城市共享单车调度分配及社会影响研究

摘要

近期，共享单车蓬勃发展，迅速占领了各大学校、城市街道，给人们的生活带来了便利，同时也带来了诸多社会、治安问题，如：出租车生意被严重打击、私占共享单车等。本文针对：因分配不当而导致单车供不应求、共享单车对出租车生意的冲击等问题建立模型进行研究，以给出改善问题的建议。

问题一中本文通过对附件 1 数据的统计得到与附件 2 中的需求表对应的当前分布表，统计从任意一点出发，到其余任意一点的概率分布，据此预测从某一点出发到另一点的概率。在将附件 1 数据中的时间参数加入统计， 还可以任意一时间段内车子的使用情况，根据不同时段车子使用次数的多少，统计出各个地点车子使用的高峰期、低谷期等规律，从而更合理地调配车辆。

对于问题二、三的模型及优化过程，本文基于最小二乘法的误差分析，以附件二中的需求分布表和实际分布表做差得到的矩阵的每个正数元素的平方和作为“代价函数”（之所以是正数，是因为如果是负数，即此处需求<实际拥有量，是不会降低用户满意度的），通过均分法，将“需求-实际分布”最大的值和最小的值做平均再减去原矩阵，可得到“调整矩阵”，让原实际分布矩阵加上调整矩阵，这便是经过了一次优化。在多次优化后，当一次调整中代价函数的降幅<某个阈值时，即达到最优解，停止优化。

对于问题三的新增车辆，本文通过之前已得的车辆分布情况，按原有比例将新增的 100 辆车分配到各个区域，再通过已有模型进行优化。最终发现，通过模型的优化后，代价函数值极小，车辆的需求基本得到满足，这也验证了模型的准确性。

对于问题四，本文通过已知的附件 3（单车投放量和其对应的打车次数）和上网查找的资料，再经过三次样条插值后，绘制出其变化曲线。通过观察推测，发现该图像的函数服从指数函数+二次函数的形式，故据此设出含未知参数的函数式，通过 MATLAB 的 lsqcurvefit()函数进行曲线拟合，找出了近似表示数据的函数式，在进行绘制原数据函数和拟合数据的一阶导，发现两者的隐藏规律：单车投放量的建议临界值，打车的固定用户数等。最后绘制了对在不同单车投放量时的单车和出租车所占市场的比率的图像，进行剖析，给出建议。

最后，本文对模型进行了中肯的评价，并对模型的应用进行了推广。

关键词：最小二乘法、均分法、三次样条插值、曲线拟合

城市共享单车调度分配及社会影响研究

目录

1. 问题的提出

当今世界，城市机动化发展速度过快，城市交通问题日益突出。共享单车作为慢性交通系统，由于零污染、零排放、灵活便利的特性，在世界范围内蓬勃发展。它起源于 1965 年在阿姆斯特丹倡导的“白色自行车”计划，在历经了三代发展后，愈发智能化，对传统公共交通方式进行有效补充，解决“最后一公里的问题”。然而，城市共享单车系统仍存在着相关问题影响其发展和完善，例如区域间的车辆调度问题。提高区域间共享单车调度的合理性和及时性能大幅减少管理运行成本，提高用户的满意度等已成为迫切需要。

2. 问题分析

2.1 问题一分析

（1）通过对附件 1 数据的统计，可以得到当前总时间内的从任意一点到其余任意一点的出行车次，构成矩阵 Tall_src_to_dst。然后据此可得出从区域 a 出发，到其余所有区域的各自的概率，由此画出从每个点出发到其余所有点的概率扇形分布图。

（2）在将附件 1 数据中的时间参数加入统计， 还可以任意一时间段内车子的使用情况，根据不同时段车子使用次数的多少，统计出车子使用的高峰期、低谷期等规律，从而更合理地调配车辆。

2.2 问题二分析

（1）附件二给出了“从十个区域中任一区域去其余任一区域所需的人次分布表”——需求矩阵 Tneed，用 Tneed-Tall_src_to_dst 得到距离矩 Tdistance。让用户的满意，就是将 Tneed 和 Tall_src_to_dst 的相似程度更高，也就是 Tdistance 接近 zeros(10)。本文通过对车辆的分布进行调整，使车使用次数多的地方得到更多的车，使用次数少的地方得到相对少的车。通过使用最小二乘法，进行误差评测，确定改进效果。

2.3 问题三分析

（1）度量指标：距离矩阵 Tdistance 所有正元素的平方和。

（2）通过均匀化处理，每次根据 Tdistance 得到 Tadjust，将 Tall_src_to_dst 加上 Tadjust，多次调整车辆分布，使得 Tall_src_to_dst 和 Tneed 的“距离”尽可能地小，直至收敛，最终根据得到的分布矩阵，确定每个区域适合的车辆数，使模型更高效、稳定。

（3）对于增加 100 辆车的情况，我们对 Tall_src_to_dst 的数据进行了按比例的扩展，然后沿用上述方法进行优化。将最终结果补在原图上，可发现如果加入新的 100 辆车后，用户的需求已基本得到了满足。

2.4 问题四分析

（1）通过附件三给出的共享单车数和对应的打车人次，在上网查找数据后，进行三次样条插值处理，使样本数充足。

（2）观测已有数据，可以推测出大致打车人次 Ncab 和单车投放数 Nbike 的函数关系：Ncab = f(Nbike)。

（3）确定了大致关系后，使用 lsqcurvefit 进行曲线拟合，确定具体函数关系，寻找规律，进行量化研究。

（4）利用得出的规律，更加合理地调配车辆的分布，改善模型的效率，提高用户的满意度。

3. 模型假设

1. 假设共享单车在使用的过程中不发生损坏，调度前后所有区域的车辆总数保持不变，即调度过程只是把车辆在各个区域之间进行重分布；
2. 假设每个人骑行的速度没有很大的偏差；
3. 假设车辆只能停在规定的十个区域；
4. 假设任意一个区域调入调出的车辆都能一次完成，具有可行性；

4. 符号说明

符号	含义
1{}	判断符号，如果{}里的表达式为真，+1，为假，+0
Source_area	出发区域
Start_time	出发时间
Arrive_time,	到达时间
target_area	到达区域
i	出发区域号
j	到达区域号
S_E	出发点到目的点的矩阵
S_Eij	从区域 i 到区域 j 的在总时间内的车辆出发的次数
n	当前正在遍历数据的行数
T	一个时间段内的总的车辆使用的次数
colsize	Ttime_table 表中各行的列数
K	Ttime_table 表的行数

L	Ttime_table 表的列数
col(l)	Ttime_table 表第 l 行的数据值
N	需求矩阵
R	实际矩阵
X	距离矩阵
Y	代价函数
Rij	矩阵 R 第 i 行第 j 列的值
Hi	R 第 i 行的所有的值的和
Rij	R 第 i 行第 j 列的值
Sum	R 中所有的值的和
Pi	R 中第 i 行数据和占总数据和的百分比
Pij	R 中第 i 行第 j 列数据占第 i 行数据的的百分比
Partij	R 中第 i 行第 j 列数据占总数据和的百分比

