一个新的弱分类器，直到达到某个预定的足够小的错误率或达到预先指定的最大迭代次数。

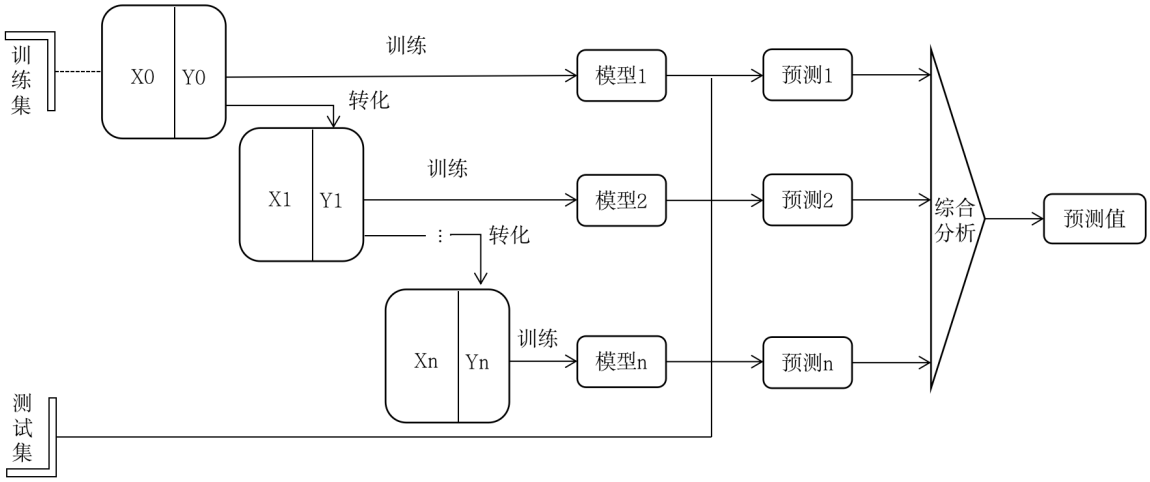


图 6.32　Boosting 框架图

6.5 算法性能对比与结果分析

以上我们介绍了基于误差建模的分类算法、基于机器学习的分类算法和基于集成学习的分类算法，接下来本文将结合实验场景和具体数据对各种方法从时间代价、准确率等指标进行对比，以选取最优的算法作为作为最终的误差分类算法。

1、测试说明

（1）数据修正

在问题一中，我们对数据进行的清洗，去除了了数据中的异常点，提高了数据的质量，这将大大提高我们定位模型的精度。其中在对正常数据（未受到干扰）清洗时我们采用拉依达准则（3σ 准则）将偏离测距中心的点做了剔除处理，保证了数据的集中性；然而我们在对异常数据（受到干扰）可视化时发现，部分异常情况下的测量数据与正常数据是基本一致的，也就是说，异常文件中可能也会存在正常数据，这会对于我们的分类模型产生误导性，导致分类性能的下降。

为了进一步提高分类模型的准确率,我们按照(3σ 准则)对异常情况下测量得到的数据进行了筛查,对于靶点 T 在有干扰和无干扰情况下的测试序列分别为 $D_1 = \{d_1^1, d_1^2, \cdots, d_1^n\}(n =$

$1,2,3)$ 和 $D_0 = \{d_0^1, d_0^2, \cdots, d_0^n\}(n = 1,2,3)$，按照下式计算序列 D_0 的均值 μ_0 和标准差 σ_0。

$$\mu_0 = \frac{1}{N}\sum_{n=1}^{N} d_0^n \tag{6.46}$$

$$\sigma_0 = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}\left(d_0^n - \mu_0\right)^2} \tag{6.47}$$

在对异常数据中的正常数据筛选时，按照下面的公式进行判断：

$$D_0 \leftarrow \begin{cases} D_0 + \{d_0^n\}, \text{ if } \mu_0 - 3\sigma_0 \le d_1^n \le \mu_0 + 3\sigma_0 \\ D_0, \text{ otherwise} \end{cases} \tag{6.48}$$

$$D_1 \leftarrow \begin{cases} D_1 - \{d_0^n\}, \text{ if } \mu_0 - 3\sigma_0 \le d_1^n \le \mu_0 + 3\sigma_0 \\ D_1, \text{ otherwise} \end{cases} \tag{6.49}$$

具体地，只有当同一靶点异常文件中四个锚点的测距值都符合正常文件测距值的 3σ 准则时，我们才将之视为正常数据。经过分析发现，这类错标的数据并不多，大概占到异常数据的 1%，如下图所示。

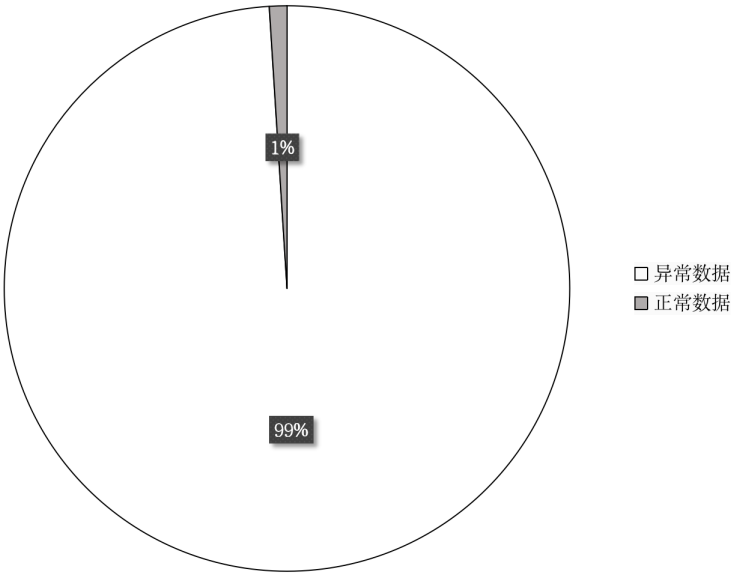


图 6.33　异常文件中正常数据占比

在后续的实验环节我们将对数据清洗后的原始数据及数据修正的数据分别进行实验，对比观察是否对分类结果有所影响。

（2）五折交叉验证在机器学习的建模过程中，通常的做法是将数据分为训练集和测试集，二者是完全独立的数据，测试集完全不参加训练，只用于模型的评估，在训练的过程中，经常会出现过拟合的问题，所谓的过拟合就是模型很好对训练集中的数据进行了建模，

但是在测试集上的泛化性却很差。为了避免这种情况，可以在训练数据中再分出一部分数据作为验证数据，用来评估模型的训练效果。

验证数据取自训练数据，但不是参与训练，这就可以相对客观的评估模型对训练集之外的数据的拟合程度。本文在模型训练时，采用五折交叉验证的方法，将原始数据分为 5 组，其中一个子集作为验证集，其他四个子集作为训练集，这样就得到了五个模型，分别评估在验证集中的表现，通过加权平均的方法就得到模型整体的表现。

2、性能对比

我们使用 python 中的 sklearn 机器学习包，自己实现了基本的机器学习方法和模型集成学习方法，在一台 macbookpro 16G M1 设备上分别对原始数据和修正后的数据进行了训练和验证，二者的分类准确率分别用分类准确率和分类准确率* 标识，结果如表所示。

表 6.10　不同分类算法的表现

序号	算法	运行时间/ms	分类准确率	分类准确率*
a	误差建模	2000	0.76	0.77
b	逻辑回归	400 ± 32.1	0.65	0.65
c	KNN	859 ± 47.1	0.80	0.80
d	朴素贝叶斯	469 ± 37.9	0.74	0.75
e	支持向量机	454 ± 36.2	0.80	0.80
f	决策树	508 ± 44.9	0.76	0.76
g	随机森林	738 ± 12	0.67	0.68
h	多层感知机	564 ± 44.9	0.71	0.71
i	集成-Voting	1100 ± 19.3	0.75	0.77
j	集成-Stacking	1341 ± 54.6	0.88	0.87
k	集成-Blending	1723± 78.1	0.85	0.85
l	集成-Boosting	1422 ± 34.9	0.81	0.81

从上表中我们可以看出，就运行时间而言，误差建模所需要的运行时间最长，因为他需要遍历数据计算判别阈值才可以做出判断；运行时间最短的是逻辑回归。就准确率而言，效果最好的是集成学习中的 Stacking 方法，达到了最高 0.88 的准确率，效果最差的是逻辑回归，其准确率仅有 0.65；基于误差建模的异常数据识别的准确率在 0.77，基于机器学习异常分类算法中表现最好的好似支持向量机，达到了 0.80 的准确率。另外，我们还发现，数据修正对模型的效果有一定的提升，但是非常有限，平均提高 0.01，原因大概在于数据异常文件中的正常数据非常少，只占到 1%，因而数据修正对模型的增益较低。

