“互联网+”时代的出租车资源配置模型

摘　要

“打车难”一直是近年来社会关注的问题，随着“互联网+”时代的到来，建立在手机和移动互联网基础之上的打车服务平台实现了乘客与出租车之间信息互通，也影响着广大人民的打车问题。本文就出租车的资源配置现状，以及出租车公司补贴方案对打车难易程度的影响进行讨论与分析，进而推出更优的补贴方案。

针对问题（1），根据滴滴、快的智能出行平台搜集了 2015 年 9 月 9 日成都市 1 点至 24 点出租车的分布情况和乘客的需求情况，根据数据，绘制图像，得到成都市乘客需求的高峰期发生在早上 8-9 点和晚上 19 点；另外，为查看各时段的出租车和乘客的匹配情况，我们按每个时点计算了出租车分布和乘客需求的均值，并进行描图和分析。在此基础上，建立匹配模型：引入“最近邻”思想，以乘客为中心，按一定的半径画圆，以圈内出租车的总数量和乘客需求量来定义匹配度，并构建度量指标$mN_i(t)$，根据该指标得到打车由难到易所需的最小半径，即得出打车难易程度。根据度量指标和描出的图形，得出了不同“时空”出租车资源配置的“供求匹配”程度：从空间上，成都东、双流北、以及都江堰中心地带较易出现打车难问题；从时间上，打车较难的时间点集中在 8-10 点，下午 16 点，傍晚 19 点。

针对问题(2)，我们搜集了滴滴打车和快的打车的补贴方案数据，出租车司机和乘客在使用打车软件后，出租车的空驶率、乘客的等待时间和出租车司机与乘客的经济收益变化等数据；根据分析，补贴方案的存在，能充分调动司机和乘客使用软件的积极性，使得使用软件的人数持续增加，这既缩短了乘客打车等待的时间，也减少了司机的空驶率，故可以有效的缓解打车难的问题。但随着补贴的降低，乘客和司机对打车软件使用的激情开始降低，尤其在高峰时段，司机不需要使用软件接单，其空驶率也很低，而乘客即便使用打车软件，也会在高峰期打不到车，即出现打车难的问题；另外，由于老年人群不会使用打车软件，所以无论什么补贴政策，都不能缓解他们打车难的问题。

针对问题（3），首先对问题（2）所搜集的数据进行处理，计算出乘客的累计补贴和司机的累计补贴，利用 SPSS 软件，以乘客累计补贴和司机累计补贴为自变量，以日均订单量/用户数的百分比为因变量进行 S 曲线拟合，且效果非常显著，并得到相应的经验公式。在此基础上提出新的补贴方案 1：低峰时段出行并成功打到车的给予一定的现金或积分奖励；但根据前面的 S 曲线模型进行模拟，发现影响率稳定在 5%左右，基本不能缓解打车难问题，接着我们分析了不能缓解的根本原因，并在此基础上提出了补贴方案 2：鼓励出租车前往人流密集处，打车软件可以通过大数据的实时分析，告诉司机哪个区域打车需求大，通过一定的现金或积分奖励，鼓励司机抢人流密集处的定单。通过举例，论证了方案 2 能较好的缓解打车难问题，得出方案 2 是可行的。

关键词：匹配度　打车软件　打车难易程度　补贴方案

一．问题重述

城市人们在出行过程中，出租车是其重要的交通工具之一，但随着城市人口密度和车密度的增加，对交通运行带来了不便，所以即便一个城市的出租车数量大于需求量，也会出现打车难的问题。为了缓解这个问题，加上“互联网+”时代的到来，有多家公司比如滴滴、快的等依托移动互联网建立了打车软件服务平台，乘客可以事先预约打车，司机可以根据预约地点和当时所处位置，进行信息互通，为了争取用户使用，各家公司推出了不同的补贴方案。问题要求搜集相关数据，完成如下问题：

（1）根据搜集的数据，构造匹配指标，并分析在不同时间和不同空间的出租车资源的“供求匹配”程度；

（2）分析滴滴、快的等打车软件公司的出租车补贴方案是否对“缓解打车难”有帮助，并得出结论；

（3）根据问题（1）和（2）的结果，设计合理的补贴方案，并论证给出方案的合理性，提出建议。

二．模型假设

1. 假设通过滴滴、快的智能出行平台搜集 2015 年的 9 月 9 号 1 点至 24 点出租车分布数据和乘客需求量数据是可靠的；
2. 假设搜集的出租车分布数据当时全部都是空载；
3. 假设搜集的乘客打车的需求都不会因为主观或者客观因素的改变而改变；
4. 在问题（1）中计算邻域半径时，按理应该采用地图距离，最好是基于可达的地图距离，但是，由于数据难于得到，故而此处暂且考虑地图球面距离；
5. 在论证问题（3）的方案 2 时，假设所有的出租车分布点都有 1%的出租车进行迁移。

三．符号说明

符号	说明	符号	说明
p_{x_i}	第 i 个乘客数据点的经度	i	第 i 个乘客数据点
p_{y_i}	第 i 个乘客数据点的纬度	j	第 j 个出租车数据点
p_i	第 i 个乘客数据点的需求量	n	乘客数据点的个数
$p_{x_i}(t)$	第 t 个时点第 i 个乘客数据点的经度	m	出租车数据点的个数

$p_{y_i}(t)$	第 t 个时点第 i 个乘客数据点的纬度	$d(A,B)$	A,B 两点的球面距离
$p_i(t)$	第 t 个时点第 i 个乘客数据点的需求量	C	指定邻域半径
$T_{x_j}(t)$	第 t 个时点第 j 个出租车数据点的经度	$A_i(t)$	第 t 个时点第 i 个乘客数据点
$T_{y_j}(t)$	第 t 个时点第 j 个出租车数据点的纬度	$B_j(t)$	第 t 个时点第 j 个出租车数据点
$T_j(t)$	第 t 个时点第 j 个出租车数据点的出租车分布数量	$N_i(t)$	$A_i(t)$ 匹配度
LonA	点 A 的经度	$N_i^{c_k}(t)$	邻域半径取 C_k 时，t 时刻点第 i 个乘客数据点 $A_i(t)$ 的匹配度
LatA	点 A 的纬度	k	领域半径的个数
LonB	点 B 的经度	$mN_i(t)$	度量指标
LatB	点 B 的纬度	C_k	第 k 个领域半径
R_1	地球半径	N	城市人群
N_1	老年人群	N_2	一般人群
D_1	日均订单量	D_2	乘客累计补贴
D_3	司机累计补贴	D_4	用户数

