队伍编号	MC2202906
题号	D

基于多目标动态规划的基站集群选址优化分析

摘 要

随着通信技术的发展，移动通信网络的站点位置和种类选择成为了当下的主要难题．本文主要基于 2500x2500 空间内点的坐标等参数设定，在满足门限条件等的约束情况下，利用 0-1 背包动态规划、加权粒子群启发式算法、DBSCAN 为代表的多种算法建立模型，得出有关移动通信网络站址选择和区域聚类的优化求解．

针对问题一，我们首先对弱覆盖点进行统计分析，排除了 $traffic \leqslant 20$ 的点，然后，根据新建基站之间和新建基站与现基站之间门限距离不小于 10 的设定对数据进行了进一步清洗，并最终得到 42863 个弱覆盖栅格数据点作为新建基站的选址范围．数据预处理后，我们基于 0-1 背包动态规划和线性规划，通过回溯和迭代，最终得到遍历后的最优宏基站和微基站的数量和选址，分别为宏基站 508 个，微基站 5786 个，最小成本为 10866，并且满足规划建设基站覆盖的业务量占总业务量的 90.05%．

针对问题二，基于第一问对数据的清洗，首先我们根据惠及周边基站的边际建设成本最小确定了在一处弱覆盖点中以 30 为半径建宏基站还是以 10 为半径建微基站更优，然后我们运用 0-1 背包动态规划、线性规划和加权粒子群启发算法，遍历了基站的旋转角度和选址，最终得到规划建设基站覆盖的业务量占总业务量的 90.06%，确定了应该建 983 个宏基站，3993 个微基站，并得到了宏、微基站的选址坐标、对应的旋转角度以及最小成本 13823．

针对问题三，我们选取了 CURE 层次聚类、DBSCAN、K-means 算法等聚类算法对弱覆盖点进行聚类分析．我们通过肘部法则、经验公式等对关键参数进行选取，其中 DBSCAN 的参数 Eps 为 20，MinPts 为 500．对比三种算法的时间复杂度和聚类结果，由于 DBSCAN 有不受初始聚类中心和聚类值影响的特性，且对密集大规模数据有良好适应能力，时间复杂度为 O(nlogn) 处于居中水平，我们最终选取 DBSCAN 为最佳聚类方法．

关键词：0-1 背包动态规划, 加权粒子群启发算法, DBSCAN, 运筹优化, 规划选址, 边际成本

目录

一、 问题重述 1
1.1 问题背景 1
1.2 问题提出 1
二、 数据预处理 2
2.1 描述性统计 2
2.2 数据清洗 3
三、 问题分析 4
3.1 问题一的分析 4
3.2 问题二的分析 4
3.3 问题三的分析 5
四、 基本假设 5
五、 符号约定和说明 5
六、 模型的建立与求解 5
6.1 问题一：0-1 背包动态规划，线性规划 5
6.1.1 问题分析 5
6.1.2 模型建立 7
6.1.3 模型求解 8
6.2 问题二：0-1 背包动态规划，赋权粒子群算法，线性规划 10
6.2.1 问题分析 10
6.2.2 模型建立 10
6.2.3 模型求解 12
6.3 问题三：CURE 层次聚类，DBSCAN,K-means 算法 14
6.3.1 问题分析 14
6.3.2 CURE 层次聚类 14
6.3.3 DBSCAN 15
6.3.4 K-means 算法 17
6.3.5 时间复杂度 19
七、 模型的优缺点 21
7.1 优点 21
7.2 缺点 21
参考文献 22
附录 1： 问题一的 MATLAB 代码 23
附录 2： 问题二的 MATLAB 以及 PYTHON 代码 26
附录 3： 问题三的 MATLAB 代码 27

一、问题重述

1.1 问题背景

随着移动通信技术的飞速发展，无线网络的需求也与日俱增，根据《2022 年 H1 中国移动通信消费市场研究报告》，2021 年国内 4G 用户数为 10.69 亿户，5G 用户达到了 3.55 亿户．与此同时, 网络规划变得越来越重要，5G 的普及在给用户带来更好体验的同时，由于基站覆盖面积缩小[1] 等原因，通信网络规划发挥着更加突出的作用[2]．平衡好提高网络运营覆盖面积与降低综合建网成本[3] 这二者的关系，需要我们进一步优化移动通信网络站址的选取．

降低综合建网成本是基站规划部署的指导原则[3]．精准的成本控制是基站选址规划的总体考虑，基站的选址与建设规模都要紧紧以它为核心．在无线基站选址时，需要建立合适的基站模型，用以估算站点数目、小区面积和容量等关键因素，以实现资源的最优化配置．基站的选址，辐射范围的确定，成本的考量……完成网络站点的合理部署，从而有效解决弱覆盖问题，是本文建模的重点考虑方面．

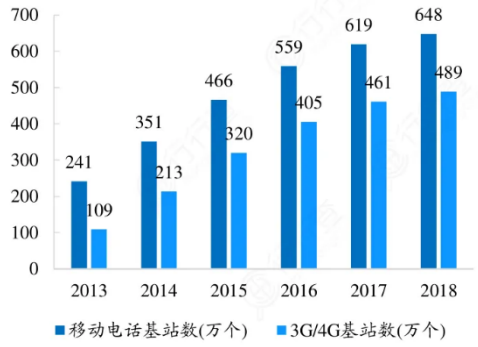

图 1　基站数持续增长，5G 推动基站建设

1.2 问题提出

本文主要利用 matlab、excel、eviews 等对数据进行分析处理，综合运用 matlab、python 等多种软件进行建模求解，最后给出针对移动通信网络站址规划与弱覆盖点区域聚类的最优方案．

问题一：针对给定区域的特定情况，合理规划移动通信网络的选址与每个站址选择的基站种类，使得弱覆盖点总业务量的 90% 被规划基站所覆盖，并考虑基站成本等其他因素．

问题二：在问题一的基础上，考虑基站扇区角度与覆盖面积发生变化时，分析新建基站所能覆盖的弱覆盖点业务量的情况．

问题三：若对不同弱覆盖区域分开管理以更好解决弱覆盖问题，试对所有弱覆盖点进行聚类，要求聚类所用方法的总时间复杂度尽量低．

二、数据预处理

2.1 描述性统计

首先，我们对弱覆盖栅格数据（附件一）进行描述性统计，研究该区域中各栅格 $traffic$ 值的分布特征，通过下列图表我们可知，各栅格 $traffic$ 值存在着较大差异，低 $traffic$ 值的栅格所占比例极大．具体见图 2．

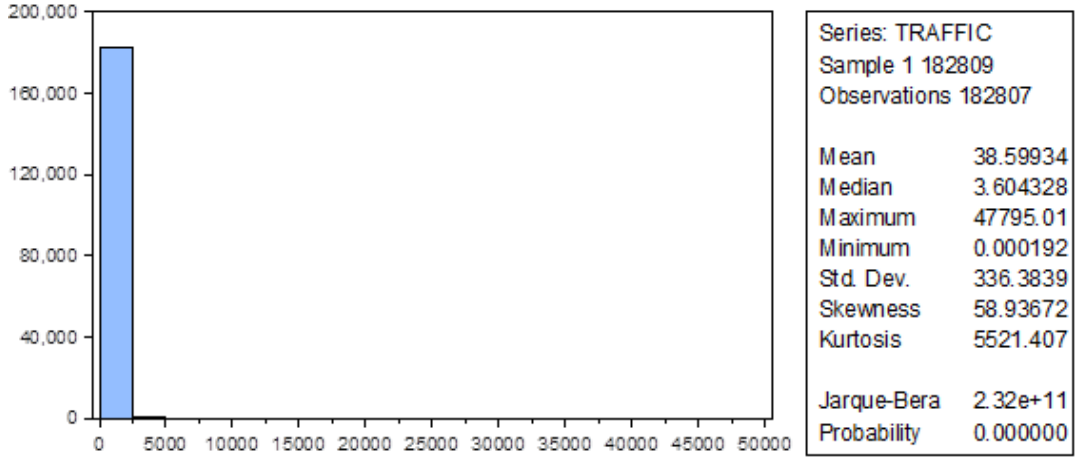

图 2 描述性统计

2.2 基于 $traffic$ 值的筛选

进一步统计分析我们可以观察到，$traffic$ 值小于等于 16 的点所占的业务量份额达到了总量的 6.25%，而这一部分的数据量却超过了 70%，进一步扩大了 $traffic$ 的阈值．$traffic$ 值大于 26 的弱覆盖点虽然只占了总体弱覆盖点的 21.05%，但整体的 $traffic$ 值却大于总体的 90%.

因此，考虑基站建设的建设成本与其他成本因素，在保证建模效果且能更好实现弱覆盖点总业务量 90% 的目标下，，我们借鉴无线基站规划的行业经验[4]，决定首先以 $traffic$ 值作为标准对弱覆盖栅格进行筛选，选取 $traffic$ 值大于 20 的弱覆盖栅格作为基站规划地点．

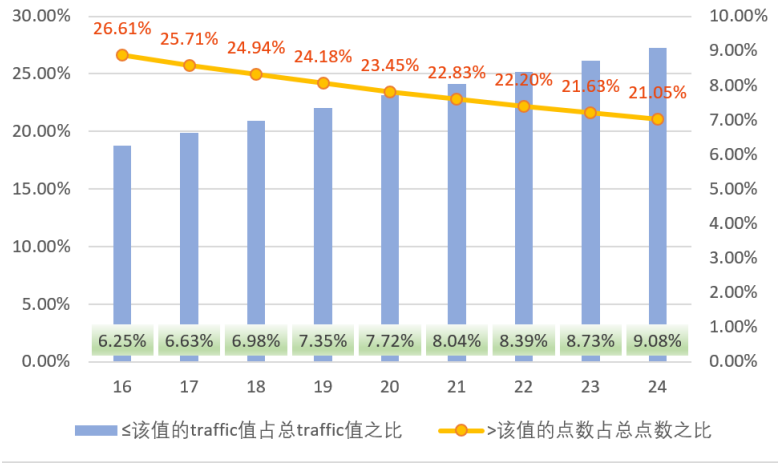

图 3　数据分布特征

2.3 基于门限的筛选

出于对现有站址与新建站址之间的距离不能小于等于门限这一条件的考量，我们根据二范数公式对现有基站之间的距离和现有基站与弱覆盖点之间的距离进行了欧氏距离的判断，设选择基站的覆盖范围为 d，基站所规划的点的坐标为：$P_0(x_0,y_0)$，则对于坐标为：$P(x,y)$ 的点，$\|P-P_0\|_2 \leqslant \mathrm{d}$，则认为该点被该基站覆盖，否则认为该点没有被该基站覆盖．基于门限条件进行数据清洗的效果见图 4.