5. 模型建立

5.1 问题一模型建立与求解

5.1.1 时空分布情况

问题 1 中，首先在 MATLAB 软件中对附件 1 中的数据进行处理，得到<start_arrive 表>，列为 Source_area， Start_time， Arrive_time, target_area 四列变量的各个值，总共有 11629 行数据，再设置一个从出发点到目的点的矩阵 S_E=zeros(10,10);接着对该表进行遍历，利用表达式

$$S\_E_{ij}=\sum_{n=1}^{1169}1\{Source\_area=i,target\_area=j\}$$

对表里的每一行数据进行判断，如果满足出发点为 i,目的点为 j,则+1，否则+0，对所有的行都遍历完成，则最后得到的求和的总值便是矩阵第 i 行第 j 列的值，也就是从区域 i 到区域 j 的在总时间内的车辆出发的次数；(具体代码见附件)，最后利用工具得到扇形图（见图 5.1.1），可以直观的看出在一天内从所有的区域出发去往各个其他区域的比例，形象的得到一天内从某个区域出发到达其他区域的分布分布情况；

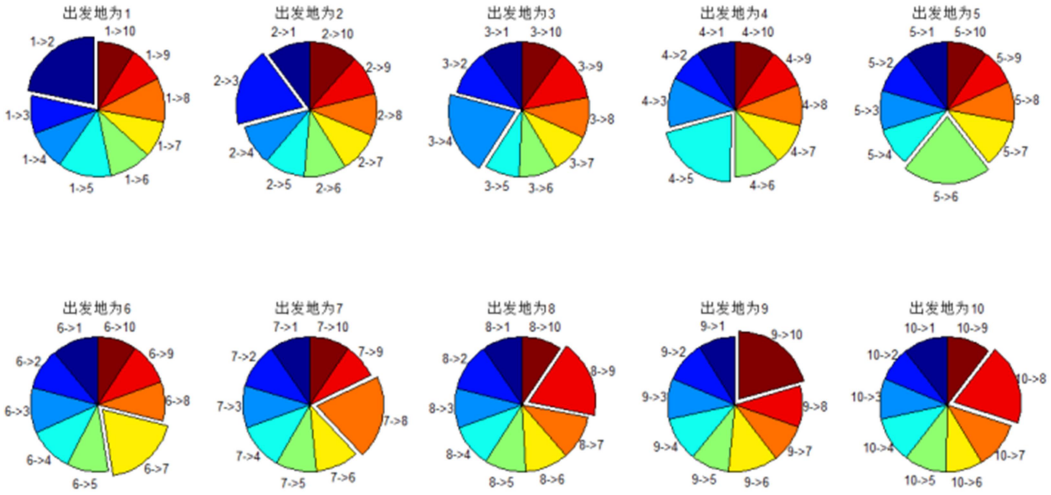

图 5.1.1 各出发点到目的点的分布

然后，把 11629 条使用车辆的数据再次用 MATLAB 进行处理，得到<Ttime_table 表>，共有 1000 行数据，每一行的数据代表的是每一辆车在一天内的使用时间情况（出发时间→到达时间→出发时间→到达时→……），选定一个 15min 的时间间隔 [t,t+15] ,通过

$$T_{[t,t+15]}=\frac{\sum_{K=1}^{1000}\sum_{l=1}^{colsize}1\{t\le col(l)\le t+15\}}{2}$$

$$s.t.\begin{cases}col(l)\in(Start\_time\,,Arrive\_time)\\ 1\le colsize\le m,\text{为每一行的列数},m\in R\end{cases}$$

对每一行内的每一列数据值进行判断，如果列值在时间段内，则+1，否则+0，判断所有的数据，得到一个时间段内的总的车辆使用的次数，循环计算，最终得到所有的时间内车辆被使用的分布情况，得到如下的折线图（图 5.1.2）和柱状图（图 5.1.3）

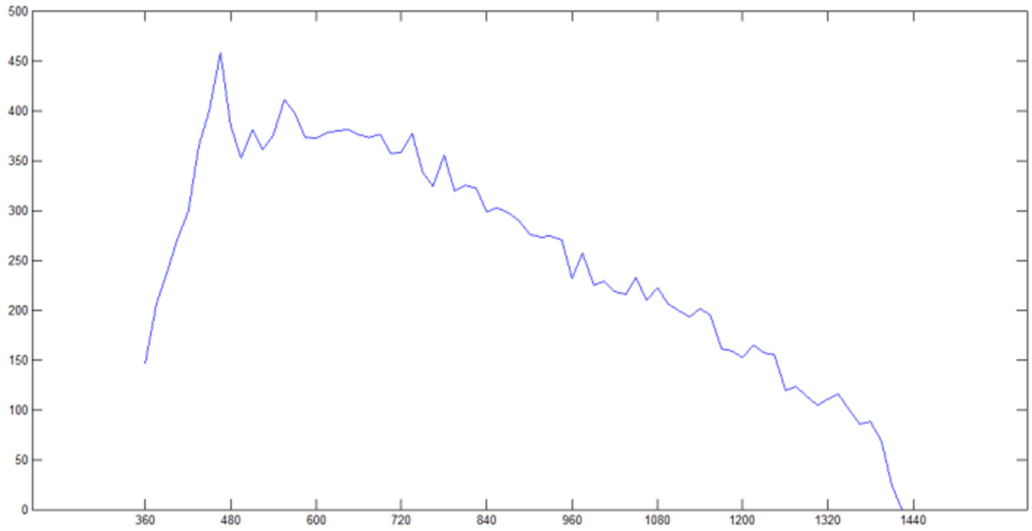

图 5.1.2 每个(15min)时间段内车使用次数（折线图）

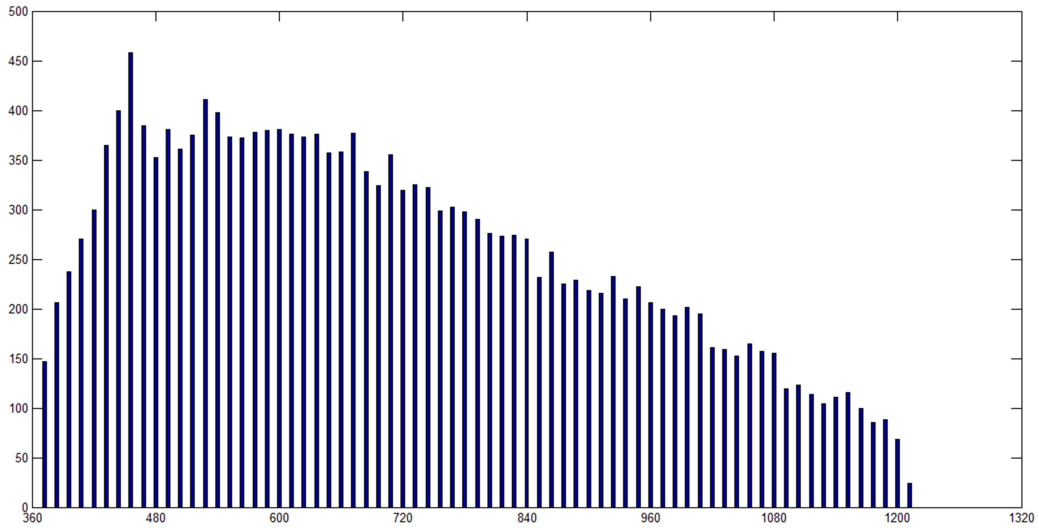

图 5.1.3 每个(15min)时间段内车使用次数(柱状图)