四．问题分析

打车难包含出租车揽不到活，乘客打不到车这两层意思，也就是资源配置不合理。实际中，有的地方比如机场，扎堆了太多的出租车，因为机场往往离市区比较远，出租车载客去机场，往往不愿空车回来，可是航班到达有时间安排，乘客分布也是有疏有密，因此，往往导致出租车扎堆等待现象，最后能成功载上一个客人，一般要等待 1-2 小时。这就出现了非常严重的资源浪费问题。因此，搭建乘客与出租车之间的信息互通平台，合理分配出租车资源，有效缓解乘客和出租车的平均等待时间，就显得意义非凡。

问题(1) 的分析：为了衡量不同时空出租车资源“供求匹配”程度，首先根据滴滴、快的智能出行平台搜集了 2015 年 9 月 9 日成都市 1 点至 24 点出租车的分布情况和乘客的需求情况，根据这些数据，作图同时反应出租车位置和乘客位置的分布情况，得到乘客需求较大的高峰期出现在早上 8-9 点和晚上 19 点；为更清晰查看各个时段的出租车

和乘客的匹配情况，文中按每个时点计算了出租车分布和乘客需求的均值，并进行描图和分析。在此基础上，建立匹配模型：引入“最近邻”思想，以乘客为中心，按一定的半径画圆，以圈内出租车的总数量和乘客需求量来定义匹配度，并定义度量指标$mN_i(t)$，根据该指标找到打车由难到易所需的最小半径，得出打车难易程度。根据指标和描出的图形，得出了空间上，成都东，双流北，以及都江堰中心地带较易出现打车难问题。另外，我们还绘制了不同时点，乘客打车的难易度条形图，从时间上得到打车难易程度的分布。

问题(2) 的分析：我们搜集了滴滴打车和快的打车的补贴方案数据、滴滴打车在实施补贴方案以后，用户数和订单数的变化情况，出租车司机和乘客在使用打车软件后，出租车的空驶率，乘客的等待时间以及出租车司机和乘客的经济收益等数据；根据分析，由于补贴的存在，能充分调动乘客和司机使用软件的积极性，使得使用软件的人数会持续增加，这样既缩短了乘客打车时间的等待，也减少了司机的空驶率，故可以有效的缓解打车难的问题。但随着补贴的降低，乘客和司机对打车软件使用的激情开始降低，尤其在高峰时段，司机不需要使用软件接单，其空驶率也很低，而乘客即便使用打车软件，也会在高峰期打不到车，即出现打车难的问题；另外，老年人群占城市人口的 20%左右，由于他们不会使用打车软件，所以无论什么补贴政策，都不能缓解他们打车难的问题；以上这些问题为问题（3）中我们提出新的补贴方案提供依据。

问题(3) 的分析：首先对问题（2）所搜集的数据进行处理，计算出乘客的累计补贴和司机的累计补贴，利用 SPSS 软件，以乘客累计补贴和司机累计补贴为自变量，以日均订单量为因变量进行 S 曲线拟合，通过检验，效果非常显著，但若以用户量为因变量进行 S 曲线的拟合，拟合效果不显著，所以我们只考虑以日均订单量为因变量的随机模拟，并且把日均订单量对应到问题(1)中的乘客需求量，由于计数单位的不同，后面重新提出以日均订单量/用户数的百分比为因变量的 S 曲线进行拟合，通过检验且其效果非常显著，并得到相应的经验公式，在此基础上提出了新的补贴方案 1：低峰时段出行并成功打到车的给予一定的现金或积分奖励；但根据前面的 S 曲线模型进行模拟，发现影响率稳定在 5%左右，基本不能缓解打车难问题，接着我们分析了不能缓解的根本原因：无论哪个时段，出租车的分布规律基本上处于一个不变的趋势，大致都是那些区域，在此基础上，提出了补贴方案 2：鼓励出租车司机前往人流密集处，打车软件可以通过大数据的实时分析，告诉司机哪个区域打车需求量大，通过一定的现金或积分奖励，鼓励司机抢人流密集处的定单。以高峰时段的早上 8 点为例，根据计算和分析，论证了方案 2 能较好的缓解打车难问题，说明该方案可行。

五．数据搜集

数据主要来源于互联网，其中问题(1)的出租车分布数据和乘客打车需求数据主要来源于滴滴快的智能出行平台(网址：http://v.kuaidadi.com/)。

六． 模型的建立与求解

6.1． 问题（1）的建模与求解

我们搜集了成都市 2015 年 9 月 9 号从 1 点至 24 点，每隔 1 个小时的出租车分布数

据以及乘客打车需求数据，经过整理，数据如表 1-1 所示。

表 1-1 成都市 2015 年 9 月 9 号从 1 点至 24 点出租车和乘客分布数据

1点钟						2点钟						……	24点钟					
出租车分布			乘客			出租车分布			乘客				出租车分布			乘客		
经度	纬度	数量	经度	纬度	需求	经度	纬度	数量	经度	纬度	需求		经度	纬度	数量	经度	纬度	需求
104.27	30.562	8	121.45	31.331	282	104.08	30.623	90	104.11	30.661	4		104.07	30.631	114	103.89	30.818	5
103.62	31.008	3	104.08	30.632	116	104.08	30.612	72	103.65	30.991	3		104.09	30.658	110	104.03	30.565	11
104.17	30.834	4	103.64	30.978	69	104.08	30.634	55	103.84	30.678	3		104.03	30.645	73	104.42	30.866	7
104.17	30.719	5	104.05	30.633	69	104.03	30.649	55	104.04	30.644	4		104.08	30.635	71	103.66	30.972	4
104.32	30.602	8	104.07	30.632	68	103.97	30.581	55	104.07	30.654	9		104.06	30.639	71	103.99	30.665	5
103.66	31.001	17	103.84	30.697	67	104.08	30.654	55	104.26	30.55	5		104.06	30.627	71	104.17	30.805	3
104.02	30.662	8	104.02	30.646	67	104.09	30.648	54	104.2	30.828	3		104.07	30.651	71	103.66	30.997	13
103.82	30.685	5	104.1	30.644	67	104.04	30.637	54					103.85	30.695	70	104.28	30.566	11
103.97	30.763	5	104.09	30.67	66	103.84	30.693	54					104.05	30.689	69	103.64	30.999	10
103.91	30.799	7	103.65	30.994	66	104.02	30.67	54					104.1	30.656	69	103.65	30.856	3
104.12	30.675	11	104.1	30.631	66	104.04	30.693	54					103.64	31.002	68	104.06	30.544	5
103.63	30.993	11	104.07	30.656	65	104.07	30.661	53					104.06	30.637	67	104.26	30.892	7
103.73	30.591	4	104.1	30.655	64	104.07	30.644	52					103.65	30.995	66	104.09	30.649	3
103.9	30.807	3	104.07	30.652	64	104.02	30.687	52					104.1	30.665	66	103.93	30.573	3