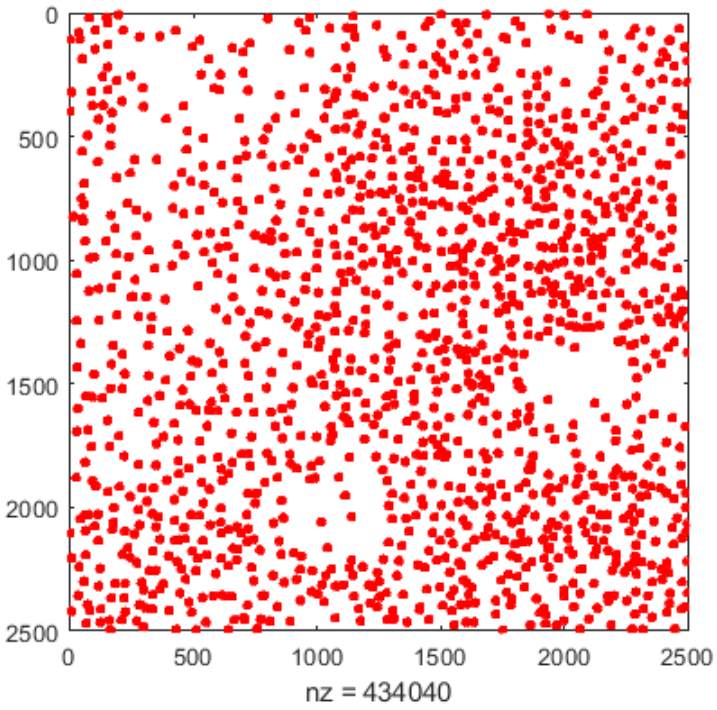

图 4　基于门限条件的数据清洗

2.4 $traffic$ 分布展示

根据弱覆盖栅格 $traffic$ 值的大小与站址距离大于门限这两个约束条件，在数据清洗后筛选出 42863 个弱覆盖栅格数据点，其 $traffic$ 值分布特征见图 5．我们在后文仍然考虑了新建站址与新建站址之间的距离同样需要大于门限的约束条件，将在后续章节展开叙述．

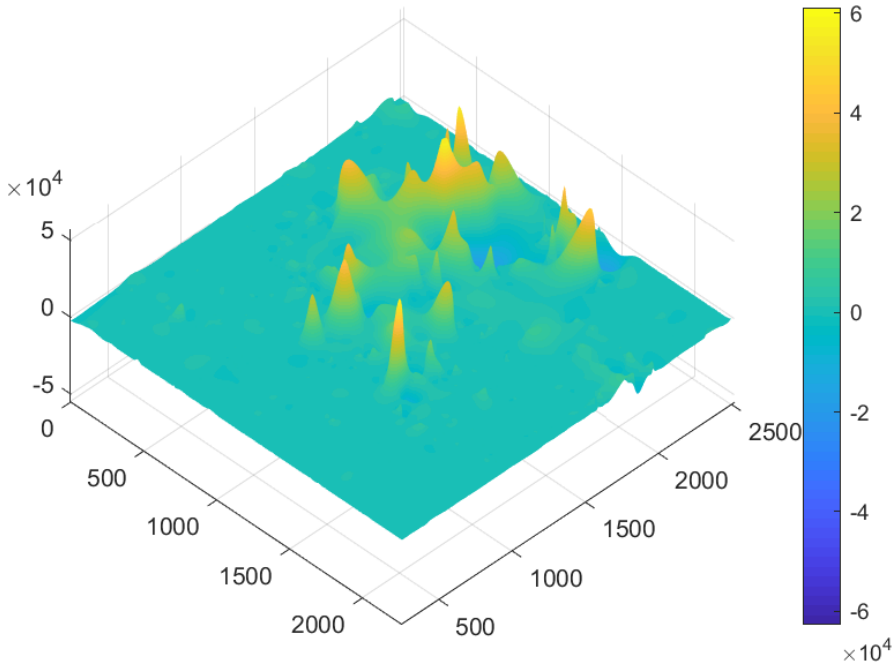

图 5 $traffic$ 分布展示

三、问题分析

3.1 问题一的分析

在移动通信网络站址规划问题中，对规划基站位置和基站类型的选择需要综合考虑建设成本与基站覆盖业务量等因素，并且由于覆盖密度过高会造成信号出现交叉干扰等负面情况[4]，我们同时对基站之间的门限进行了条件约束．本题通过选择规划基站站址的坐标以及每个站址选择的基站种类，来实现弱覆盖点总业务量的 90% 被规划基站覆盖的目标．

解决此类问题，我们选择采用 0-1 背包动态规划来构建各观测点是否建设基站、基站类型等因素针对覆盖业务量的优化方案．在问题一中，我们构建基站建设成本函数作为目标函数，考虑基站建设门限、覆盖业务量比重、基站类型建设优先级等因素作为约束条件，以此得到移动通信网络站址规划的优化方案．

3.2 问题二的分析

问题二在问题一的基础上更改了基站覆盖面的形状，并且添加了基站覆盖角度的选项．我们首先排除一部分不满足门限条件的点，再根据单个基站惠及周围弱覆盖点的成

本最小，判断出了每个弱覆盖点最优的基站角度和基站种类，最后通过 0-1 背包动态规划，赋权粒子群算法和线性规划，选择对哪些弱覆盖点进行基站建立.

3.3 问题三的分析

问题三中我们需要对弱覆盖点进行聚类，并要求所选方法时间复杂度尽可能的低.我们首先选取了 CURE 层次聚类、DBSCAN、K-means 等聚类算法对弱覆盖点进行聚类，然后比较了它们的聚类效果和时间复杂度，最终选择了 DBSCAN 的聚类结果.

四、基本假设

1. 指定区域为同一高度的无干扰区域；
2. 同类型基站（宏基站或微基站）的发射功率与覆盖范围是相同的；
3. 不考虑人为因素对于规划基站发射功率与覆盖范围的影响；
4. 基站的辐射时间忽略不计；
5. 基站服务能力不受天气等自然因素的影响；
6. 短时间内通信量不会改变；
7. 通信交换时长忽略不计；
8. 边缘弱覆盖点的性质不会随 2500×2500 栅格外的新建基站的改变而改变；
9. 新建基站的性能（辐射范围）和成本在短时间内不会产生较大波动；

五、符号约定和说明

符号	符号说明
d	规划基站的覆盖范围
r_1	单个宏基站覆盖半径，值为 30
r_2	单个微基站覆盖半径，值为 10
c_1	单个宏基站建设成本，值为 10
c_2	单个微基站建设成本，值为 1
n_1	以 r_1 为半径做圆内含的弱覆盖点
n_2	以 r_2 为半径做圆内含的弱覆盖点
tr_{ij}	弱覆盖点（x_i,y_j）的业务量

六、模型的建立与求解

6.1 问题一：0-1 背包动态规划，线性规划

6.1.1 问题分析

在移动通信网络站址规划问题中，对规划基站位置和基站类型的选择需要综合考虑建设成本与基站覆盖业务量等因素，并且由于覆盖密度过高会造成信号出现交叉干扰等负面情况[4]，我们同时对基站之间的门限进行了条件约束．本题通过选择规划基站站址的坐标以及每个站址选择的基站种类，来实现弱覆盖点总业务量的 90% 被规划基站覆盖的目标．栅格被基站覆盖的定义是：设选择基站的覆盖范围为 d，基站所规划的点的坐标为：$P_0(x_0,y_0)$，则对于坐标为：$P(x,y)$ 的点，$\|P-P_0\|_2 \leqslant \mathrm{d}$，则认为该点被该基站覆盖，否则认为该点没有被该基站覆盖．

解决此类问题，我们选择采用 0-1 背包动态规划算法来构建各观测点是否建设基站、基站类型等因素针对覆盖业务量的优化方案．在问题一中，我们构建基站建设成本函数作为目标函数，考虑基站建设门限、覆盖业务量比重、基站类型建设优先级等因素作为约束条件，以此得到移动通信网络站址规划的优化方案．