从图中可以看出，大约在 480min~600min，即大约 8 点到 10 点的时候，车辆使用是最多的，我们可以据此在出现高峰期的地点补充车辆，以防此地的实际分布与需求量偏差太大。

5.2 问题二模型建立与求解

5.2.1 数据预处理

通过条件附件 2 和骑行数据，经过 MATLAB 的处理统计得到以下两个矩阵（图 5.2.1、图 5.2.2）。

	1	2	3	4	5	6	7	8	9	10
1	0	240	119	123	145	126	115	127	112	102
2	135	0	225	129	120	126	139	127	121	140
3	125	132	0	251	110	117	126	137	158	126
4	116	115	148	0	261	144	132	141	124	119
5	128	133	124	116	0	273	138	120	103	132
6	158	128	143	140	144	0	244	132	140	116
7	129	140	125	146	135	138	0	237	116	135
8	134	168	145	134	142	139	145	0	244	128
9	105	122	129	138	123	143	114	119	0	237
10	138	108	131	136	121	113	144	243	134	0

图 5.2.1 各区域需求数据矩阵 N

	1	2	3	4	5	6	7	8	9	10
1	0	229	95	101	134	105	92	113	87	96
2	122	0	222	115	116	120	115	121	113	137
3	116	127	0	230	97	107	112	113	140	116
4	110	98	141	0	247	134	118	117	113	116
5	117	122	110	107	0	248	132	115	100	113
6	133	119	131	126	120	0	221	114	122	112
7	111	125	116	130	112	120	0	231	97	110
8	124	146	121	130	129	132	134	0	237	122
9	99	112	109	121	103	134	104	110	0	232
10	122	94	112	126	118	101	132	225	121	0

图 5.2.2 各区域实际数据矩阵 R

比较上两幅图，现在图 5.2.2 的情况并非已有条件下最好的调度分配，因此本文比较需求矩阵和实际矩阵的距离，并通过最小二乘法最小化距离的平方和，寻找数据的最佳函数匹配。

5.2.2 距离矩阵相关模型

令需求矩阵为 N,实际矩阵为 R,距离矩阵为 X，其中

$$X_{ij} = N_{ij} - R_{ij} \qquad 0 \le i \le 10, 0 \le j \le 10$$

可得代价函数为

$$y = \sum_{i=1}^{10} \sum_{j=1}^{10} X_{ij}^2 (-1)^{1\{x_{ij}<0\}}$$

如图 5.2.1，在未优化的情况下，代价函数 Y=19790

下面将通过平均法优化矩阵

（1）先找出 X 中最大值最小值，分别为 max，min。

$$avg = \frac{\max + \min}{2}$$

$$\max = \max - avg$$
$$\min = \min - avg$$

（2）其次，找出大小仅次于 max，min 的值，重复进行上述操作。

（3）依次类推，经过一次优化，得出距离矩阵 X_1；将距离矩阵 X_1 加到实际矩阵 R 上，得到一次优化后的矩阵 R_1。

（4）经过 n 次优化后，得出距离矩阵 X_n，其代价函数 Y 已收敛并趋于稳定。此时将距离矩阵 X_n 加到上一次优化后的距离矩阵 R_{n-1} 上得出矩阵 R_n，如下图所示（图 5.2.3）。

	1	2	3	4	5	6	7	8	9	10
1	0	226.6250	105.6250	109.6250	131.6250	112.6250	101.6250	113.6250	98.6250	88.6250
2	121.7500	0	211.7500	115.7500	106.7500	112.7500	125.7500	113.7500	107.6250	126.7500
3	111.6250	118.7500	0	237.7500	96.7500	103.7500	112.7500	123.7500	144.7500	112.6250
4	102.7500	101.7500	134.7500	0	247.7500	130.7500	118.7500	127.7500	110.7500	105.7500
5	114.7500	119.7500	110.7500	102.7500	0	259.7500	124.7500	106.7500	89.7500	118.7500
6	144.7500	114.7500	129.7500	126.7500	130.7500	0	230.7500	118.7500	126.7500	102.7500
7	115.7500	126.7500	111.7500	132.7500	121.7500	124.7500	0	223.7500	102.7500	121.7500
8	120.7500	154.7500	131.7500	120.7500	128.7500	125.7500	131.7500	0	230.7500	114.7500
9	91.7500	108.7500	115.7500	124.7500	109.7500	129.7500	100.7500	105.7500	0	223.7500
10	124.7500	94.7500	117.7500	122.7500	107.7500	99.7500	130.7500	229.7500	120.7500	0

图 5.2.3 各区域实际最优解

5.2.3 车辆模拟优化结果

将图 5.2.3 横向叠加除以总数得出各自相应的比例，按此比例分配 1000 辆共享单车，可得到各区域放置共享单车数量最优解，如下图（5.2.4）

区域	1	2	3	4	5	6	7	8	9	10
分配车数	93	98	100	101	99	105	101	108	95	99

图 5.2.4 各区域放置共享单车数量最优图

5.3 问题三的求解

利用问题二通过最小二乘法建立的车辆调度模型，通过代价函数 Y，如果代

价函数得到最优解的话，说明各个区域的满足程度已经趋于最优，那么就可以用代价函数 Y 来近似的作为各区域所需共享单车的满足程度的度量指标；若增加 100 辆单车，为了使其投放最优，可以通过边缘密度函数（出发和到达），对各区域实际数据矩阵 R（如图 5.2.2）进行处理。

$$H_i=\sum_{j=1}^{10} R_{ij}$$

$$Sum=\sum_{i=1}^{10} H_i$$

$$P_i=\frac{H_i}{Sum}$$

$$P_{ij}=\frac{R_{ij}}{H_i}$$

$$Part_{ij}=P_{ij}P_i$$

通过上述的公式对每一行的数值取和，得到每一行的数据和 Hi ，即得到每一个区域去往其他各个区域的总的车辆次数，然后取每行的总值得到矩阵 R 的实际的数据总和 Sum，即一天之内车辆的实际需求的次数，于是就可以得到每个区域在一天之内的需求占所有区域总需求的百分比，然后又计算得到某一个区域到其他某个区域占该区域一天需求的百分比，这样循环就可以得到每一个区域到其他所有区域的百分比（如图 5.3.1）；根据以往统计，1000 辆车一天之内被使用 11649 次，所以 100 辆车大概是被调用 1164 次，把这 1164 次按 Partij 的比例进行发放调度次数，

$$R'_{ij}=R_{ij}+Part_{ij}\times 1164$$

得到一个调整过了的新的实际需求的矩阵 R',对该矩阵重新进行计算

$$X_{ij}=N_{ij}-R'_{ij}$$

继续对 X 进行问题二中描述的平均法优化该矩阵，得到如图 5.3.2 的表，可以看出，通过该方法进行投放，可以使代价函数 Y 降到 371.2239，据此发现如果加入新的 100 辆车后，用户的需求已基本得到了满足。