3、结果分析

通过性能对比，我们评估了不同模型的分类效果，接下来我们将对附件 4 中的十组数据进行分类，为了避免模型的过拟合现象以及模型测试过程中的偶然误差，我们分别使用基于误差建模的分类算法 (a)、支持向量机 (e) 和集成学习-Stacking 模型 (j) 进行预测，预测结果如下所示。

表 6.11　不同分类算法的预测结果

算法	组 1	组 2	组 3	组 4	组 5	组 6	组 7	组 8	组 9	组 10
a	0	1	0	1	1	0	0	1	1	0
e	0	1	0	1	1	0	0	0	1	0
j	0	1	0	1	1	0	0	1	1	0

从上表中可以看出，误差建模和集成学习的方法预测的结果是一致的，SVM 预测的结果在八组中出现了差异，经过综合分析，我们对附件 4 中的数据的预测结果依次是：正常、异常、正常、异常、异常、正常、正常、异常、异常、正常，即正常数据编号为 1、3、6、7、10，异常数据编号为 2、4、5、8、9。

7 问题五——运动轨迹定位

7.1 问题五分析

运动轨迹定位是 UWB 重要应用之一，可以通过计算被测物体的位置坐标，获得其在一定时间范围内的运动轨迹，适合于室内静止或移动的人员、物体的定位跟踪与导航。在前面的问题中，我们完成了在信号有干扰和无干扰的条件下定位模型的建立，可以通过测量得到靶点到四个锚点的距离得到靶点的精确定位坐标，其误差基本可控制在 1CM 之内。

在本题的场景下，我们预先并不知道所采集的数据是否受到了干扰，因此首先需要使用问题四中建立的分类模型识别动态靶点数据是否存在干扰因素，然后再使用问题二中提出的误差搜索及有偏卡尔曼滤波的方式针对异常测量数据下的靶点位置进行校正，并通过在问题二中表现更优的 Chan 定位方式得到精确的定位值。

在本节中，题目要求为绘制靶点的运动轨迹，因此除了在异常数据分类后针对于静态节点的测量数据进行校正外，需要将测量时间戳作为影响模型的变量。因此对时间维度进行建模，利用上一时刻的状态对下一时刻进行估计，采用适用于非线性状态估计的扩展卡尔曼滤波优化获得靶点在空间中的运动轨迹模型。

7.2 LOS/NLOS 环境中的移动目标定位模型

由于问题五提供的数据包括存在 NLOS 干扰的异常数据和 LOS 环境中的正常数据，因此在进行靶点定位之前，我们首先将获得的测二中对于每个锚点的拟合曲线进行随机误差矫正。之后根据问题四中的异常数据分类模型对测量的数据进行分类。由于该问题中对于靶点运动轨迹的估计是在时间维度上的，因此我们建立了针对非线性空间的扩展卡尔曼滤波模型，优化问题二中设计的三维空间定位模型，得到了较为精确的时间序列上靶点估计位置，并在空间坐标系中绘制了靶点运动轨迹曲线。

7.3 基于扩展卡尔曼滤波的轨迹校准

卡尔曼滤波是一种常用的线性最小方差统计估算方法，扩展卡尔曼滤波是标准卡尔曼滤波器的一种拓展，利用 Talor 级数展开将非线性系统线性化，一般取一阶导和而阶导就可以很好地近似为线性方程。

7.3.1 原始数据测量误差校正

由于在本任务环境中，靶点的运动轨迹为非线性，因此采用适用于非线性空间的扩展卡尔曼滤波方法对于估计的靶点位置进行过滤。由于实验环境与问题二相同，因此借助问题二中拟合的四个锚点随机误差曲线对附件 5 的数据进行测量校正。

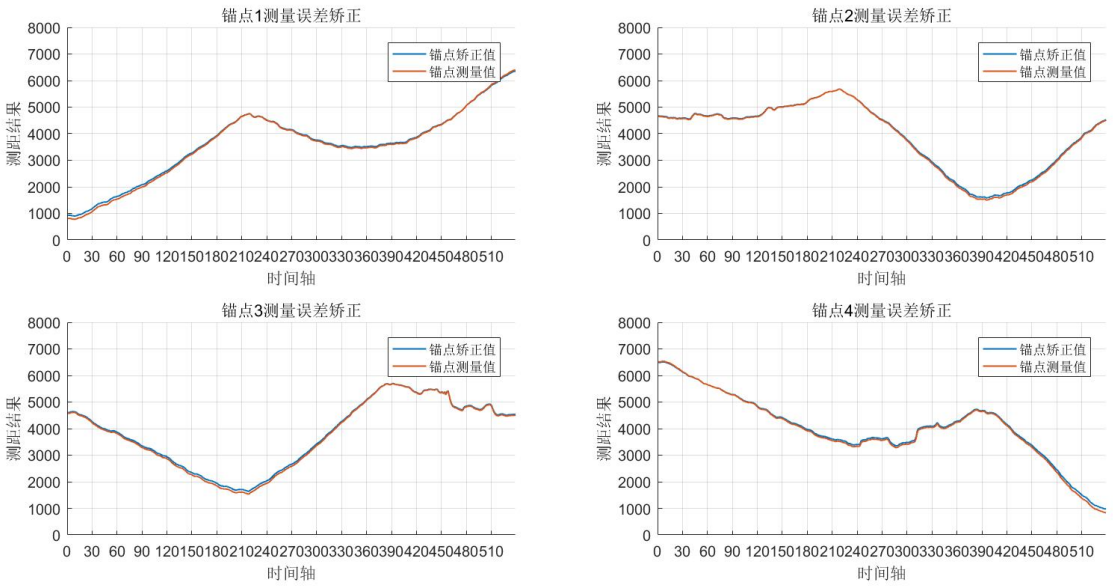


图 7.34　四维数据矫正

可以看出，校正后数据发生了微小的偏移，该偏移代表锚点在当前实验环境中可能由于环境因素产生的测量偏差。

7.3.2 等采样间隔线性插值

在扩展卡尔曼滤波模型中，模型能够通过上一时刻的状态估计下一时刻的位置，因此时刻之间应当是均匀变化的，状态之间的时间差 d_t 为自定义参数。由于需要符合不同状态转移时间差的要求，我们在将数据输入模型之前进行了等采样间隔的线性插值，插值后结果如图 7.35 所示。

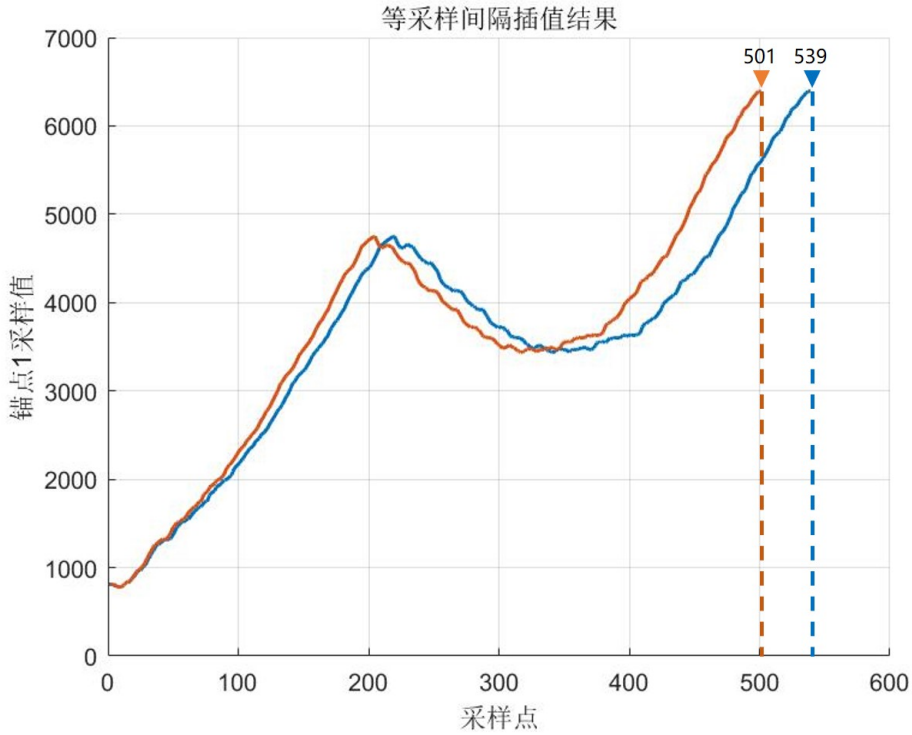


图 7.35　锚点 1 测量数据等采样间隔插值

7.3.3 数据分类及定位

将插值后的数据进行分类得到正常/异常数据标签，根据标签选择适合的定位算法。对于正常数据，我们采用了在问题二中表现较好的三维 Chan 模型；对于异常数据，我们采用了能够适应多种实验环境的有偏卡尔曼滤波方法进行数据校正。对正常和异常数据均进行定位后，通过扩展的卡尔曼滤波消除运动轨迹误差。通过上述在数据校正及坐标估计两个阶段使用不同的卡尔曼滤波的方法，对 NLOS 环境中产生的误差进行修正，能够得到较为准确且平滑的靶点运动轨迹。