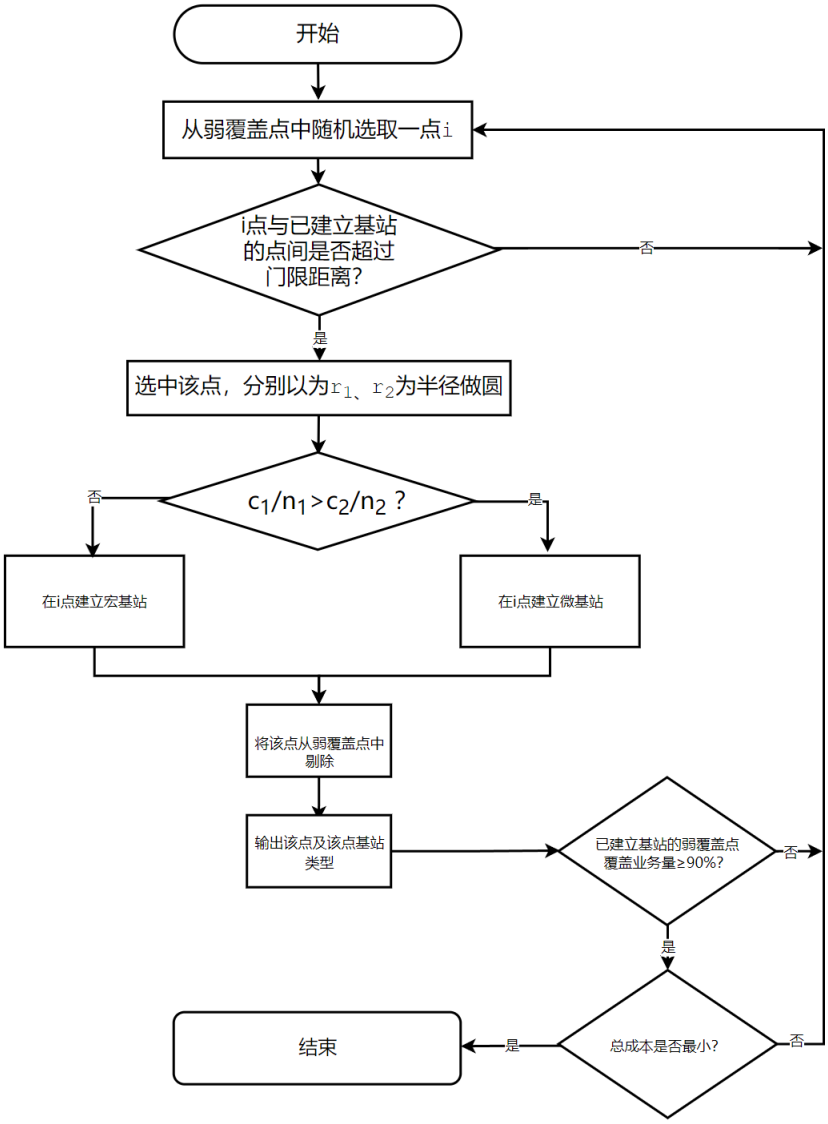

图 6　问题一流程图

6.1.2 模型建立

本模型的建立，主要是为了解决实现规划基站的建设能够覆盖弱覆盖点 90% 的总业务量，并且考虑经济效益因素，尽可能花费较小的建设成本, 因此我们选择 0-1 背包动态规划算法[6] 进行求解.

动态规划算法通常是用来解决某种最优性质的问题. 基本思想是将待求解问题划分为若干个子问题，先求解子问题，然后从子问题的解得到原问题的解. 基于动态规划方法求解 0-1 背包问题，即 n 个物体，1 个背包. 对物品 i，其价值为 vi, 重量为 Wi，背包的容量为 W，在选取物品的重量小于等于背包重量的，尽可能使背包中物品的总价值最

大，即可设计如下的约束条件和目标函数：

$$\max \sum_{i=1}^{n} v_i x_i$$

$$s.t. \quad \begin{cases} \sum_{i=1}^{n} w_i x_i \geqslant W \\ x_i \in 0,1,2<i<n \end{cases}$$

在本问中，首先，设一共需要建设 N 个基站，为使得弱覆盖点总业务量的 90% 被规划基站覆盖，对选择宏基站还是微基站进行规划，引入 0-1 变量 b_k, s_k，

满足条件：

$$b_k = \begin{cases} 0 & \text{未建设宏基站} \\ 1 & \text{建设宏基站} \end{cases} \quad s_k = \begin{cases} 0 & \text{未建设微基站} \\ 1 & \text{建设微基站} \end{cases} \tag{1}$$

因为一个宏基站覆盖范围为 30，成本为 10；而单个微基站覆盖范围为 10，成本为 1．易得微基站的单位建设成本更低，在其他条件不变的情况下，优先选择微基站．

若 $\frac{c_1}{n_1} > \frac{c_2}{n_2}$, 则在 i 点建设微基站；若 $\frac{c_1}{n_1} \leqslant \frac{c_2}{n_2}$, 则在 i 点建设宏基站．

假定在给定区域中任意一点 D 坐标为（x_i，y_j，tr_{ij},b_k，s_k），其中 tr_{ij} 表示该点的业务量. 在假设既定任务（弱覆盖点 90% 的业务量被规划基站所覆盖）完成的条件下，为尽可能降低基站建设所要花费的成本，构造目标函数：

$$W_{min} = 10 \sum_{k=1}^{N} b_k + \sum_{k=1}^{N} s_k \tag{2}$$

目标函数需要满足如下四个约束条件：

1. 规划基站的建设能够覆盖弱覆盖点 90% 的总业务量：

$$\sum_{i=1}^{n} {}_1 b_k tr_{ij} + \sum_{j=1}^{n} {}_2 s_k tr_{ij} \geqslant 90\% \sum tr_{ij} \tag{3}$$

2. 给定区域中各点至多只能建设一个基站：

$$b_k + s_k \leqslant 1 \tag{4}$$

3. 新建基站与新建基站之间距离需要大于门限：

$$\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \geqslant 10 \tag{5}$$

4. 新建基站只有两种选择，即宏基站与微基站：

$$N = n_1 + n_2 = \sum^{N} b_k + \sum^{N} s_k \tag{6}$$

根据以上目标函数与约束条件，建立以下模型：

$$W_{min} = 10\sum_{k=1}^{N} b_k + \sum_{k=1}^{N} s_k$$

$$s.t. \quad \begin{cases} \sum_{i=1}^{n}{}_1 b_k tr_{ij} + \sum_{j=1}^{n}{}_2 s_k tr_{ij} \geqslant 90\% \sum tr_{ij} \\ b_k + s_k \leqslant 1 \\ \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \geqslant 10 \\ N = n_1 + n_2 = \sum^{N} b_k + \sum^{N} s_k \end{cases}$$

6.1.3 模型求解

. 经过 MATLAB 运算程序，求解得到共需要建设 508 个宏基站，5786 个微基站，建设成本 W 为 10866 元，规划建设基站的业务量占总业务量的 90.05%．在给定区域内基站建设示意图如下：

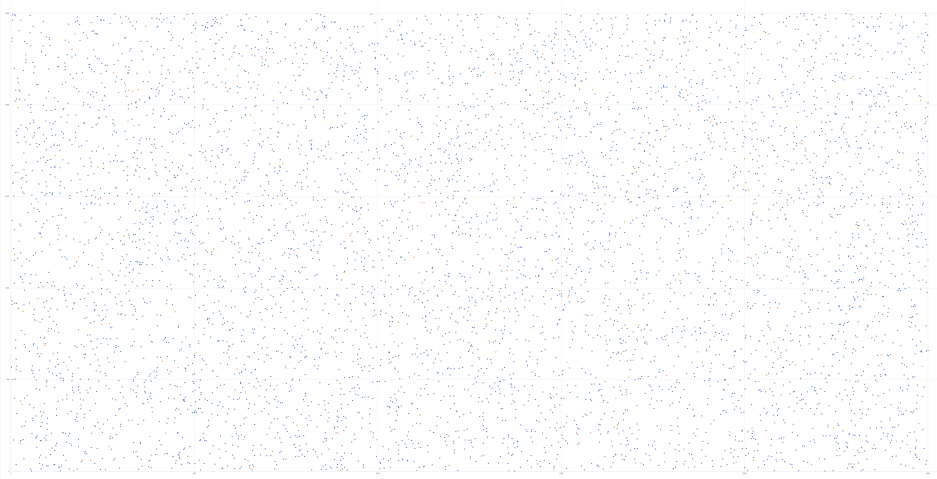

图 7 基站分布图

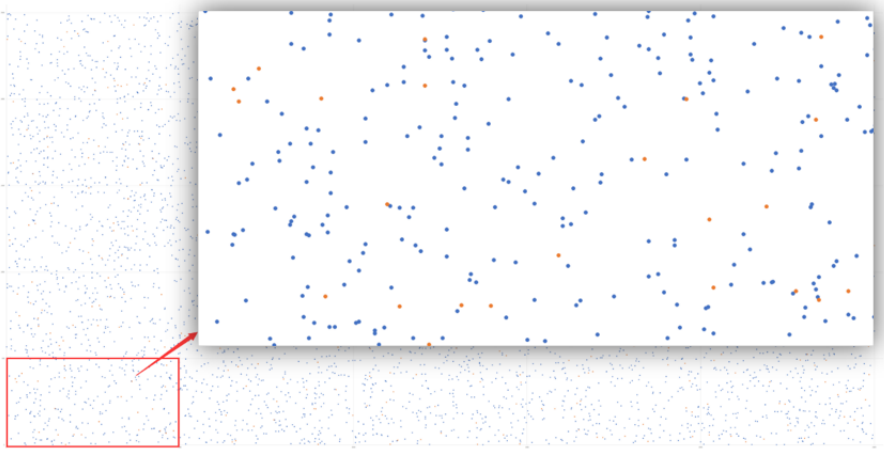

图 8 基站分布（局部放大）

注：橙色点为宏基站规划建设点，蓝色点为微基站规划建设点

并且，我们可以得到规划基站选择站址的坐标与选择站址的类型，限于篇幅有限，正文部分仅展示 50 个宏基站与 50 个微基站的站址选取坐标，完整的站址选取名单详见附件.

