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	福州大学
参赛队号	21103860005
队员姓名	1. 汤婉红
	2. 周玮蜜
	3. 罗曦

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　信号干扰下的超宽带精确定位问题

摘　　要：

UWB 技术具有穿透能力，但当信号受到较强干扰时，仍然会产生误差，基本无法完成室内定位。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。

针对问题一，对附件 1 的原始数据集进行预处理，首先利用 Python 读取每个文本文件中的测量数据，再对数据文件的 4 个变量进行正态性检验，判断是否可以使用拉以达准则 (也称 3σ 准则) 剔除异常值，随后删除重复的样本数据。接着，利用**基于距离阈值的系统聚类分析法**对每个文件进行聚类，其中“正常数据”距离阈值设定为 15 毫米、“异常数据”距离阈值设定为 100 毫米，并将正常样本聚类成 1-3 类、异常样本聚类成 2-4 类，每一类中保留一个距离样本均值最近的样品，其余样品全部删去。最后得到**“正常数据”共 582 组数据，“异常数据”共 713 组数据**。

针对问题二和问题三，在给定场景下的锚点坐标和测量距离的基础上，利用**最小二乘算法**和**残差加权误差抑制算法**分别计算“正常数据”文件和“异常数据”的近似坐标，随后分别用**极限学习机**和 **BP 神经网络**对近似坐标进行优化，得到较为精确的定位坐标。最后得到该模型在信号无干扰情况下测试集的一维 x, y, z 精度的平均误差分别为 **25.7mm，27.0mm，130.0mm**，二维 (x, y) 精度为 **41.3mm**，三维 (x, y, z) 精度为 **141.7mm**；在信号有干扰情况下测试集的一维 x, y, z 精度的平均误差分别为 **137.6mm，127.9mm，364.8mm**，二维 (x, y) 精度为 **199.1mm**，三维 (x, y, z) 精度为 **419.4mm**。

针对问题四，本文对所有数据应用最小二乘算法计算出近似坐标点，为了寻找数据的差异性，通过计算近似坐标点到四个锚点之间的欧氏距离作为 4 个新特征指标、该距离与测量距离之差作为 4 个新特征指标，以及距离差之和作为 1 个新特征指标，共提炼出这 9 个新特征指标。接着，对这 9 个新特征指标进行重要性评估，选择重要性较大的 5 个特征指标建立**基于最小二乘算法的随机森林分类模型**，对信号有无干扰的数据进行分类，测试集的正确率达到 **85.1%**。最后

本文预测附件 4 中的**第 1、3、6、9、10 组数据是来自信号无干扰下采集的，第 2、4、5、7、8 组数据是来自信号有干扰下采集的**。

针对问题五，本文先利用问题四的分类模型，判别附件 5 中的数据是不是在信号有干扰下采集的，然后再分别利用问题二中的正常数据的定位模型和异常数据的定位模型确定位置坐标点。在静态点的定位坐标的基础之上，考虑到靶点自身运动规律，在相邻间隔很短的两个时间内靶点的位置差异不会太大，所以对坐标点进行修正，并且利用**轨迹平滑化模型**进行处理。最后得出靶点在空间中的运动轨迹：**在约为 1500mm 的高度上呈现出一个“N”字形**。

关键字：　系统聚类法　最小二乘算法　残差加权算法　极限学习机　随机森林　BP 神经网络

目录

1 问题重述

1.1 问题背景

UWB(Ultra-Wideband) 技术也被称之为“超宽带”，又称之为脉冲无线电技术。这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，并且信号传输过程中的功耗仅仅有几十 μW。UWB 因其独有的特点，使其在军事、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。

UWB 的定位技术有多种方法，本文仅考虑基于飞行时间 (Time of Flight, TOF) 的测距原理，它是 UWB 定位法中最常见的定位方法之一。TOF 测距技术属于双向测距技术，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，这个距离肯定有不同程度的误差，但其精度已经比较高。在室内定位的应用中，UWB 技术可以实现厘米级的定位精度 (一般指 2 维平面定位), 并具有良好的抗多径干扰和衰弱的性能以及具有较强的穿透能力。但由于室内环境复杂多变 UWB 通信信号极易受到遮挡，虽然 UWB 技术具有穿透能力，但仍然会产生误差，在较强干扰时，数据会发生异常波动 (通常是时间延时)，基本无法完成室内定位，甚至会造成严重事故。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 实验场景和数据采集

如图所示，在 5000mm*5000mm*3000mm 的测试环境中，分别在 4 个角落 A0，A1，A2，A3 放置 UWB 锚点 (anchor)，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。

实验在实验场景 1 中采集了 Tag 在 324 个不同位置，在信号无干扰和信号干扰下的 UWB 数据，即每个位置各测试 (采集)2 次，一次信号无干扰，另一次信号有干扰 (锚点与靶点间有遮挡)，注意：每次采集数据时，由于 Tag 在同一位置会停留一会儿时间，而锚点与 Tag 之间每 0.2—0.3 秒之间就会发送、接收信号一次，所以在同一位置点，UWB 会采集到多组数据 (多组数据都代表同一位置的信息)，组数的多少视 Tag 在同一位置的时间而定，停留的时间越长，组数就越多。数据见文件夹“附件 1：UWB 数据集”。

实验场景 1：靶点 (Tag) 范围：5000mm*5000mm*3000mm；锚点 (anchor) 位置 (单位：mm)：A0(0，0，1300)、A1(5000，0，1700)、A2(0，5000，1700)、A3(5000，5000，1300)

1.3 问题目标

为解决信号干扰下的超宽带 (UWB) 精确定位问题，我们通过实际场景实测，采集到一定数量的数据，即利用 UWB 的定位技术 (TOF)，采集到锚点 (anchor) 与靶点 (Tag) 之间的距离，再通过数学建模 (或算法) 方法，无论信号是否干扰，都可以给出目标物 (靶点) 的精确定位 (3 维坐标)。

1.4 问题描述

试根据题目给定的数据，完成如下任务：

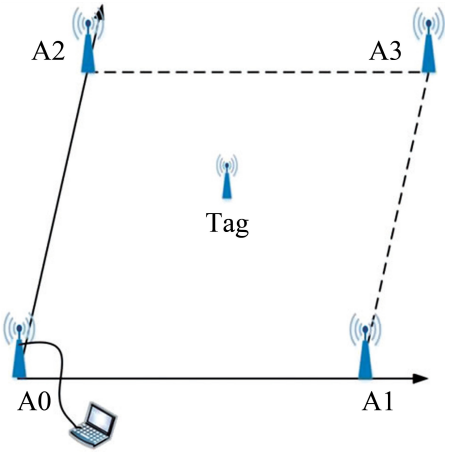

图 **1-1**　实测环境示意图

问题一：数据预处理 (清洗)

无论是信号无干扰下采集数据，或信号干扰下采集数据，Tag 在同一坐标点上都采集多组数据 (见附件 1 中 648 个数据文件)，请用某种方法把每个数据文件相应数值抓取出来，并转换成二维表 (矩阵) 形式 (txt、Excel 或其他数据格式)，每一行代表一组数据 (即一个样品)，然后对这些数据文件进行预处理 (清洗)，删除掉一些“无用”(异常、缺失、相同或相似的数据 (样品)。

经处理后，“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多少组（多少个样品）数据，并重点列出以下 4 个数据文件，经处理后保留的数据（矩阵形式）：“正常数据”文件夹中：24. 正常.txt、109. 正常.txt“异常数据”文件夹中：1. 异常.txt、100. 异常.txt。

问题二：定位模型

利用任务一处理后的数据，分别对“正常数据”和“异常数据”，设计合适的数学模型 (或算法)，估计 (或预测) 出 Tag 的精确位置，并说明你所建立的定位模型 (或算法) 的有效性；同时请利用你的定位模型 (或算法) 分别对附件 2 中提供的前 5 组 (信号无干扰) 数据和后 5 组 (信号有干扰) 数据进行精确定位 (3 维坐标)。

问题三：不同场景应用

我们的训练数据仅采集于同一实验场景 (实验场景 1)，但定位模型应该能够在不同实际场景上使用，我们希望你所建立的定位模型能够应用于不同场景。附件 3 中 10 组数据采集于下面实验场景 2(5 组数据信号无干扰，后 5 组数据信号有干扰)，请分别用上述建立的定位模型，对这 10 组数据进行精确定位 (3 维坐标)。

实验场景 2：靶点 (Tag) 范围：5000mm*3000mm*3000mm；锚点 (anchor) 位置 (单位：mm)：A0(0，0，1200)、A1(5000，0，1600)、A2(0，3000，1600)、A3(5000，3000，1200)

问题四：分类模型

上述定位模型是在已知信号有、无干扰的条件下建立的，但 UWB 在采集数据时并不知道信号有无干扰，所以判断信号有无干扰是 UWB 精确定位问题的重点和难点。利用任务一处理后的数据，建立数学模型 (或算法)，以便区分哪些数据是在信号无干扰下采集的数据，哪些数据是在信号干扰下采集的数据？并说明你所建立的分类模型 (或算法) 的有效性；同时请用你所建立的分类模型 (或算法) 判断附件 4 中提供的 10 组数据 (这 10 组数据同样采集于实验场景 1) 是来自信号无干扰或信号干扰下采集的。

问题五：运动轨迹定位

运动轨迹定位是 UWB 重要应用之一，利用静态点的定位模型，加上靶点自身运动规律，希望给出动态靶点的运动轨迹。附件 5 是对动态靶点采集的数据(一段时间内连续采集的多组数据)，请注意，在采集这些数据时，会随机出现信号干扰，请对这个运动轨迹进行精确定位，最终画出这条运动轨迹图 (数据采集来自实验场景 1)。

2 基本假设与符号系统

2.1 基本假设

为了合理简化分析过程，在建立基于超宽带 (UWB) 技术的三维室内定位模型时，我们提出如下假设：

(1) 假设无线通信锚点发射出信号的强度是稳定的，且本身是不会产生剧烈的波动，因为本文仅考虑基于飞行时间的测距原理。
(2) 假设无线通信在靶点和锚点之间的接收过程中不存在时间延迟，即通信信号只在传播过程中受到遮挡产生传播时间延迟。
(3) 假设同一组数据是在相同实验场景以及相同实验环境 (包括温度、气压、湿度等) 中测量得来。

2.2 符号系统

本文中正常数据是指在信号无干扰下采集的样本数据，异常数据是指存在信号有干扰下采集的样本数据，符号说明见表 2.1。

表 2.1　符号说明

符号	说明
d_0	靶点到锚点 A0 的测量距离 (mm)
d_1	靶点到锚点 A1 的测量距离 (mm)
d_2	靶点到锚点 A2 的测量距离 (mm)
d_3	靶点到锚点 A3 的测量距离 (mm)
d_0'	靶点近似点坐标到锚点 A0 坐标的欧式距离 (mm)
d_1'	靶点近似点坐标到锚点 A1 坐标的欧式距离 (mm)
d_2'	靶点近似点坐标到锚点 A2 坐标的欧式距离 (mm)
d_3'	靶点近似点坐标到锚点 A3 坐标的欧式距离 (mm)
e_i	测量距离 d_i 与欧式距离 d_i' 的绝对误差 (mm)
e	绝对误差之和 (mm)
X	靶点三维估计坐标
x	估计靶点 x 轴坐标
y	估计靶点 y 轴坐标
z	估计靶点 z 轴坐标

3 问题一的分析与求解

3.1 问题一的分析

本文数据来自于实际场景实测，采集了靶点在 324 个不同位置点时信号无干扰和信号有干扰下的两类 UWB 数据，共 648 个数据文件。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、相对准确的距离测量数据作为支撑，但是实际操作的数据测量、数据记录、数据导出等过程中难免会存在一些问题。例如，每次采集数据时，由于靶点在同一位置会停留一会儿时间，而锚点与靶点之间每 0.2—0.3 秒之间就会发送和接收信号一次，所以在同一位置点，UWB 会采集到多组同一位置的测量数据。因此，对数据进行科学有效地预处理，对于在信号有干扰和信号无干扰情况下的定位模型的构建有着决定性意义。

读取 UWB 数据集中每个文本文件相应的数值，我们将其转换成二维表形式，每一行代表一组数据，构成 1 个 $n\times4$ 的矩阵，其中 n 是每个数据文件中的数据组数，4 个指标是指靶点到 4 个锚点的测量距离。首先判断原始数据中的数据是否存在缺失值，其次为了降低随机误差带来的影响，我们要删除异常值，可以利用 SPSS 软件对测量距离进行正态性检验，判断是否可以用拉依达准则 (3σ 准则) 剔除 n 个样品中的异常值。在处理完异常值后，对剩余的样品进行系统聚类，通过距离阈值和聚类类数范围来确定聚类数。最后，在每一类中保留 1 个距离样本均值最近的样品，其余样品全部删去，以此达到删去“无用”和“冗余”数据的效果。

基于对问题一的分析，绘制如下的数据预处理流程图 (图3-1)，以此展示数据预处理的具体过程。

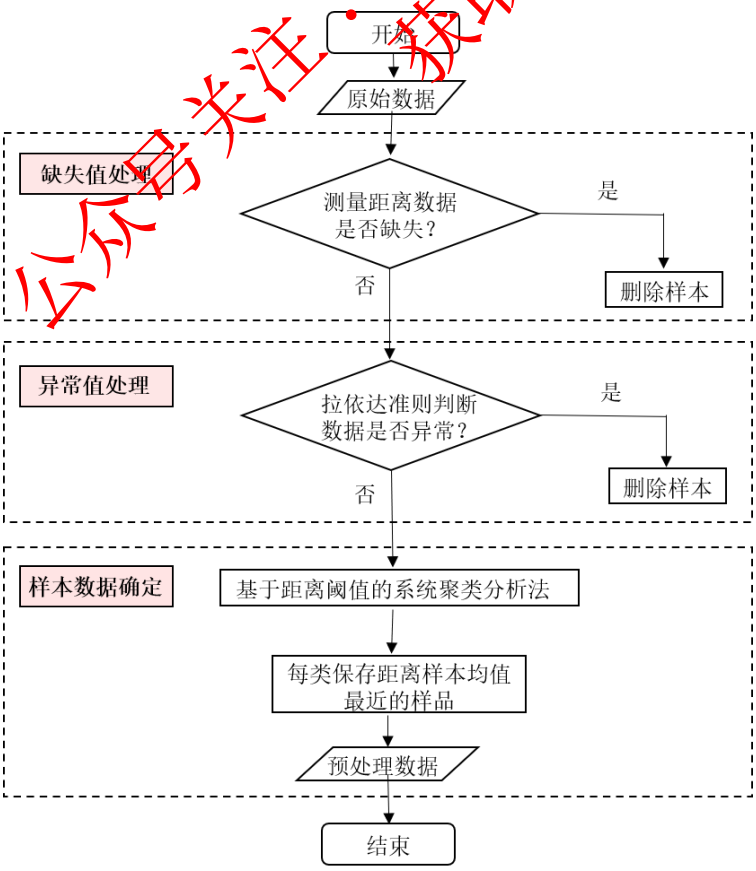

图 **3-1**　数据预处理流程图

3.2 问题一的求解

3.2.1 基于 3σ 准则检测异常值

本文采用 Python 程序对二维表形式数据信息进行数据预处理，对原始数据中所有可能存在的空缺值进行检查，经过检查发现该数据中不存在任何的数据缺失。由于正态分布具有非常典型的中间高、两边低的图形特征，如果样本数据不服从正态分布，我们可以通过直方图很快的分辨出来，所以对于数据分布的正态性研究，我们首选方法是图形观察。利用直方图和 P-P 图进行观察，发现大部分测量距离变量的直方图有正态分布的趋势，如图3-2，我们随机选取两个正常文件数据和两个异常文件数据，分别生成四个测量距离变量的带有正态曲线的直方图，从图中可以看出正常样本数据中的四个测量距离分布是非常符合正态分布形状的，近乎紧贴生成的正态曲线。