7.4 结果分析

在本问题中，我们采用了窗口为 5 的滑动平均和扩展卡尔曼滤波两种方法对异常数据定位模型输出的估计位置进行滤波处理，以消除 NLOS 环境中的定位误差，以此来验证扩展卡尔曼滤波在本问题场景中的优越性。

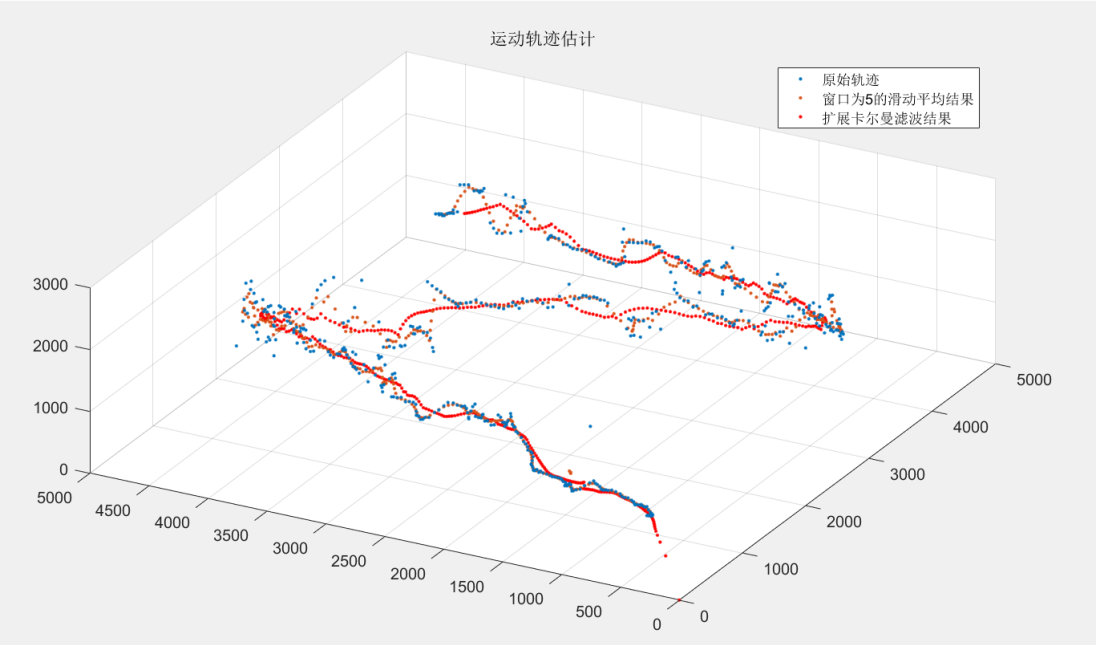


图 7.36　三维空间中的运动轨迹

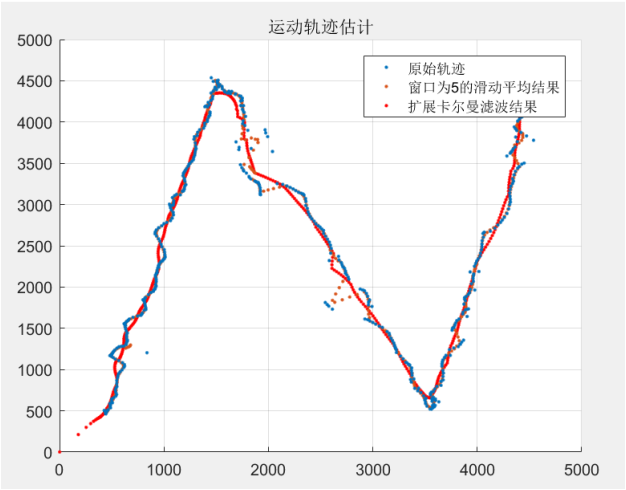


图 7.37　XY 平面运动轨迹

如图 7.37 所示，原始轨迹为仅经过异常数据定位模型的位置估计后绘制出的靶点运动轨迹。虽然能够较为直观的看出靶点运动趋势，但是定位坐标存在相对较大的偏移，且在部分区域的位置估计值出现离群点，存在的数据波动较大。使用窗口为 5 的滑动平均后，在一定程度上靶点运动轨迹的偏移减少，坐标离群点被平滑处理。但在原始靶点坐标出现较大偏移时，滑动窗口的预测结果并不是十分准确。不仅如此，如果通过选择更大的窗口来平滑原始靶点坐标，会导致定位敏感度的下降。针对该问题我们采用的扩展卡尔曼滤波算法能够估计出较为稳定的靶点运动轨迹 (图 7.37 中的红色轨迹)，在避免离群点和轨迹波动的情况下对靶点的每个位置进行较为准确的估计，在保证定位敏感度的情况下保证了定

位的精确程度。从 X_Y 截面来看，靶点的运动轨迹为“N”型，而且能够明显看出通过扩展卡尔曼滤波，靶点的运动轨迹明显得到了平滑，离群点和异常点数量减少。这也验证了扩展卡尔曼滤波在该问题中的有效性。

参考文献

[1] 张宝军, 田奇, 王珩, 等, 基于 CNN 和在线学习的 UWB 室内定位算法, 传感技术学报(4): 511-516, 2020.

[2] 赵红梅, 赵杰磊, 超宽带室内定位算法综述, 电信科学, 34(09):130-142, 2018.

[3] 陈燕, 基于 UWB 的高精度室内三维定位技术研究, 电子科技大学.

[4] 面向非视距环境的室内定位算法, 电子学报(5 期):1174-1179.

[5] 房秉毅, 基于超宽带技术的室内定位系统, 电子技术应用(07):124-127, 2006.

[6] 赵永翔, 周怀北, 陈淼, 等, 卡尔曼滤波在室内定位系统实时跟踪中的应用, 武汉大学学报 (理学版)(06):77-81, 2009.

[7] 孔欣然, 机器学习综述, 电子制作, 000(024):82-84,38, 2019.

[8] 曹霞, 牛顿法在求解计算中的应用研究, 价值工程, 32(17):196-196, 2013.

[1, 2, 3, 4, 5, 6, 7, 8]

附录 A　问题结果

A.1　问题一结果

1、处理前后数据量对比

表 1.12　处理前后数据量

文件类型	原始数据量	处理后数据量
正常文件	77122	37028
异常文件	79176	65824

“正常数据”所有数据文件和“异常数据”所有数据文件最后各保留 37028 组和 65824 组数据。

2、24-正常数据预处理结果

A0	A1	A2	A3
3280	4660	2600	3910
3280	4670	2600	3920
3280	4670	2600	3910
3280	4660	2600	3900
3280	4660	2610	3920
3280	4660	2600	3920
3280	4660	2610	3910
3290	4660	2600	3920
3290	4650	2600	3910
3290	4650	2600	3920
3300	4650	2590	3920
3290	4660	2590	3910
3290	4660	2590	3900
3280	4660	2590	3890
3280	4650	2590	3880
3280	4660	2590	3880
3280	4660	2590	3920
3280	4660	2590	3910
3270	4670	2600	3900
3270	4660	2600	3910
3290	4660	2600	3910
3280	4660	2610	3900
3290	4660	2610	3910
3290	4670	2610	3900
3290	4670	2610	3910
3280	4660	2620	3920
3280	4670	2610	3920
3280	4670	2610	3930
3280	4660	2610	3930
3270	4660	2610	3910
3270	4660	2590	3880
3270	4650	2590	3890
3280	4660	2590	3900
3270	4670	2590	3900
3270	4680	2590	3900
3270	4670	2590	3910
3270	4660	2590	3910
3280	4650	2590	3900

(a) 24-正常-1

3280	4650	2600	3910
3280	4650	2590	3890
3280	4640	2600	3900
3270	4660	2600	3890
3270	4660	2600	3900
3270	4670	2600	3890
3290	4660	2600	3900
3290	4660	2590	3920
3290	4650	2590	3900
3290	4650	2590	3890
3270	4660	2590	3890
3270	4660	2590	3900
3270	4680	2590	3910
3270	4670	2590	3920
3280	4660	2590	3930
3270	4660	2590	3940
3270	4650	2590	3930
3270	4650	2590	3910
3270	4650	2600	3910
3270	4660	2600	3920
3280	4660	2600	3890
3280	4670	2590	3900
3290	4660	2590	3890
3290	4670	2590	3900
3270	4660	2590	3920
3280	4650	2600	3920
3270	4670	2600	3920
3270	4670	2600	3910
3280	4670	2600	3890
3280	4660	2600	3880
3270	4660	2600	3880
3270	4660	2610	3890
3280	4670	2610	3890
3290	4660	2610	3900
3290	4650	2610	3890
3290	4650	2610	3900
3300	4650	2610	3900
3290	4650	2620	3900
3280	4650	2600	3900