宏基站	宏基站	微基站	微基站
(1981, 1184)	(1827, 1352)	(267, 103)	(2274, 2428)
(1191, 1011)	(167, 1224)	(1905, 1711)	(987, 2465)
(742, 443)	(1125, 96)	(2079, 1976)	(699, 1302)
(1921, 546)	(429, 2222)	(713, 624)	(1086, 896)
(2022, 2071)	(978, 360)	(93, 2072)	(675, 2205)
(1000, 308)	(261, 1786)	(2102, 125)	(998, 2043)
(190, 486)	(2098, 170)	(2381, 676)	(1797, 977)
(2157, 582)	(510, 412)	(2499, 969)	(745, 343)
(2256, 464)	(1377, 806)	(777, 200)	(22, 232)
(544, 1526)	(1727, 1104)	(610, 1671)	(2367, 1353)
(2344, 577)	(2165, 2431)	(2472, 434)	(1369, 235)
(1075, 1125)	(2160, 1323)	(1894, 2084)	(1234, 892)
(2176, 1183)	(2224, 954)	(1888, 1017)	(1502, 329)
(1955, 1771)	(2238, 1611)	(2051, 1015)	(1391, 1053)
(2105, 2403)	(1286, 1401)	(776, 1641)	(1379, 1171)
(1279, 2321)	(392, 898)	(1242, 284)	(571, 2168)
(286, 1449)	(593, 725)	(1587, 1281)	(1000, 2206)
(570, 1723)	(2418, 505)	(368, 679)	(168, 805)
(298, 1761)	(1935, 1621)	(531, 668)	(507, 1497)
(1795, 40)	(1175, 2437)	(1690, 1488)	(1041, 1132)
(2367, 1340)	(1977, 2036)	(2293, 2425)	(7, 1125)
(995, 2313)	(1760, 1042)	(237, 2119)	(314, 277)
(1239, 2308)	(1589, 2009)	(1264, 2045)	(290, 1392)
(236, 2088)	(842, 2130)	(2472, 2427)	(9, 104)
(1373, 515)	(858, 1606)	(889, 37)	(648, 1583)

图 9 规划基站选址与类型（部分）

6.2 问题二：0-1 背包动态规划，赋权粒子群算法，线性规划

6.2.1 问题分析

问题二是对问题一的进一步延展，在问题一中我们已经得到使得弱覆盖点总业务量的 90% 被规划基站覆盖时的站址规划. 在问题二中我们结合移动通信网络站址实际，考虑到基站并不是完全的圆形覆盖，而是以三个不同扇区的形态对弱覆盖点进行覆盖，在问题一的基站成本等因素仍然成立的前提条件下，寻找最优站址与扇区角度，并验证在此条件下新建站能否覆盖弱覆盖点总业务量的 90%.

6.2.2 模型建立

首先，不难得出在三个扇区的主方向夹角互为 120 度时，基站的覆盖面积达到最大[7]，相较于扇区角度处于其他状态时，主方向夹角互为 120 度时基站效益达到最大. 单个扇区如图 1 所示，扇区整体形状如图 2 所示.

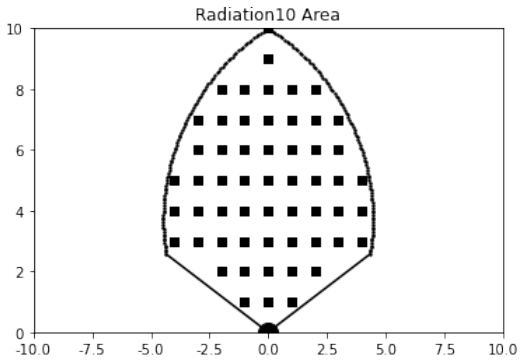

图 10 微基站单个扇区

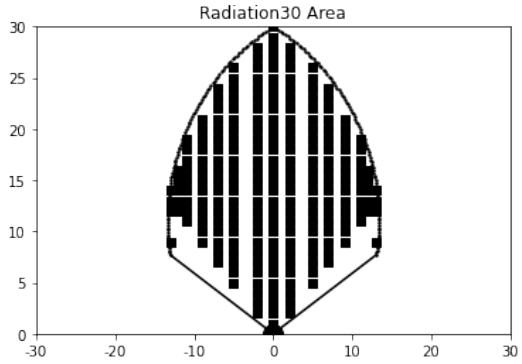

图 11 宏基站单个扇区

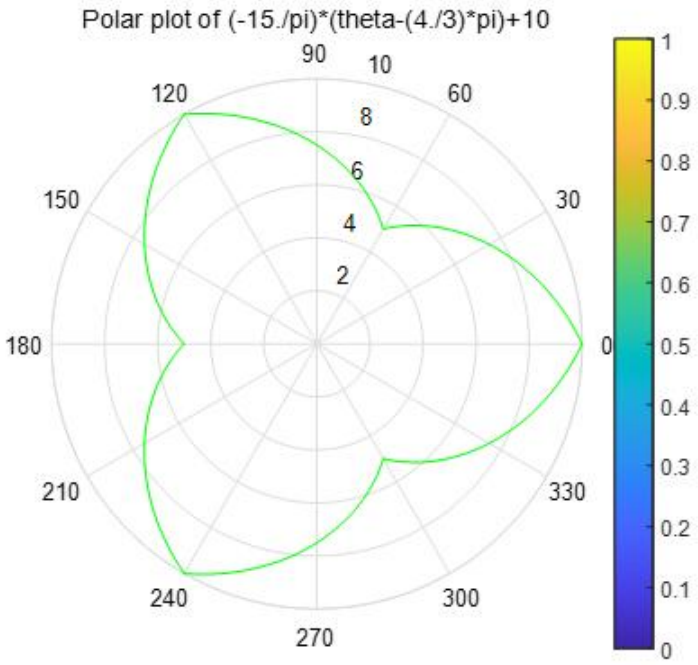

图 12 扇区整体形状

在问题二中，对移动通信站址选择需要同时考虑基站建设种类（宏基站或微基站）和扇区指向两个因素，一方面基站建设种类决定了对弱覆盖区域的最长辐射距离，另一方面由于覆盖范围在主方向左右按线性逐渐缩小，主方向的选择对基站的覆盖能力也极为重要[7].

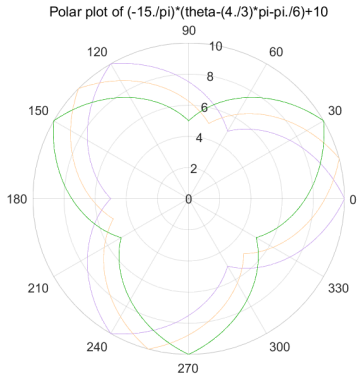

图 13 扇区指向变化效果展示

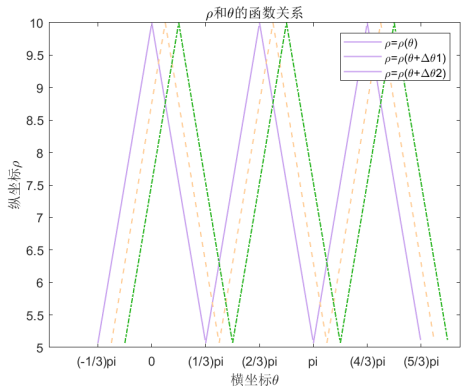

图 14 扇区指向变化时参数变化

与问题一相同，在主方向保持不变情况下，若 $\frac{c_1}{n_1} > \frac{c_2}{n_2}$, 则在 i 点建设微基站；若 $\frac{c_1}{n_1} < \frac{c_2}{n_2}$, 则在 i 点建设宏基站.

放开主方向保持不变这一条件，令 θ=0，1，2，$\cdots$, 359

令 $W(\theta_m, \mathrm{b}_m, s_m) = \min\{w(\theta_1, b_1, s_1), w(\theta_2, b_2, s_2), \cdots, w(\theta_{359}, b_{359}, s_{359})\}$, 该点最优建站方案即确定.

对给定区域内符合条件的各点均重复此过程，则能确定各基站选址建设种类与扇区指向两个关键因素.

综上所述，基于以上最优建站方案可以确定目标函数：

$W(\theta_m, \mathrm{b}_m, s_m) = \min\{w(\theta_1, b_1, s_1), w(\theta_2, b_2, s_2), \cdots, w(\theta_{359}, b_{359}, s_{359})\}$

结合问题一，目标函数需要满足以下四个约束条件：

1. 规划基站的建设所能覆盖的业务量为宏基站与微基站所能呢覆盖的业务量之和：

$$\sum_{i=1}^{n} {}_1 b_k tr_{ij} + \sum_{j=1}^{n} {}_2 s_k tr_{ij} \geqslant 90\% \sum tr_{ij} \tag{7}$$

2. 给定区域中各点至多只能建设一个基站：

$$b_k + s_k \leqslant 1 \tag{8}$$

3. 新建基站与新建基站之间距离需要大于门限：

$$\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \geqslant 10 \tag{9}$$

4. 新建基站只有两种选择，即宏基站与微基站：

$$N = n_1 + n_2 = \sum^{N} b_k + \sum^{N} s_k \tag{10}$$

根据以上目标函数与约束条件，建立以下模型：

$$\mathrm{W}(\theta_m, \mathrm{b}_m, s_m) = \min\{w(\theta_1, b_1, s_1), w(\theta_2, b_2, s_2), \cdots, w(\theta_{359}, b_{359}, s_{359})\}$$

$$s.t. \quad \begin{cases} \sum_{i=1}^{n} {}_1 b_k tr_{ij} + \sum_{j=1}^{n} {}_2 s_k tr_{ij} = \sum tr_{ij} \\ b_k + s_k \leqslant 1 \\ \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \geqslant 10 \\ N = n_1 + n_2 = \sum^{N} b_k + \sum^{N} s_k \end{cases}$$