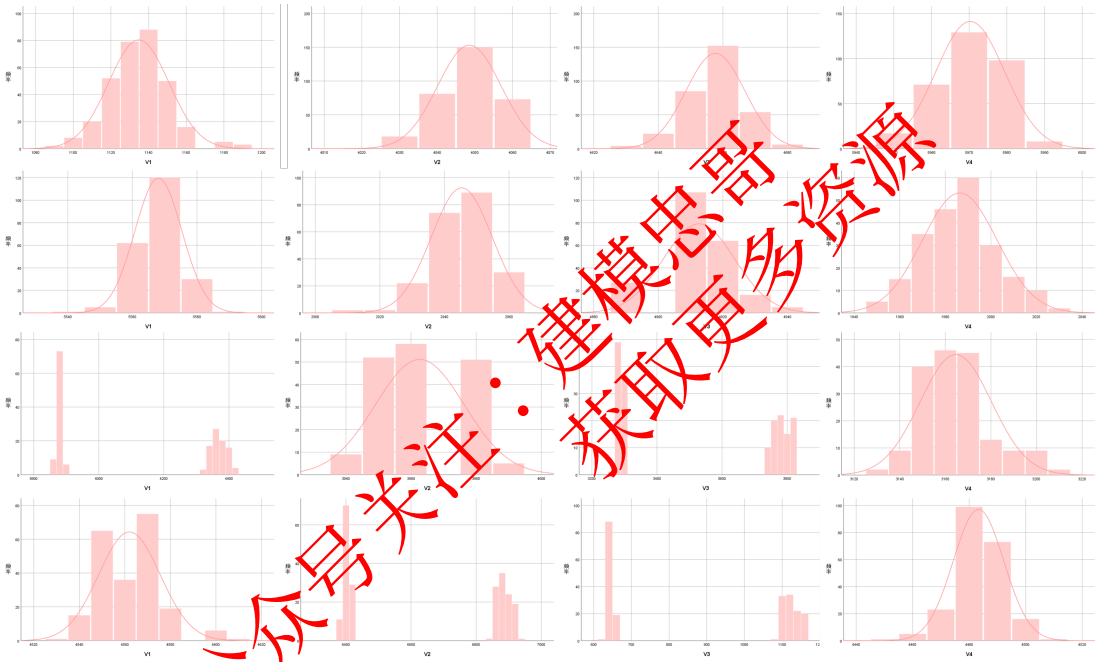

图 **3-2**　带正态曲线的直方图

由上分析，我们判断样本数据服从正态分布或近似正态分布，因此可以利用 3σ 准则又称为拉以达准则来剔除样本数据中的异常值。假设 4 个测量距离变量的数据只含有随机误差，锚点与靶点之间每 0.2-0.3 秒之间测量一次，设靶点到同一个锚点的测量距离数值分别为 $x_1, x_2, \cdots, x_n$，计算其算术平均值 $\bar{x}$、以及剩余误差 $v_i = x_i - \bar{x}(i = 1, 2, \cdots, n)$，利用贝塞尔公式计算出其标准误差 σ，若某个测量值 x_a 的剩余误差 $v_a(1 \le a \le n)$ 满足 $|v_a| = |v_a - \bar{x}| > 3\sigma$，则认为该 x_a 是含有粗大误差的坏值，应予以剔除。其中贝塞尔公式如下：

$$\sigma = (\frac{1}{n-1}\sum_{i=1}^{n} v_i^2)^{\frac{1}{2}} = \frac{\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2/n}{(n-1)^{1/2}}.$$

3.2.2 基于距离阈值的系统聚类分析法

聚类[1] 是将某个对象划分为若干组的过程，使得同一个组内的数据对象具有较高的相似度，而不同组中的对象是不相似的。对于同一个坐标的多组测量距离，我们采用系统聚类分析进行处理。

该系统聚类算法的基本思想是首先将同一个坐标的 n 个样本视作各自成为一类，然后计算类与类之间的距离，选择距离最小的一对样本合并成一个新类，计算在新产生的类别分划下各类之间的距离，再将距离最近的两类合并。停止合并条件设置为：**“正常数据”距离阈值设定为 15 毫米、“异常数据”距离阈值设定为 100 毫米，并且满足正常样本聚类成 1-3 类、异常样本聚类成 2-4 类**，直到所有样本满足这两个停止合并的条件为止。

针对问题一的数据，我们以欧氏距离作为衡量样本间距离的标准，以最短距离法作为衡量类间距离的标准进行聚类。其中欧式距离的计算公式为:

$$d_{ij} = \sum_{k} |x_{ik} - x_{jk}|.$$

类间距离采用的是最短距离法，即两类中相距最近的两样本间的距离：

$$D_{pq} = \min_{x_i \in G_p, x_j \in G_q} d_{ij}.$$

通过以上对问题二的分析，我们可以通过算法 1 对清洗后的数据 (即判断是否存在缺失值，检测异常值后的数据) 进行聚类，算法 1 的迭代框架如下：

算法 1: 基于距离阈值的系统聚类分析法

步骤 1：初始分类。令 $k = 0$，同一坐标点的每个样本自成一类，即 $G_i^{(0)} = \bar{x}_i (i = 1, 2, \cdots, n)$。

步骤 2：计算各类之间的距离 D_{ij}，生成一个对称的距离矩阵 $D^{(k)} = (D_{ij})_{m \times m}$，$m$ 为类的个数。

步骤 3：找出前一步求得的矩阵 $D^{(k)}$ 中的最小元素，设它是 $D_i^{(k)}$ 和 $D_j^{(k)}$ 间的距离，若正常数据的该距离小于 15 毫米（异常数据的该距离小于 100 毫米），则将 $D_i^{(k)}$ 和 $D_j^{(k)}$ 两类合并成一类，于是产生新的聚类 $D_1^{(k+1)}, D_2^{(k+1)}, \cdots$，令 $m = m - 1$，转至 Step4；否则，转至 Step5。

步骤 4：检查类的个数。如果正常数据类数 $m > 1$(异常数据类数 $m > 2$)，令 $k = k + 1$，转至步骤 2；否则，停止。

步骤 5：对于正常数据 (异常数据)，置 $m = 3 (m = 4)$，停止。

如果某一循环中具有最小类之间的距离不止一个类对，则对应这些最小距离的类可以同时合并。上述算法步骤给出了本题从 n 类聚类到理想类数的完全聚类过程。

3.3 问题一的结果

经过上述基于 3σ 准则剔除异常值和基于距离阈值和聚类数的系统聚类分析方法对原始数据进行处理后，最终**“正常数据”保留 582 组数据，“异常数据”保留 713 组数据**。其中正常数据和异常数据的聚类数见表3.1。

表 **3.1**　聚类数

聚类数	正常数据文件号
1	1 3 5 8 11 16 17 19 21 22 23 24 26 27 28 29 30 35 37 4 40 41 44 46 50 54 55 58 64 73 74 79 81 82 83 84 86 89 91 93 94 95 97 98 100 101 105 106 109 110 111 112 114 115 116 117 119 120 122 123 127 128 131 134 135 136 137 138 139 140 143 145 146 147 148 149 150 151 152 153 154 155 157 160 164 166 169 170 173 175 176 177 178 179 180 181 182 184 185 186 187 188 190 191 192 197 199 201 202 207 210 213 215 220 223 224 227 228 233 235 236 238 239 240 241 242 243 245 247 251 252 253 258 267 271 272 273 274 278 279 281 282 285 286 290 291 294 296 297 300 303 305 313 314 320 321 322 324
2	6 10 12 13 14 20 25 31 32 33 34 36 38 39 43 48 49 52 53 56 66 68 70 72 75 77 78 85 88 92 96 102 121 124 129 132 133 156 158 161 168 171 193 203 204 205 206 208 211 212 216 217 221 222 225 226 229 230 231 234 249 250 257 261 268 277 287 289 293 295 299 304 317 318 323
3	2 7 9 15 18 42 45 47 51 57 59 60 61 62 63 65 67 69 71 76 80 87 90 99 103 104 107 108 113 118 125 126 130 141 142 144 159 162 163 165 167 172 174 183 189 194 195 196 198 200 209 214 218 219 232 237 244 246 248 254 255 256 259 260 262 263 264 265 266 269 270 275 276 280 283 284 288 292 298 301 302 306 307 308 309 310 311 312 315 316 319
聚类数	异常数据文件号
2	1 10 100 101 102 103 104 105 106 107 108 109 11 110 111 112 113 116 117 118 119 12 120 121 122 123 124 126 127 128 129 13 130 132 133 135 136 137 139 14 140 141 142 143 144 145 146 147 148 15 151 152 153 154 155 157 16 160 161 164 165 166 167 168 169 17 170 171 172 173 174 175 177 178 179 18 181 183 185 188 189 19 190 191 192 193 195 196 199 2 20 200 201 202 205 206 207 208 209 21 211 212 213 214 215 216 218 219 220 221 222 223 224 225 227 228 229 23 230 231 232 233 234 235 236 237 238 239 24 240 241 243 244 245 246 247 248 249 25 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 27 270 271 272 273 274 275 276 277 278 279 28 280 281 282 283 284 285 286 287 288 289 29 290 291 292 293 294 295 296 297 298 299 3 30 300 301 302 303 304 305 306 307 308 31 310 311 312 313 314 315 316 317 318 319 32 320 321 322 323 324 33 35 36 37 38 4 41 42 44 45 46 47 48 5 50 51 52 53 54 55 57 58 59 60 61 62 63 64 67 68 69 7 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 86 87 88 89 9 90 91 92 93 96 97 98 99
3	115 125 134 149 150 159 162 163 180 186 194 197 204 217 226 242 34 39 40 49 8 94 95
4	114 131 138 156 158 176 182 184 187 198 203 210 22 26 309 43 56 6 65 66 85

所有的数据不存在缺失值，其中，通过拉依达准则处理得到**24. 正常样本中有 1 个异常值，109. 正常样本中有 2 个异常值，1. 异常样本中有 4 个异常值，100. 异常样本中有 6 个异常值**，具体包含异常值的样本数据见表3.2。

表 **3.2**　问题一的异常值样本

文件	距离 1	距离 2	距离 3	距离 4
24. 正常	3290	4650	2590	70
109. 正常	4910	5320	2020	70
	4910	5330	2024	70
1. 异常	1250	4540	4580	6290
	1270	4540	4580	6280
	1280	4540	4580	6280
	1270	4550	4600	6280
100. 异常	1240	4550	4580	6290
	1280	4550	4580	6300
	1510	3560	5350	5700
	1500	3570	5880	5690
	1510	3560	5660	5700
	1510	3560	5400	5700

我们剔除所有异常值后，对剩余的样品进行系统聚类，通过距离阈值和聚类类数范围来确定聚类数，**4. 正常样本聚为 1 类，109. 正常样本聚为 1 类，1. 异常样本聚为 2 类，100. 异常样本聚为 2 类**。最后在每一类中保留一个距离样本均值最近的样品，其余样品全部删去。得到的具体数据如表3.3。

表 **3.3**　问题一的数据结果

文件	距离 1	距离 2	距离 3	距离 4
24. 正常	3280	4660	2600	3910
109. 正常	4890	5330	2030	2880
1. 异常	1250	4550	4550	6300
	770	5030	4550	6300
100. 异常	1510	3560	4930	5700
	1510	3770	4680	5710

4 问题二的分析与求解

4.1 问题二的分析

4.1.1 概念分析

TOF 测距技术是通过计算信号在两个锚点和靶点之间的飞行时间，再乘以光速得出它们之间的距离，建立关系以获取靶点的三位精确定位坐标值。为了简化说明，首先给出对于信号无干扰的情况的几何原理图。

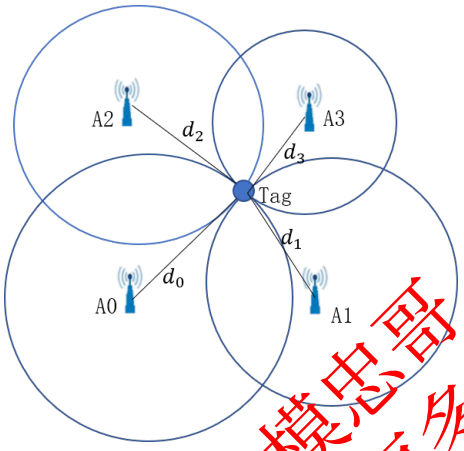

图 **4-1** 基于 **TOF** 的三维定位原理投影图

如图4-1所示，A0、A1、A2、A3 为四个锚点，其与靶点的距离分别为 d_0、d_1、d_2、d_3，分别以 A0、A1、A2、A3 为球心，d_0、d_1、d_2、d_3 为半径画球，假设测量距离是精确的，那么这四个球就会相交于一个点，则该点为靶点 Tag 所在位置，对应的三维坐标即为锚点定位出的靶点的三维坐标。

对于有干扰的情况，由于测量过程信号被干扰，导致到达时间变长，从而使得测量距离会有较大误差，干扰过程如下，信号有干扰的情况下对三维定位坐标的精度影响示意图如图4-2所示。

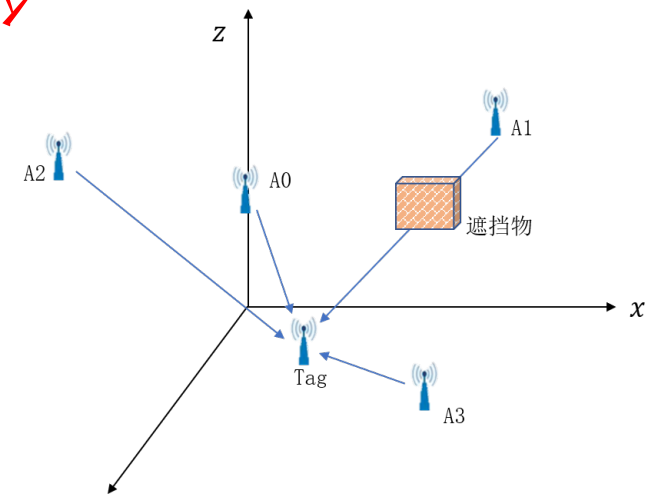

图 **4-2** 信号有干扰的室内三维定位示意图 **(A1** 被遮挡时**)**

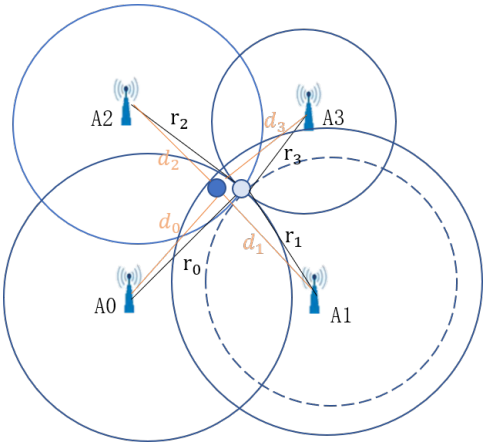

图 **4-3**　信号有干扰的三维定位原理投影图 **(A1** 被遮挡时**)**

如图4-3所示，以 A0、A1、A2、A3 为球心，d_0、d_1、d_2、d_3 为半径画球，假设锚点 A1 被遮挡，显然基站 A1 到靶点的测量距离 d_1 大于真实距离 r_1，则以 A0、A1、A2、A3 为球心或以 A0、A1、A3 为球心或以 A1、A2、A3 为球心的三个球有个公共的相交区域，靶点的位置则位于该区域之中，但无法确定其具体的三维坐标值。如何在已知的空间区域中抑制或消除各类误差，获得最终靶点的精确位置，是求解信号有干扰下定位问题的关键。