6.1.1. 数据的描述分析

首先，针对表 1-1 所示的数据，我们绘制了二维散点图，散点图的点阵大小揭示了出租车数量或者乘客需求量的多少。

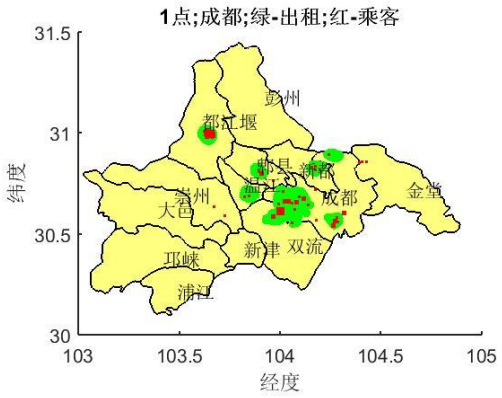


图 1-1 成都市 1 点钟的分布图

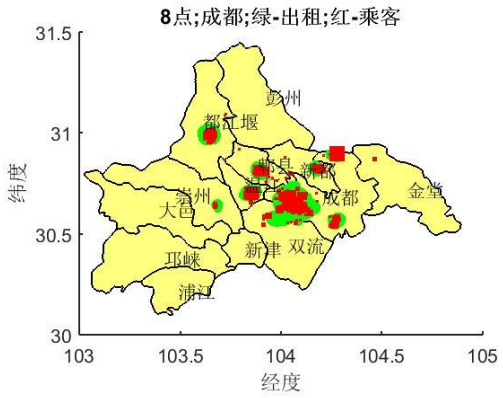
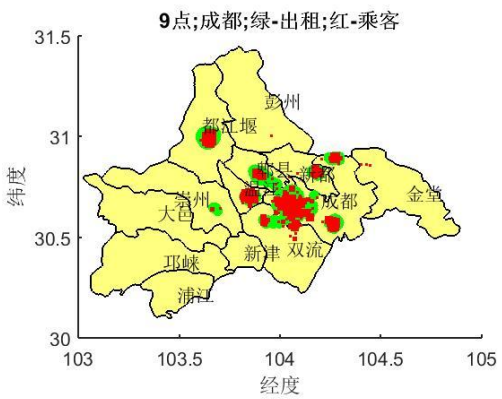


图 1-2 成都市 8 点钟的分布图



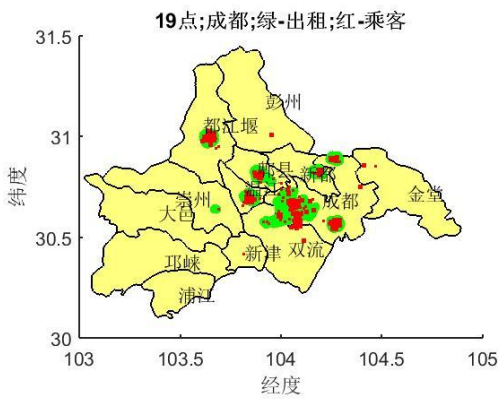


图 1-3 成都市 9 点钟的分布图　　图 1-4 成都市 19 点钟的分布图

图 1-1—图 1-4 绘出了 1 点，8 点，9 点和 19 点的成都市的出租车和乘客的分布图，全天 24 小时的分布图见附件 A。从这些图可以看出，无论何时，均存在一定的出租车分布和乘客需求，甚至午夜时分也存在着较大的乘客需求。通观全天，从这些图中，可以发现，乘客需求较大的高峰期发生在早上 8-9 点和晚上 19 点，因为此时红色区域所占的面积较大，较易发生“打车难”现象。

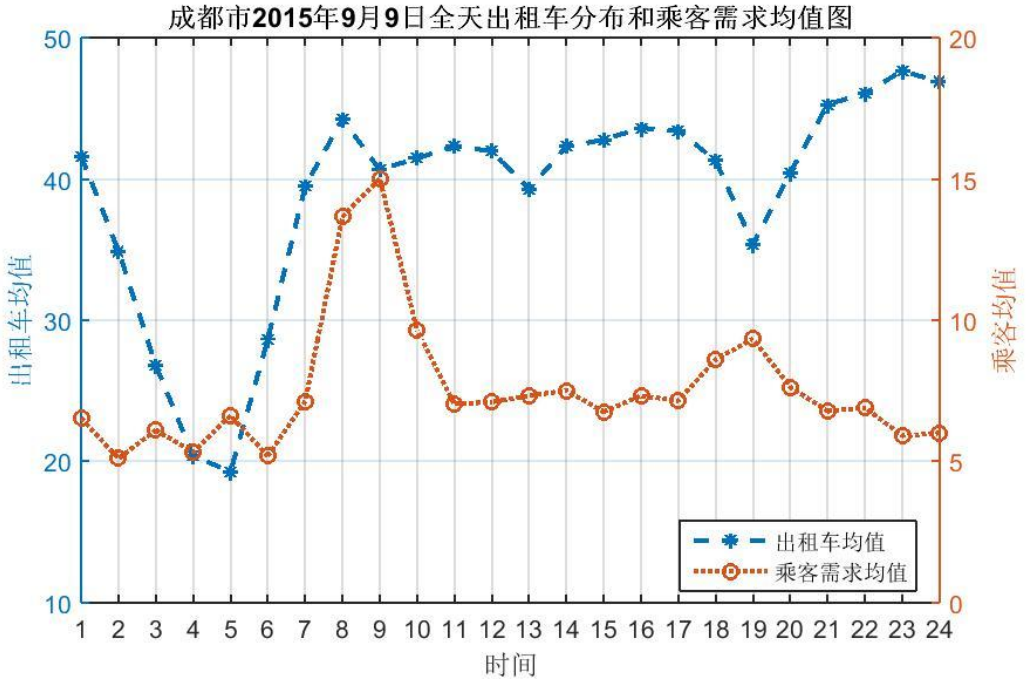


图 1-5 成都市 2015 年 9 月 9 日全天出租车分布和乘客需求均值图

为了更好的查看各个时段的出租车和乘客的匹配情况，我们按每个时点计算了出租车分布和乘客需求的均值，均值图见图 1-5。从图 1-5 可以看出，总体而言，成都市的各个时点的出租车平均数量大于乘客的平均需求量，大约相差 2 倍左右。因此，平均而言，不会出现“打车难”问题。