	1	2	3	4	5	6	7	8	9	10
1	0	0.0197	0.0082	0.0087	0.0115	0.0090	0.0079	0.0097	0.0075	0.0082
2	0.0105	0	0.0191	0.0099	0.0100	0.0103	0.0099	0.0104	0.0097	0.0118
3	0.0100	0.0109	0	0.0197	0.0083	0.0092	0.0096	0.0097	0.0120	0.0100
4	0.0094	0.0084	0.0121	0	0.0212	0.0115	0.0101	0.0100	0.0097	0.0100
5	0.0100	0.0105	0.0094	0.0092	0	0.0213	0.0113	0.0099	0.0086	0.0097
6	0.0114	0.0102	0.0112	0.0108	0.0103	0	0.0190	0.0098	0.0105	0.0096
7	0.0095	0.0107	0.0100	0.0112	0.0096	0.0103	0	0.0198	0.0083	0.0094
8	0.0106	0.0125	0.0104	0.0112	0.0111	0.0113	0.0115	0	0.0203	0.0105
9	0.0085	0.0096	0.0094	0.0104	0.0088	0.0115	0.0089	0.0094	0	0.0199
10	0.0105	0.0081	0.0096	0.0108	0.0101	0.0087	0.0113	0.0193	0.0104	0

图 5.3.1 各路径所占比例图

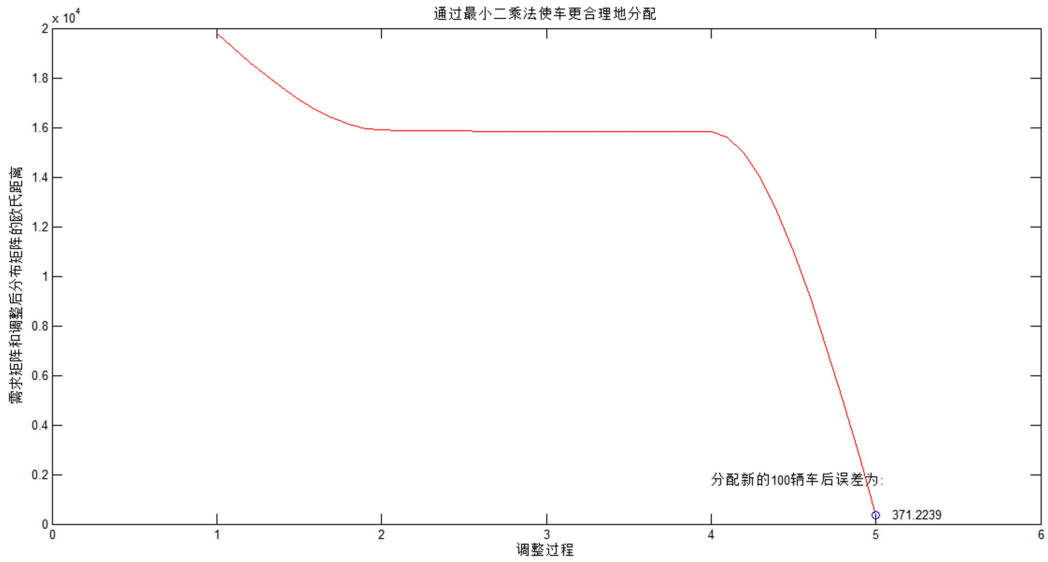

图 5.3.2 误差曲线图

5.4 问题四的模型建立与求解

5.4.1 数据收集及预处理

通过查阅收集数据，并对数据进行三次样条插值，根据插值画出的图像，估测单车投放量和打车人次的关系的大致形式为 y = a*exp(-c/x) + d*x^2 + e*x+f(1000 <= x <= 5000)。
之后用 lsqcurvefit()函数对已有数据进行拟合，得到最终的拟合函数为：

$$y = 22177e^{-\frac{451}{x}} + 0.00076x^2 - 7.8x + 9599$$

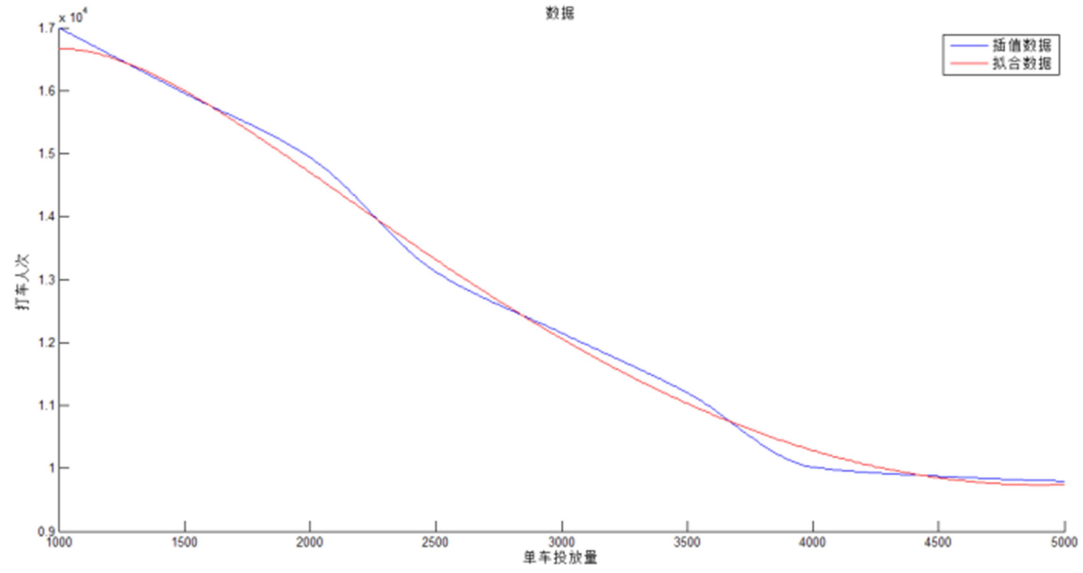

图 5.4.1　单车投放量-打车人次关系图

5.4.2 数据模型分析

附件 1 中数据统计可知：当单车投放量为 1000 时，单车使用人次约为 11600；又如图 5.4.1，此时打车人次约为 17000；可估算此地区有此两种用车需求的总人口约为 28600；进而，可以由公式