4.1.2 定位分析

本题给出无干扰和有干扰环境下靶点到 4 个锚点的距离测量数据以及所有锚点的三维坐标，要求根据这些已知数据估计出靶点的三维位置坐标，并要求对附件 2 中提供的前 5 组信号无干扰的数据和后 5 组信号有干扰的数据进行三维坐标精确定位。

对于正常数据，可以直接用数据给定的靶点到 4 个锚点的测量距离、4 个锚点坐标和建立最小二乘模型求解出一个近似点坐标，然后用这个近似点的 3 维坐标作为特征建立 ELM 极限学习机优化模型，使得输入近似点的 3 维坐标，输出对应的靶点定位的 3 维坐标。

对于异常数据，因为干扰会造成靶点的定位结果和真实坐标之间的误差较大。所以为了减小干扰带来的误差，需要采取一些措施来抑制干扰带来的误差。我们可以选取 3 个测量距离求解出一个近似坐标点，由于题目中给出四个锚点，有四个测量距离，所以一共可以定位 $C_4^3 = 4$ 个近似点。为了抑制遮挡误差对定位的影响，我们对这 4 个坐标点建立残差加权误差抑制模型，得出一个近似点坐标，然后用这个 3 维近似点坐标作为特征建立 BP 神经优化模型，使得输入近似点的 3 维坐标，输出对应的靶点定位的 3 维坐标。

4.2 信号无干扰定位模型

4.2.1 最小二乘算法 (LS)

空间中 4 个锚点 A_0, A_1, A_2, A_3, 待测靶点为 $X = [x, y, z]^T$，若不存在测量误差，则测量距离的平方等于待测靶点到每个锚点之间的距离平方，即

$$d_i^2 = (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2, \tag{1}$$

展开得

$$d_i^2 = k_i - 2x_i x - 2y_i y - 2z_i z + x^2 + y^2 + z^2,$$

其中 $k_i = x_i^2 + y_i^2 + z_i^2, i = 0, 1, 2, 3.$ 且 d_i 为靶点到第 i 个锚点的测量距离。

设第 1 个锚点到其他锚点的距离平方差为 $d_{i,0}$，那么

$$\begin{aligned} d_{i,0} &= d_i^2 - d_0^2 \\ &= [(x-x_i)2 + (y-y_i)^2 + (z-z_i)^2] - [(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2] \\ &= 2\begin{bmatrix} x_0 - x_i & y_0 - y_i & z_0 - z_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + k_i - k_0, \end{aligned}$$

转换成矩阵形式表示：

$$AX = b,$$

其中

$$A = \begin{bmatrix} x_0 - x_1 & y_0 - y_1 & z_0 - z_1 \\ x_0 - x_2 & y_0 - y_2 & z_0 - z_2 \\ x_0 - x_3 & y_0 - y_3 & z_0 - z_3 \end{bmatrix}, b = \frac{1}{2}\begin{bmatrix} d_{1,0} - k_1 + k_0 \\ d_{2,0} - k_2 + k_0 \\ d_{3,0} - k_3 + k_0 \end{bmatrix}.$$

实际上，测量误差是存在的，等式 (1) 不一定成立，我们的目标是测量距离与真实距离的误差达到最小，同时保证坐标在题目给定的范围之内，所以考虑如下最小二乘优化 (LS)[3] 问题：

$$\begin{aligned} \min_X \quad & ||AX - b||_2^2 \\ s.t. \quad & \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \leq X \leq \begin{bmatrix} 5000 \\ 5000 \\ 3000 \end{bmatrix} \end{aligned}$$

最后我们用 MATLAB 中的 lsqlin 函数对该优化问题进行求解。

4.2.2 极限学习机 (ELM)

极限学习机是一种单隐藏层前馈神经网络。ELM 通过随机选取输入权值和隐藏层节点参数，求解隐藏层输出权值的伪逆矩阵得到输出权值矩阵，就完成了神经网络的训练过程[10]，具体的训练过程如下：

假设有 $N(N = 582)$ 个样本 (X_i, t_i), 其中 $X_i = [x_{i1}, x_{i2}, x_{i3}]^T$ 为有最小二乘算法求出来的近似点坐标，$t_i = [t_{i1}, t_{i2}, t_{i3}]^T$ 为期望输出即靶点的真实坐标，则极限学习机的神经网络为

$$\sum_{i=1}^{L} \beta_i G(W_i \cdot X_j + b_i) = o_j, j = 1, 2, ..., N$$

其中 n 表示输入神经元的个数,L 表示隐藏层神经元的个数，m 表示输出神经元的个数，$G(.)$ 表示隐藏层的激活函数,$W_i = [w_{i1}, w_{i2}, ..., w_{in}]^T$ 为输出权重，β_i 为输出权重，b_i 为第 i 个隐藏神经元的偏置,o_j 表示第 i 个样本的输出。

当隐藏层节点数足够时，ELM 具有较好的泛化能力，通常取隐藏层神经元 L 大于输入神经元 n, 单隐层神经网络学习的目标是是的输出的误差最小，即使得

$$\sum_{j=1}^{N} ||o_j - t_i|| = 0,$$

也就是存在 β_i, W_i, b_i, 使得：

$$\sum_{i=1}^{L} \beta_i G(W_i \cdot X_j + b_i) = t_j, j = 1, 2, ..., N$$

用矩阵可以表示为

$$H\beta = T,$$

其中

$$H = \begin{bmatrix} G(w_1x_1+b_1) & G(w_2x_1+b_2) & \cdots & G(w_Lx_1+b_L) \\ G(w_1x_2+b_1) & G(w_2x_2+b_2) & \cdots & G(w_Lx_2+b_L) \\ \vdots & \vdots & \ddots & \vdots \\ G(w_1x_N+b_1) & G(w_2x_N+b_2) & \cdots & G(w_Lx_N+b_L) \end{bmatrix}, \beta = \begin{bmatrix} \beta_1^T \\ \beta_2^T \\ \cdots \\ \beta_L^T \end{bmatrix}. T = \begin{bmatrix} T_1^T \\ T_2^T \\ w_i \cdots \\ T_N^T \end{bmatrix}.$$

单隐藏层神经网络学习的目标是寻找 $\hat{W}_i, \hat{b}_i, \hat{\beta}_i$ 使得

$$(\hat{W}_i, \hat{b}_i, \hat{\beta}_i) = \arg\min_{W,b,\beta} ||H(W_i, b_i)\beta_i - T||, i = 1, 2, ..., L$$

即最小化以下的损失函数

$$E = \sum_{j=1}^{N} \left[\sum_{i=1}^{L} \beta_i G(W_i \cdot X_j + b_i) - T_j \right]^2$$

在 ELM 神经网络中，随机确定权重 W_i 和隐藏层偏置 b_i，H 也就被唯一确定。所以线性系统 $H\beta = T$ 的最小二乘解为

$$\hat{\beta} = H^{\dagger}T.$$

4.2.3 正常数据的定位算法

算法 2: 正常数据的定位算法

输入: $d_0, d_1, d_2, d_3, A0, A1, A2, A3$

输出: x, y, z

步骤 1： 将 $d_0, d_1, d_2, d_3, A0, A1, A2, A3$ 代入最小二乘算法，求解出近似点坐标。

步骤 2： 将近似点坐标代入 ELM 算法中进行训练，调整参数，当测试集的定位点坐标与真实坐标的平均误差达到最优时，停止训练，输出靶点的定位坐标 (x, y, z)。

4.3 信号有干扰定位模型

4.3.1 残差加权误差抑制算法 (RWGH)

由于 4 个基站中都只有一个基站被干扰，我们只用 3 个到基站的距离求解出一个近似坐标点，所以建立残差加权 (RWGH) 算法模型[4] 通过穷举基站个数为 3 的分组，利用通过最小二乘算法得到的每种组合下的靶点的初始估计坐标，同时求得该组合的残差[1]。

假设总共有 N 个基站 (此时 $N = 4$)，我们考虑每种组合的基站数目为 3 的情况，那么共有

$$M = C_N^3 = 4$$

种组合，其索引集为 $S_k, k = 1, 2, \cdots, M$。

残差定义为：

$$R_{es} = \sum_{i \in S_k} (r_i - ||X_k - X_i||),$$

其中，r_i 为锚点 A_i 的测量距离，X_k 表示在该组合下的靶点估计位置坐标，X_i 表示锚点 A_i 的坐标。为了消除不同组合下基站数目对残差大小的影响，定义归一化残差为：

$$R_{eS_k} = \frac{R_{es}(X, S_k)}{Size(S_k)}.$$

最后，以归一化残差的倒数作为权值，对 M 个组合数的估计结果进行残差加权，得到靶点的最终估计坐标为：

$$X = \frac{\sum_{k=1}^{M} X_k (R_{eS_k})^{-1}}{\sum_{k=1}^{M} (R_{eS_k})^{-1}}.$$

信号无干扰的时候定位出来的坐标点与真实坐标的误差较小，当信号有干扰的时候定位出来的坐标点与真实坐标的误差会比较大。残差加权算法主要是通过以残差的倒数进行加权求和，来消除信号干扰误差所带来的影响。

4.3.2 BP 神经网络优化坐标

1. **BP 算法的定义值**

   BP 人工神经网络算法（Back Propagation Artificial Neural Network）也被称为误差反向传播神经网络算法。其本质是一种网络连接权值的训练方法，且为监控式学习算法，而利用该方法所建层状神经网络称为 BP 网络[2]。从结构上讲，BP 人工神经网络具有输入层、隐含层和输出层三层结构，同一层单元之间不存在相互连接，层与层之间多采用全连接的方式。基本思想：首先计算网络实际输出与期望输出的方差，按降低其减小的原则，将其反向传播回去，并借以修正权值。其中反向传播是指误差信号的反向传播，而非信息流的反向传播[6]。

2. **网络的前馈计算**

   考虑一个二层神经网络. 设网络的某层中第 j 个节点在给定一个训练样本时，其总输入为：

$$net_j = \sum_i O_i \omega_{ij},$$

   其中，第 j 个节点的输出是其总输出的一个交换：

$$O_j = f_s(net_j) = \frac{1}{1 + e^{-net_j}} f_s'(net_j) = f_s(net_j)(1 - f_s(net_j)).$$

3. **BP 网络其权值的调整规则**

   网络的误差函数：$e = \frac{1}{2}\sum_{j=1}^{m}(t_j - O_j)^2$, 权值的修改量：$\triangle\omega_{ij} = -\eta\frac{\partial e}{\partial \omega_{ij}}$，其中

$$\omega_{ij}(N) = \omega_{ij}(N-1) + \triangle\omega_{ij},$$

$$\frac{\partial e}{\partial \omega_{ij}} = \frac{\partial e}{\partial O_j} \cdot \frac{\partial O_j}{\partial net_j} \cdot \frac{\partial net_j}{\partial \omega_{ij}},$$

$$\frac{\partial net_j}{\partial \omega_{ij}} = \frac{\partial\left(\sum_i O_i \omega_{ij}\right)}{\partial \omega_{ij}} = O_i,$$

$$\frac{\partial O_j}{\partial net_j} = \frac{\partial\left(f_s\left(net_j\right)\right)}{\partial net_j} f_s'\left(net_j\right) = f_s\left(net_j\right)\left(1 - f_s\left(net_j\right)\right) = O_j\left(1 - O_j\right),$$

$$\frac{\partial e}{\partial O_j}=\frac{\partial\left(\frac{1}{2}\sum_{j=1}^{m}\left(t_j-O_j\right)^2\right)}{\partial O_j}=-\left(t_j-O_j\right),$$

$$\frac{\partial e}{\partial \omega_{ij}}=-O_i\cdot O_j\left(1-O_j\right)\left(t_j-O_j\right).$$

4. **BP 学习算法的步骤**

(i) 初始化：依据实际问题，设置输入、输出及隐层节点神经网络个数，同时设初始权重任意小。

(ii) 提供训练样本：$(x_1,x_2,\cdots,x_n;t_1,t_2,\cdots,t_m)$.