从图 1-5 我们还可以看出，出租车在凌晨 5 点的时候分布最少，平均大约 20 辆左右，这是 1 天中的最低点。而出租车分布的最高点出现在午夜 23 点左右，大约有近 50 辆出租车，表明成都市民的夜生活还是蛮丰富的，从早 7 点到下午的 18 点，出租车基本稳定在 40 辆左右。

相对于出租车而言，乘客的需求量呈现出不一样的规律。早上 8-9 点是乘客需求最大的时候，平均而言，有近 15 个需求；此外，下午的 18-19 点也出现了乘客需求的小高峰，平均大约 10 个左右；其余时段中，白天的 11 点至下午的 17 点，以及晚上的 20-22 点基本稳定在 8 个需求左右；从午夜 23 点至第二天凌晨 6 点，乘客需求最低，大约在 5 个左右。

6.1.2. 出租车和乘客的匹配模型

从前面的分析可知，虽然平均而言，不会出现“打车难”问题，但是从图 1-4 以及附

录 A 的所有图形，可以看出，存在出租车和乘客需求分布不均衡现象。有的区域非常集中，而个别乘客分布点则非常孤立，周边基本上不存在出租车，显然，这也会导致打不到车的现象。

为此，下面我们主要以乘客为中心，建立匹配模型，以及衡量“打车难”程度的度量指标，分析成都市的出租车问题。

不难看出，每个时点，我们所采集的数据的具体位置不同。这一方面体现了数据的动态性和随机性，另一方面，也为我们进行出租车和乘客的匹配增加了难度。如何建立一个合理的模型以及指标，来衡量“打车难”的程度呢？

这里，我们引入“最近邻”的思想来进行匹配。所谓“最近邻”的思想，就是基于每个时点，每个乘客需求点，寻找离它最近的出租车进行匹配。但是这会导致一个问题，比如图 1-6 中的 A 点，即彭州南部的某点，该点处的乘客距离所有的出租车似乎都较远，如果硬要安排最近一辆出租车去接他，或许没有问题；但是从市场经济的角度出发，出租车司机应该不会空载那么长距离，单纯为了接这位乘客。

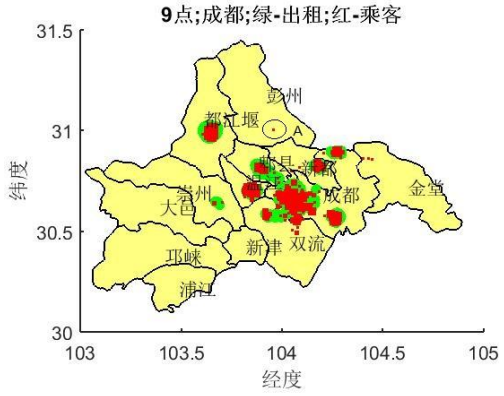


图 1-6　成都市 9 点的出租车和乘客需求图

所以，我们认为比较合理的匹配模型，应该是基于地图距离（当然，最好是基于可达的地图距离，但是，由于数据难于得到，故而此处暂且考虑地图球面距离），按照离乘客所在点一定距离的出租车圈来进行匹配，比如 500 米，1 千米，或者 2 千米等。

记乘客数据结构为$\{(P_{x_i}, P_{y_i}, P_i), i=1, 2, \ldots, n.\}$这里，$P_{x_i}$表示第 i 个乘客数据点的经度，P_{y_i}表示第 i 个乘客数据点的纬度，P_i表示第 i 个乘客数据点的需求量。n 表示搜集到的乘客数据点个数，这里我们分不同时点进行考虑，因为不同的时点，会得到不同的乘客数据点集，以及不同的乘客数据点个数 n，为了表述的方便，我们在符号中可加入时间这个因素，得到的乘客数据点集合为

$$\{(P_{x_i}(t), P_{y_i}(t), P_i(t)), \quad i=1, 2, \ldots, n(t).\}$$

其中 $t=1, 2, \ldots, 24.$

相应的，我们可以得到出租车的数据点集

$$\{(T_{x_j}(t), T_{y_j}(t), T_j(t)),\quad j=1, 2, \ldots, m(t).\}$$

其中，$T_{x_j}(t)$表示第 t 个时点第 j 个出租车数据点的经度，$T_{y_j}(t)$表示第 t 个时点第 j 个出租车数据点的纬度，$T_j(t)$表示第 t 个时点第 j 个出租车数据点的出租车分布数量。

记 d(A,B)表示 A，B 两点的球面距离，C 为指定的邻域半径，定义第 t 个时点第 i 个乘客数据点$A_i(t)$的匹配度为

$$N_i(t) = \begin{cases} 1, & \sum_{j:d(A_i(t),B_j(t))\le C} T_j(t) < P_i(t) \\ 0, & \sum_{j:d(A_i(t),B_j(t))\le C} T_j(t) = P_i(t) \\ -1, & \sum_{j:d(A_i(t),B_j(t))\le C} T_j(t) > P_i(t) \end{cases}$$

其中，t=1, 2, …, 24，$B_j(t)$表示第 t 个时点第 j 个出租车数据点。不难看出，如果与乘客点$A_i(t)$的距离不超过 C 的出租车之和大于乘客需求，即不会出现打车难问题，定义匹配度$N_i(t)$等于-1。如果与乘客点$A_i(t)$的距离不超过 C 的出租车之和小于乘客需求，则表明会出现打车难问题，定义此时的匹配度$N_i(t)$等于 1。如果恰好相等，则定义匹配度等于 0.