6.2.3 模型求解

经过 MATLAB 运算程序，求解得到在最优站址与扇区角度的条件下，共需要建设 983 个宏基站，3993 个微基站，建设成本合计 13823 元，覆盖弱覆盖点的总业务量为 90.06%．在给定区域内基站建设示意图如下：

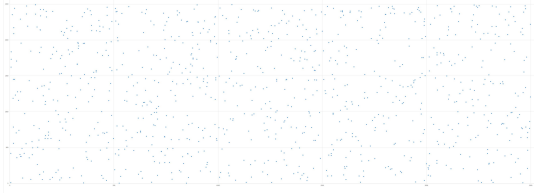

图 15 微基站规划选址

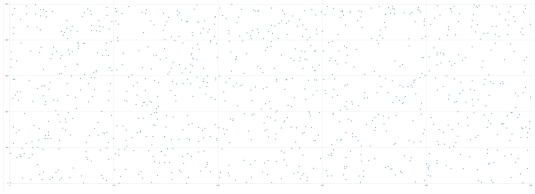

图 16 宏基站规划选址

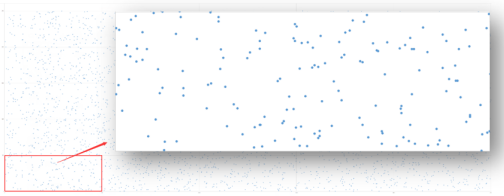

图 17 微基站规划选址（局部放大）

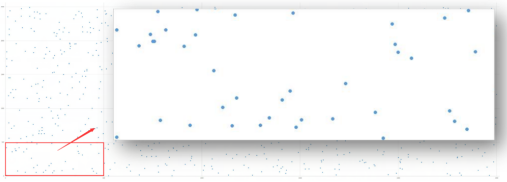

图 18 宏基站规划选址（局部放大）

并且，我们可以得到规划基站选择站址的坐标与选择站址的类型，限于篇幅有限，正文部分仅展示其中 25 个微基站和 25 个宏基站的站址选取坐标，完整的站址选取名单详见附件.

宏基站	旋转角度	微基站	旋转角度
(962,1614)	52	(1374,775)	289
(961,1613)	98	(2327,1497)	219
(2243,1726)	202	(873,1220)	317
(294,1092)	139	(1499,1721)	272
(2408,2011)	231	(2155,714)	266
(533,558)	337	(1004,2267)	54
(1333,2068)	299	(2110,2430)	247
(522,406)	182	(1868,943)	50
(784,267)	13	(172,1545)	193
(1943,1195)	249	(811,1068)	265
(1543,2327)	118	(127,1768)	304
(1360,1305)	48	(1764,2282)	321
(525,988)	97	(2472,1090)	330
(1022,458)	40	(963,1552)	101
(906,2239)	123	(2477,1491)	312
(125,564)	52	(557,114)	299
(2482,1583)	165	(1748,2069)	219
(2320,881)	236	(694,46)	202
(735,2364)	281	(547,1846)	337
(982,1305)	207	(878,2107)	299
(1650,781)	26	(778,2315)	246
(2315,2486)	313	(208,2390)	293
(523,359)	115	(1221,1849)	261
(1880,2126)	41	(1493,1829)	229
(929,1565)	17	(830,67)	341

图 19 规划基站选址与类型（部分）

6.3 问题三：CURE 层次聚类，DBSCAN,K-means 算法

6.3.1 问题分析

实际工作中，基于人力物力的考量，时间空间的限制，对弱覆盖点进行聚类后分区管理可以更好的解决弱覆盖问题．考虑类间相关性，2 个弱覆盖点的欧式距离不大于 20 聚为一类；考虑聚类的传递性，若 A、B 为一类，B、C 为一类，则 A、C 也为一类．此外时间复杂度要在满足条件的情况下尽可能的低．我们综合时间复杂度和问题实际解决能力的判断，对多种聚类算法进行考量，确定了最合适的聚类方法．

6.3.2 CURE 层次聚类

算法介绍：　层次聚类是一种直观算法．通过计算不同类别数据点间相似度创建一颗有层次的嵌套聚类树，可以自下而上将小的 cluster 合并聚集，也可以自上而下将大的 cluster 进行分割[9]．它可以不需要指定具体类别数目，聚类完成之后，可在任意层次横切一刀，得到指定数目的簇．由于数据量庞大，我们采用了 CURE 多阶段层次聚类法，这是一种基于质心和基于代表对象方法之间的中间策略，具体思路如下：

1. 最开始，每个对象就是一个独立的类
2. 为了处理大数据，采用“随机抽样”和“分割手段”
3. 传统算法通常采用一个对象来代表类，cure 采用“多个中心”代表类．
4. 在对噪声点的处理中，对聚类过程中增长缓慢的直接剔除；在聚类快结束的时候，把类内个数明显少的类剔除．

中心点计算公式：

$$w.mean = \frac{|u|u.mean+|v|v.mean}{|u|+|v|} \tag{11}$$

代表点计算公式：

$$w.rep = p + \alpha * (w.mean - p) \tag{12}$$

当代表点个数超过阈值时，需要选择阈值数的代表点：首先选择距离聚类中心最远的数据点，然后选择距离前一个选出的代表最远的点．依次选择至数量 = 阈值．对以上选择出来的点进行收缩因子处理之后，才能是真正的代表点[10]．

聚类结果：　我们一共设置了八个聚类点，具体聚类结果如下图，由于数据量过于庞大，树形图难以表示，聚类结果如下图所示，

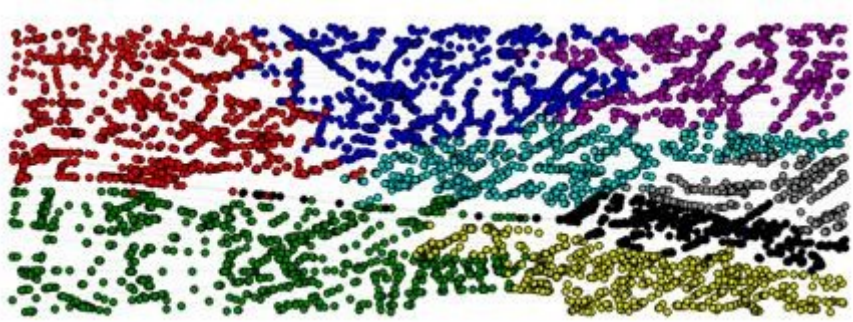

图 20 层次聚类的聚类结果

6.3.3 DBSCAN

算法介绍： 不同于层次聚类、K-means 算法、K-means++ 算法，DBSCAN 是一种基于密度的聚类算法．在 DBSCAN 中，点被分为核心点、边界点、密度可达点、异常点等类型．算法根据制定的邻域密度参数找出范围内的核心点，再根据核心点集合推导其密度相连的样本集合从而生成聚类簇．

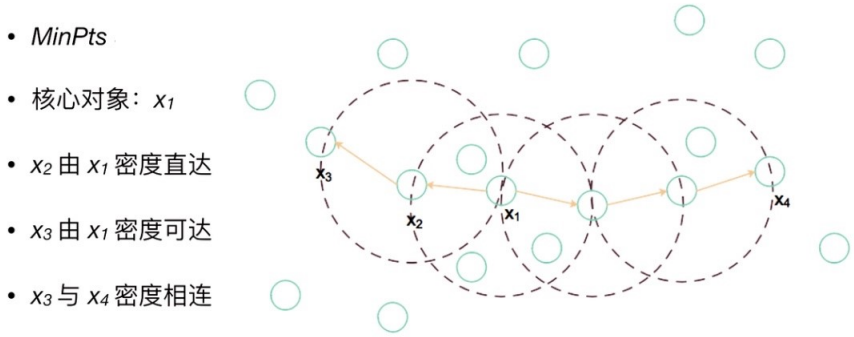

图 21 自制 DBSCAN 原理介绍

参数选取： DBSCAN 聚类算法极大取决于两个超参数 eps 和 MinPts 的选择[11]．

我们采用欧氏距离作为距离的度量方法，Eps 根据要求选定为 20．根据 matlab 判断计算出半径为 20 的圆内（包括边界）的坐标点有 1257 个，因此 MinPts 我们折中选取为 500．