(b) 24-正常-2

3300	4660	2610	3920
3290	4650	2610	3920
3290	4660	2610	3920
3280	4670	2610	3900
3270	4660	2610	3880

(c) 24-正常-3

图 1.38　24-正常数据预处理结果

3、109-正常数据预处理结果

A0	A1	A2	A3
4910	5310	2030	2870
4920	5310	2020	2870
4920	5310	2020	2880
4910	5310	2020	2880
4910	5310	2020	2870
4890	5310	2020	2880
4880	5310	2020	2880
4890	5310	2020	2870
4920	5310	2030	2870
4910	5310	2020	2860
4910	5320	2020	2860
4880	5320	2020	2860
4870	5320	2020	2860
4860	5320	2020	2870
4870	5330	2020	2870
4860	5330	2020	2870
4860	5320	2030	2870
4850	5320	2030	2870
4870	5310	2030	2870
4870	5300	2020	2880
4880	5320	2030	2880
4890	5330	2030	2880
4910	5320	2020	2880
4910	5320	2030	2880
4910	5320	2040	2880
4920	5310	2010	2880
4890	5320	2030	2880
4880	5310	2030	2880
4880	5300	2030	2880
4880	5300	2030	2870
4890	5300	2020	2870
4890	5310	2020	2860
4890	5310	2010	2860
4880	5310	2010	2870
4890	5310	2030	2870
4880	5320	2030	2870
4880	5320	2020	2870
4890	5320	2020	2880
4890	5330	2020	2880

(a) 109-正常-1

4890	5330	2010	2880
4890	5320	2010	2880
4880	5320	2010	2880
4910	5320	2010	2880
4910	5310	2010	2880
4910	5330	2030	2880
4910	5330	2020	2880
4880	5340	2010	2880
4880	5330	2030	2880
4880	5330	2040	2880
4880	5320	2050	2880
4880	5330	2050	2880
4870	5330	2050	2870
4870	5330	2050	2880
4880	5330	2020	2880
4880	5330	2020	2890
4880	5320	2010	2890
4890	5320	2010	2890
4870	5320	2010	2890
4860	5320	2030	2890
4870	5320	2030	2890
4870	5320	2040	2890
4890	5320	2040	2890
4880	5330	2040	2870
4890	5330	2040	2880
4910	5330	2040	2880
4890	5320	2040	2880
4890	5320	2030	2870
4870	5320	2030	2880
4880	5320	2040	2880
4910	5340	2010	2890
4910	5330	2010	2890
4910	5330	2020	2890
4910	5320	2010	2890
4910	5330	2010	2880
4890	5330	2020	2870
4890	5330	2040	2870
4920	5330	2030	2870
4930	5330	2040	2870
4920	5340	2040	2870

(b) 109-正常-2

4920	5340	2040	2880
4910	5340	2040	2880
4920	5340	2040	2890
4920	5330	2040	2880
4910	5320	2030	2870
4910	5330	2040	2870
4910	5320	2050	2880
4910	5340	2060	2880
4910	5340	2050	2880
4890	5340	2040	2880
4890	5340	2040	2890
4890	5330	2040	2890
4920	5330	2030	2880
4920	5330	2030	2890
4920	5340	2030	2890
4910	5340	2030	2890
4910	5330	2030	2890
4910	5330	2040	2900
4890	5330	2040	2900
4880	5330	2040	2890
4870	5320	2040	2880
4910	5330	2040	2890
4910	5320	2030	2890
4910	5320	2030	2900
4910	5320	2040	2890
4910	5330	2030	2900
4890	5330	2020	2890
4890	5330	2030	2890
4890	5340	2030	2890
4890	5340	2030	2880
4920	5330	2020	2880
4920	5340	2030	2880
4920	5340	2020	2880
4920	5340	2020	2890
4910	5340	2020	2890
4930	5340	2020	2880
4940	5340	2030	2880
4930	5340	2030	2880
4930	5340	2030	2890
4910	5340	2030	2880

(c) 109-正常-3

图 1.39　109-正常数据预处理结果

4910	5340	2040	2890
4920	5330	2040	2890
4920	5340	2030	2900
4890	5340	2020	2880
4910	5350	2030	2880
4910	5350	2040	2880
4910	5350	2050	2870
4910	5350	2040	2870
4910	5340	2040	2870

图 1.40: 109-正常数据预处理结果

4、1-异常数据预处理结果

A0	A1	A2	A3
1280	4550	4550	6300
1260	4550	4550	6300
1250	4550	4550	6300
1240	4550	4550	6300
1230	4550	4550	6300
1290	4550	4550	6310
1290	4550	4550	6300
1240	4550	4560	6310
1230	4550	4560	6310
1250	4550	4560	6300
1260	4550	4560	6300
1230	4550	4570	6300
1240	4550	4560	6300
1280	4550	4560	6300
1220	4550	4550	6300
1270	4550	4560	6300
1240	4550	4560	6290
1230	4550	4570	6290
1230	4550	4560	6300
1300	4550	4550	6300
1280	4540	4550	6300
1220	4540	4550	6300
1240	4540	4550	6310
1230	4540	4550	6310
1270	4540	4550	6310
1240	4540	4550	6300
1210	4540	4560	6300
1200	4550	4550	6300
1200	4550	4570	6300
1220	4550	4570	6300
1220	4550	4570	6310
1240	4550	4570	6300
1250	4550	4570	6300
1270	4540	4560	6300
1220	4540	4560	6300
1270	4550	4570	6300
1260	4550	4570	6290
1220	4550	4560	6290
1250	4550	4560	6290

(a) 1-异常-1

1280	4550	4560	6290
1290	4550	4560	6290
1270	4550	4550	6290
1260	4550	4550	6290
1240	4550	4550	6290
1280	4550	4550	6290
1270	4550	4550	6300
1290	4550	4550	6290
1230	4550	4550	6290
1220	4540	4550	6290
1280	4540	4550	6290
1210	4540	4560	6280
1250	4540	4560	6290
1270	4550	4560	6290
1210	4550	4560	6290
1210	4550	4560	6300
1210	4550	4550	6300
1250	4540	4540	6300
1290	4540	4540	6310
1230	4540	4540	6310
1280	4540	4550	6310
1250	4540	4550	6310
1270	4540	4550	6300
1250	4560	4550	6300
1230	4550	4540	6290
1220	4550	4560	6300
1270	4540	4560	6310
1240	4540	4560	6310
1220	4540	4570	6310
1290	4540	4570	6290
1270	4550	4570	6290
1210	4550	4570	6290
1210	4550	4570	6300
1270	4550	4570	6310
1260	4550	4570	6300
1220	4550	4570	6290
1290	4550	4560	6300
1240	4550	4550	6310
1220	4550	4550	6310
1230	4550	4550	6310

(b) 1-异常-2

1250	4550	4550	6310
1280	4560	4560	6310
1230	4560	4550	6310
1250	4550	4540	6300
1260	4550	4540	6300
1260	4550	4540	6310
1280	4550	4540	6310
1250	4540	4550	6300
1230	4540	4550	6290
1290	4540	4550	6290
1270	4540	4550	6280
1220	4550	4550	6290
1280	4550	4540	6300
770	5040	4540	6300
770	5000	4540	6290
760	5060	4540	6290
770	5030	4540	6290
770	5030	4550	6290
770	5070	4550	6290
760	5050	4550	6290
770	5030	4550	6300
780	5070	4550	6300
780	5030	4550	6300
770	5050	4550	6290
770	5010	4550	6290
760	5030	4550	6290
760	5010	4550	6290
760	5010	4550	6300
770	5010	4550	6280
750	5010	4550	6280
760	5020	4550	6290
770	5040	4550	6290
770	5020	4550	6290
770	5000	4550	6290
760	5030	4550	6300
760	5030	4550	6310
750	5010	4550	6310
750	5070	4550	6310
750	5030	4550	6310
760	5010	4550	6310

(c) 1-异常-3

图 1.41　1-异常数据预处理结果

750	5030	4550	6300
750	5070	4550	6300
760	5050	4550	6300
740	5050	4550	6310
740	5050	4550	6300
770	5070	4550	6310
760	5050	4540	6310
760	5070	4540	6300
760	5070	4540	6290
750	5010	4550	6290
750	5070	4550	6290
750	5050	4550	6300
780	5030	4550	6310
780	5000	4550	6310
770	5000	4550	6300
770	5020	4550	6300
760	5020	4550	6300
770	5030	4560	6280
770	5010	4550	6300
780	5050	4550	6300
750	5010	4550	6300
760	5070	4550	6300
770	5010	4550	6310
760	5050	4550	6310
740	5030	4550	6310
740	5070	4550	6310
780	5070	4550	6290
780	5010	4540	6290
780	5010	4540	6300
770	5010	4540	6300
770	5070	4540	6290
750	5050	4550	6290
770	5050	4550	6300
770	5070	4550	6300
780	5010	4550	6300
760	5070	4550	6310
760	5060	4550	6310
760	5020	4550	6310
770	5020	4550	6310
770	5040	4550	6310