$$\eta=\frac{S_i}{S}$$

可估算出两种交通方式的市场占有率。
其中：

η：市场占有率　i：选择的出行方式　S：出行总人口

如下图

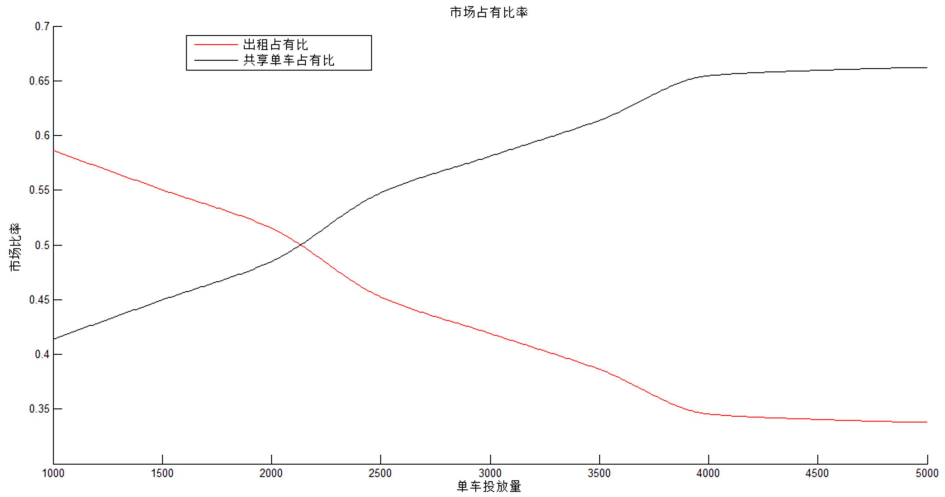

图 5.4.2 单车投放量对两种交通方式市场占有率的影响图

可得：

（1）随着共享单车投放量的增加，打车人次呈下降趋势，市场占有率从原先的 57.5%降至 33.3%

（2）随着共享单车投放量的增加，打车人次的下降趋势逐渐变缓。

以上现象可说明：共享单车对打车市场具有一定的影响，但此种影响并不是无-限制的。

为了进一步研究共享单车的投入对打车市场的影响，我们作出图 5.4.1 的导数图像，如下图。

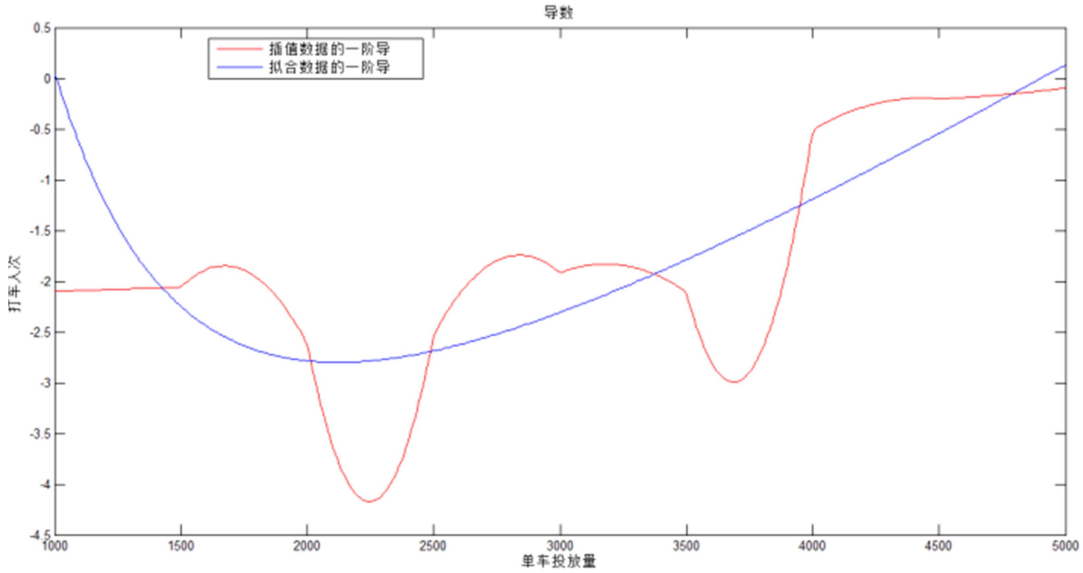

图 5.4.2　单车投放量-打车人次导数关系图

从图 5.4.2，可以观察到：当单车投放量增加到 4500 后，插值数据和拟合数据的导数都趋向 0，波动范围约为[-0.5,+0.5]。依此，可以得到以下结论：打车市场有其固定用户，人次大约占总人口的 33.3%，约 10000 人。他们几乎不会因单车市场的竞争而影响其选择。

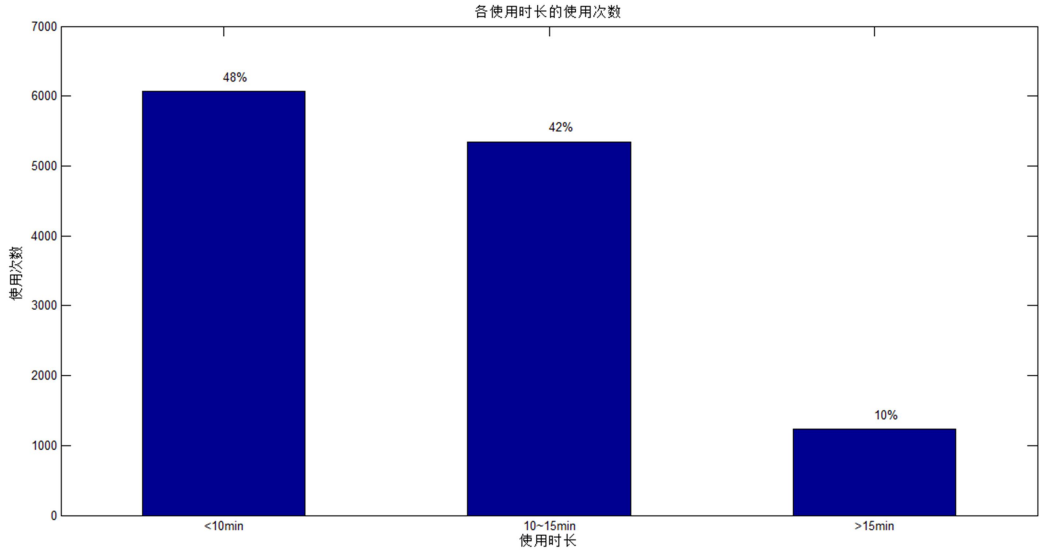

图 5.4.3　单车使用时长比例图

设用户骑单车速度为 15km/h

（1）短距离出行即 S<2.5km 约占总出行的 48%

（2）中距离出行即 2.5<S<3.75km 约占总出行的 42%

（3）长距离出行即 S>3.75km 约占总出行的 10%

由此可见，短、中距离出行共享单车的优势明显。

5.4.3 市场模型模拟结论

（1）可由图 5.4.2 观测出在投放大约 4100 辆单车之后继续投放的使用率大不如

之前，因此建议投放的单车数目不超过 4100 辆。
（2）由图 5.4.3 可见，短、中距离出行共享单车的优势明显，而长途旅行的旅客依然会选择出租。同时，根据此，我们建议：将相邻的单车存放站点的距离设为不超过 3.75km，这样会大大增加单车的被使用的次数。

6. 总结

6.1 优点

本模型基于最小二乘法的误差分析，采用均分法寻找最优解，迭代步数少，降低了模型的开销。

本模型不依赖于某特定时间，较为灵活、可重用性高。

6.2 缺点

本模型忽略了对车辆进行调度时的路程、时间、经费开销，若结合最短路径问题考虑送货，可以提升模型的鲁棒性。

6.3 推广

本模型主要依据车辆的分布进行建模，但可用对象不仅仅限于车辆的摆放，其中的距离的推算、车辆的分布可以引申到诸如公共设施建设等问题上。

7. 参考文献

[1]焦玉涛.公共自行车借还特性分析及调度模型研究[D].东南大学,2015
[2]卢蝶.中小城市公共交通与个人交通服务属性及竞争关系研究[D].华南理工大学，2013
[3]管娜娜.公共自行车调度路径优化问题研究[D].西南交通大学,2015
[4]王恺.公共自行车调度优化研究[D].西南交通大学,2016
[5]刘凯强.”互联网+”范式下出租车行业利益博弈及发展路向[N].太原理工大学学报，2016 年 4 月第 34 卷第二期