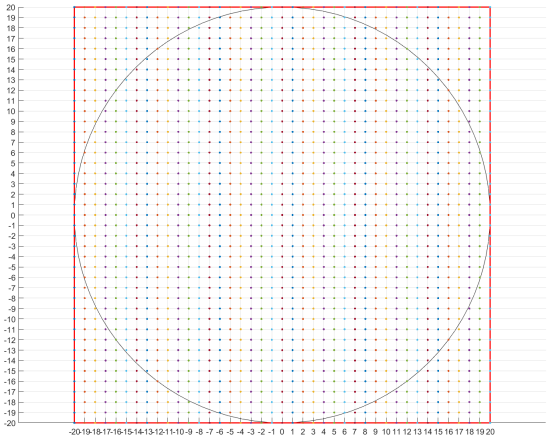

图 22 半径为 20 的圆内的坐标点可视化

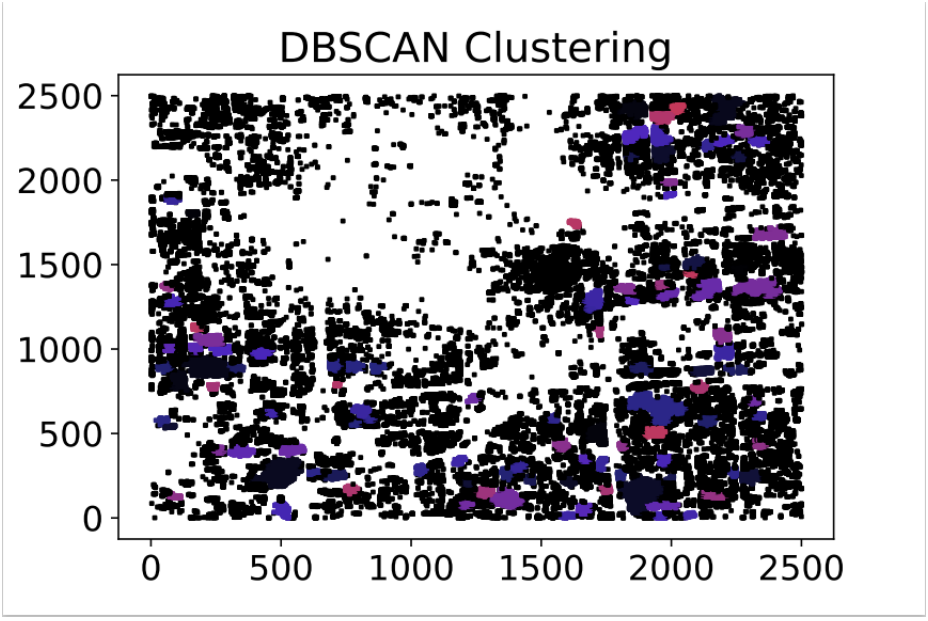

图 23 DBSCAN 聚类结果

聚类结果： 我们得出了聚类种类和聚类中心值的坐标，由于篇幅有限，仅展示前 50 个，

聚类种类	中心值坐标	聚类种类	中心值坐标
1	(555.9655,231.7745)	26	(642.499,550.159)
2	(2265.41,1415.677)	27	(964.639,1367.652)
3	(1950.288,679.8404)	28	(975.803,2072.27)
4	(140.8965,1745.155)	29	(26.813,2389.829)
5	(2246.384,2313.032)	30	(1559.95,2450.967)
6	(213.6522,914.4919)	31	(1659.227,2092.553)
7	(1530.638,1478.644)	32	(2012.245,1217.754)
8	(1017.576,639.2166)	33	(705.29,2184.312)
9	(1892.684,153.0206)	34	(454.026,1202.434)
10	(1884.897,2290.418)	35	(1580.578,199.439)
11	(1284.22,170.0278)	36	(2432.104,490.578)
12	(286.0278,452.6078)	37	(1284.712,1427.286)
13	(2233.873,913.6722)	38	(1079.881,1496.955)
14	(1826.718,1335.434)	39	(2229.024,787.237)
15	(699.389,937.9839)	40	(1447.42,1005.426)
16	(2294.397,1841.933)	41	(1421.461,316.922)
17	(357.8255,2307.901)	42	(715.456,703.566)
18	(1621.517,430.9443)	43	(2475.76,1031.269)
19	(2299.081,331.3792)	44	(1161.416,2263.665)
20	(170.469,1282.69)	45	(2018.162,549.891)
21	(1760.258,1965.4)	46	(1981.854,1869.16)
22	(233.435,1123.355)	47	(2267.492,680.778)
23	(1157.942,1491.202)	48	(245.042,1480.867)
24	(1746.238,1915.622)	49	(2005.412,2114.395)
25	(705.213,1783.801)	50	(1600.787,1668.154)

图 24 DBSCAN 得出的聚类种类和聚类中心值的坐标

6.3.4 K-means 算法

算法介绍： K-means 聚类是一种基于划分的聚类算法，它认为两个目标距离越近，相似度越大．基本步骤是按照给定样本之间距离大小，将样本集划分为 K 个簇，并计算 K 个聚类中心，然后计算各个对象与各个种子聚类中心点的距离，并根据距离将对象分配．每分配一个对象，聚类中心会被重新计算，重复直至达到终止条件[12]．

K-Means 的核心目标是将给定的数据集划分成 K 个簇（K 是超参），并给出每个样本数据对应的中心点．具体步骤非常简单，可以分为 4 步：

1. 数据预处理．主要工作是标准化、异常点过滤．

2. 随机选取 K 个中心，记为 $u_1^{(0)}, u_2^{(0)}, \cdots, u_k^{(0)}$

3. 定义损失函数，$Jc, u = \min \sum_{i=1}^{M} \|x_i - u_{ci}\|^2$

4. 令 t=0,1,2,$\cdots J$ X_i ，将其分配到距离最近的中心; 对于每一个类中心 k，重新计算该类的中心.

K-Menas 具有高效可伸缩，计算复杂度低，收敛速度快，原理相对通俗易懂，可解释性强.

同时，K-Means 也有一些明显的缺点：受初始值和异常点影响，聚类结果可能不是全局最优而是局部最优；K 是超参数，一般需要按经验选择；样本点只能划分到单一的类中等[13].

参数选取： K 值的选择一般基于实验和多次实验结果. 例如采用手肘法，尝试不同 K 值并将对应的损失函数画成折线. 手肘法认为图上的拐点就是 K 的最佳值.

由于数据量太大，Elbow method 拐点表现特征不明显. 如果将区域 20 等分，每片区域范围大约为 560×560 个栅格，比较合理. 因此我们初始参数选取 K=20.

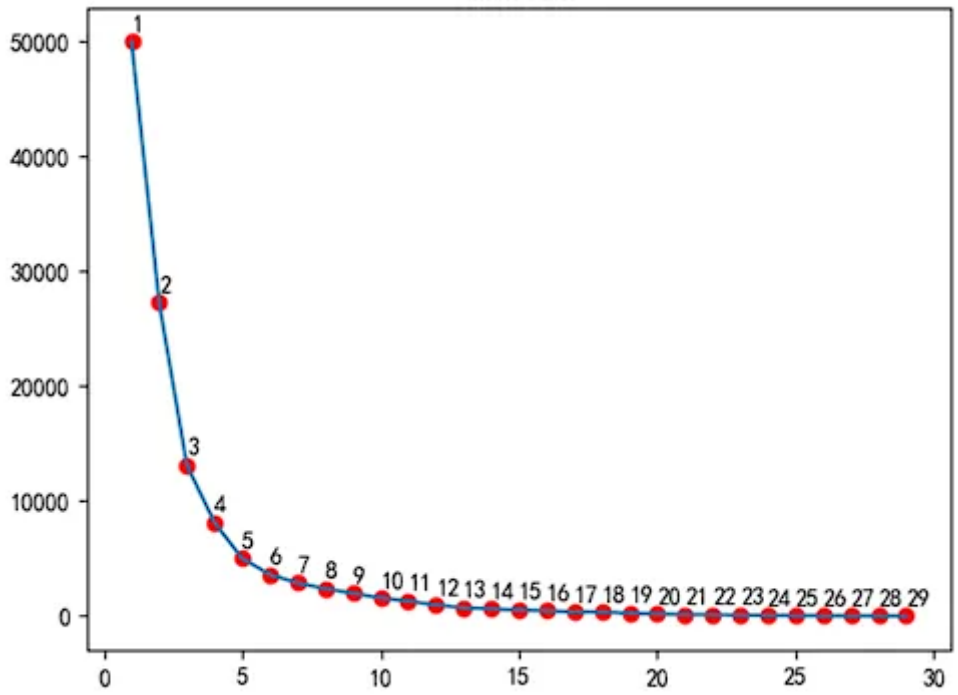

图 25 K-means 的肘部法则图

聚类结果： 我们得到了各点的分类以及聚类中心点的坐标，由于篇幅有限，分类结果只展示部分（前 50 个点）

聚类种类	中心值坐标	聚类种类	中心值坐标
1	(555.9655,231.7745)	26	(1284.22,170.0278)
2	(2265.41,1415.677)	27	(286.0278,452.6078)
3	(1950.288,679.8404)	28	(2233.873,913.6722)
4	(140.8965,1745.155)	29	(1826.718,1335.434)
5	(2246.384,2313.032)	30	(699.389,937.9839)
6	(213.6522,914.4919)	31	(2294.397,1841.933)
7	(1530.638,1478.644)	32	(357.8255,2307.901)
8	(1017.576,639.2166)	33	(1621.517,430.9443)
9	(1892.684,153.0206)	34	(2299.081,331.3792)
10	(1884.897,2290.418)	35	(170.469,1282.69)

图 26 K-means 得出的聚类中心点坐标

聚类种类	坐标	聚类种类	坐标
20	(66,1486)	7	(1643,1486)
20	(67,1486)	7	(1644,1486)
20	(177,1486)	7	(1645,1486)
20	(187,1486)	7	(1646,1486)
20	(284,1486)	14	(1951,1486)
20	(309,1486)	14	(1953,1486)
7	(1400,1486)	14	(1958,1486)
7	(1418,1486)	14	(1959,1486)
7	(1419,1486)	14	(1961,1486)
7	(1464,1486)	14	(1963,1486)
7	(1483,1486)	14	(1965,1486)
7	(1554,1486)	14	(1967,1486)
7	(1571,1486)	14	(1968,1486)
7	(1582,1486)	14	(1969,1486)
7	(1584,1486)	14	(1976,1486)
7	(1610,1486)	14	(1977,1486)
7	(1611,1486)	14	(1981,1486)
7	(1615,1486)	14	(1983,1486)
7	(1618,1486)	14	(1984,1486)
7	(1624,1486)	14	(1985,1486)
7	(1635,1486)	14	(1986,1486)
7	(1639,1486)	14	(1999,1486)
7	(1640,1486)	14	(2007,1486)
7	(1641,1486)	14	(2008,1486)
7	(1642,1486)	14	(2009,1486)