因此，$N_i(t)$值越大，打车就越难。

考虑到本题我们获取的是经度纬度数据，可根据网站（网址：http://www.cnblogs.com/ycsfwhh/archive/2010/12/20/1911232.html）所述方法计算地球上任意两点的间的距离。

设第一点 A 的经，纬度为(LonA, LatA)，第二点 B 的经、纬度为(LonB, LatB)，按照 0 度经线的基准，东经取经度的正值(Longitude)，西经取经度负值(-Longitude)，北纬取 90-纬度值(90- Latitude)，南纬取 90+纬度值(90+Latitude)，则经过上述处理过后的两点被计为(MLonA, MLatA)和(MLonB, MLatB)。那么根据三角推导，可以得到计算两点距离的如下公式：

C_0 = sin(MLatA)×sin(MLatB) ×cos(MLonA-MLonB) + cos(MLatA) ×cos(MLatB)

Distance = R_1 ×Arccos(C_0) ×Pi/180

这里，R_1和 Distance 单位相同，如果是采用 6371.004 千米作为半径，那么 Distance 的单位就为千米。

前面所述的 d(A,B)可相应的替换成这里的距离公式 Distance(A,B)。从而，邻域半径 C 可取一系列值。本文取 100 米，200 米，…，1000 米，2 千米，…，50 千米等 59 个刻度，并分别标号为 1,2,3,…,59。如表 1-2 所述。

表 1-2 邻域半径 C（单位：千米）的格点取法以及对应序号

序号	1	2	3	…	9	10	11	…	58	59
半径 C（单位：千米）	0.1	0.2	0.3	…	0.9	1	2	…	49	50

可以预计，领域半径越小，匹配度取 1 的概率越大，随着邻域半径逐步增大，匹配度会从 1 变向-1，也就是，在一个小的范围内打不到车，但如果较大范围的出租车都参与乘客订单竞争的话，乘客能顺利打到车的概率会明显增加。

由此，我们定义如下的打车难易度的度量指标：

$$mN_i(t) = \min_k\{\mathrm{N}_i^{c_k}(t) = -1, k = 1,2,3,\dots,59\}$$

其中，$\mathrm{N}_i^{c_k}(t)$表示邻域半径取c_k时，第 t 个时点第 i 个乘客数据点$A_i(t)$的匹配度。度量指标$mN_i(t)$，即为使得打车由难到易所需的最小半径所对应的序号，序号越大表明需要较大半径的出租车参与乘客数据点的运输，才能满足乘客的打车需求。即$mN_i(t)$越大，表明成功打到车的难度越大；而$mN_i(t)$越小，表明成功打到车的难度越小。

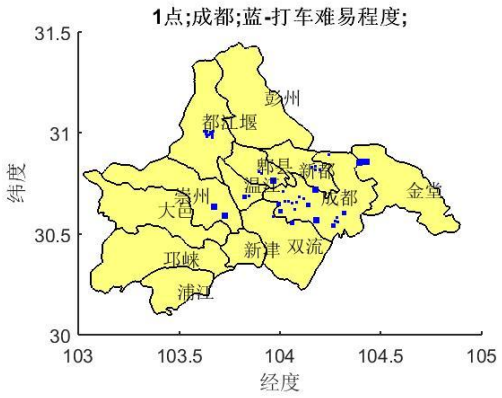


图 1-7 成都市 1 点的打车难易度图

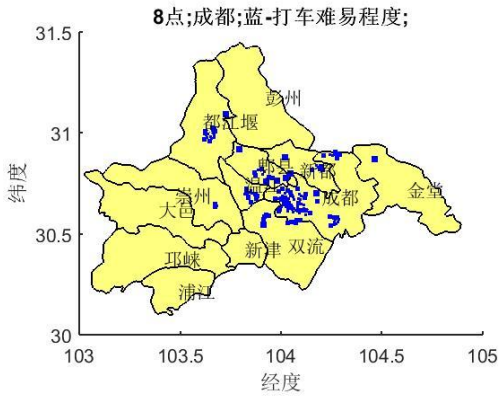


图 1-8 成都市 8 点的打车难易度图

图 1-7—图 1-10 给出了成都市 2015 年 9 月 9 日，1 点，8 点，9 点和 19 点的难易度分布图。更多时点的图形参考附录 B。

从这些图形可以看出，数据点面积最大的集中在早上 8-9 点和晚上 19 点，即早晚高峰上下班时段。这些时段打车难的区域主要集中在成都东这片区域。

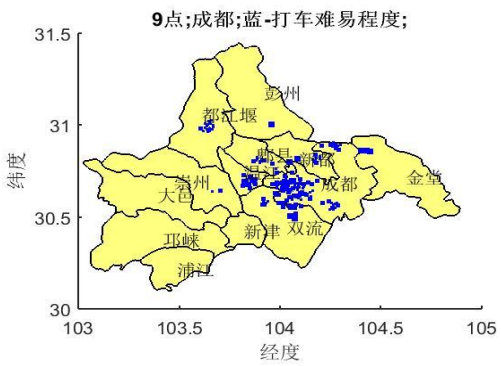


图 1-9 成都市 9 点的打车难易度图

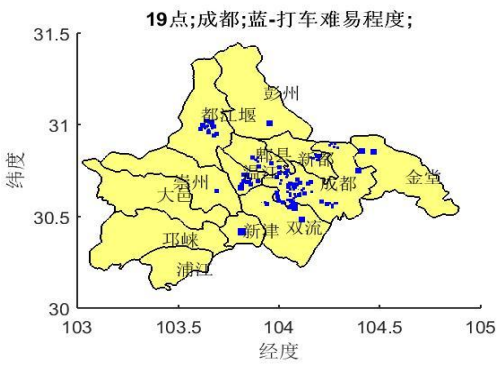


图 1-10 成都市 19 点的打车难易度图

我们还绘制了所有时段乘客数据点的难易度散点图，见图 1-11。从图 1-11 可以看出，一般在 2 公里范围内的出租车即可满足乘客的打车需求，大部分在 1 公里之内即可满足，只有极少部分数据点，需要 10 公里左右甚至更大，比如 30 多公里的出租车才能满足。这些数据点就是打车最难的数据点。