附录

附录 1：main_model.m 主模型代码，可得信息包括所有的柱状图、饼状图、表格数据，基于 Matlab-2014a

```
clear, clc;
close all;
%% Get the data formatted
data = xlsread('all_data\data1.xlsx');
[h, w] = size(data);
for i = 1 : h
    if isnan(data(i, 3))
        data(i, 3) = data(i, 2);
    end
end
data(:, 2) = data(:, 3);
data(:, 3) = data(:, 4);
data(:, 4) = data(:, 2) - data(:, 1);
counter = 0;
maxer = counter;
for i = 1 : h
    if isnan(data(i, 3))
        p = data(i, 1);
        q = data(i, 2);
        if counter > maxer
            maxer = counter;
        end
        counter = 0;
        continue;
    else
        counter = counter + 1;
        data(i, 5) = p;
        data(i, 6) = q;
    end
end

%% Do the statistics on src->dst concerning direction_calculation
% i 行 j 列为从 i 地到 j 地
all_src_to_dst = [zeros(1, 10); zeros(1, 10); zeros(1, 10); zeros(1, 10); ...
    zeros(1, 10); zeros(1, 10); zeros(1, 10); zeros(1, 10); zeros(1, 10); zeros(1, 10); ];
t_table = zeros(1000, maxer+2);
```

```
line_idx = 0;
t = 1;
for i = 1 : h
    if isnan(data(i, 3))
        line_idx = line_idx + 1;
        t = 1;
        t_table(line_idx, t) = data(i+1, 6);
        continue;
    else
        t = t + 1;
        t_table(line_idx, t) = data(i, 3);
    end
end
for i = 1 : 1000
    t = size(t_table(i, :));
    for j = 1 : t(2)
        if t_table(i, j+1) == 0
            break;
        end
        all_src_to_dst(t_table(i, j), t_table(i, j+1)) = ...
            all_src_to_dst(t_table(i, j), t_table(i, j+1)) + 1;
    end
end
all_src_to_dst_initial = all_src_to_dst;
%% plot the pies
for k = 1 : 10
    explode = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0];
    if k < 10
        explode(k+1) = 1;
    else
        explode(8) = 1;
    end
    explode(k) = 1;
    subplot(2, 5, k)
    pie(all_src_to_dst(k, :),explode,{[num2str(k) '->1'], [num2str(k) '->2'], ...
        [num2str(k) '->3'], [num2str(k) '->4' ], [num2str(k) '->5'], ...
        [num2str(k) '->6'], [num2str(k) '->7'], [num2str(k) '->8'], ...
        [num2str(k) '->9'], [num2str(k) '->10'],})
    title(['出发地为', num2str(k)]);
end

%% Do the statistics on src<->dst concerning the average_time-on-travelling
% i 行 j 列为从 i 地到 j 地
travelling_time = [zeros(1, 10); zeros(1, 10); zeros(1, 10); zeros(1, 10); ...
```

```
        zeros(1, 10); zeros(1, 10); zeros(1, 10); zeros(1, 10); zeros(1, 10); zeros(1,
10); ];
    for i = 1 : h
        if isnan(data(i, 3))
            src_idx = data(i, 2);
        else
            travelling_time(src_idx, data(i, 3)) = travelling_time(src_idx, data(i, 3))
+ data(i, 4);
        end
    end
    [tt_h, tt_w] = size(travelling_time);
    for i = 1 : tt_h
        for j = i : tt_w
            travelling_time(i, j) = travelling_time(i, j) + travelling_time(j, i);
        end
    end
    for i = 1 : tt_h
        travelling_time(i, 1 : i - 1) = 0;
    end
    all_src_to_dst_copy = all_src_to_dst;
    [sd_h, sd_w] = size(all_src_to_dst_copy);
    for i = 1 : sd_h
        for j = i : sd_w
            all_src_to_dst_copy(i, j) = all_src_to_dst_copy(i, j) +
all_src_to_dst_copy(j, i);
        end
    end
    for i = 1 : sd_h
        all_src_to_dst_copy(i, 1 : i - 1) = 0;
    end
    for i = 1 : tt_h
        for j = i : tt_w
            travelling_time(i, j) = travelling_time(i, j) / all_src_to_dst_copy(i, j);
        end
    end
    travelling_length = travelling_time .^ -1;
    travelling_length = travelling_length * 10000;
    travelling_length = round(travelling_length);
    t = travelling_length';
    for i = 1 : 10
        travelling_length(i, 1 : i - 1) = t(i, 1 : i - 1);
    end
    %% Divide the time
    % 360 ~ 1440之间分割: 360 | 480 | 600 | 720 | 840 | 960 | 1080 | 1200 | 1320
```