(iii) 计算实际输出：$O_j=\left[1+exp\left(-\sum_i O_i\omega_{ij}\right)\right]^{-1}$，$e=\frac{1}{2}\sum_{j=1}^{m}\left(t_j-O_j\right)^2$

(iv) 权值调整：$\omega_{ij}\left(N-1\right)=\omega_{ij}\left(N\right)+\triangle\omega_{ij}$

对于隐层–输出层：

$$\triangle\omega_{ij}=-\eta\frac{\partial e}{\partial\omega_{ij}}=\eta O_i\cdot O_j\left(1-O_j\right)\left(t_j-O_j\right)$$

对于输入层–隐层：

$$\delta_k=-O_k\left(1-O_j\right)$$

(v) 返回 (ii) 步，重复步骤，直到误差满意为止。

5. **BP 神经网络的建立**

本文采用 BP 神经网络作为定位目标预测模型。通过上述基于残差加权误差抑制算法可以分别得出异常数据的靶点近似坐标点，再利用神经网络的学习能力建立精确定位模型，进行位置预测，为室内定位提供辅助支撑。

最基本的 BP 神经网络是包括输入层、隐含层 (隐含层可以有一层或多层) 和输出层这三层节点的前馈网络，我们采用两层神经元网络 (只有一层隐含层) 来建立室内定位目标预测模型，如图4-4所示。

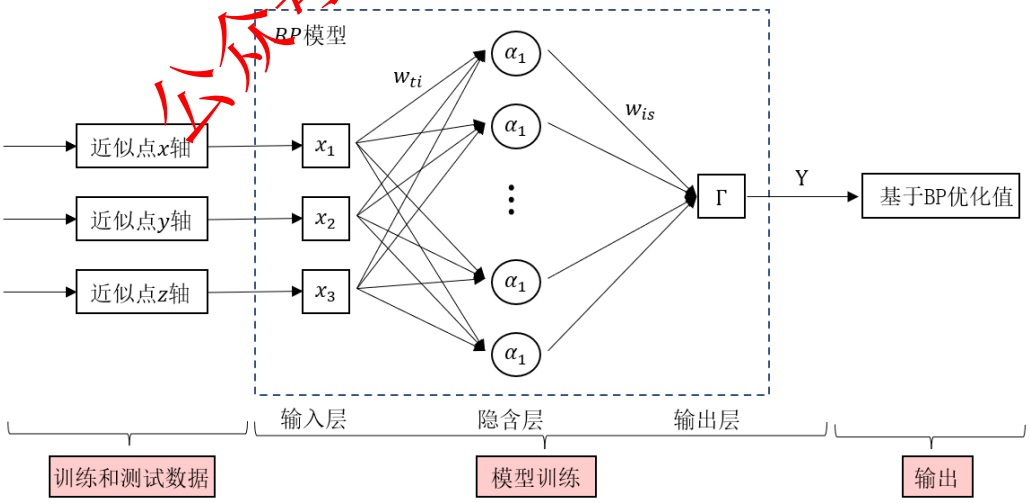

图 **4-4**　**BP** 神经网络预测模型

(1) 输入层：x_1,x_2,x_3 为网络的输入，分别是正常数据由最小二乘算法 (异常数据由残差加权误差抑制算法) 得出的近似点的三维坐标。

(2) 输出层：靶点精确的三维坐标 Y。

(3) 隐含层：Robert 证明只要隐含层的节点数足够多，只含有一个隐含层的神经网络模型就可以任意的精度逼近一个非线性函数，因此我们实际问题的 BP 神经网络模型选择采用一个隐含层。w_{ti}, w_{is} 分别代表输入层与隐含层、隐含层和输出层之间的权值，其中 $i = 1, 2, \cdots, 7$(w_{1i} 为输入 x_1 与隐含层之间的个权值)。α_i 和 Γ 分别代表输入层与隐含层、隐含层和输出层之间的阈值。

(4) 隐含层节点数的确定：经过大量实验，我们决定隐含层的节点数为 7 个节点。

6. **激励函数的选取**

BP 神经网络模型，网络选用 S 型传递函数：$f(x) = \frac{1}{1+e^{-x}}$, 通过反向传播误差函数

$$e = \frac{\sum_{i=1}^{n} t_i + O_i}{2},$$

其中 t_i: 为期望输出,O_i: 为网路的计算输出。不断调解网络权值和阈值使误差函数 e 达到极小。

BP 神经网络通常采用可微函数和现性函数作为网络的激励函数。本 BP 网络模型选择 S 型正切函数 tansig 作为隐层神经元的激励函数。我们利用 mapminmax 函数来进行归一化，由于网络输出归一化到 [−1, 1] 范围内，所以我们以线性函数 purelin 作为输出层神经元的激励函数。

4.3.3 异常数据的定位算法

算法 3: 异常数据的定位算法

输入: $d_0, d_1, d_2, d_3, A0, A1, A2, A3$

输出: x, y, z

步骤 1： 将 $d_0, d_1, d_2, d_3, A0, A1, A2, A3$ 代入残差加权误差抑制算法，求解出近似点坐标。

步骤 2： 将近似点坐标代入 BP 神经网络算法中进行训练，调整参数，当测试集的定位点坐标与真实坐标的平均误差达到最优时，停止训练，输出靶点的定位坐标 (x, y, z)。

4.4 定位精度评价指标

平均定位误差 ALE 是最常用的一种评估定位误差的指标，在本节中，主要考虑平均误差对定位算法的 3 维精度、2 维精度以及 1 维精度进行评估。假设第 k 个样本的靶点的真实坐标为 (x_k, y_k, z_k)，其对应的估计坐标位置为 (x_k', y_k', z_k')。

(1) 3 维 (x, y, z) 精度评价指标

平均定位误差 (ALE) 定义为:

$$ALE = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_k - x_k')^2 + (y_k - y_k')^2 + (z_k - z_k')^2}$$

(2) 2 维 (x, y) 精度评价指标

平均定位误差 (ALE) 定义为:

$$ALE = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_k - x_k')^2 + (y_k - y_k')^2}$$

(3) 1 维 x, y, z 精度评价指标
平均定位误差 (ALE) 定义为:

$$ALE_x = \frac{1}{N}\sum_{i=1}^{N}|x_k - x_k^{'}|, ALE_y = \frac{1}{N}\sum_{i=1}^{N}|y_k - y_k^{'}|, ALE_z = \frac{1}{N}\sum_{i=1}^{N}|z_k - z_k^{'}|$$

4.5 模型的求解

在 Matlab 编程平台建立室内定位预测模型的过程中，我们先将正常数据和异常数据分开确定测试集和训练集，在聚类之后，**从 324 个位置中均匀选 32 个位置，即每个高度选 8 个点，作为测试集**，用于验证预测模型，剩余的的样本数据用于训练预测模型，具体的样本数分配见表4.1。

表 **4.1**　训练样本和验证样本分配

数据类型	样本组数	训练组数	验证组数
正常数据	582	437	145
异常数据	713	535	178

对于正常距离数据的定位，首先建立最小二乘算法模型，解得 x, y, z 的平均误差分别为 41.87mm，34.29mm，396.18mm；对于异常距离数据的定位，利用残差加权误差抑制算法，得到 x, y, z 的平均误差分别为 155.82mm，150.18mm，474.02mm。为了优化模型的误差，我们分别利用 ELM 极限学习机和 BP 神经网络进行优化。

完成上面两个步骤，得出结果如图4-5至图4-8所示。由图4-5和图4-7可以看出，对于二维来说，LS+ELM(RWGH+BP) 复合模型的误差较比之前 LS(RWGH) 单个模型的误差没有明显的变化。由图4-6和图4-8可以看出对于三维来说，LS+ELM (RWGH+BP) 复合模型的误差较比之前 LS(RWGH) 单个模型有明显减小，说明用复合模型能够对 x, y, z 起到良好的修正效果，其中对 z 的修正效果最为明显。

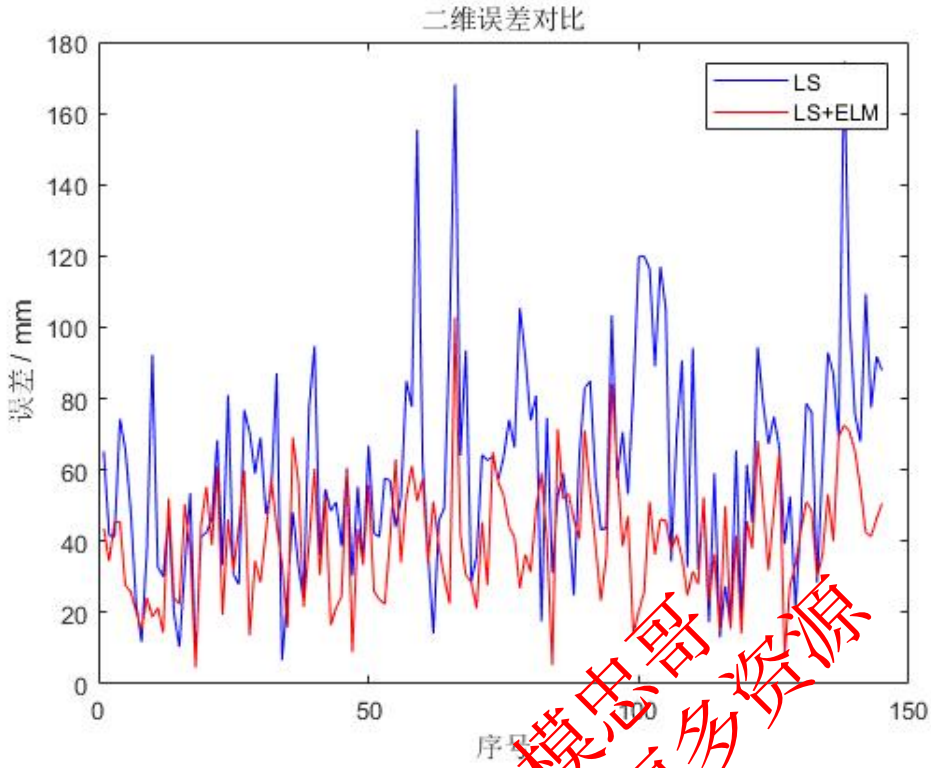

图 **4-5** 正常数据定位坐标的二维误差

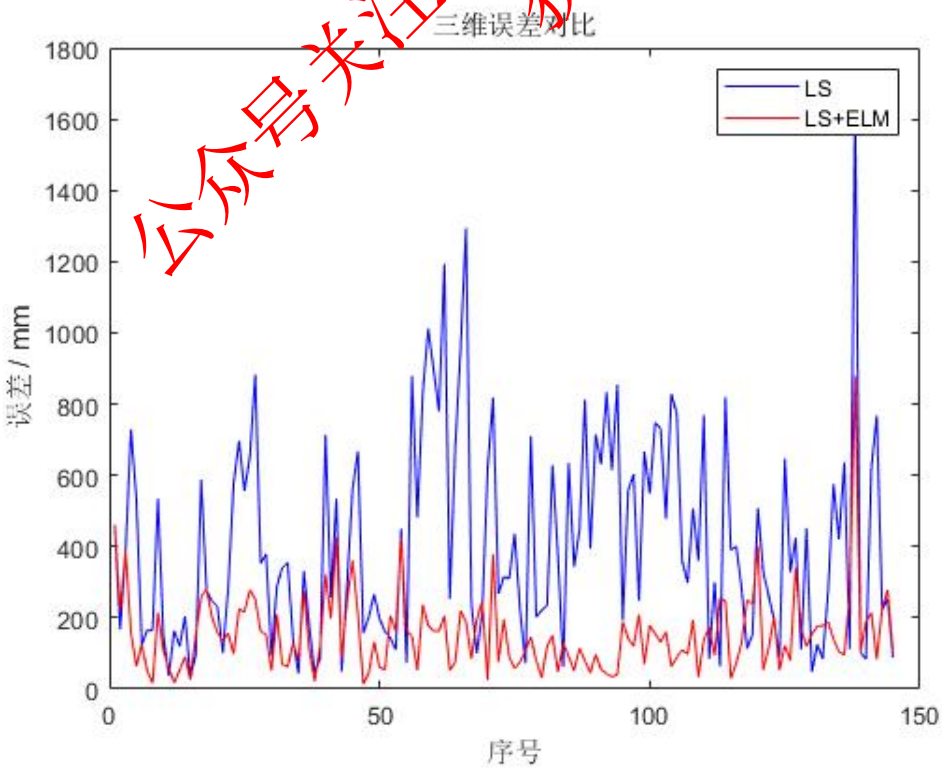

图 **4-6** 正常数据定位坐标的三维误差

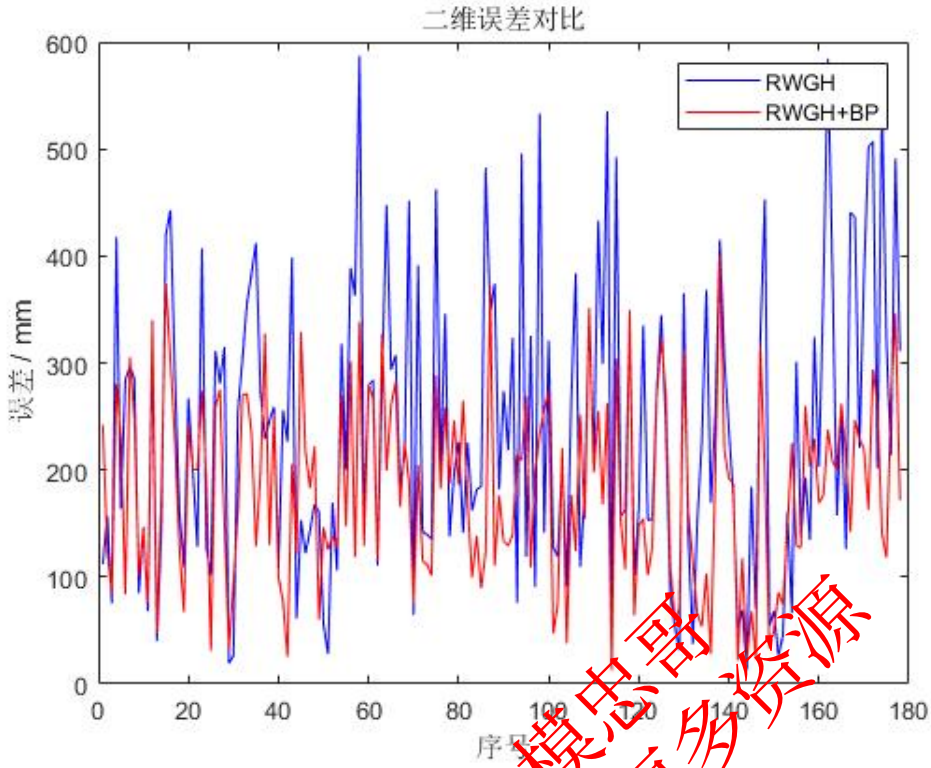

图 **4-7**　异常数据定位坐标的二维误差

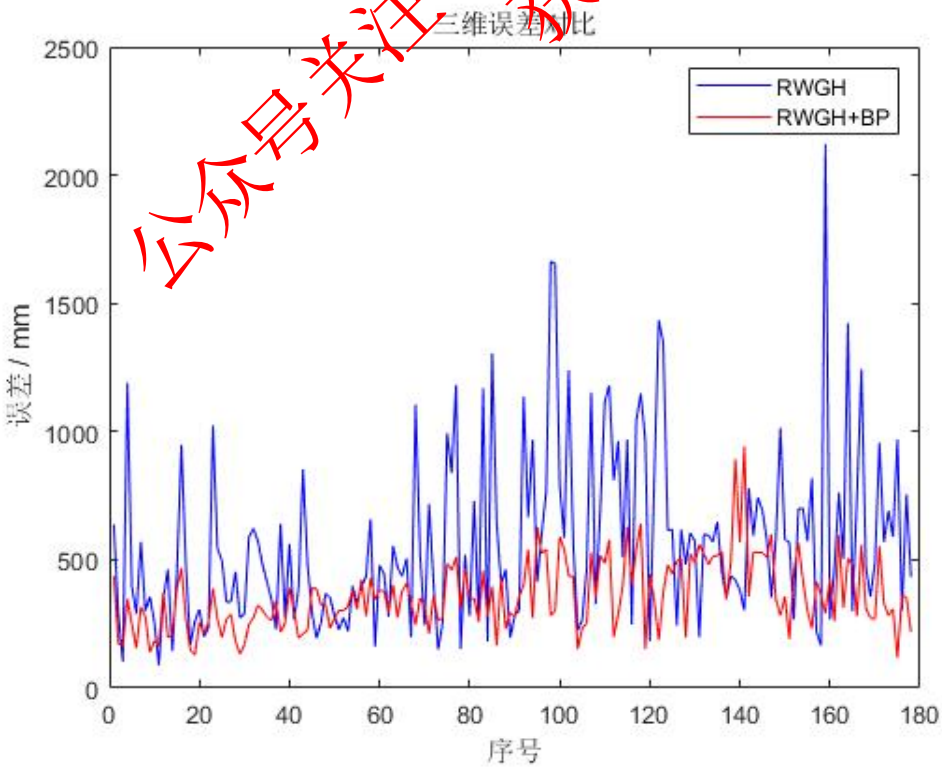

图 **4-8**　异常数据定位坐标的三维误差

经过计算，对于正常数据的定位，利用 LS+ELM 神经网络的模型得到 x, y, z

的平均误差分别为 25.67mm，27.02mm，130.29mm；对于异常数据的定位，利用 RWGH+BP 神经网络的模型得到 x, y, z 的平均误差分别为 137.62mm，127.94mm，364.84mm。正常数据定位模型和异常数据定位模型的 3 维 (x, y, z) 精度、2 维 (x, y) 精度以及 1 维 x, y, z 的精度见表4.2。