(a) 1-异常-4

770	5040	4550	6300
770	5030	4550	6310
780	5070	4550	6310
790	5070	4540	6310
790	5030	4540	6310
790	5010	4550	6310
780	5030	4550	6290
780	5010	4550	6280
790	5010	4550	6280
790	5030	4550	6290
790	5050	4550	6300
790	5010	4550	6290
750	5030	4550	6290
740	5010	4540	6300
750	5050	4540	6310
770	5070	4540	6310
770	5010	4540	6310
770	5050	4540	6310
770	5050	4540	6300
780	5050	4550	6310
770	5050	4550	6310
760	5070	4560	6310
760	5030	4560	6300
760	5070	4560	6300

(b) 1-异常-4

图 1.42　1-异常数据预处理结果

5、100-异常数据预处理结果

A0	A1	A2	A3
1510	3750	4690	5710
1520	3720	4680	5720
1530	3710	4690	5720
1510	3730	4690	5720
1500	3750	4690	5710
1510	3730	4680	5710
1510	3720	4680	5710
1520	3750	4680	5710
1530	3760	4690	5700
1520	3770	4690	5700
1510	3770	4690	5700
1510	3740	4690	5700
1510	3750	4690	5700
1510	3730	4680	5700
1510	3760	4680	5710
1510	3790	4680	5700
1510	3820	4680	5700
1510	3840	4670	5710
1500	3840	4670	5710
1510	3830	4680	5710
1520	3830	4680	5700
1520	3840	4680	5700
1520	3860	4680	5710
1510	3870	4680	5710
1510	3860	4680	5710
1510	3860	4680	5700
1510	3850	4680	5700
1510	3840	4680	5700
1500	3830	4680	5700
1510	3840	4680	5710
1510	3830	4680	5700
1500	3810	4680	5710
1500	3800	4680	5720
1500	3780	4680	5710
1500	3750	4680	5710
1490	3780	4670	5700
1490	3790	4670	5690
1500	3780	4670	5700
1500	3780	4660	5700

(a) 100-异常-1

1500	3770	4670	5700
1500	3770	4680	5710
1510	3770	4680	5710
1520	3770	4670	5700
1520	3780	4670	5700
1520	3790	4670	5700
1510	3800	4670	5700
1520	3810	4680	5700
1510	3810	4680	5700
1500	3800	4680	5710
1510	3800	4680	5710
1520	3800	4680	5720
1510	3810	4680	5710
1510	3820	4680	5710
1510	3790	4670	5700
1510	3780	4670	5700
1510	3770	4660	5700
1510	3760	4660	5700
1510	3750	4660	5710
1510	3740	4660	5700
1510	3730	4670	5700
1510	3720	4670	5700
1510	3710	4670	5700
1510	3700	4670	5700
1510	3690	4670	5700
1510	3690	4680	5700
1510	3680	4690	5700
1510	3700	4680	5700
1510	3710	4680	5700
1510	3740	4680	5700
1510	3740	4680	5690
1500	3740	4680	5690
1500	3750	4680	5700
1510	3750	4680	5700
1500	3750	4670	5710
1510	3750	4680	5710
1500	3740	4680	5710
1490	3740	4680	5710
1500	3730	4680	5710
1500	3730	4680	5700

(b) 100-异常-2

1510	3740	4680	5710
1510	3740	4680	5720
1500	3730	4690	5700
1510	3720	4680	5700
1500	3720	4680	5700
1520	3740	4680	5700
1520	3750	4680	5700
1510	3760	4680	5700
1520	3760	4680	5700
1520	3760	4680	5690
1520	3760	4670	5700
1520	3750	4670	5710
1520	3750	4670	5720
1530	3760	4680	5710
1530	3760	4680	5720
1520	3760	4680	5720
1500	3760	4680	5710
1500	3760	4680	5720
1500	3720	4680	5710
1500	3710	4680	5700
1510	3560	5290	5710
1510	3560	5230	5710
1510	3560	5190	5700
1510	3560	5110	5700
1510	3560	5060	5710
1510	3560	5010	5710
1510	3560	4970	5710
1510	3550	4960	5710
1510	3550	4950	5710
1510	3560	4930	5720
1510	3560	4930	5710
1510	3560	4930	5700
1510	3560	4910	5700
1520	3560	4890	5700
1520	3560	4870	5700
1520	3560	4860	5710
1520	3560	4850	5710
1530	3560	4840	5720
1520	3560	4820	5710
1520	3560	5090	5710

(c) 100-异常-3

图 1.43　100-异常数据预处理结果

1520	3560	5000	5710
1510	3560	5030	5710
1510	3560	5140	5710
1510	3560	5040	5710
1510	3560	4980	5700
1510	3550	5190	5700
1510	3550	5080	5710
1520	3550	5020	5710
1530	3560	4970	5710
1520	3560	4940	5720
1510	3560	4910	5720
1510	3560	4880	5710
1510	3560	4850	5710
1520	3560	4860	5700
1510	3560	4880	5700
1510	3560	4890	5700
1520	3560	4910	5690
1520	3560	4920	5690
1520	3560	4930	5690
1520	3570	4920	5690
1510	3570	4890	5690
1510	3560	4910	5690
1510	3570	4910	5690
1510	3570	4910	5680
1520	3570	4910	5680
1520	3570	4890	5680
1510	3570	4890	5680
1520	3570	4890	5690
1520	3570	4890	5700
1530	3570	4910	5700
1520	3570	4910	5700
1510	3570	4910	5700
1500	3560	4910	5700
1500	3570	4910	5700
1510	3570	4890	5700
1520	3560	4880	5700
1510	3560	4870	5700
1510	3560	4840	5700
1510	3560	4830	5700
1520	3560	4820	5700

(a) 100-异常-4

1510	3560	4810	5700
1500	3560	4790	5690
1510	3560	4790	5690
1520	3560	5240	5700
1520	3560	5130	5700
1520	3560	5040	5710
1500	3560	4990	5700
1500	3560	4970	5690
1510	3560	4970	5690
1520	3570	4960	5690
1520	3560	4970	5690
1520	3560	4980	5690
1520	3560	4940	5690
1530	3560	4910	5690
1530	3550	4880	5690
1530	3550	4860	5700
1530	3550	4860	5710
1520	3550	4840	5700
1520	3550	4840	5710
1510	3550	4840	5710
1510	3560	4840	5710
1520	3560	4830	5710
1510	3560	4870	5710
1510	3550	4850	5700
1510	3560	4820	5710
1520	3570	4810	5710
1520	3570	4840	5710
1520	3560	4840	5700
1520	3560	4830	5700
1520	3560	4800	5700
1510	3560	4790	5700
1520	3560	4790	5700
1520	3550	4800	5700
1510	3550	4810	5700
1510	3550	4880	5700
1510	3550	4910	5700
1510	3560	4920	5710
1520	3560	4930	5710
1530	3560	4930	5710
1530	3550	4940	5700

(b) 100-异常-5

1520	3550	4940	5690
1510	3560	4930	5690
1510	3560	4920	5690
1500	3560	5060	5690
1510	3560	5000	5690
1510	3560	4960	5690
1510	3560	4940	5690
1500	3560	4940	5690
1500	3560	4930	5690
1520	3560	4940	5700
1510	3560	4940	5710
1500	3560	4930	5700
1510	3550	4930	5700
1510	3550	4920	5700
1520	3550	4930	5690
1520	3550	4920	5700
1510	3550	4920	5710
1500	3550	4920	5710
1500	3550	4930	5700

(c) 100-异常-日 6

图 1.44　100-异常数据预处理结果

A.2　问题二结果

1、附件 2 的精确定位

表 1.13　实验环境 1 附件 2 靶点位置坐标

序号	X 轴	Y 轴	Z 轴	是否为正常测距
1	1165	668	967	是
2	3168	1721	1025	是
3	2726	1170	1099	是
4	2460	1009	1977	是
5	1480	2552	1759	是
6	1165	668	967	否
7	3168	1721	1025	否
8	2726	1170	1099	否
9	2460	1009	1977	否
10	1480	2552	1759	否

2、测量精度

表 1.14　各维度坐标估计均方根误差

维度	迭代法	Chan 算法	牛顿法
3 维	33.0438	14.4381	35.4136
2 维	4.4662	3.3840	6.7690
1 维 Z 轴	32.6971	13.9547	34.1890
1 维 Y 轴	2.6424	2.0476	4.1343
1 维 X 轴	3.5507	2.6539	4.5955

A.3 问题三结果

表 1.15 实验环境 2 中的靶点位置坐标

序号	X 轴	Y 轴	Z 轴	是否为正常测距
1	3660	2237	1396	是
2	4196	1726	1093	是
3	3172	1745	1311	是
4	2591	1884	2151	是
5	586	67	1489	是
6	3649	2222	1116	否
7	4241	1747	1214	否
8	3243	1793	1445	否
9	2600	1882	2105	否
10	539	119	738	否