| 1440

```
    for i = 1 : h
        if isnan(data(i, 3))
            data(i, 1 : 4) = NaN;
        end
    end

    time_table = zeros(1000, maxer);
    t = 1;
    bike_idx = 0;
    for i = 1 : 1 : h
        if isnan(data(i, 3))
            bike_idx = bike_idx + 1;
            t = 1;
            continue;
        end
            time_table(bike_idx, t) = data(i, 1);
            t = t + 1;
            time_table(bike_idx, t) = data(i, 2);
            t = t + 1;
    end
    % 计算每个车子的辗转的地点数
    counter = zeros(1000, 1);
    t_count = 0;
    for i = 1 : h
        if isnan(data(i, 3))
            t_count = t_count + 1;
        end
        counter(t_count) = counter(t_count) + 1;
    end

    % 行数是"车站", 列数是分割时刻的序号.
    time_divide = [360 : 15 : 1440];
    % 即: 6:00, 8:00, 10:00, 12:00, 14:00, 16:00, 18:00, 20:00, 22:00
    [td_h, td_w] = size(time_divide);
    tmp_bike_place = zeros(10, td_w);
    arr = zeros(td_w-1, 1);
    for k = 1 : td_w - 1
        cursor = 0;
        for i = 1 : 1000
            t = find(time_table(i, :)>time_divide(k) &
time_table(i, :)<time_divide(k+1));
            if isempty(t)
```

```
                continue;
            end
            t_size = size(t);
            if t_size == 1
                t = t_size(1);
            else
                t = t_size(2);
            end
            arr(k) = arr(k) + ceil(t/2);
        end
end

figure;
res = sum(tmp_bike_place(:,:));
plot(time_divide(1:td_w-1), arr', 'b');
set(gca, 'xtick', [360 : 120 : 1440])
figure
bar(arr, 0.3)
set(gca, 'xticklabel', {360 : 120 : 1440})

%% 第二题
% 数据(附件二)
need = [
  0 240 119 123 145 126 115 127 112 102;
  135 0 225 129 120 126 139 127 121 140;
  125 132 0 251 110 117 126 137 158 126;
  116 115 148 0 261 144 132 141 124 119;
  128 133 124 116 0 273 138 120 103 132;
  158 128 143 140 144 0 244 132 140 116;
  129 140 125 146 135 138 0 237 116 135;
  134 168 145 134 142 139 145 0 244 128;
  105 122 129 138 123 143 114 119 0 237;
  138 108 131 136 121 113 144 243 134 0 ];

t_matrix = [
0, 216, 86, 98, 125, 103, 86, 107, 79, 94;
108, 0, 214, 109, 112, 116, 113, 114, 109, 129;
107, 116, 0, 215, 96, 104, 110, 111, 97, 114;
109, 96, 126, 0, 238, 121, 117, 114, 111, 113;
115, 121, 110, 104, 0, 239, 126, 112, 97, 111;
130, 115, 126, 124, 111, 0, 216, 110, 114, 108;
109, 122, 115, 114, 109, 119, 0, 227, 94, 105;
121, 134, 117, 121, 122, 126, 124, 0, 197, 116;
97, 111, 107, 116, 101, 129, 101, 106, 0, 224;
```

```
114, 90, 108, 118, 106, 96, 124, 208, 116, 0
];

the_diff = [
0,1.5,-9.5,-8.0,1.5,-7.5,-8.5,-1.5,-11.0,8.5;
-0.5,0,11.0,-1.5,10.0,8.5,-9.5,8.0,5.0,11.0;
4.5,9.5,0,-7.5,-0.5,3.5,-1.5,-10.0,-4.5,3.0;
8.0,-3.5,6.0,0,-1.5,3.0,-1.5,-10.0,1.5,11.0;
1.5,1.5,-2.0,4.5,0,-11.0,7.5,9.5,11.0,-6.0;
-11.0,4.5,0.5,-2.0,-10.0,0,-8.5,-4.5,-4.5,10.0;
-4.5,-2.5,4.5,-3.0,-9.0,-4.5,0,7.5,-6.0,-11.0;
2.5,-8.0,-10.5,10.0,-0.5,6.0,1.0,0,6.0,7.0;
7.0,2.0,-7.0,-4.0,-7.0,4.5,2.0,4.0,0,9.0;
-3.0,-2.0,-6.0,2.0,10.5,0.5,0.5,-5.0,-1.0,0
];

the_diff2 = [
0,0.75,-1.25,-0.5,0.75,0.0,-1.25,0.75,-0.5,-1.25;
0.75,0,-0.75,0.75,-0.75,-1.25,-1.25,-0.75,0.5,-0.75;
0.0,-1.25,0,-0.25,0.75,-0.25,0.75,-0.75,-0.25,0.5;
-0.75,-0.25,0.25,0,0.75,0.25,0.75,-0.75,0.75,-0.75;
0.75,0.75,1.25,-0.25,0,-0.75,-0.25,-1.25,-0.75,0.25;
-0.75,-0.25,0.75,1.25,-0.75,0,-1.25,-0.25,-0.25,-0.75;
-0.25,0.75,-0.25,0.25,-0.75,-0.25,0,-0.25,0.25,-0.75;
0.75,-0.75,-0.25,-0.75,0.75,0.25,1.25,0,0.25,0.25;
0.25,1.25,0.25,0.25,0.25,-0.25,1.25,0.25,0,-0.75;
0.25,1.25,0.25,1.25,-0.25,0.75,0.75,0.25,1.25,0
];

the_diff3 = [
0,0.125,0.125,-0.125,0.125,-0.125,0.125,0.125,-0.125,0.125;
0.0,0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,-0.125,0.0;
-0.125,0.0,0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,-0.125;
0.0,0.0,0.0,0,0.0,0.0,0.0,0.0,0.0,0.0,0.0;
0.0,0.0,0.0,0.0,0,0.0,0.0,0.0,0.0,0.0,0.0;
0.0,0.0,0.0,0.0,0.0,0,0.0,0.0,0.0,0.0,0.0;
0.0,0.0,0.0,0.0,0.0,0.0,0,0.0,0.0,0.0,0.0;
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0,0.0,0.0,0.0;
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0,0.0,0.0;
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0,0
];
%% 计算各地每个时间段的需求和实际分布
arr_ratio = zeros(size(arr));
for i    = 1 : size(arr_ratio)
```

```
        arr_ratio(i) = arr(i) / sum(arr);
end
% 需求
zone_need = sum(need(:, :)');
need_time_dist = zeros(size(arr));
for j = 1 : 10
        for i    = 1 : size(arr_ratio)
                need_time_dist(i) = arr_ratio(i) * zone_need(j);
        end
end
figure
bar(need_time_dist, 0.3)
xlabel('时间段');
ylabel('距离需求的差距');
set(gca,'xticklabel',{360 : 15 : 1440});
title('需求分布');
% 实际分布
zone_dist = sum(all_src_to_dst(:, :)');
real_time_dist = zeros(size(arr));
for j = 1 : 10
        for i    = 1 : size(arr_ratio)
                real_time_dist(i) = arr_ratio(i) * zone_dist(j);
        end
end
figure
bar(real_time_dist, 0.3)
xlabel('时间段');
ylabel('实际分布量');
set(gca,'xticklabel',{360 : 15 : 1440});
title('实际分布');
figure
bar(need_time_dist - real_time_dist, 0.3)
xlabel('时间段');
ylabel('距离需求的差距');
set(gca,'xticklabel',{360 : 135 : 1440});
title('需求减去实际之差的分布');


%% 最小二乘法改进
res_qe = [];
qe_idx = 1;
res_qe(qe_idx) = get_quar_error(all_src_to_dst, need);
qe_idx = qe_idx + 1;
all_src_to_dst = all_src_to_dst - the_diff;
```

```
    res_qe(qe_idx) = get_quar_error(all_src_to_dst, need);
    qe_idx = qe_idx + 1;
    all_src_to_dst = all_src_to_dst - the_diff2;
    res_qe(qe_idx) = get_quar_error(all_src_to_dst, need);
    qe_idx = qe_idx + 1;
    all_src_to_dst = all_src_to_dst - the_diff3;
    res_qe(qe_idx) = get_quar_error(all_src_to_dst, need);
    qe_idx = qe_idx + 1;
    % 根据最终的 all_src_to_dst 确定车辆的分布配比
    final_sum = sum(all_src_to_dst')';
    final_sum_sum = sum(final_sum);
    ratio_final_sum = final_sum ./ final_sum_sum;
    final_all_src = round(ratio_final_sum .* 1000); % 最终确定的分布
    %% 按比例分配新的 100 辆车
    initial_dist = sum(all_src_to_dst_initial')' ./ sum(sum(all_src_to_dst_initial));
    sum_each_src = sum(all_src_to_dst_initial')';
    ratio_in_each_src = zeros(10);
    for i = 1 : 10
        for j = 1 : 10
            ratio_in_each_src(i, j) = initial_dist(i) .* (all_src_to_dst_initial(i, j) /
sum_each_src(i));
        end
    end
    add_matrix = ratio_in_each_src .* 1165;
    t = all_src_to_dst_initial + add_matrix;
    error = get_quar_error(t, need);
    %%
    res_qe(qe_idx) = error;
    sz_res_qe = size(res_qe);
    xx = [1 : 0.1 : sz_res_qe(2)];
    yy=interp1([1:sz_res_qe(2)],res_qe,xx,'cubic');
    figure

    plot(xx, yy, 'r');
    title('通过最小二乘法使车更合理地分配')
    xlabel('调整过程')
    ylabel('需求矩阵和调整后分布矩阵的欧氏距离');
    text(xx(end)-1, yy(end)'+1400, '分配新的 100 辆车后误差为: ');
    text(xx(end)+0.1, yy(end)', num2str(yy(end)));

    %%
    hold on
    plot(qe_idx, error, 'o');
    axis([0, 6, 0, 20000])
```

```
%% 求各个用车时长的使用次数
t_arr = zeros(3, 1);
for i = 1 : h
    if data(i, 4) < 10
        t_arr(1) = t_arr(1) + 1;
    else
        if data(i, 4) < 15
            t_arr(2) = t_arr(2) + 1;
        else
            t_arr(3) = t_arr(3) + 1;
        end
    end
end
t_arr_porpotion = t_arr ./ sum(t_arr);
figure
bar(t_arr, 0.5)
title('各使用时长的使用次数');
xlabel('使用时长');
ylabel('使用次数');
set(gca, 'xticklabel', {'<10min', '10~15min', '>15min'})
for i = 1 : size(t_arr)
    text(i, t_arr(i)+200, strcat(num2str(round(t_arr_porpotion(i)*100)), '%'));
end
```