图 27 K-means 得出的各点分类结果

6.3.5 时间复杂度

时间复杂度： 判断一个算法所编程序运行时间的多少，并不是将程序编写出来，通过在计算机上运行所消耗的时间来度量．一方面，解决一个问题的算法可能有很多种，一

一实现的工作量无疑是巨大的，得不偿失；另一方面，不同计算机的软、硬件环境不同，即便使用同一台计算机，不同时间段其系统环境也不相同，程序的运行时间很可能会受影响，严重时甚至会导致误判．因此，引入算法的时间复杂度概念，用以对算法运算时间进行度量，它的本质上是一个函数，来表示算法的层次和计算量．根据这个函数，我们能在不用测试完所有进程的情况下，粗略估算算法的执行效率，通俗来说其表示的只代码执行时间随数据规模增长的一种变化趋势．

假设一个算法的执行次数为 $T(n)$，只保留最高次项且忽略最高项的系数后可以得到函数 $f(n)$，用大 记号表示算法的时间性能，此时算法的时间复杂度就是 $O(f(n))$．时间复杂度的关系为：

$O(1) < O(\log 2n) < O(n) < O(n\log 2n) < O(n^2) < O(n^3) < \cdots < O(2^n) < O(n!) < O(n^n)$

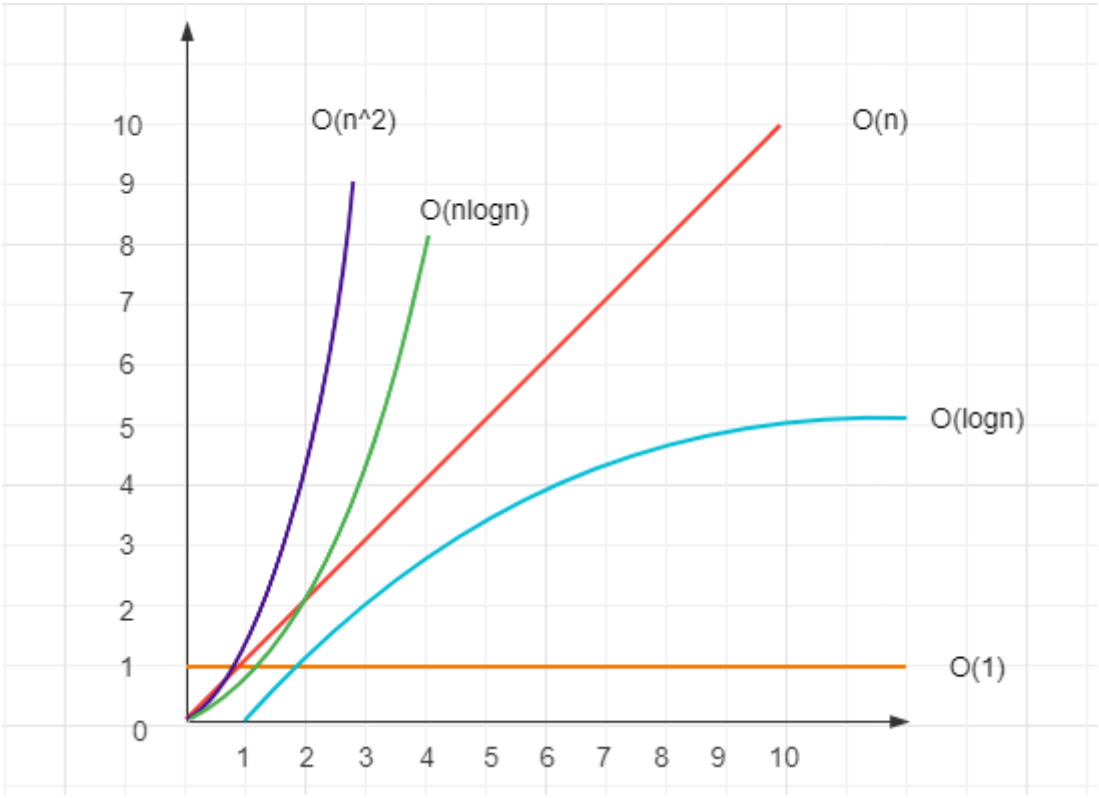

图 28 时间复杂度对比

通过计算，我们所用的算法中，CURE 层次聚类法的时间复杂度为 $O(n^3)$，DBSCAN 聚类法的时间复杂度为 $O(nlogn)$，K-means 聚类法的时间复杂度为 On．

选择 **DBSCAN** 的说明： 层次聚类法虽然直观，并可以一次性得到整个积累过程，但是计算量太大，每次计算都要计算多个 cluster 内所有数据点的两两距离，时间复杂度太高，因此不被纳入考虑的范围．

K-means 虽然时间复杂度比 DBSCAN 低，但 K-means 受初始聚类中心和 K 值影响较大，随机性较强，而 DBSCAN 多次运行会产生相同的结果，较为稳定．并且由于弱覆盖点数据量太大，肘部法呈二次曲线关系，不起作用，确定不了确切的 K-means 的 K

值．另外，K-means 易受异常值的影响，并且很难处理非球形的簇和不同大小的簇，而 DBSCAN 在这些方面表现要比 K-means 更优越．

综上，基于实际情况和时间复杂度的考察，我们倾向于选择 DBSCAN 作为弱覆盖点的聚类算法，相关的聚类结果已在前文给出．

七、模型的优缺点

7.1 优点

1. 逻辑思维清晰，思路明确；
2. 运用了多次遍历算法，综合全面考虑到所有情形；
3. 相对于静态规划，动态规划模型能够得到全局最优解[14]；
4. 将统计学、几何学、运筹学和具体数学算法灵活结合，突出了学科交叉性；
5. 第二问在第一问 0-1 背包动态规划的基础上加入粒子群启发式算法优化，使得覆盖率和成本达到双向最优；
6. 第三问运用了 CURE 层次聚类、DBSCAN、K-means++ 等多种聚类算法，并通过进行时间复杂度和聚类效果的二维对比从而得出最优聚类算法．

7.2 缺点

1. 没有很好地进行回溯，从而得出确切的最优算法；
2. 可能陷入局部最优解，结果没有收敛到全局最优；
3. 由于反复遍历欧氏距离，计算量较大，代码的运行速度较慢；
4. 部分聚类算法随机性较强，聚类算法中一些参数是人为设定，存在主观性[15]；
5. 第二问中固定了三个主方向夹角 120 度，且为简化模型基站旋转角度设置成离散点而非连续函数，可能错失成本最低解．

参考文献

[1] 刘家欣. 基于带权极小模理想点法的 5G 网络基站选址优化 [D]. 西安电子科技大学,2019.DOI:10.27389/d.cnki.gxadu.2019.003217.

[2] 朱思峰, 陈国强, 张新刚, 李强. 多目标优化量子免疫算法求解基站选址问题 [J]. 华中科技大学学报 (自然科学版),2012,40(01):49-53.DOI:10.13245/j.hust.2012.01.001.

[3] 刘娅汐. 移动通信网络覆盖计算与优化方法研究 [D]. 北京科技大学,2021.DOI:10.26945/d.cnki.gbjku.2021.000123.

[4] 赵文涛. 异构蜂窝无线网络规划和优化算法 [D]. 南京大学,2013.

[5] 颜陆红, 郭文普, 徐东辉, 杨海宇. 基于 NSGA-II 算法的机动通信系统站址规划方法 [J]. 计算机应用研究,2022,39(01):226-230+235.DOI:10.19734/j.issn.1001-3695.2021.06.0232.

[6] 党俊肖. 基于 GIS 的特频专用 CDMA 移动通信站传播覆盖预测研究 [D]. 兰州大学,2016.

[7] 石建. 面向 5G 的移动通信技术及其优化研究 [D]. 天津大学,2017.

[8] 沈亮. 数据挖掘在移动通信网络优化中的应用 [D]. 上海交通大学,2009.

[9] 肖伟雄. 移动通信网络覆盖优化分析方法与系统实现 [D]. 中南大学,2010.

[10] 宋军. 移动通信网络规划与网优技术研究 [D]. 北京邮电大学,2007.

[11] 任雨婷. 面向 5G 超密集网络的动态分簇算法研究 [D]. 北京交通大学,2020.DOI:10.26944/d.cnki.gbfju.2020.002716.

[12] 韩佳成. 面向临时热点资源分配的用户聚类研究 [D]. 北京交通大学,2020.DOI:10.26944/d.cnki.gbfju.2020.002589.

[13] 刘旭. 基站天线效率相关技术研究 [J]. 移动通信,2022,46(03):21-25.

[14] 冯幸, 钟其铿.5G 无线基站部署方式研究 [J]. 中国新通信,2021,23(17):61-62.

[15] [1] 程松贵, 张文群, 王福松. 基于 UDP 的 AGV 无线通信系统设计 [J]. 电子技术与软件工程,2021(11):3-4.