A.4 问题四结果

1、算法有效性说明

见论文 6.5 小节。

2、数据分类结果

数据 1、3、6、7、10 为正常数据；数据 2、4、5、8、9 为异常数据。

A.5 问题五结果

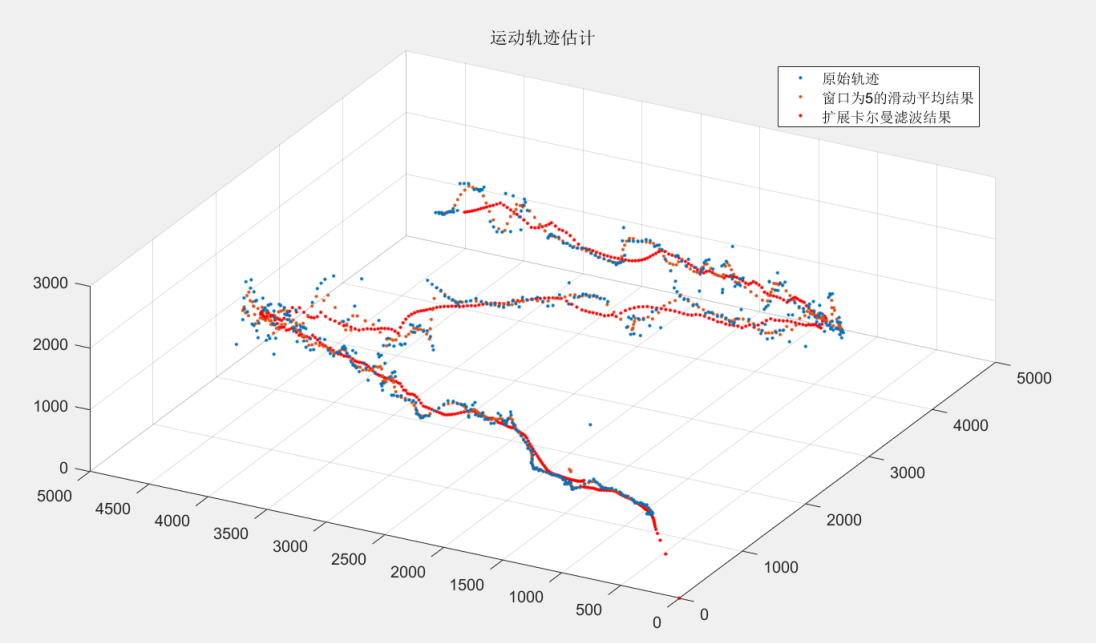


图 1.45 三维空间中的运动轨迹

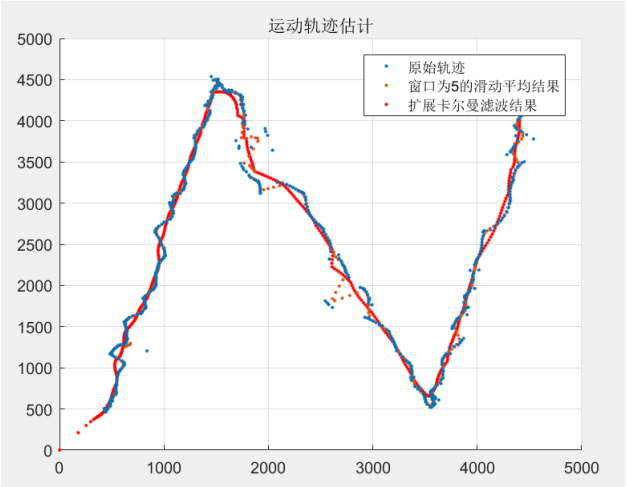


图 1.46 XY 平面运动轨迹

附录 B　Python 源程序

B.1　第 1 问程序

data.py

```
from tqdm import tqdm
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import kstest
from sklearn.cluster import SpectralClustering
from sklearn.cluster import KMeans
import warnings
warnings.filterwarnings('ignore')
plt.style.use('seaborn-whitegrid')



# 读取靶点的坐标
tag_path = "/Desktop/华为杯数学建模/2021年E题/附件1：UWB数据集
    /Tag坐标信息.txt"
teg_data = np.loadtxt(tag_path, skiprows=2, usecols=[1,2,3])
# print(len(teg_data))
# print(teg_data[23])

def get_real_dist(tag_idx):
    pos1 = 0 ,0, 1300
    pos2 = 5000, 0, 1700
    pos3 = 0, 5000, 1700
    pos4 = 5000, 5000, 1300
    teg_pos = teg_data[int(tag_idx-1)]*10
    d1 = np.linalg.norm(pos1-teg_pos)
    d2 = np.linalg.norm(pos2-teg_pos)
    d3 = np.linalg.norm(pos3-teg_pos)
    d4 = np.linalg.norm(pos4-teg_pos)

    return d1, d2, d3, d4
```

```
def get_uwb_dist(tag_idx, normal=True):
    if normal:
        file_path = "/Desktop/华为杯数学建模/2021年E题/附件1:
           UWB数据集/正常数据/"+str(tag_idx)+".正常.txt"
        df = pd.read_csv(file_path, skiprows=1, usecols=[1, 4,
            5], names=["t", "num", "d"], delimiter=":", dtype={
           "t":str, "num":int, "d":int})
    else:
        file_path = "/Desktop/华为杯数学建模/2021年E题/附件1:
           UWB数据集/异常数据/"+str(tag_idx)+".异常.txt"
        df = pd.read_csv(file_path, skiprows=1, usecols=[1, 4,
            5], names=["t", "num", "d"], delimiter=":", dtype={
           "t":str, "num":int, "d":int})
    # t_unique = df["t"].unique()
    # data = []
    # for t in t_unique:
    #     n = (df["t"]==t).sum()
    #     df_group = df[df["t"]==t]
    #     if n!=4:
    #         for i in range(4):
    #             if i not in df_group["num"].values:
    #                 data.append(-1)
    #             else:
    #                 filter_i = (df_group["num"]==i)
    #                 data.append(df_group.loc[filter_i, "d"
       ].values[0])
    #     else:
    #         for i in range(4):
    #             filter_i = (df_group["num"]==i)
    #             data.append(df_group.loc[filter_i, "d" ].
       values[0])
    data = df["d"].values
    data = np.array(data)

    if data.shape[0]%4 !=0:
        raise IndexError
    group_num = int(data.shape[0]/4)
```

```
    data = data.reshape(group_num, -1)
    tag = tag_idx*np.ones(len(data), dtype=int).reshape(-1, 1)

    data = np.concatenate([tag, data], axis=1)

    return data

def plot_dist(real_dist, uwb_dist):

    x = range(uwb_dist.shape[0])
    plt.figure(figsize=(8,8))

    plt.subplot(2, 2, 1)
    plt.plot(x, uwb_dist[:, 1])
    plt.axhline(y=real_dist[0] , c="red", ls="--", lw=1)
    plt.ylabel("d(mm)")
    plt.xlabel("time")
    plt.ylim([0, 7000])
    plt.legend(labels=['d0', 'real position'],loc='best')
    plt.title("d0-t")


    plt.subplot(2, 2, 2)
    plt.plot(x, uwb_dist[:, 2])
    plt.axhline(y=real_dist[1] , c="red", ls="--", lw=1)
    plt.legend(labels=['d1', 'real position'],loc='best')
    plt.xlabel("time")
    plt.ylabel("d(mm)")
    plt.ylim([0, 7000])
    plt.title("d1-t")

    plt.subplot(2, 2, 3)
    plt.plot(x, uwb_dist[:, 3])
    plt.axhline(y=real_dist[2] , c="red", ls="--", lw=1)
    plt.legend(labels=['d2', 'real position'],loc='best')
    plt.xlabel("time")
    plt.ylabel("d(mm)")
    plt.ylim([0, 7000])
    plt.title("d2-t")
```

```
    plt.subplot(2, 2, 4)
    plt.plot(x, uwb_dist[:, 4])
    plt.axhline(y=real_dist[3] , c="red", ls="--", lw=1)
    # x_a = [ 72, 111]#109
    # plt.plot(x_a, uwb_dist[:, 4][x_a], "ro")
    plt.legend(labels=['d3', 'real position', 'abnormal'],loc=
        'best')
    plt.xlabel("time")
    plt.ylabel("d(mm)")
    plt.ylim([0, 7000])
    plt.title("d3-t")




def check_data(normal=True):
    print(normal)
    total_n = 0
    rest_n = 0
    na_n = 0
    error_n = 0
    dp_n = 0
    ab_file_n = 0
    ab_n = 0
    df_ls = []

    for i in tqdm(range(1,325)):


        real_dist = get_real_dist(i)
        uwb_dist = get_uwb_dist(i, normal)
        total_n += len(uwb_dist)

        x = range(uwb_dist.shape[0])
        # for j in range(1,5):
        #     plt.subplot(2,2,j)
        #     plt.plot(x, uwb_dist[:, j] )
        # plt.show()
```

```
# 缺失值检查,KNN填充
# na_idx, anchor_idx =  np.where(uwb_dist==-1)
# na_n += len(na_idx)
# # 去除异常值
# uwb_dist = np.delete(uwb_dist, np.unina_idx, axis =
   0)
# from sklearn.impute import KNNImputer
# imputer = KNNImputer(n_neighbors=5, missing_values
   =-1)
# uwb_dist = imputer.fit_transform(uwb_dist)

# 转为df
df = pd.DataFrame(uwb_dist, columns=["tag", "d1", "d2"
   , "d3", "d4"])
d = ["d1", "d2", "d3", "d4"]

error_idxs = []
for col_name in d:
    # 计算均值
    u = df[col_name].mean()
    # 计算标准差
    std = df[col_name].std()
    # 计算P值
    p=kstest(df[col_name], 'norm', (u, std))[1]
    print(p)


    if p<0.06:  # 文件55p值为0.05xxx, 可以保留
        #3igma原则, 提取超过范围的数据行索引
        error_idx = df[np.abs(df[col_name] - u) > 3 *
           std].index
        if len(error_idx.values) > 0:
            error_n += len(error_idx)
            error_idxs.append(error_idx)

    else:
        print("file{:d} is not normal distribution in
           {:s} column, p is {:.2f}".format(i, col_name
```

```
                , p))
            ab_file_n +=1
            ab_n += len(df)
            df = None
            break

    if df is not None:
        # 去除超过3sigma分布的数据
        if len(error_idxs)!=0:
            error = np.concatenate(error_idxs)
            df = df.drop(error)
            print(error)
        # for j in range(1,5):

        #     plt.subplot(2,2,j)
        #     plt.plot(x, uwb_dist[:, j] )
        #     if len(error_idxs)!=0:
        #         plt.plot(error, uwb_dist[:, j][error], "
           ro") # 画出不在正太点
        #     plt.legend(labels=['original', 'missing', '
           delete'],loc='upper right')

        # 去除重复值

        dp_n += df.duplicated().sum()