表 **4.2**　模型的精度 **(**单位：**mm)**

数据类型	3 维 (x,y,z) 精度		2 维 (x,y) 精度		1 维 x,y,z 精度					
	LS (RWGH)	**LS+ELM (RWGH+BP)**	LS (RWGH)	**LS+ELM (RWGH+BP)**	LS (RWGH)			**LS+ELM (RWGH+BP)**		
正常	406.0	**141.7**	60.2	**41.3**	41.9	34.3	396.2	**25.7**	**27.0**	**130.3**
异常	564.1	**419.4**	237.1	**199.1**	155.8	150.2	474.0	**137.6**	**127.9**	**364.8**

通过对表 4.2 分析，我们发现利用 LS+ELM(RWGH+BP) 复合模型计算出靶点的定位坐标的误差比只用单一的 LS(RWGH) 模型计算出来的定位坐标误差有较为显著的下降，说明我们利用 ELM 极限学习机 (BP 神经网络) 对 LS(RWGH) 算法定位出来的三维坐标进行优化是合理的，同时表4.3的模型训练集和测试集的精度对比也说明了我们的模型较为合理。

表 **4.3**　复合模型训练集和测试集的精度对比 **(**单位：**mm)**

误差	正常		异常	
	训练集	测试集	训练集	测试集
1 维 x	21.5	25.7	127.8	137.6
1 维 y	24.3	27.0	120.9	127.9
1 维 z	127.6	130.3	356.7	364.8
2 维 (x,y)	37.4	41.3	193.1	199.1
3 维 (x,y,z)	136.2	141.7	411.3	419.4

4.6 模型对比分析

4.6.1 对比方法介绍

为了进一步验证正常数据的极限学习机 (ELM) 和异常数据的 BP 神经网络模型对近似坐标点的优化效果。我们尝试了其他优化模型的算法：广义回归神经网络 (GRNN) 和人工蜂群算法 (ABC)。

(1) 广义回归神经网络[11]: 广义回归神经网络基本上是一种基于神经网络的函数。广义回归神经网络属于概率神经网络的范畴。概率神经网络的使用是特别有利的，因为网络通过数据一次“学习”，并且一旦存储，就可以从例子中进行归纳。换句话说，网络是一根电缆，它只需几个可用的训练样本就能收敛到数据的底层函数。在广义回归神经网络方法中，如果训练集可用，因变量 y 对自变量 x 的回归估计 y 的最大可能值。

(2) 人工蜂群算法[12][13]: 人工蜂群算法是一种基于流行的群体智能的随机算法，由 Karaboga 于 2007 年提出，用于解决各种优化问题。该算法模仿智能蜜蜂的行

为，目标是搜索食物来源。在一个群体中，蜜蜂根据它们的角色被分成三组:雇佣蜂、旁观蜂、侦察蜂。ABC 算法执行四种不同的选择流程:(i) 全局选择过程: 被旁观蜂利用；(ii) 本地选择过程: 在一个地区由雇佣的和旁观者蜜蜂进行；(iii) 贪婪选择过程: 被所有蜜蜂使用；(iv) 随机选择过程: 侦察蜂使用。

4.6.2 主要参数说明

设置合适的参数对于机器学习方法的模型训练至关重要，为了切实保证实验结果的客观性和可对比性，本题中所用到的所有模型均采用其默认参数，各个模型的主要参数设置如表4.4所示。

表 **4.4** 主要参数统计表

方法	参数	赋值
神经网络 (BP)	隐含层激励函数	tansig
	输出层激励函数	purelin
	网络训练函数	trainlm
	网络性能	mse
	隐含层神经元数	7
	网络迭代数 epochs	1000
	期望误差 goal	0.00001
	学习效率 lr	0.01
极限学习机 (ELM)	隐含层神经元数	30
	激活函数	Sigmoidal
广义回归神经网络 (GRNN)	desired_spread	0.2
人工蜂群算法 (ABC)	最大迭代次数	20
	单次保存的蜜源最大数量	100
	跟随蜂数目	100
	蜜源搜索范围扩大系数	1

4.6.3 结果展示及分析

首先利用最小二乘算法和残差加权误差抑制算法分别计算出正常数据和异常数据的近似坐标点，然后我们采用上面四种模型分别对其进行训练，待模型训练好以后，我们将测试集输入模型，将最终得到的结果与真实值比较，计算出他们的平均定位根误差，误差的值越小代表优化结果越好，所有的实验结果如表4.5和4.6所示。

表 **4.5**　正常数据不同模型的精度对比 **(**单位：**mm)**

误差 / 方法	1 维 x	1 维 y	1 维 z	2 维 (x,y)	3 维 (x,y,z)
ELM	**25.67**	**27.02**	**130.29**	**41.30**	**141.67**
BP	33.90	29.70	139.70	48.30	156.10
GRNN	77.56	78.65	135.00	112.87	199.40
ABC	50.07	41.32	215.96	69.90	235.08

从表4.5可以看出，对于正常数据，ELM 模型预测出来的靶点坐标与真实坐标的一维、二维、三维误差相比于其他三种模型的误差小，说明相比于其他三种模型，ELM 模型对近似点坐标的优化效果是最好的。

表 **4.6**　异常数据不同模型的精度对比 **(**单位：**mm)**

误差 / 方法	1 维 x	1 维 y	1 维 z	2 维 (x,y)	3 维 (x,y,z)
BP	**137.60**	**127.90**	**364.80**	**199.10**	**419.40**
ELM	142.12	130.43	375.29	207.53	456.85
GRNN	163.74	164.67	375.28	252.34	484.98
ABC	170.25	209.39	551.26	305.81	663.56

同样地，从表4.6可以看出，对于异常数据，BP 模型预测出来的靶点坐标与真实坐标的一维、二维、三维误差相比于其他三种模型的误差小，说明相比于其他三种模型，BP 模型对近似点坐标的优化效果是最好的。

4.7 问题二的结果

正常数据定位模型和异常数据定位模型的 3 维 (x,y,z) 精度、2 维 (x,y) 精度以及 1 维 x,y,z 的精度见表4.7。

表 **4.7**　模型的精度 **(**单位：**mm)**

数据类型	3 维 (x,y,z) 精度	2 维 (x,y) 精度	1 维 x,y,z 精度		
正常	**141.7**	**41.3**	**25.7**	**27.0**	**130.3**
异常	**419.4**	**199.1**	**137.6**	**127.9**	**364.8**

利用两种定位模型分别对附件 2 中提供的前 5 组信号无干扰的数据和后 5 组信号有干扰数据进行精确定位，得到的 3 维坐标如表4.8和表4.9所示。

表 **4.8**　附件 **2** 信号无干扰数据的定位结果 **(**单位：**mm)**

序号	测量距离				定位坐标	
					LS	**LS+ELM**
1	1320	3950	4540	5760	(1179,678,691)	**(1188,703,960)**
2	3580	2580	4610	3730	(3175,1715,763)	**(3210,1693,1033)**
3	2930	2600	4740	4420	(2738,1167,809)	**(2722,1216,1316)**
4	2740	2720	4670	4790	(2449,1008,2278)	**(2478,1005,1803)**
5	2980	4310	2820	4320	(1480,2542,2134)	**(1512,2529,1927)**

表 **4.9**　附件 **2** 信号有干扰数据的定位结果 **(**单位：**mm)**

序号	测量距离				定位坐标	
					RWGH	**RWGH+BP**
6	2230	3230	4910	5180	(1856,901,2821)	**(2091,838,1241)**
7	4520	1990	5600	3360	(4284,1570,524)	**(4155,1592,1212)**
8	2480	3530	4180	5070	(1788,1322,1533)	**(1762,1291,1341)**
9	4220	2510	4670	3490	(3557,1975,1634)	**(3578,2056,1598)**
10	5150	2120	5800	2770	(4765,1887,1585)	**(4731,2099.1582)**

5 问题三的分析与求解

5.1 问题三的分析

针对问题三，虽然我们的训练数据仅采集于同一实验场景 (实验场景 1)，但定位模型的参数包含锚点的坐标、靶点的范围、靶点到锚点的测量距离，所以我们的定位模型在改变这些参数时依旧是适用的，也就是说能够应用于不同实际场景上。题目要求我们分别用上述建立的定位模型对附件 3 中在实验场景 2 采集的 10 组数据进行精确定位。在上述模型的基础上，我们改变锚点的坐标、靶点的范围，当信号无干扰时利用最小二乘算法求出初始近似坐标点，当信号有干扰时利用残差加权算法求出初始近似坐标点，然后分别用 ELM 极限学习机和 BP 神经网络去优化得到最终较为精确的坐标点。

5.2 问题三的结果

我们用上述建立的定位模型，对附件 3 中在实验场景 2 采集的信号无干扰和信号有干扰数据分别定位，得到结果见表 5.1 和表 5.2。

表 **5.1** 附件 **3** 信号无干扰数据的定位结果 **(**单位：**mm)**

序号	测量距离				定位坐标	
					LS	**LS+ELM**
1	4220	2580	3730	1450	(3648,2204,988)	**(3684,2204,1387)**
2	4500	1940	4420	1460	(4195,1695,826)	**(4202,1706,1014)**
3	3550	2510	3410	2140	(3168,1725,934)	**(3194,1718,1032)**
4	3300	3130	2900	2790	(2586,1874,1692)	**(2573,1931,1464)**
5	720	4520	3050	5380	(522,58,1231)	**(519,412,1575)**

表 **5.2** 附件 **3** 信号有干扰数据的定位结果 **(**单位：**mm)**

序号	测量距离				定位坐标	
					RWGH	**RWGH+BP**
6	5100	2220	4970	800	(4711,2080,1095)	**(4690,2138,1624)**
7	2900	3210	3140	2890	(2503,1547,1078)	**(2505,1323,1355)**
8	2380	3530	2320	3760	(1789,1511,1702)	**(1779,1291,1386)**
9	2150	3220	3140	3640	(2117,942,1943)	**(2173,856,1263)**
10	1620	3950	2580	4440	(1212,872,1463)	**(1192,867,1316)**

同样地，由第二问知复合定位模型的定位精度较高，所以我们确定附件三中靶点的定位结果为表 5.1 和表 5.2 的最后一列。

6 问题四的分析与求解

6.1 问题四的分析

问题二的定位模型是在已知信号有、无干扰的条件下建立的，但由于 UWB 在采集数据时并不知道信号有无干扰，在判断信号有无干扰时，我们应该将所有采集的数据使用同一个定位模型，即用最小二乘算法或者残差加权误差抑制算法计算近似坐标点。同时从第二问的数据处理中可以发现，对于正常数据和异常数据来说，使用同一个定位模型计算出来的误差还是有差别的，这有利于后续分类。又由于测量距离变量之间具有高度非线性和相互强藕联的关系，而随机森林具有优秀的高维和非线性数据集处理能力。针对以上问题，本文使用随机森林算法建立分类模型，以便区分哪些数据是在信号无干扰下采集的数据，哪些数据是在信号干扰下采集的数据。

对于第四问，应该在利用最小二乘算法或者残差加权误差抑制算法计算近似坐标点的基础上，继续寻找这二类数据的差异性，提炼出具有差异性的一些特征指标，用这些指标进行随机森林训练，只要前面求近似点坐标比较准确，这二类数据一定有差异的地方。最后我们选取正确率较高的分类模型来判断附件 4

中提供的 10 组数据是来自信号无干扰或信号干扰下采集的。

6.2 问题四的求解

6.2.1 特征指标的提炼

我们直接用原始数据中每个样品的 4 个测量距离指标去建立判别模型，发现其效果并没有明显差异性，所以应该再从原始数据中提炼出一些有用的特征指标。由于对于正常数据和异常数据用同一模型算出来定位坐标的误差有较大的区别，利用最小二乘算法求解出近似点的定位坐标，其与靶点真实坐标的三维误差如图 6-1 所示；利用残差加权误差抑制算法求解出近似点的定位坐标，其与靶点真实坐标的三维误差如图 6-2 所示。

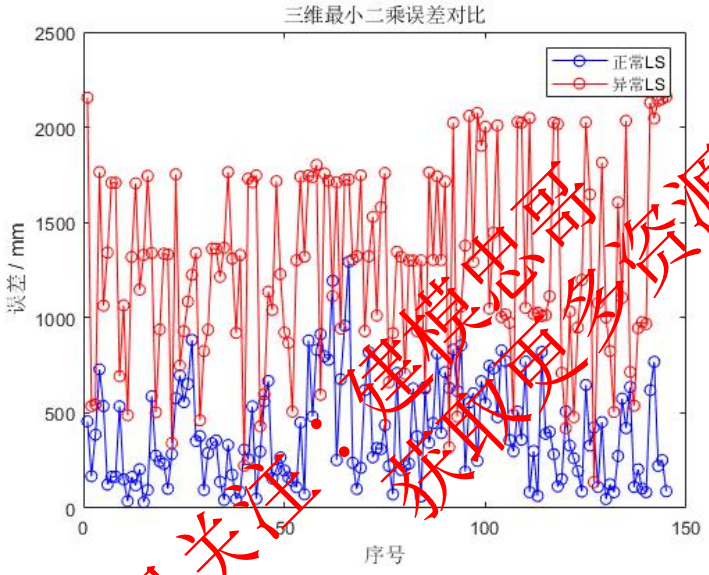

图 **6-1**　基于最小二乘算法定位坐标的三维误差

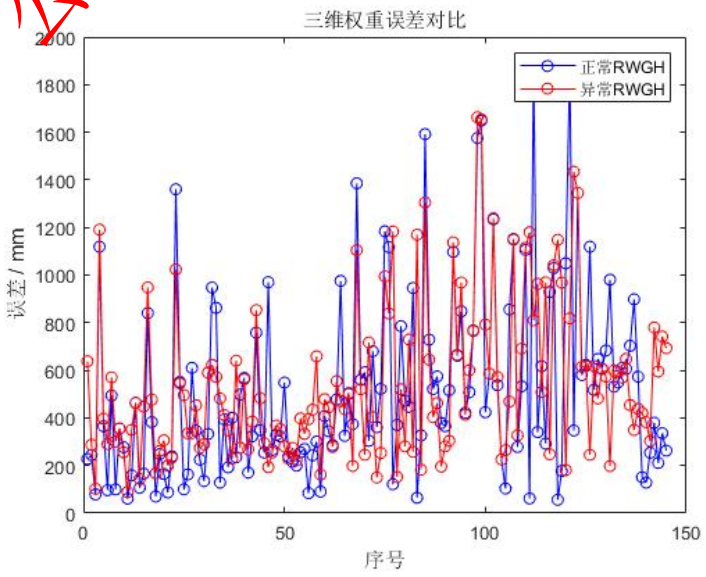

图 **6-2**　基于残差加权误差抑制算法定位坐标的三维误差

由图 6-1 可知，基于最小二乘算法得到所有数据的定位坐标，其中正常数据和异常数据的三维误差有较为明显的区别，所以近似点得定位坐标到锚点的欧式距离肯定也会有较大的区别。由图 6-2 可知，基于残差加权误差抑制算法得到所有数据的定位坐标，其中正常数据和异常数据的三维误差差别较小，所以猜测基于最小二乘算法的随机森林分类模型的正确率比较高。