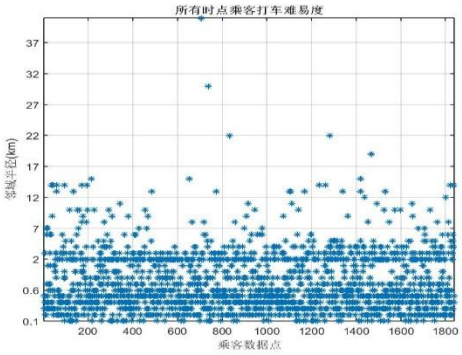


图 1-11 所有时段乘客打车难易度散点

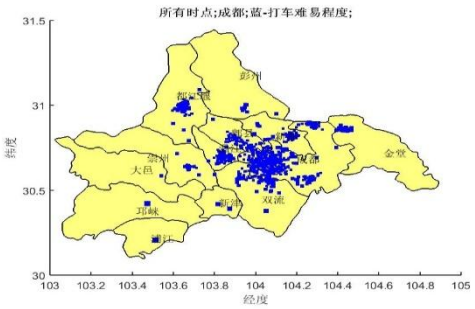


图 1-12 所有时段乘客打车难易度分布

从所有时段的乘客打车难易度的分布图 12 来看，成都东，双流北，以及都江堰中心地带较易出现打车难问题。

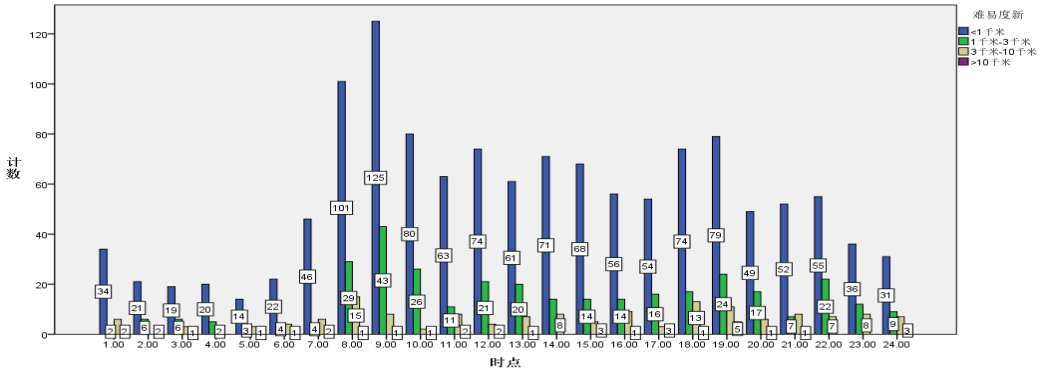


图 1-13 不同时段乘客打车难易度条图

图 1-13 绘制了不同时点，乘客打车的难易度条形图。由图可知，任何时点在不超过 1 千米的距离内都有出租车满足大部分乘客需求。早高峰时段出现在早上的 8 点-9 点，傍晚 19 点会出现一个晚高峰时段；白天基本持平，深夜 1 点-5 点的频数最低。

6.2．问题(2)的讨论

该部分通过搜集的信息和数据，主要讨论打车软件的补贴政策与打车难易程度之间的关系，并找出现有打车软件补贴政策变化带来的新的问题，为问题(3)构建新的打车方案做准备。详细分析如下：

6.2.1．滴滴打车和快的打车对乘客进行补贴的比较分析

根据收集的数据，滴滴打车和快的打车对乘客的补助如下表 2-1 所示。（数据来源：http://chuansong.me/n/651901）

表 2-1　2014 年 2-1 月滴滴打车和快的打车对乘客的优惠额度对比

时间	滴滴打车补贴方案	快的打车补贴方案
1 月 10 日	每单减 10 元	减 0 元
1 月 20 日	持续每单减 10 元	持续每单减 10 元
2 月 11 日	每单减 5 元	每单减 10 元
2 月 17 日	每单减 10 元，新司机首单 50 元	每单减 11 元
2 月 18 日	每单减 12-20 元	每单减 13 元
3 月 3 日	每单减 12-20 元	每单减 10 元
3 月 18 日	每单减 5 元	每单减 5 元
3 月 22 日	每单减 5 元	每单减 3-5 元
3 月 23 日	每单减 3-5 元	持续每单减 3-5 元

纵观补贴方案的数据，可以得出如下结论：滴滴打车对司机的补贴 1 月 10 日为 10 元，而快的为 0 元。然后滴滴打车出现先减后增的趋势，直到 3 月 3 日后，补贴开始狂减至 0 元，这说明公司开始为了争取用户使用，增大补贴力度，而当用户群体达到一定份额后，对乘客的奖励逐渐取消，此时大部分乘客已经习惯使用打车软件进行打车，所以取消补贴，乘客的对打车软件使用率会降低，但不会明显下降，尤其是高峰时期为了减少等待时间乘客更愿意使用打车软件，以减少打车难的问题；而快的打车的补贴从开始就持续增加，目的是为了与滴滴打车竞争，争取更多的用户，后面开始减少补贴后其用户的变化和滴滴打车类似。

6.2.2．滴滴打车和快的打车对司机进行补贴的对比分析

表 2-2　2014 年 1-3 月滴滴打车和快的打车对司机的补贴额度对比

时间	滴滴打车补贴方案	快的打车补贴方案
1 月 10 日	每单减 10 元	减 0 元
1 月 20 日	持续每单减 10 元	每单减 10 元
1 月 21 日	持续每单减 10 元	每单减 15 元
2 月 12 日	持续每单减 10 元	每单减 5 元
2 月 17 日	每单减 5 元	持续每单减 5 元

2 月 18 日	持续每单减 5 元	每单减 5-11 元
2 月 19 日	持续每单减 5 元	每单减 5-13 元
3 月 4 日	每单减 10 元	持续每单减 5-13 元
3 月 22 日	持续每单减 10 元	每单减 2 元
3 月 23 日	每单减 2 元	持续每单减 2 元

滴滴打车和快的打车对司机的补助如上表 2-2 所示（数据来源：http://chuansong.me/n/651901）。纵观补贴方案的数据，滴滴打车对司机的补贴开始稳定在 10 元/单，后面开始逐渐减少，而快的打车的补助总体上比滴滴打车多，目的是抢占市场份额，当达到一定规模时，补贴降为 0。这其中包括：

(1).当补贴很高的时候，司机更愿意去接收订单，这一方面可以减少乘客的等待时间，但是对于没有使用软件下订单的乘客，他们很难打到车，对这一部分乘客增加了打车难度，而对其中老人小孩群体，会出现根本打不到车的局面。