        df = df.drop_duplicates()


        rest_n += (total_n - dp_n - na_n -error_n)
        df_ls.append(df)

print("缺失值组数", na_n)
print("在正态分布外的组",error_n)
print("不属于整态分布的文件数量", ab_file_n)
print("不属于整态分布的组", ab_n)
print("重复值", dp_n)
print("总组数", total_n)
print("剩余组数", rest_n)
```

```
    return df_ls




# 处理正常.txt
df_ls = check_data(normal=True)
data_normal = pd.concat(df_ls, axis=0)
data_normal.to_csv('./data_normal_d.csv', index=False)

# 处理异常.txt
df_ls = check_data(normal=False)
data_abnormal = pd.concat(df_ls, axis=0)
data_abnormal.to_csv('./data_abnormal_d.csv', index=False)


# 画出处理后正/异常数据比较图
def check_csv(idx, normal=True):
    if normal:
        df = pd.read_csv("./data_normal.csv")
    else:
        df = pd.read_csv("./data_abnormal.csv")
    real_dist = get_real_dist(idx)
    uwb_dist = df[df["tag"]==idx].values

    plot_dist(real_dist, uwb_dist)


    return uwb_dist




def spilt_abnormal(idx):

    df_n = pd.read_csv("./data_normal_d.csv")
    df_ab = pd.read_csv("./data_abnormal_d.csv")
```

```
real_dist = get_real_dist(idx)
uwb_dist = df_n[df_n["tag"]==idx].values
ab_uwb_dist = df_ab[df_ab["tag"]==idx].values

x_n = range(uwb_dist.shape[0])
x_ab =  range(ab_uwb_dist.shape[0])


de_ls = [] # 记录去掉的索引下标
for j in range(1,5):
    plt.subplot(2,2,j)
    u = np.mean(uwb_dist[:, j] )
    sigma = np.std(uwb_dist[:, j] )

    condition1 = ab_uwb_dist[:, j]>u+3*sigma
    condition2 = ab_uwb_dist[:, j]<u-3*sigma

    de_idx = np.where(condition1 | condition2) # bool, 返
        回tulple
    if(len(de_idx[0])!=0): # 行索引, 去掉的数据
        de_ls.append(de_idx[0])


#     plt.plot(x_n, uwb_dist[:, j] )
#     plt.plot(x_ab, ab_uwb_dist[:, j] )
#     if(len(de_idx[0])!=0): # 行索引, 画出被去掉的数据
#         plt.plot(de_idx[0], ab_uwb_dist[:, j][de_idx
   [0]], c="r", ls="-", lw=1)

#     plt.axhline(y=u+3*sigma , c="purple", ls="--", lw=1)
#     plt.axhline(y=u-3*sigma , c="purple", ls="--", lw=1)
#     #plt.ylim([0, 7000])
# plt.show()

de_ls = np.unique(np.concatenate(de_ls))# 合并array成单个
   array

new_uwb_dist = np.delete(ab_uwb_dist, de_ls, axis = 0)
```

```
    ab_uwb_dist= ab_uwb_dist[de_ls, :]


    return ab_uwb_dist, new_uwb_dist

def split_ab():
    df_ab_ls = []
    df_new_n_ls = []
    for i in tqdm(range(1,325)):
        ab_uwb_dist, new_uwb_dist = spilt_abnormal(i)
        df_ab = pd.DataFrame(ab_uwb_dist, columns=["tag", "d1"
           , "d2", "d3", "d4"])
        df_new_n = pd.DataFrame(new_uwb_dist, columns=["tag",
           "d1", "d2", "d3", "d4"])

        df_ab_ls.append(df_ab)
        df_new_n_ls.append(df_new_n)

    data_abnormal = pd.concat(df_ab_ls, axis=0)
    data_abnormal.to_csv('./data_abnormal_split.csv', index=
       False)
    df_new_n_ls = pd.concat(df_new_n_ls, axis=0)
    df_new_n_ls.to_csv('./data_normal_split.csv', index=False)
```

B.2 第 4 问程序

clf.py

```
import warnings
warnings.filterwarnings('ignore')
import itertools
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
from sklearn import datasets
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
```

```
from mlxtend.classifier import StackingClassifier
from sklearn.model_selection import cross_val_score,
   train_test_split
from mlxtend.plotting import plot_learning_curves
from mlxtend.plotting import plot_decision_regions
import sklearn.linear_model as sk_linear
import sklearn.neural_network as sk_nn
import sklearn.tree as sk_tree
from sklearn import metrics




# #X, y = X[:, 1:3], y # 使用两维可视化，或者使用PCA降低维度
# import sklearn.decomposition as sk_decomposition
# pca = sk_decomposition.PCA(n_components=2, whiten=False,
   svd_solver='auto')
# pca.fit(X)
# reduced_X = pca.transform(X) #reduced_X为降维后的数据


clf1 = KNeighborsClassifier(n_neighbors=5)
clf2 = RandomForestClassifier(random_state=1)
clf3 = GaussianNB()
clf4 = sk_linear.LogisticRegression(penalty='l2',dual=False,C
   =1.0,n_jobs=1,random_state=20,fit_intercept=True)
clf5 = sk_nn.MLPClassifier(activation='tanh',solver='adam',
   alpha=0.0001,learning_rate='adaptive',learning_rate_init
   =0.001,max_iter=200)
clf6 = sk_tree.DecisionTreeClassifier(criterion='entropy',
   max_depth=None,min_samples_split=2,min_samples_leaf=1,
   max_features=None,max_leaf_nodes=None,min_impurity_decrease
   =0)

lr = LogisticRegression()
sclf = StackingClassifier(classifiers=[clf1, clf2, clf3,clf4,
   clf5, clf6],
```

```
                              meta_classifier=lr)


label = ['KNN', 'Random Forest', 'Naive Bayes', "
   LogisticRegression", "MLPClassifier","DecisionTreeClassifier
   ",'Stacking Classifier']
clf_list = [clf1, clf2, clf3, clf4, clf5, clf6,sclf]

fig = plt.figure(figsize=(10,8))
gs = gridspec.GridSpec(4, 2)
grid = itertools.product([0,1],repeat=3)


clf_cv_mean = []
clf_cv_std = []
for clf, label, grd in zip(clf_list, label, grid):

    scores = cross_val_score(clf, X, y, cv=5, scoring='
       accuracy')
    print("Accuracy: %.2f (+/- %.2f) [%s]" %(scores.mean(),
       scores.std(), label))
    clf_cv_mean.append(scores.mean())
    clf_cv_std.append(scores.std())
    clf.fit(X_train_split, y_train_split)



    ax = plt.subplot(gs[grd[0], grd[1]])
    fig = plot_decision_regions(X=X_val, y=y_val, clf=clf,
       legend=2)
    plt.title(label)


    # y_pre = clf.predict(X_val)
    # acc=clf.score(X_val,y_val)
    # print("验证集准确率", acc)
    # %timeit  y_pre = clf.predict(X_val)
    # """预测并计算roc的相关指标"""
    # fpr, tpr, threshold = metrics.roc_curve(y_val, y_pre)
```

```
    # roc_auc = metrics.auc(fpr, tpr)
    # print('未调参前lightgbm单模型在验证集上的AUC: {}'.format
      (roc_auc))
    # """画出roc曲线图"""
    # plt.figure(figsize=(8, 8))
    # plt.title('Validation ROC')
    # plt.plot(fpr, tpr, 'b', label = 'Val AUC = %0.4f' %
      roc_auc)
    # plt.ylim(0,1)
    # plt.xlim(0,1)
    # plt.legend(loc='best')
    # plt.title('ROC')
    # plt.ylabel('True Positive Rate')
    # plt.xlabel('False Positive Rate')
    # # 画出对角线
    # plt.plot([0,1],[0,1],'r--')
    # plt.show()

    # clf.fit(X, y)
    # y_pre = clf.predict(test2)
    # print(y_pre)
    # y_pre = model.predict(test4)
    # print(y_pre)


plt.show()
```