基于以上的思想，我们进行特征的提取，用这些提取出来的特征代入随机森林分类模型中进行训练。提取特征过程为: 首先，对于正常数据和异常数据，用最小二乘算法 (或残差加权误差抑制算法) 求出相应的近似点坐标；然后求出初始近似点坐标与 4 个锚点的距离 d_0', d_1', d_2', d_3'；最后，求 4 个距离误差以及 4 个距离误差和：

$$e_0 = |d_0' - d_0|, e_1 = |d_1' - d_1|, e_2 = |d_2' - d_2|, e_3 = |d_3' - d_3|, e = \sum |d_i' - d_i|.$$

综上，我们提炼出的特征指标有 $d_0', d_1', d_2', d_3', e_0, e_1, e_2, e_3, e$。

6.2.2 基于随机森林算法的分类模型

随机森林 (Random Forest) 是一种集成学习方法，常用于分类、回归和其他机器学习任务[7]，其原理是在训练时构建大量决策树，而每一颗决策树之间是没有关联的，当有一个新的样本进入算法时，每一颗决策树都会分别进行一下判断，并各自识别这个样本应该属于哪一类别，然后根据某一类别被选择最多，就预测这个样本为哪一类别。随机森林是一种自举集成算法，基于 bagging 决策树实现了随机森林的随机分割选取，有效地纠正了决策树拟合的问题。

随机森林通过自助采样法 (Bootstrap Sampling) 随机有放回地抽取 N 组样本生成训练样本集，并建立 N 棵 CART 决策树。我们将样本的 4/5 作为袋内数据，1/5 作为袋外数据，袋外数据可以通过内部交叉验证并应用到所有决策树，估算整个随机森林的泛化误差[8]。每颗决策树的每个节点处随机抽取 m 个特征进行节点分裂。CART 决策树的节点划分是通过计算 Gini 指数来实现的，节点的 Gini 指数越小，表示集合中被选中样本被错分的概率越小，特征划分的效果越理想。Gini 指数计算公式如下：

$$Gini(P) = \sum_{k=1}^{K} P_k(1 - P_k) = 1 - \sum_{k=1}^{K} P_k^2,$$

式中，P_k 表示选中的样本属于 K 类别的概率，则这个样本被错分的概率是 $1 - P_k$，样本集合中有 K 个类别。最后由生成的多颗决策树构成随机森林分类器对数据进行分类，最终的分类结果由所有决策树投票来决定。

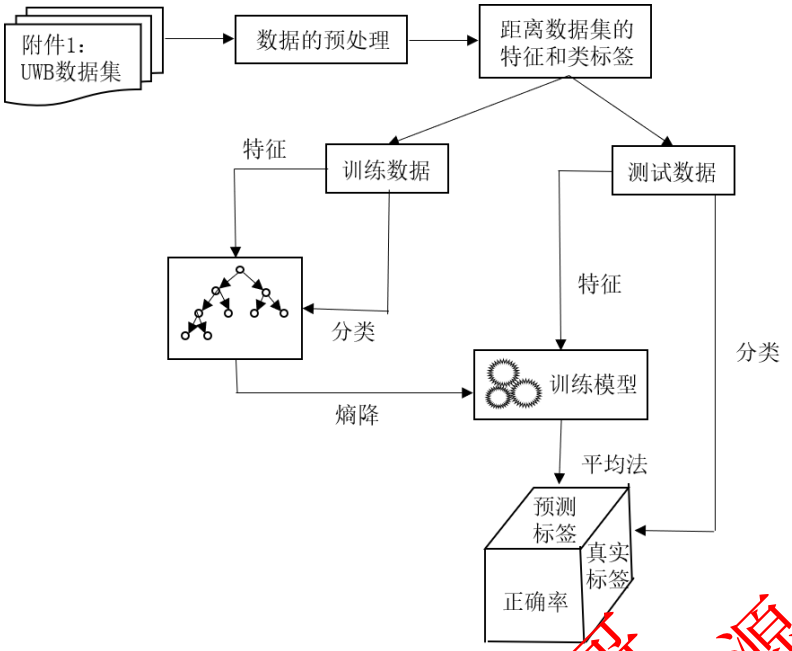

图 **6-3**　随机森林算法流程图

融合提取的特征信息，对随机森林算法进行参数调优，发现当决策树的数量 (Ntree) 和最小分割点分别为 155 和 2 时，分类精度最高。将获得的最优参数运用于分类模型进行分类，得到信号无干扰或信号干扰下的分类结果。

6.2.3 分类精度评价指标

1. **方法的选取**

当指标为 $d_0', d_1', d_2', d_3', e_0, e_1, e_2, e_3, e$ 时，我们采用混淆矩阵的方法对分类结果进行精度评价，计算分类模型评价指标为总体分类精度 (Overall Accuracy) 和 Kappa 系数 (Kappa Coefficient)。总体分类精度等于被正确分类的数除以总数，详细结果如表 6.1 和表 6.2 所示。

表 **6.1**　基于最小二乘算法的随机森林分类的混淆矩阵

数据类型	信号无干扰	信号有干扰	**正确率/%**
信号无干扰	133	44	**93.0**
信号有干扰	10	135	**75.4**

表 **6.2**　基于残差加权误差抑制算法的随机森林分类的混淆矩阵

数据类型	信号无干扰	信号有干扰	**正确率/%**
信号无干扰	126	51	**82.4**
信号有干扰	27	118	**69.8**

因此从表 6.1 和表 6.2 可以看出，用随机森林建立分类模型，当指标为

d_0'、d_1'、d_2'、d_3'、e_0、e_1、e_2、e_3、e 时，基于最小二乘算法建立随机森林分类模型，该模型的测试集正确率为 83.2%，基于残差加权误差抑制算法建立随机森林分类模型，该模型的测试集正确率为 75.8%，这也验证 6.2.1 节的猜想：基于最小二乘算法的随机森林分类模型的正确率比较高。

Kappa 系数是一种比例，代表着分类与完全随机分类产生错误减少的比例，其计算公式为

$$k = \frac{p_0 - p_e}{1 - p_e},$$

其中，p_0 是每一类正确分类的样本数量之和除以总样本数，也就是分类精度。假设每一类的真实样本个数分别为 $a_1, a_2, \cdots, a_c$, 而预测出来的每一类的样本个数分别为 $b_1, b_2, \cdots, b_c$, 总样本个数为 n, 则有

$$p_e = \frac{a_1 \times b_1 + a_2 \times b_2 + \cdots + a_c \times b_c}{n \times n}.$$

由上述公式求得基于最小二乘算法的随机森林分类模型的 kappa 系数为 0.7044，基于残差加权误差抑制算法的随机森林分类模型的 kappa 系数为 0.5180，所以基于最小二乘算法的随机森林分类模型优于基于残差加权误差抑制算法的随机森林分类模型。

基于以上分析和求解结果，我们最后选用基于最小二乘算法的随机森林分类模型对附件 4 进行分类。

2. **指标的选取**

将提取出来的九个指标代入基于最小二乘法的随机森林模型中进行训练，得出各个指标的重要性。

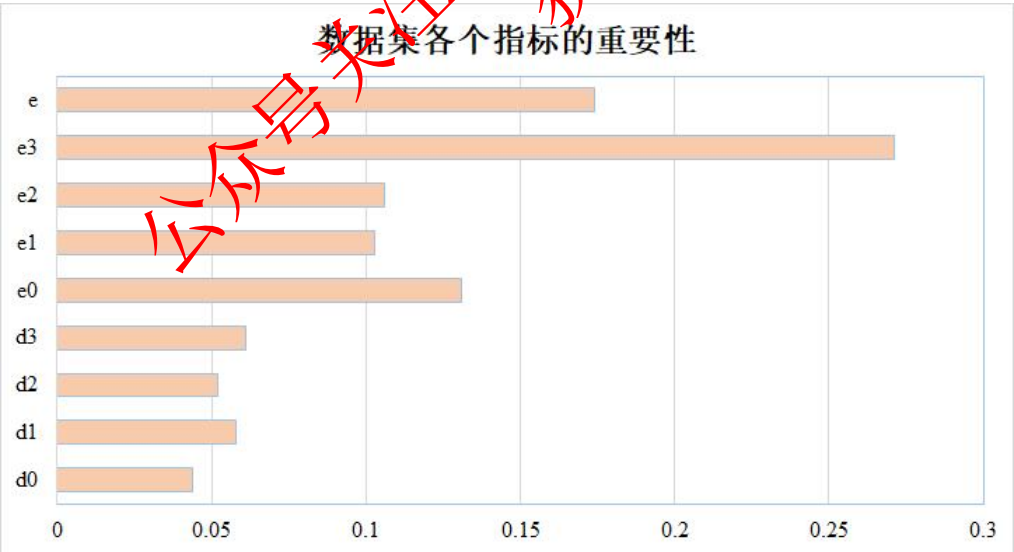

图 **6-4**　各指标的重要性

由图 6-4 可以发现 e_0, e_1, e_2, e_3, e 这五个指标比较重要，所以我们考虑用这五个指标去建立基于最小二乘法的随机森林模型。

表 **6.3**　九个指标和五个指标分类正确率的对比表

指标	分类正确率/%
$d_0', d_1', d_2', d_3', e_0, e_1, e_2, e_3, e$	83.2
e_0, e_1, e_2, e_3, e	85.1

通过表 6.3 发现虽然选取五个指标比选取九个指标建立出来的分类模型的分类准确率只高了一点点，但是从模型的角度来看，模型相对来说会简单一点，而且避免了过拟合的情况，所以最终我们决定选取 e_0, e_1, e_2, e_3, e 这五个指标来建立分类模型。

根据以上的分析，基于最小二乘算法并且选取 e_0, e_1, e_2, e_3, e 这五个指标进行随机森林分类，此时的分类模型正确率更高。

3. **数据集的分类结果**

利用五个指标建立基于最小二乘法的随机森林模型进行分类，表 6.4 是训练集和测试集的分类结果准确率。

表 **6.4**　模型的训练集和测试集分类正确率的对比表

数据集	分类正确率/%
训练集	100
测试集	85.1

为了更直观呈现出分类结果，我们绘制如下的分类结果散点图如图 6-5。

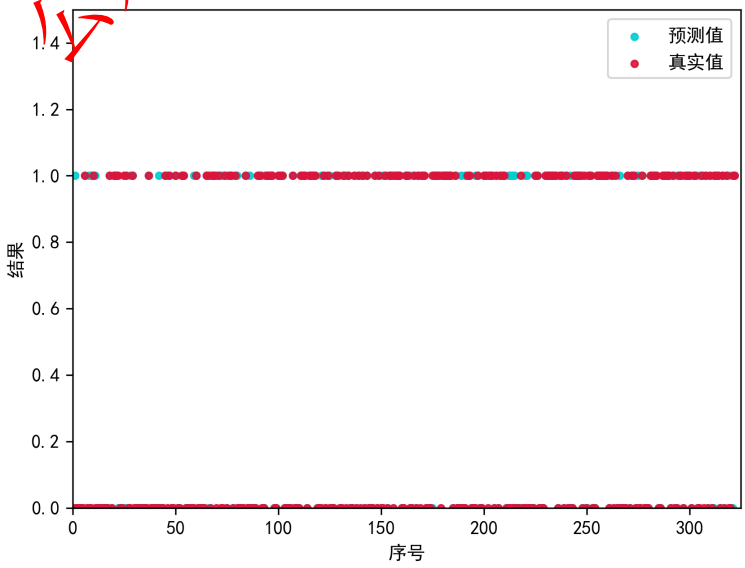

图 **6-5**　分类结果散点图

6.2.4 分类算法

算法 4: 基于 LS(或 RWGH) 的随机森林分类算法

输入: $d_0, d_1, d_2, d_3, A0, A1, A2, A3$
输出: 输出分类结果 0 或 1
步骤 1： 对于所有数据，用最小二乘算法计算出相应的近似点坐标。
步骤 2： 计算近似点坐标与 4 个锚点的欧式距离 $d_0^{'}, d_1^{'}, d_2^{'}, d_3^{'}$ 及其与相应的测量距离之间的绝对误差 e_0, e_1, e_2, e_3, 绝对误差之和 e 这些特征。
步骤 3： 将提取的特征 e_0, e_1, e_2, e_3, e 代入随机森林分类模型进行训练，调整参数，当测试集分类的正确率达到最优时，停止训练，输出分类结果。

6.3 问题四的结果

对比上述两个模型，我们最终利用五个指标 e_0, e_1, e_2, e_3, e 建立基于最小二乘算法的随机森林分类模型，对附件 4 中提供的 10 组数据进行分类，得到的第 1、3、6、9 和 10 组数据是来自信号干扰下采集的，第 2、4、5、7 和 8 组数据是来自信号干扰下采集的，具体情况可见表6.5，其中分类结果 1 表示来自信号无干扰下采集的数据、0 表示来自信号有干扰下采集的数据。

表 **6.5**　附件 **4** 的分类结果

序号	测量距离				分类结果
1	2940	4290	2840	4190	**1**
2	5240	5360	2040	2940	**0**
3	4800	2610	4750	2550	**1**
4	5010	4120	3810	2020	**0**
5	2840	4490	2860	4190	**0**
6	5010	5320	1990	2930	**1**
7	5050	3740	3710	2070	**0**
8	5050	4110	3710	2110	**0**
9	4840	2600	4960	2700	**1**
10	2740	2720	4670	4790	**1**

7 问题五的分析与求解

7.1 问题五的分析

针对问题五，其中附件 5 是一段时间内连续采集的多组数据，由于在采集这些数据时，会随机出现信号干扰，所以我们首先需要对数据进行处理，应用第四问中基于最小二乘算法的随机森林分类模型，对数据类型进行判别，区分信号无干扰下采集的数据和信号干扰下采集的数据。接着我们应用第二问中建立的定位模型，对靶点在不同时间点以及信号有无干扰的状态下的位置分别进行定位。最后得出所有静态点的定位坐标后，考虑靶点自身运动规律以及靶点的原始轨