附录 2：qu_zhuduijiaoxian.m，可去除矩阵的主对角线并进行拼接，基于 Matlab-2014a

```
function [ res ] = qu_zhuduijiaoxian( A )
    B=[];
    [m,n]=size(A);
    for i=1:m
        B=[B;A(i,[1:i-1 i+1:n])];
    end
    t = B';
    res = t(1: 90)';
end
```

附录 3：question4.m 用于解决问题四，基于 Matlab-2014a

```
clear, clc;
close all;
data_bike = [1000 1500 2000 2500 3000 3500 4000 4500 5000 ];
data_cub = [17000 15960 14940 13120 12150 11200 10017 9875 9796 ];
```

```
xx = [1000 : 10 : 5000];
x = data_bike;
result = data_cub;
yy=interp1(x,result,xx,'cubic');
c0 = [0, 0, 0, 0, 0, 0];
c = lsqcurvefit (@the_formula, c0, xx, yy);
c0 = c; %  最终确定的参数
figure;
x_fit = xx;
res_fit = ( c(1) - c(2) ) .* exp( ( -c(3) .* ( 1 ./ x_fit ) ) ) + c(5) .* x_fit.^2 + c(6).*x_fit
+ c(4);
hold on;
plot(xx, yy, 'b', x_fit, res_fit', 'r');
legend('插值数据', '拟合数据');
title('数据');
xlabel('单车投放量');
ylabel('打车人次');
figure
hold on
t = yy ./ 29000;
plot(xx, t, 'r' );
hold on
plot(xx, 1-t, 'k');
title('市场占有比率')
xlabel('单车投放量');
ylabel('市场比率');
hleg1 = legend('出租占有比', '共享单车占有比');
set(hleg1, 'Position', [.13,.85,.4,.05]);
figure
z=gradient(yy)./gradient(xx);
plot(xx,z,'r')
hold on
z=gradient(res_fit)./gradient(x_fit);
plot(x_fit,z,'b')
hleg1 = legend('插值数据的一阶导', '拟合数据的一阶导');
set(hleg1, 'Position', [.13,.85,.4,.05]);
title('导数');
xlabel('单车投放量');
ylabel('打车人次');
```

附录 4：get_quar_error.m 用于求平方和，基于 Matlab-2014a

```
function [ t_r ] = get_quar_error( A, B )
    % A 是输入, B 是参照
    t_r = 0;
```

```
        for i = 1 : 10
            for j = 1 : 10
                t = A(i, j) - B(i, j);
                if t < 0
                    t_r = t_r + t * t;
                else
                    t_r = t_r - t * t;
                end
            end
        end
    end
```

附录 5：the_formula**.m 用于拟合曲线，基于 Matlab-2014a**

```
function [ res ] = the_formula( c, x )
    res = ( c(1) - c(2) ) .* exp( ( -c(3) .* ( 1 ./ x ) ) ) + c(5).*x.^2 + c(6).*x + c(4);
end
```

附录 5：averaging**.m 用于均分化，基于 JAVASE**

```
package mathorcup;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.LinkedHashSet;
import java.util.Scanner;
import java.util.Set;

public class text2 {

    public static void main(String[] args) throws NumberFormatException,
IOException {
        // TODO Auto-generated method stub

        BufferedReader reader=new BufferedReader(new
FileReader("D://java_file_lesson//6.txt"));

        ArrayList<data> set=new ArrayList<>(90);
        int i=90;
        int h=1,z=1;
        int zero=1;
        while(i>0){
            String s1,s2,s3;
```

```
        int x=0,y=0;

        double value = 0;
        if(z<=9){
            x=h;
        }else{
            h++;
            x=h;
        }
        if(z<=9){

            y=z;
            z++;
        }else{
            z=1;
            y=z;
            z++;
        }
        if((s3=reader.readLine())!=null){

            String s31=s3.trim();
            value=Double.parseDouble(s31);
        }
        data d1=new data(value, x, y);
        set.add(d1);
        i--;
    }
    for(int j=0;j<89;j++){
        for(int k=0;k<89-j;k++){
            if(set.get(k).value>set.get(k+1).value){
                data d=set.get(k);
                set.set(k, set.get(k+1));
                set.set(k+1,d);
            }
        }
    }
    for(int m=0;m<=44;m++){
        double ave=(set.get(m).value+set.get(89-m).value)/2;
        set.get(m).setValue(ave-set.get(m).value);
        set.get(89-m).setValue(ave-set.get(89-m).value);
    }
    for(int j=0;j<89;j++){
        for(int k=0;k<89-j;k++){
            if(set.get(k).x>set.get(k+1).x){
```

```
                        data d=set.get(k);
                        set.set(k, set.get(k+1));
                        set.set(k+1,d);
                    }
                    if(set.get(k).x==set.get(k+1).x){
                        if(set.get(k).y>set.get(k+1).y){
                            data d=set.get(k);
                            set.set(k, set.get(k+1));
                            set.set(k+1,d);
                        }
                    }
                }
            }
            int sum=1;
            for(int n=0;n<90;n++){
//              System.out.print(set.get(n).x+" ");
//              System.out.print(set.get(n).y+" ");
//              System.out.println(set.get(n).value);
                if(sum<=9){
                    if(sum==9){

if(n==0||n==10||n==20||n==30||n==40||n==50||n==60||n==70||n==80){
                            System.out.print("0,");
                        }
                        if(n==89){
                            System.out.print(set.get(n).value+","+"0;");
                            break;
                        }
                        System.out.print(set.get(n).value+";");

                    }
                    else{

if(n==0||n==10||n==20||n==30||n==40||n==50||n==60||n==70||n==80){
                            System.out.print("0,");
                        }
                        System.out.print(set.get(n).value+",");
                    }
                    sum++;
                }else{

                    sum=1;
                    System.out.println();
```

```
if(n==0||n==10||n==20||n==30||n==40||n==50||n==60||n==70||n==80){
                        System.out.print("0,");
                    }
                    System.out.print(set.get(n).value+",");
                    sum++;
                }

            }

        }

}
class data{
    public double value;
    public int x;
    public int y;
    public data(double value,int x,int y){
        this.value=value;
        this.x=x;
        this.y=y;
    }
    public void setValue(double a){
        value=a;
    }
}
```