附录 C　Matlab 源程序

C.1　第 2 问程序

```
% Chan方法进行三维定位
cnt = 1;
for i = 1:size(index,1)
    locpred = [];
    err_chan = [];
    for j = 1:size(nors(i).data,1)
        Obsz(1,:) = fitresult1(nors(i).data(j,1));
        Obsz(2,:) = fitresult2(nors(i).data(j,2));
        Obsz(3,:) = fitresult3(nors(i).data(j,3));
        Obsz(4,:) = fitresult4(nors(i).data(j,4));
        locresult(cnt,:) = chan3d(anchorpos,Obsz');
        locpred(j,:)=locresult(cnt,:);
        [err1_chan(cnt,:),err2_chan(cnt),err3_chan(cnt)]=errcal(
           locresult(cnt,:),tag(i,:)*10);
        err_chan(j,:) = [err1_chan(cnt,:),err2_chan(cnt),err3_chan(
           cnt)];
        cnt = cnt +1;
    end
    errmean_chan(i,:) = mean(err_chan,1);
    pred_chan(i).data = locpred;
end
% Jacobi法进行三维定位
para.Thr = 100;
para.DataLen=1;
para.BsNum=4;
para.BsLoc=anchorpos;
para.MaxIters=25;
para.xinit = mean(anchorpos,1);
para.dRef=2500;
cnt = 1;
for i=1:size(index,1)
    err_itre = [];
    Locpred = [];
    for j = 1:size(nors(i).data,1)
        Obsz(1,:) = fitresult1(nors(i).data(j,1));
```

```
        Obsz(2,:) = fitresult2(nors(i).data(j,2));
        Obsz(3,:) = fitresult3(nors(i).data(j,3));
        Obsz(4,:) = fitresult4(nors(i).data(j,4));
        % Jacobi迭代法
        LocEst(cnt,:)= JacoIters(Obsz',para);
        Locpred(j,:)=LocEst(cnt,:);
        [err1_itre(cnt,:),err2_itre(cnt),err3_itre(cnt)]=errcal(
            LocEst(cnt,:),tag(i,:)*10);
        err_itre(j,:) = [err1_itre(cnt,:),err2_itre(cnt),err3_itre(
            cnt)];
        cnt=cnt+1;
    end
    errmean_itre(i,:) = mean(err_itre,1);
    pred_itre(i).data = Locpred;
end
% 牛顿法进行三维定位
parak.Thr = 100;
parak.DataLen=1;
parak.BsNum=4;
parak.BsLoc=anchorpos;
parak.MaxIters=25;
parak.xinit = mean(anchorpos,1);
parak.dRef=2500;
cnt = 1;
for i=1:size(index,1)
    err_new = [];
    Locpred = [];
    for j = 1:size(nors(i).data,1)
        Obsz(1,:) = fitresult1(nors(i).data(j,1));
        Obsz(2,:) = fitresult2(nors(i).data(j,2));
        Obsz(3,:) = fitresult3(nors(i).data(j,3));
        Obsz(4,:) = fitresult4(nors(i).data(j,4));
        newEst(cnt,:)= Newton(Obsz',parak);
        newpred(j,:)=newEst(cnt,:);
        [err1_new(cnt,:),err2_new(cnt),err3_new(cnt)]=errcal(newEst
            (cnt,:),tag(i,:)*10);
        err_new(j,:) = [err1_new(cnt,:),err2_new(cnt),err3_new(cnt)
            ];
        cnt=cnt+1;
```

```
    end
    errmean_new(i,:) = mean(err_new,1);
    pred_new(i).data = newpred;
end
% 有偏卡尔曼滤波异常数据定位
para.Thr = 100;
para.DataLen=1;
para.BsNum=4;
para.BsLoc=anchorpos;
para.MaxIters=25;
para.anchorpos = anchorpos;
para.xinit = mean(anchorpos,1);
para.dRef=2500;
cnt = 1;
for i=1:size(index,1)
    err_itre = [];
    abLocpred = [];
    for j = 1:size(abnors(i).data,1)
        Obsz(1,:) = fitresult1(abnors(i).data(j,1));
        Obsz(2,:) = fitresult2(abnors(i).data(j,2));
        Obsz(3,:) = fitresult3(abnors(i).data(j,3));
        Obsz(4,:) = fitresult4(abnors(i).data(j,4));
        % GammaKF_t有偏卡尔曼滤波
        abLocEst_new(cnt,:)= GammaKF_t(Obsz',para);
        abLocpred_new(j,:)=abLocEst_new(cnt,:);
        [err1_itre(cnt,:),err2_itre(cnt),err3_itre(cnt)]=errcal(
           abLocEst_new(cnt,:),tag(i,:)*10);
        err_itre(j,:) = [err1_itre(cnt,:),err2_itre(cnt),err3_itre(
           cnt)];
        cnt=cnt+1;
    end
    aberrmean_itre_new(i,:) = mean(err_itre,1);
    abpred_itre_new(i).data = abLocpred_new;
end
```

C.2 第 5 问程序

```
%对观测进行等采样间隔插值，线性插值
T2=size(pro3(:,1),1);
```

```
Bsnum = 4;
dt=T2(end)/500;
tseq=0:dt:T2(end);
for ii=1:5
     outres2(:,ii)=interp1(T2, outres(:,ii) ,tseq,'linear');
end
figure;
title('等采样间隔插值结果')
hold on;grid on;
plot(outres(:,2),'-', 'linewidth', 1.5, 'markeredgecolor','k');
plot(outres2(:,2),'-', 'linewidth', 1.5, 'markeredgecolor','k');
xlabel('采样点');ylabel('锚点1采样值');
size(outres)
Obsz2 = [];
for jj=1:size(outres2,1)
    Obsz2(1,:) = 0.9707*outres(jj,1)+144.5;
    Obsz2(2,:) = 0.9803*outres(jj,2)+110.4;
    Obsz2(3,:) = 0.9774*outres(jj,3)+135.8;
    Obsz2(4,:) = 0.9716*outres(jj,4)+161;
end
tag = importdata('taginfo.mat');
for jj=1:size(outres,1)
    if jj < 5
        LocEst3(jj,:)= UWBLocIters_t(outres(jj,:),para);
    else
        LocEst3(jj,:)= UWBLocIters(outres(jj,:),para);
    end
end

LocMatrix = LocEst3;
% 扩展卡尔曼滤波
LocEst6 = EKF(LocMatrix);
LocEst6 = LocEst6';

parak.T = 0.25;
parak.BsNum=Bsnum;
parak.BsLoc=Bsloc;
parak.DataLen=size(outres,1);
parak.Q = eye(6)*25;
```

```
parak.Rz = eye(4)*25;
parak.xinit=[LocEst3(2,1:2) 720 0 0 0];
type = round(type);
LocMatrix = LocEst3;
% 标准卡尔曼滤波
LocEst4 = KF(outres2(:,1:4),type,parak);

%%
% 滑动平均用于轨迹平滑
for ii = 1:size(LocEst3,2)
    x=LocEst3(:,ii);
    N = 5;
    for j=1:(length(x)-N+1)
        LocEst5(j,ii)=mean(x(j:(j+N-1)));
    end
end

figure;hold on;grid on;
title('运动轨迹估计')
plot3(LocEst3(:,1),LocEst3(:,2),LocEst3(:,3),'.');
plot3(LocEst4(:,1),LocEst4(:,2),LocEst4(:,3),'.');
plot3(LocEst6(:,1),LocEst6(:,2),LocEst6(:,3),'r.');
legend('原始轨迹','窗口为5的滑动平均结果','扩展卡尔曼滤波结果');
view(3);
```