迹由于测量误差可能存在运动抖动的情况，对数据进行平滑处理以形成动态靶点的运动轨迹。

7.2 问题五的求解

7.2.1 靶点轨迹定位算法

我们首先判别数据是否在信号有干扰下采集的，然后应用相对应的模型对其进行定位，具体处理方式如下：

步骤 1：判断数据类型

(1) 用最小二乘算法求出附件 5 数据的初始近似点坐标；
(2) 求初始近似点坐标与 4 个基站的距离 d_0', d_1', d_2', d_3'；
(3) 求 4 个距离误差以及 4 个距离误差和：$e_0 = |d_0' - d_0|, e_1 = |d_1' - d_1|, e_2 = |d_2' - d_2|, e_3 = |d_3' - d_3|, e = \sum |d_i' - d_i|$；
(4) 用随机森林建立分类模型，指标为 $d_0', d_1', d_2', d_3', e_0, e_2, e_2, e_3, e$；
(5) 输出分类结果。

步骤 2：确定位置坐标点

判断完数据类型之后，基于最小二乘算法的 ELM 极限学习机对信号无干扰的数据进行精确定位，基于残差加权算法的 BP 神经网络对信号有干扰的数据进行精确定位，由此得到靶点运动轨迹中每个时间点的定位坐标。

步骤 3：靶点运动轨迹平滑处理

考虑到靶点自身运动规律，在相邻间隔很短的两个时间内靶点的位置差异不会太大，我们从轨迹位置一般不会发生突变对坐标点进行修正。利用靶点静态点定位模型求得靶点在不同时刻的位置坐标后，分别找出 x, y, z 坐标对应的所有跳跃较大的间断点，运用两个相邻时刻点的均值替换该点的新位置坐标，随后利用移动平均法对其轨迹进行光滑化处理[8]。简单的移动平均法公式如下：

$$X(t) = (X(t-1) + X(t-2) + X(t-3) + \cdots + X(t-n))/n,$$

式中，$X(t)$ 表示对下一时刻的预测值，n 表示移动平均的时刻个数，$X(t-1)$ 表示前一时刻的实际值，$X(t-2), X(t-3), \cdots, X(t-n)$ 表示前两期、前三期直至前 n 期的实际值。

7.3 问题五的结果

根据上述的靶点轨迹定位算法，首先判断数据的类型是有干扰还是无干扰，分别记为分类结果 0 或分类结果 1，判断结果见表 7.1 的最后一列，其次根据数据类型使用不同的定位模型求出靶点的定位坐标，定位结果如表 7.1。

表 **7.1**　附件 **5** 靶点的定位坐标 **(**单位：**mm)**

序号	测量距离				定位坐标			分类结果
1	810	4650	4570	6520	432	429	1628	1
2	810	4650	4579	6510	434	432	1594	1
3	810	4640	4580	6510	442	422	1595	1
…		…	…			…		…
315	3600	3230	3800	3960	2700	2288	1495	0
316	3600	3210	3810	3990	2531	2095	1369	0
317	3590	3190	3830	4000	2678	2138	1494	0
318	3580	3170	3860	3990	2691	2169	1494	0
…		…	…			…		…
538	6390	4480	4500	840	4648	4477	1511	1
539	6390	4510	4500	830	4608	4497	1516	1

根据靶点轨迹定位算法求出每个时刻靶点的定位坐标，分别绘制出 x, y, z 的初始运动轨迹，绘制结果如图 7-1，其中平滑前的 x、平滑前的 y、平滑前的 z 分别表示未进行平滑处理前的 x, y, z 的初始运动轨迹。

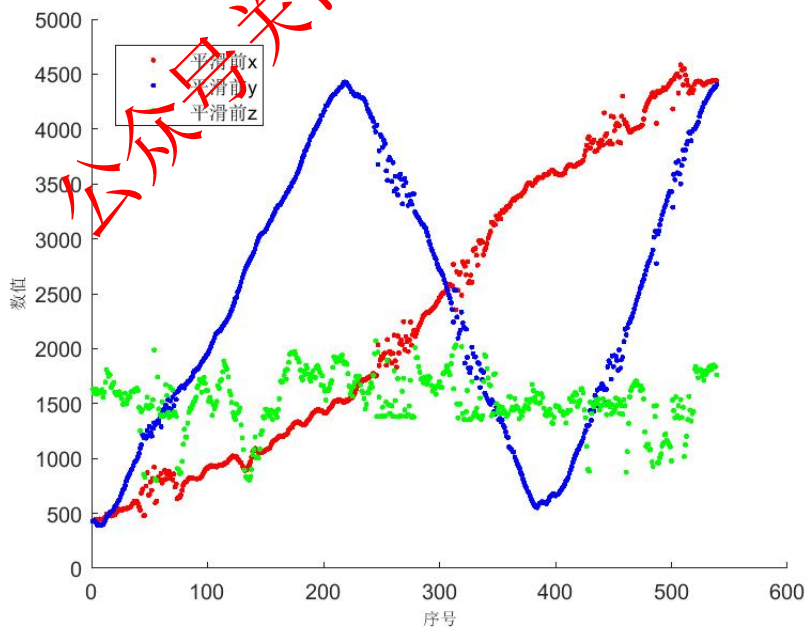

图 **7-1**　平滑化处理前 x, y, z 一维的运动轨迹

对 x, y, z 初始运动轨迹进行平滑处理后，绘制图 7-2，其中平滑后 x、平滑后 y、平滑后 z 表示在初始运动轨迹基础上对 x, y, z 平滑处理后的运动轨迹。

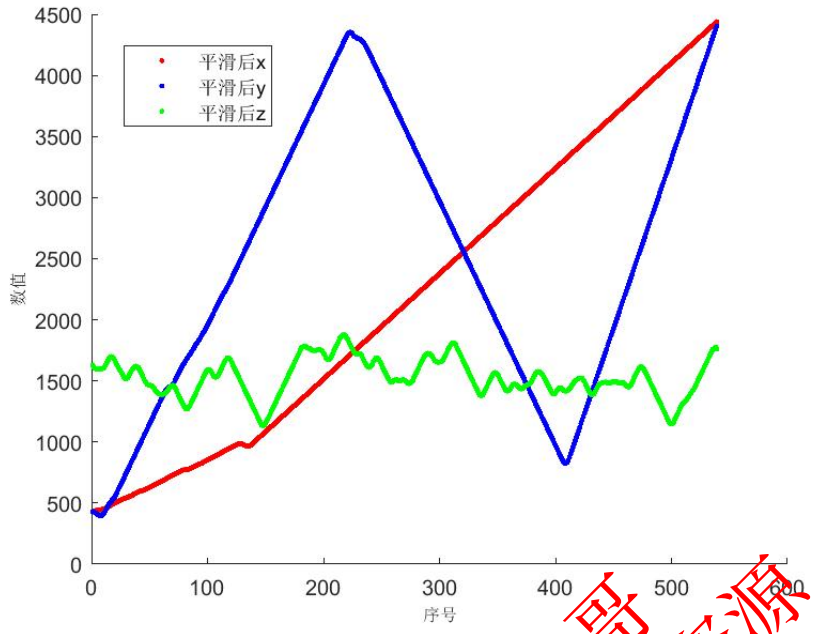

图 **7-2**　平滑化处理后 x, y, z 一维的运动轨迹

由图 7-2 一维坐标的运动趋势可知，x 轴坐标随着时间的增加而增大，y 轴坐标随着时间的增加呈现出“N”字形状，而 z 轴坐标随着时间的增加在 1500mm 附近波动。

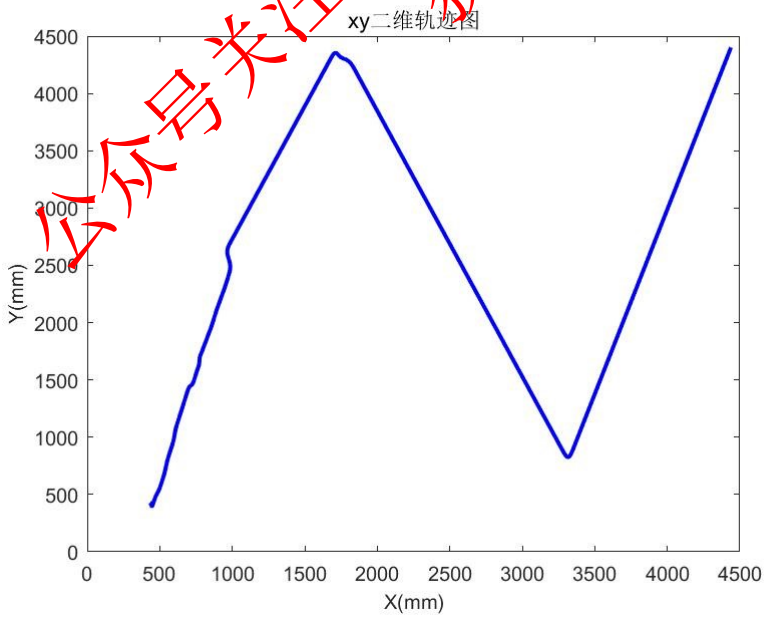

图 **7-3**　(x, y) 二维运动轨迹

图 7-3 是 (x, y) 的二维平面运动轨迹图，由二维图可知，该运动轨迹在 xoy 平面上的投影呈现出“N”字形状。

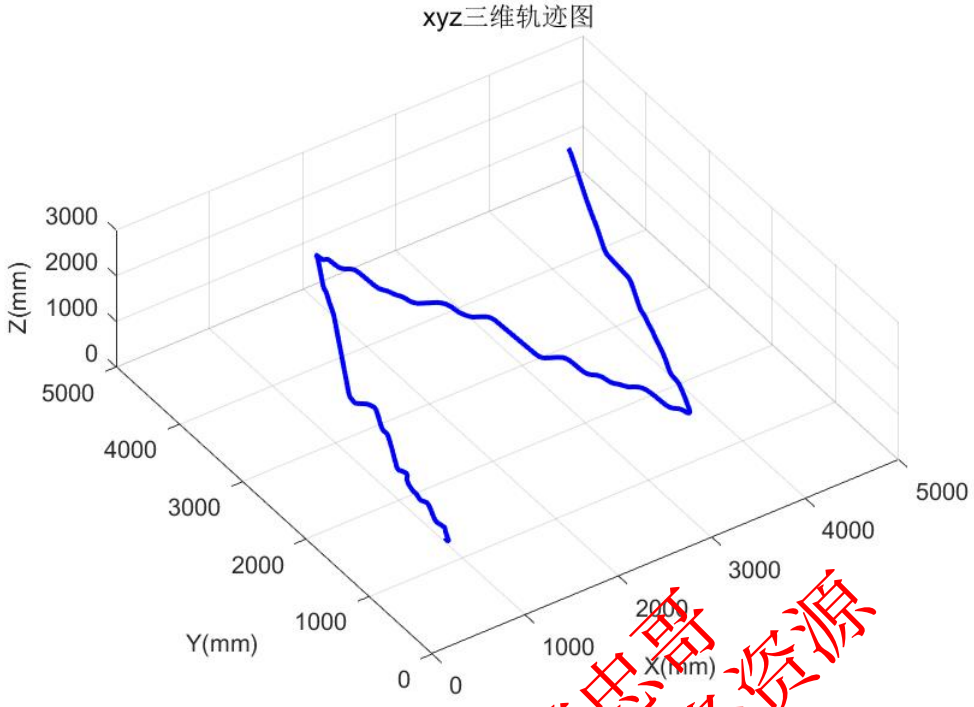

图 **7-4** (x, y, z)三维运动轨迹

图 7-4 是靶点在场景 1 中的三维运动轨迹图，观察图 7-4，靶点在空间中的运动轨迹是在约为 1500mm 的高度上走了一个“N”字形。

8 模型的评价

8.1 模型的优点

本文采用了较为创新的方法，首先我们系统聚类分析结合了距离阈值和类数，在数据预处理中达到了理想的效果。

其次，本文在考虑问题时使用了多个模型，比如针对问题二使用了基于最小二乘算法的 ELM 极限学习机模型和基于残差加权误差抑制算法的 BP 神经网络模型，从而实现了的三维坐标定位，并从高度入手有技巧的选取了训练集和测试集，选取了效果较为好的训练集比例，发现复合模型能够较好的修正 z，在一定程度上降低误差。

最后，我们使用了锚点坐标、测量距离和初始近似点坐标提取出多个特征指标，再使用随机森林算法对数据是否在有干扰信号下采集的进行分类，得到的检验效果较好，想法创新，是本文的一大创新点。

8.2 模型的缺点

第一，对题目所给的数据异常值处理不够，由于使用基于 3σ 准则检测异常值前要验证总体样本是否服从或者近似服从正态分布，我们验证过绝大部分文件样本是满足条件的，但同时可能也有极少数不满足，在聚类之前样本中可能仍然存在着个别的异常值未处理干净情况。

第二，ELM 极限学习机、BP 神经网络模型随机性较大，我们通过不断调节参数训练模型，使得测试集的精度误差达到最小，再对附件中的数据进行定位坐标预测。

第三，定位模型的 x, y 精度还不错，但 z 相对来说误差比较大，在一定程度上会影响后面分类模型以及运动轨迹模型的效果。

8.3 模型的改进

后期可以考虑对数据清洗时使用箱线图剔除异常值，然后再对数据进行后续的处理。另外，由于残差加权误差抑制算法在求解近似点坐标时，只起到了抑制误差的作用，并不能很好的消除信号干扰所带来的误差，所以后续我们需要对该模型进行改进，可以考虑结合几何方法找出信号被干扰的测量距离数据后，从而实现更精准地抑制干扰误差的效果。

9 参考文献

[1] 司守奎，孙兆亮，数学建模算法与应用，国防工业出版社，216-223，2015.

[2] 邓锴，基于 NLOS 识别和误差消除的无线定位算法研究 [D]，西南交通大学，2018.

[3] I. Guvenc, S. Gezici, F. Watanabe and H. Inamura, Enhancements to Linear Least Squares Localization Through Reference Selection and ML Estimation, 2008 IEEE Wireless Communications and Networking Conference, pp. 284-289, 2008.

[4] L. Jiao, J. Xing, X. Zhang, J. Zhang and C. Zhao, LCC-Rwgh: A NLOS Error Mitigation Algorithm for Localization in Wireless Sensor Network, 2007 IEEE International Conference on Control and Automation, pp. 1354-1359, 2007.

[5] Li, Bing, Wei Cui, and Bin Wang. A Robust Wireless Sensor Network Localization Algorithm in Mixed LOS/NLOS Scenario, Sensors 15, no. 9: 23536-23553, 2015.

[6] Bo Yan, Yao Cui, Lin Zhang, Chao Zhang, Yongzhi Yang, Zhenming Bao, Guobao Ning, Beam Structure Damage Identification Based on BP Neural Network and Support Vector Machine, Mathematical Problems in Engineering, vol. 2014, Article ID 850141, 8 pages, 2014.

[7] Huan, Ruohong & Chen, Qingzhang & Mao, Keji & Pan, Yun. A three-dimension localization algorithm for wireless sensor network nodes based on SVM, 2010.

[8] 于军琪，虎群，赵安军，高之坤，张娜，基于随机森林的大型公共建筑能耗混合预测模型 [J/OL]，控制工程：1-9，2021.

[9] 王燕，时间序列分析——基于 R，中国人民大学出版社，117-124，2015.

[10] Huang G B, Zhu Q Y, Siew C K. Extreme Learning Machine:Theory and Applications [J]. Neurocomputing, 70(1-3): 489-501, 2006.