(2).当补贴降为 0 以后，根据调查（http://www.ithome.com/html/it/98292.htm）司机抢单热情下降，特别是打车高峰期，随时随地都可能拉到客人，所以总体上讲，打车的难易局面与没有打车软件的情形差不多。

6.2.3．从乘客的等待时间和出租车的空驶率讨论对“打车难易程度”的影响

针对滴滴打车，自补贴从 2014 年 1 月 10 日开始以来，2014 年 1 月-3 月的注册用户量变化情况和日均订单量的增长情况如表 2-3 所示。数据来源：（http://news.mydrivers.com/1/298/298626.htm）

表 2-3　滴滴打车用户、日均订单占用户数据

日期	用户数	日均订单	日均订单/用户数
1 月 10 日	2200 万	35 万	1.6%
2 月 9 日	4000 万	183 万	4.6%
2 月 24 日	8260 万	316 万	3.8%
3 月 27 日	1 亿	521.83 万	5.2%

通过表 2-3 可以得出，滴滴打车从 2014 年 1 月至 3 月，随着补贴的进行，除了注册用户持续增加以外，订单数也持续增加，“订单/用户数”的百分比稳定在 4%-5%之间。通过市场份额的估算，截至 2014 年 12 月，中国打车 APP 累计账户规模达 1.72 亿，其中快的打车市场份额为 56.5%，滴滴打车为 43.3%，这说明补贴政策极大的吸引了打车用户的注册和下单；快的打车的变化情况和滴滴打车类似。

（数据来源：http://www.askci.com/news/chanye/2015/02/13/172139y4la.shtml）。

调查显示，使用打车软件后，94.0%的乘客打车等候时间在 10 分钟内，等候时间在 10 分钟以上的乘客比重下降了 29.9%。而使用打车软件后，90.3%的司机认为降低了空驶率，其中 55.0%的司机认为每月空驶率下降 10%以下，41.2%的司机认为每月空驶率下降 10-30%，3.9%的司机认为每月空驶率下降 30%以上。（数据来源：http://chuansong.me/n/651901）。

综上分析，司机的空驶率下降，说明载客率增加，乘客的等待时间减少，说明打车变得容易，所以通过前面的分析，有补贴政策，且补贴较高，司机和乘客都更愿意去使用打车软件，这样司机空驶率下降，乘客等待时间也下降，两个下降都使得打车变得容易。

6.2.4. 从乘客和司机的经济收益讨论补贴方案对“打车难易”程度的影响

调查显示，2014 年 1 月以来，快的打车与滴滴打车在补贴金额上一直互相较劲，双方补贴金额合计在 25 亿以上。除补贴司机外，大部分补贴均被乘客获得。调查显示，81.0%的乘客认为节约了打车费用。

由于打车软件为司机每单提供补贴，这为司机提供加价收入来源。调查显示，在使用打车软件后，92.0%的司机认为平均每单收入有增加。其中，48.0%的司机认为平均每单收入增加 10-30%，51.8%的司机认为平均每单收入增加 10%以下，0.2%的司机认为平均每单收入增加 30%以上。(数据来源：http://chuansong.me/n/651901)。

因此，无论从乘客的角度，还是从司机的角度，在经济方面都获得了收益，所以在这种经济补贴的环境下，双方都更愿意去使用打车软件，根据上面的结论，软件使用率越高，对使用软件打车的乘客，打车变得容易。

通过 1-4 的分析，我们将城市人群 N 分类：老年人群 N_1 和一般人群 N_2；

(1). 对于老年人群 N_1（20%左右），由于这部分人群使用智能手机的较少，也就是即便打车软件提供补贴，对他们的“打车难”问题仍然很难解决。

(2). 对于一般人群 N_2，可以得出如下关系图 2-1：

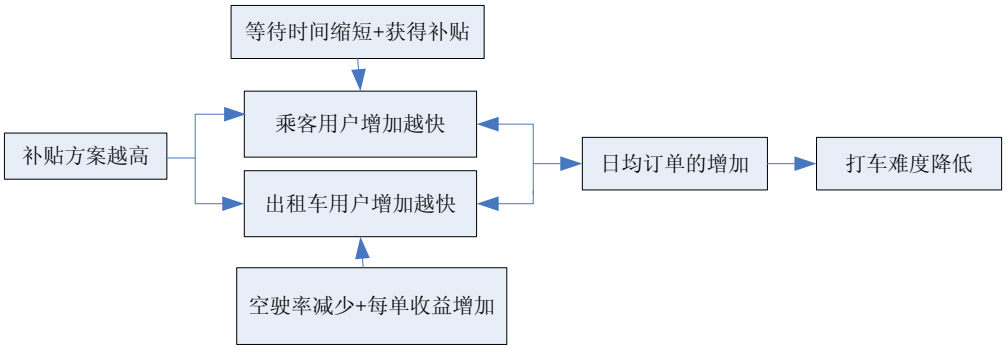


图 2-1　补贴政策与打车难易程度关系图

也就是说针对人群 N_2，随着乘客和出租车司机补贴的存在或者提高，充分调动了他们使用软件的积极性，使得用软件的人数会持续增加，这样既缩短了乘客打车时间的等待，也减少了司机的空驶率，故可以有效的缓解打车难的问题。但随着补贴的降低，乘客和司机对打车软件使用的激情开始降低，尤其在高峰时段，司机不需要使用软件接单，其空驶率也很低；而乘客基本使用打车软件，在高峰期打车依然困难，这说明对补贴方案的改变和调整需要进一步讨论。

由于使用打车软件预约打车，尤其在高峰时段，司机由于可以享受打车补贴。所以没有预约需要打车的人，在高峰时段几乎打不到车，所以很多城市比如上海、济南等城市严禁高峰期使用打车软件。这也说明对打车软件公司的补贴政策需要进一步讨论。(数据来源: http://news.sohu.com/20140226/n395713186.shtml;

http://readmeok.com/2014-3/6_29733.html）

6.3．问题(3)的设计方案

6.3.1．影响“打车难”可能的因素

根据北京晨报的报道，影响打车难的主要因素包括如下几个方面：

原因一：人多车少司机挑活儿；

原因二：司机收车常现空当儿；

原因三：禁限行路段宁愿空跑；

原因四：车辆调配不当效率差；

原因五：道路拥堵运营太耗时；

原因六：部分区域“黑车”当道；

原因七：新手不认路效率太低。

纵观以上因素，这些原因分别与职业道德、信息畅通、调度、信用、道路拥挤，外部竞争，职业素养等因素有关。而其中的调度直接和现有的打车软件有密切关系，所以在新的补贴方案中应考虑其他影响打车难的因素，比如错开高峰时段打车进行补贴，调动出租车去密集地区给予补助等。