附 录

附录 1：问题一的 MATLAB 代码

```
%%第一问
%MATLAB 做圆
rec = zeros(21,21);
for i = 1:21
for j = 1:21
if((i-11)^2+(j-11)^2 <= 100)%圈定圆形区域
rec(i,j)=1;
end
end
end


%MATLAB 排除门限
load('data_give.mat','old_x');
load('data_give.mat','old_y');
load('data_give.mat','x');
load('data_give.mat','y');
load('data_give.mat','traffic');
station1 = zeros(2500, 2500);
traffic = zeros(2500, 2500);
station2 = zeros(2500, 2500);
l=10;
old_x=old_x+1;
old_y=old_y+1;
x=x+1;
y=y+1;

for i = 1:length(old_x)
station(old_x(i),old_y(i))=1;
begin_row=0;end_row=0;begin_col=0;end_col=0;
if(old_x(i)-10<=0)
begin_row=1;
else
begin_row=old_x(i)-10;
end
if(old_x(i)+10>2500)
end_row=2500;
else
end_row=old_x(i)+10;
end
```

```
if(old_y(i)-10<=0)
begin_col=1;
else
begin_col=old_y(i)-10;
end
if(old_y(i)+10>2500)
end_col=2500;
else
end_col=old_y(i)+10;
end
for m = begin_row:end_row
for n=begin_col:end_col
if(station2(m,n)==0)
station2(m,n)=rec(m-x(i)+11,n-y(i)+11);%将问题化为稀疏矩阵模式
end
end
end
end
for i = 1:length(x)
traffic(x(i),y(i))=traffic(i);
end
sum(station2(:))%统计0值的个数
spy(station2,"black",2)%标注稀疏矩阵图


#清洗Traffic小于20的数据（python）
import pandas as pd
import random
df=pd.read_csv("附件1 弱覆盖栅格数据(筛选).csv")
a=df[df["traffic"]>20]
a.to_excel("清洗后.xlsx")
%画Traffic三维图
x=xlsread('清洗后.xlsx','Sheet1','B2:B42836');
y=xlsread('清洗后.xlsx','Sheet1','C2:C42836');
traffic=xlsread('清洗后.xlsx','Sheet1','D2:D42836');
xmax=max(x);
xmin=min(x);
ymax=max(y);
ymin=min(y);
[X,Y]=meshgrid(xmin:5:xmax,ymin:5:ymax);
Z=griddata(x,y,traffic,X,Y,'v4');
figure(1)
FontS = 16;
FontW = 'bold';
```

```
L(1) = xlabel('$x$','interpreter','latex','FontSize',FontS,'FontWeight',FontW);
L(2) = ylabel('$y$','interpreter','latex','FontSize',FontS,'FontWeight',FontW);
L(3) = zlabel('$z$','interpreter','latex','FontSize',FontS,'FontWeight',FontW);
L(4) = title('$Traffic','interpreter','latex','FontSize',FontS,'FontWeight',FontW)
    ;
surf(X,Y,Z)
shading interp
colorbar


%第一问求解
clc, clear, a1 = readmatrix('清洗后.csv',head=None);
a1(isnan(a1))=0;
a2= xlsread('清洗后.csv');
a2(isnan(a2))=0;
check=a1-a2;
c1=10;
c2=1;
r1=30;
r2=10;
int sum,i,j,n;
for i=[1:];
for j=[1:]
sum1=10i+j;
end
end
for j=[1:];
for i=[1:];
sum2=i.*tr+j.*tr;
end
end
if sum2>=sum(traffic)*0.9%判断是否满足达到总交通量的90%
for i=[j[j-1:1]
sum_total=i+j;
end
end
if c1/r1<=c2/r2%判断是否满足经济效益
m+=1
else
n+=1
x=matrix(m,n)
sum_total
x.to_excel("基站分布1.csv")
```

附录 2：问题二的 MATLAB 以及 PYTHON 代码

```
%%第二问
#python作弧，预处理
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import cm
from sklearn.preprocessing import scale
plt.rcParams['font.sans-serif'] = ['Times']
plt.rcParams['axes.unicode_minus'] = False
import warnings
warnings.filterwarnings('ignore')
sec_mat10 = np.zeros([11,11])
angle = np.arange(0,np.pi/3,np.pi/30)
angle = np.arange(0,np.pi/3,np.pi/300)
y = (10-angle*10/(np.pi*2/3))*np.cos(angle)
x = (10-angle*10/(np.pi*2/3))*np.sin(angle)
plt.scatter(x,y,c='black',s=2)
plt.scatter(-x,y,c='black',s=2)
plt.scatter([0],[0],c='black',s=200)
plt.plot([0,x[-1]],[0,y[-1]],c='black')
plt.plot([0,-x[-1]],[0,y[-1]],c='black')
plt.xlim(-10,10)
plt.ylim(0,10)
plt.title('Radiation10 Area')
plt.draw()
plt.savefig("Radiation10 Area.png")
plt.show()
df_ans.to_excel('问题二.CSV')

%matlab求解
clc, clear, a1 = readmatrix('清洗后.csv',head=None);
a1(isnan(a1))=0;
a2= xlsread('清洗后.csv');
a2(isnan(a2))=0;
check=a1-a2;
c1=10;
c2=1;
r1=30;
r2=10;
int sum,i,j,n,angle;
for i=[1:];
for j=[1:]
```

```
sum1=10i+j;
end
end
for j=[1:];
for i=[1:];
sum2=i.*tr+j.*tr;
end
end
if sum2>=sum(traffic)*0.9
for i=[j[j-1:1]
sum_total=i+j;
end
end
if c1/r1<=c2/r2
m+=1
else
n+=1
x=matrix(m,n)
sum_total
x.to_excel("基站分布2.csv")

#python
import pandas as pd
pd.read_csv("基站分布2.CSV")
large_angle=[]
small_angle=[]
for angle in range(360):
if c1/r1<=c2/r2:
large_angle.append(angle)
else
small_angle.append(angle)
result=[large_angle,small_angle]
pd.append.result in[4:8]
pd.to_excel("基站分布2.0.CSV")
```

附录 3：问题三的 MATLAB 代码

```
###第三问
##kmeans聚类
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets.samples_generator import make_blob
from scipy.spatial.distance import cdist
import pandas as pd
```

```
pd.read_csv("附件1 弱覆盖栅格数据(筛选)")
class K_Means(object):

def __init__(self, n_clusters=6, max_iter=300, centroids=[]):
self.n_clusters = n_clusters
self.max_iter = max_iter
self.centroids = np.array(centroids, dtype=np.float)
def fit(self, data):
if (self.centroids.shape == (0,)):
self.centroids = data[np.random.randint(0, data.shape[0], self.n_clusters), :]
for i in range(self.max_iter):
distances = cdist(data, self.centroids)
c_index = np.argmin(distances, axis=1)
for i in range(self.n_clusters):
if i in c_index:
self.centroids[i] = np.mean(data[c_index == i], axis=0)
def predict(self, samples):
distances = cdist(samples, self.centroids)
c_index = np.argmin(distances, axis=1)
return c_index
c_index.to_excel("K-means聚类标注+中心点坐标.csv")


##肘部效应的判断
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv('清洗后.CSV') # 文件目录加文件名
df.head()
#定位数据
X = df.iloc[:,2:6]
X.head()
# 标准化数据
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
standX = scaler.fit_transform(X)
standX
# 肘部法则的可视化
from sklearn.cluster import KMeans
elbow=[]
for i in range(1,30): # 创建遍历，找到最合适的k值
kmeans=KMeans(n_clusters=i,init='k-means++',random_state=101)
kmeans.fit(X)
elbow.append(kmeans.inertia_)
# 通过画图找出最合适的K值
```

```
import seaborn as sns
sns.lineplot(range(1,20),elbow,color='blue')
plt.rcParams.update({'figure.figsize':(16,10),'figure.dpi':100})
plt.title('Elbow Method')
plt.show()


##CURE层次聚类算法
def gave_data(df):
# 训练数据
df_train = df.iloc[:,1:]
# 随机抽取样本，从大样本中每次随机抽取小样本
sample_df = df_train.sample(n=1000)
# 归一化处理，对聚类数据进行归一化处理
min_max_scaler = preprocessing.MinMaxScaler()
X_minMax = min_max_scaler.fit_transform(sample_df)
# 一类一标签
# labelture = [i for i in range(sample_df.shape[0])]
# 统一标签
labelture = [0 for i in range(sample_df.shape[0])]
return X_minMax, labelture
def tree(X, labelture):
# row_clusters = linkage(pdist(df, metric='euclidean'), method='complete') #
    使用抽秘籍距离矩阵
row_clusters = linkage(X, method='complete', metric='euclidean')
print (pd.DataFrame(row_clusters, columns=['row label1', 'row label2', '
    distance', 'no. of items in clust.'],
index=['cluster %d' % (i + 1) for i in range(row_clusters.shape[0])]))
plt.show()



##DBSCAN聚类算法
import random
import utils
# 算法主体
def dbscan(Data, Eps, MinPts):
# 拿到处理好的Dataset中的"点集"
num = len(Data)
# 设置所有点为未访问的点
unvisited = [i for i in range(num)]
# 设置已访问点的列表
visited = []
# C为输出的簇结果，初始化为全-1(默认认为全是噪声点)
```

```
C = [-1 for i in range(num)]
# 使用k标记不同object，k=-1代表噪声点
k = -1

# 开始循环(结束条件为所有点都被访问过了)
while len(unvisited) > 0:
# 随机选出一个未访问的点p
p = random.choice(unvisited)
unvisited.remove(p)
visited.append(p)
# 设N为p的epsilon邻域中objects的集合
N = []
for i in range(num):
if utils.get_distance(Data[i], Data[p]) <= Eps:
N.append(i)

# 接下来判断如果p点的epsilon邻域中的objects对象数大于指定阈值，说明p是一个核心对
    象
if len(N) >= MinPts:
# 某个簇对应不同的值...
k = k+1
C[p] = k
# 对于p的epsilon邻域中的每一个object
for pi in N:
if pi in unvisited:
unvisited.remove(pi)
visited.append(pi)
# 判断每个pi的所属类型
# 如果是核心再去找该pi的邻域内的objects，再对这些objects迭代讨论...
# 设M是位于pi的邻域内的objects的集合
M = []
for j in range(num):
if utils.get_distance(Data[j], Data[pi]) <= Eps:
M.append(j)
if len(M) >= MinPts:
# 将该核心object的邻域内objects放入N循环迭代讨论
for t in M:
if t not in N:
N.append(t)
# 若pi还不属于任何簇，C[pi] == -1说明C中pi值没有改动(此object为边界点)
if C[pi] == -1:
C[pi] = k
# 没有到达最低阈值 --> 噪声点
else:
```

```
C[p] = -1
# 算法结束
return C
```