[11] Ling Ding, Prasad Rangaraju, Amir Poursaee, Application of generalized regression neural network method for corrosion modeling of steel embedded in soil, Soils and Foundations, Volume 59, Issue 2, Pages 474-483, 2019.

[12] D. Karaboga and B. Basturk, On the performance of artificial bee colony (ABC) algorithm, Applied Soft Computing, vol. 8, no. 1, pp. 687–697, 2008.

[13] M. Koudil, K. Benatchba, A. Tarabet, and E. B. Sahraoui, Using artificial bees to solve partitioning and scheduling problems in codesign, Applied Mathematics and Computation, vol. 186, no. 2, pp. 1710–1722, 2007.

附录 A 程序代码

附录 1：系统聚类法

```
import math
import pandas as pd
import os

def hierarchical_cluster(data, t):
result = [[aData] for aData in data]
step2(result, t)
return result

def step2(result, t):
min_dis = min_distance(result[0], result[1])

index1 = 0
index2 =

for i in range(len(result)):
for j in range(i + 1, len(result)):
dis_temp = min_distance(result[i], result[j])
if dis_temp < min_dis:
min_dis = dis_temp
index1 = i
index2 = j

if min_dis <= t:

result[index1].extend(result[index2])
result.pop(index2)
step2(result, t

def min_distance(list1, list2):

min_dis = get_distance(list1[0], list2[0])
for i in range(len(list1)):
for j in range(len(list2)):
dis_temp = get_distance(list1[i], list2[j])
if dis_temp < min_dis:
min_dis = dis_temp
return min_dis

def get_distance(data1, data2):
distance = 0
for i in range(len(data1)):
distance += pow((data1[i] - data2[i]), 2)
return math.sqrt(distance)

t = 15

file_dir = 'C:/Users/Admin/Desktop/dataSet/normalC';
li = []
all_file_list = os.listdir(file_dir)
```

```
for single_file in all_file_list:
data=[]
single_data_frame = pd.read_excel(os.path.join(file_dir,
   single_file), encoding='utf-8')
data = single_data_frame.iloc[:, 0:4]
data = data.values.tolist();
try:
result = hierarchical_cluster(data, t)
opt = len(result)
if opt > 3:
opt = 3
li.append(opt)
except IndexError:
opt = 1
li.append(opt)
wdf = pd.DataFrame(li)
wdf.to_excel('C:/Users/Admin/Desktop/dataSet/norma15.xlsx',
    index=False,header=None)

print("OK")
```

附录 2：最小二乘算法

```
clc;
clear all;

data = xlsread('C:\Users\Admin\Desktop\dataSet...
\abnormalC1.异常权重测试.xlsx');
[m ,n]=size(data);
xwrite =[];
for k=1:m
R0=data(k,1);R1=data(k,2);R2=data(k,3);R3=data(k,4);
h1=R1^2-R0^2;h2=R2^2-R0^2;h3=R3^2-R0^2;
k0=1300^2;k1=5000^2+1700^2;k2=5000^2+1700^2;
k3=5000^2+5000^2+1300^2;
G=[-5000 0 -400
0 -5000 -400
-5000 -5000 0];
H=1/2*[h1-k1+k0
h2-k2+k0
h3-k3+k0];
A=[];
b=[];
lb=[0;0;0];
ub=[5000;5000;3000];
[x,resnorm,residual,exitflag,output]=lsqlin(G,H,A,b,[],[],lb,ub)
xx=[data(k,1:7),x'];
xwrite = [xwrite;xx];
end
```

附录 3：残差加权算法

```
clc;
clear;
```

```
A0= [0 0 1300]';
A1=[5000 0 1700]';
A2=[0 5000 1700 ]';
A3=[5000 5000 1300]';
data = xlsread('C:\Users\Admin\Desktop\dataSet\csv...
\normalC1.正常最小二乘测试.xlsx');

[m ,n]=size(data);
xwrite =[];
for k=1:m
R0=data(k,1);R1=data(k,2);R2=data(k,3);R3=data(k,4);

h1=R1^2-R0^2;h2=R2^2-R0^2;
k0=1300^2;k1=5000^2+1700^2;k2=5000^2+1700^2;
G=[-5000 0 -400
0 -5000 -400];
H=1/2*[h1-k1+k0
h2-k2+k0];
A=[];
b=[];
lb=[0;0;0];
ub=[5000;5000;3000];
[x1,resnorm,residual,exitflag,output]=lsqlin...
(G,H,A,b,[],[],lb,ub);
RS1=abs(R0+R1+R2-norm(x1-A0,2)-norm(x1-A1,2)-norm(x1-A2));
RES1=RS1/3

h1=R1^2-R0^2;h3=R3^2-R0^2;
k0=1300^2;k1=5000^2+1700^2;k3=5000^2+5000^2+1300^2;
G=[-5000 0 -400
-5000 -5000 0];
H=1/2*[h1-k1+k0
h3-k3+k0];
A=[];
b=[];
lb=[0;0;0];
ub=[5000;5000;3000];
[x2,resnorm,residual,exitflag,output]=lsqlin...
(G,H,A,b,[],[],lb,ub);
RS2=abs(R0+R1+R3-norm(x2-A0,2)-norm(x2-A1,2)-norm(x2-A3,2));
RES2=RS2/3

h2=R2^2-R0^2;h3=R3^2-R0^2;
k0=1300^2;k2=5000^2+1700^2;k3=5000^2+5000^2+1300^2;
G=[0 -5000 -400
-5000 -5000 0];
H=1/2*[ h2-k2+k0
h3-k3+k0];
A=[];
b=[];
lb=[0;0;0];
ub=[5000;5000;3000];
[x3,resnorm,residual,exitflag,output]=lsqlin...
(G,H,A,b,[],[],lb,ub);
```

```
RS3=abs(R0+R3+R2-norm(x3-A0,2)-norm(x3-A2,2)-norm(x3-A3));
RES3=RS3/3

h1=R2^2-R1^2;h2=R3^2-R1^2
k1=5000^2+1700^2;k2=5000^2+1700^2;k3=5000^2+5000^2+1300^2;
G=[5000 -5000 0
0 -5000 0];
H=1/2*[h1-k2+k1
h2-k3+k1];
A=[];
b=[];
lb=[0;0;0];
ub=[5000;5000;3000];
[x4,resnorm,residual,exitflag,output]=lsqlin(G,H,A,b,[],[],...
lb,ub);
RS4=abs(R3+R1+R2-norm(x4-A3,2)-norm(x4-A1,2)-norm(x4-A2));
RES4=RS4/3
w1=(1/RES1)/(1/RES1+1/RES2+1/RES3+1/RES4);
w2=(1/RES2)/(1/RES1+1/RES2+1/RES3+1/RES4);
w3=(1/RES3)/(1/RES1+1/RES2+1/RES3+1/RES4);
w4=(1/RES4)/(1/RES1+1/RES2+1/RES3+1/RES4);

X=x1.*w1+x2.*w2+x3.*w3+x4.*w4;
xx=[data(k,1:7),X'];

xwrite = [xwrite;xx];
end

xlswrite('C:\Users\Admin\Desktop\dataSet\csv...
\normalC1.正常权重测试.xlsx',xwrite);
```

附录 4：极限学习机主函数

```
clear all
clc

datax =
   xlsread('C:\Users\86183\Desktop\data\normalC1.训练.xlsx');
datac =
   xlsread('C:\Users\86183\Desktop\data\normalC1.测试.xlsx');

P_train = datax(:,8:10)';
T_train = datax(:,5:7)'.*10;

P_test = datac(:,8:10)';
T_test = datac(:,5:7)'.*10;

N = size(P_test,2);

[Pn_train,inputps] = mapminmax(P_train);

Pn_test = mapminmax('apply',P_test,inputps);

[Tn_train,outputps] = mapminmax(T_train);
```

```
Tn_test = mapminmax('apply',T_test,outputps);

[IW,B,LW,TF,TYPE] = elmtrain(Pn_train,Tn_train,30,'sig',0);

tn_sim = elmpredict(Pn_test,IW,B,LW,TF,TYPE);
tn_sim1 = elmpredict(Pn_train,IW,B,LW,TF,TYPE);

T_sim = mapminmax('reverse',tn_sim,outputps);

T_sim1 = mapminmax('reverse',tn_sim1,outputps);

result = [T_test' T_sim'];

E = mse(T_sim - T_test);

error=abs(T_sim - T_test);
disp('测试集:X, Y, Z,XY,XYZ误差')
disp(sum(error(1,:))/N)
disp(sum(error(2,:))/N)
disp(sum(error(3,:))/N)

N = length(T_test);

error = error';
msexy1 = [];
msexyz1 = [];
for k=1:N
msexy1(k) = sqrt(error(k,1)^2+error(k,2)^2);
msexyz1(k) = sqrt(error(k,1)^2+error(k,2)^2+error(k,3)^2);
end
avmsexy1 = sum(msexy1)/(N)
avmsexyz1 = sum(msexyz1)/(N)

N = length(T_train);
error1=abs(T_sim1 - T_train);
disp('训练集:X, Y, Z,XY,XYZ误差')
disp(sum(error1(1,:))/N)
disp(sum(error1(2,:))/N)
disp(sum(error1(3,:))/N)

N = length(T_train);

error1 = error1';
msexy = [];
msexyz = [];
for k=1:N
msexy(k) = sqrt(error1(k,1)^2+error1(k,2)^2);
msexyz(k) = sqrt(error1(k,1)^2+error1(k,2)^2+error1(k,3)^2);
end

avmsexy = sum(msexy)/(N)
avmsexyz = sum(msexyz)/(N)
```

附录 5：BP 神经网络

```
clear all
clc

datax = xlsread('C:\Users\Admin\Desktop\dataSet...
\csv\abnormalC1.训练.xlsx');
datac = xlsread('C:\Users\Admin\Desktop\dataSet...
\csv\abnormalC1.测试.xlsx');
[row ,col]=size(datac);

input_train = datax(:,8:10)';
output_train =datax(:,5:7)'.*10;
input_test = datac(:,8:10)';
output_test =datac(:,5:7)'.*10;

inputnum=3;
hiddennum=5;
outputnum=3;

[inputn,inputps]=mapminmax(input_train,0,1);
[outputn,outputps]=mapminmax(output_train,0,1);

net=newff(minmax(inputn),outputn,hiddennum,...
{'tansig','purelin'},'trainlm');

net.trainParam.epochs=1000000;
net.trainParam.lr=0.01;
net.trainParam.goal=0.000001;
net=train(net,inputn,outputn);

inputn_test=mapminmax('apply',input_test,inputps);

an=sim(net,inputn_test);
test_simu=mapminmax('reverse',an,outputps);
error=abs(test_simu-output_test);

figure(1)
plot(output_test,'bo-')
hold on
plot(test_simu,'r*-')
hold on
plot(error,'square','MarkerFaceColor','b')
legend('期望值','预测值','误差')
xlabel('数据组数')
ylabel('值')
[c,l]=size(output_test);
MAE1=sum(abs(error))/l;
MSE1=error*error'/l;
RMSE1=MSE1^(1/2);
disp(sum(error(1,:))/row)
disp(sum(error(2,:))/row)
disp(sum(error(3,:))/row)
```

附录 6：随机森林

```
import pandas as pd
import numpy as np
from sklearn import metrics
from sklearn.ensemble import RandomForestRegressor
from sklearn.feature_selection import SelectFromModel,
   VarianceThreshold
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split

data1 =
   pd.read_excel(r'C:\Users\Admin\Desktop\全部数据训练.xlsx',...
encoding = 'gbk')
data2 =
   pd.read_excel(r'C:\Users\Admin\Desktop\全部数据测试.xlsx',...
encoding = 'gbk')
data1.columns=['dd0','dd1','dd2','dd3','c','d0','d1','d2',...
'd3','e0','e2','e3','e4','e','e4','e5','e6','e7']
data2.columns=['dd0','dd1','dd2','dd3','c','d0','d1','d2',...
'd3','e0','e2','e3','e4','e','e4','e5','e6','e7']
x_train = data1[['e0','e2','e3','e4','e']]
y_train= data1['c']
x_test = data2[['e0','e2','e3','e4','e']]
y_test= data2['c']

print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)

rf1 =RandomForestClassifier(n_estimators=155,max_depth=None,...
min_samples_split=2,random_state=0);

rf1.fit(x_train,y_train)
predicted= rf1.predict(x_test)

print('yzhi : \n',predicted)

accuracy = rf1.score(x_test, y_test)
print('决策树准确率: \n', accuracy)

la=sum(abs(predicted-y_test))/322
print('失误率: \n',la)

print(y_test.shape)

msel = metrics.mean_squared_error(y_test,predicted)
print("MSE1:%.4f" % msel)

importances = rf1.feature_importances_
importances =100.0*(importances/importances. max())
print("重要性: ", importances)
```

附录 7：靶点运动修正跳跃点

```
clc
data = xlsread('C:\Users\lenovo\Desktop\05\近似坐标bp.xlsx');
row =size(data);
sd=[];
%处理x
dax = data(:,10);
while max(diff(dax))>20
for i=2:row-1
if abs(dax(i)-dax(i-1))>20 || abs(dax(i)-dax(i-1))>20
dax(i) =(dax(i+1)+dax(i-1))/2;
end
end
end
%处理y
day = data(:,11);
while max(diff(day))>20
for i=2:row-1
if abs(day(i)-day(i-1))>20 || abs(day(i)-day(i-1))>20
day(i) =(day(i+1)+day(i-1))/2;
end
end
end
%处理z
daz = data(:,12);
while max(diff(daz))>20
for i=2:row-1
if abs(daz(i)-daz(i-1))>20 || abs(daz(i)-daz(i-1))>20
daz(i) =(daz(i+1)+daz(i-1))/2;
end
end
end
```

附录 6：轨迹平滑化以及画轨迹图

```
clc;
clear;
data =xlsread('C:\Users\lenovo\Desktop\05\画图数据.xlsx');
row =size(data,1);

x = data(:,1);
xx = smooth(data(:,4));
y = data(:,2);
yy = smooth(data(:,5));
z = data(:,3);
zz = smooth(data(:,6));

figure(1)
scatter(1:row,x,'r*')
hold on
scatter(1:row,y,'b.')
hold on
scatter(1:row,z,'gx')
```

```
legend('平滑前x','平滑前y','平滑前z')
xlabel('序列','FontSize',8);
ylabel('数值','FontSize',8);

figure(2)
scatter(1:row,xx,'r*')
hold on
scatter(1:row,yy,'b.')
hold on
scatter(1:row,zz,'gx')
legend('平滑后x','平滑后y','平滑后z')
xlabel('序列','FontSize',8);
ylabel('数值','FontSize',8);

figure(3)
plot(xx,yy,'b','LineWidth',2);
title('xy二维轨迹图')
xlabel('X(mm)','FontSize',10);
ylabel('Y(mm)','FontSize',10);

figure(4)
plot3(xx,yy,zz,'b-','LineWidth',2)
grid on
title('xyz三维轨迹图')
view(55,-70)
xlabel('X(mm)','FontSize',10);
ylabel('Y(mm)','FontSize',10);
zlabel('Z(mm)','FontSize',10);
xlim([0 5000])
ylim([0 5000])
zlim([0 3000])
```