6.3.2．借助 S 曲线模型，恰当补贴，减缓打车难

通过对现有打车软件补贴方案的比较分析，我们可以看到，目前的补贴方案对时空的区分较少，千篇一律。虽然在一定程度上可以缓解打车难问题，但是对于高峰时段，以及那些很难打到车的区域效果不是太明显。

结合问题(1)的求解可知，人们出行集中在早晚高峰时段，即早上 8-9 点，和晚上 19 点左右。其他时段，白天的打车需求相对于夜晚的较大。虽然白天的出租车分布也比夜晚的多，但二者相对来说可以持平。可是由于白天出租车流动性很大，从图 13 各个时段的分布图可以看出，白天如果人们要成功打到车，往往需要大约 3 公里之内的出租车参与运载才行。这也就意味着，人们为了成功打到车，往往需要等待一会，甚至较长的时间。所以，打车软件的补贴方案应该考虑不同时段区别对待，高峰时段和低峰时段的补贴应该有所不同，尽量引导人们避开高峰时段。

此外，我们还可以看到，就成都市而言，“打车难”容易出现的区域是有集中趋势的，比如成都东，双流北，这些地方往往是人流比较集中的地方，比如火车站（成都东站），或者机场（双流机场），或者学校，写字楼等等。因此，为了有效缓解“打车难”问题，在设计软件的时候需要加入区域的因素，也就是分流的思想。通过补贴引导乘客尽量走走路，从人流密集的，难于打到车的区域往人流稍微稀疏，较容易打到车的地方挪动。

鉴于此，我们设计了如下的补贴方案：

方案 1：低峰时段出行并成功打到车的给予一定的现金或积分奖励；

方案 2：鼓励出租车前往人流密集处，打车软件可以通过大数据的实时分析，告诉司机哪个区域打车需求大，通过一定的现金或者积分奖励，鼓励司机抢人流密集处的定单。

下面，我们利用模拟对这两种方案的实施效果进行评价。

对问题(2)所搜集的数据，我们可以计算出乘客的累计补贴和司机的累计补贴，见表3-1。一般而言，补贴刚开始投入时，人们知之甚少，因此产生的效果也不是太明显；随着补贴的逐渐加大，补贴的累计效应将逐步产生，表现在用户数和日均订单量均显著增加；但到了后期，随着市场饱和等等因素的影响，补贴的效应将明显下降，用户数和日均订单量将趋于一个平衡状态。整个变化过程，可以用 S 曲线来进行刻画。

表 3-1 滴滴打车软件的补贴数据与 S 曲线预测结果

日期	间隔	乘客补贴(元)	司机补贴(元)	乘客累计补贴(元)	司机累计补贴(元)	用户数(万)	日均订单(万)	基于乘客累计补贴的日均订单量预测(万)	基于司机累计补贴的日均订单量预测(万)
2014/1/10	0	10	10	10	10	2200	35	33.21382	33.08413
2014/1/20	10	10	10	20	20			121.33963	125.65335
2014/1/21	11	10	10	30	30			186.88047	196.04641
2014/2/9	30	10	10	40	40	4000	183	231.92339	244.87886
2014/2/11	32	5	10	45	50			249.23255	279.83803
2014/2/12	33	5	10	50	60			264.00548	305.8748
2014/2/17	38	10	5	60	65			287.8228	316.52212
2014/2/18	39	20	5	80	70			320.63838	325.9429
2014/2/19	40	20	5	100	75			342.09734	334.33409
2014/2/24	45	20	5	120	80	8260	316	357.19537	341.8534
2014/3/3	52	20	5	140	85			368.38595	348.6284
2014/3/4	53	20	10	160	95			377.00836	360.3438
2014/3/18	67	5	10	165	105			378.86315	370.11543
2014/3/22	71	5	10	170	115			380.61716	378.3872
2014/3/23	72	5	2	175	117			382.27838	379.89132
2014/3/27	76	5	2	180	119	10000	521.83	383.85397	381.35057

所谓 S 曲线模型，假定自变量为 t，因变量为 y，则 S 曲线定义为：

$$y=\exp\left(b_0+\frac{b_1}{t}\right)$$

或者

$$\log(y)=b_0+\frac{b_1}{t}$$

其中 b_0 和 b_1 为待定常数，需要通过数据拟合得到。将 S 曲线模型代入 SPSS 软件，可以得到如下拟合结果。

表 3-2 日均订单量的 S 曲线拟合的拟合度

	模型摘要				ANOVA					
	R	R 平方	调整后的 R 平方	标准估算的错误		平方和	自由度	均方	F	显著性
自变量为乘客累计补贴	.979	.959	.939	.290	回归 (R)	3.963	1	3.963	47.131	.021
					残差	.168	2	.084		
					总计	4.132	3			
	R	R 平方	调整后的 R 平方	标准估算的错误		平方和	自由度	均方	F	显著性
自变量为司机累计补贴	.976	.953	.930	.310	回归 (R)	3.939	1	3.939	40.914	.024
					残差	.193	2	.096		
					总计	4.132	3			

从表 3-2 可以看出，以乘客累计补贴和司机累计补贴为自变量，以日均订单量为因变量的 S 曲线拟合效果非常显著，R 平方非常接近于 1，方差分析的显著性水平也小于 0.05。而 S 曲线拟合的系数估计见表格 3-3. 记D_1为日均订单量，D_2为乘客累计补贴，D_3为司机累计补贴。

从表 3-3 我们可以得到如下经验公式：

$$D_1 = \exp(6.094 - \frac{25.913}{D_2})$$

或者

$$D_1 = \exp(6.168 - \frac{26.689}{D_3})$$

表 3-3　日均订单量的 S 曲线拟合的参数估计

	系数					
		非标准化系数		标准系数	t	显著性
		B	标准错误	贝塔		
自变量为乘客累计补贴	1 / 乘客累计补贴	-25.913	3.774	-.979	-6.865	.021
	（常量）	6.094	.195		31.181	.001
自变量为司机累计补贴	1 / 司机累计补贴	-26.689	4.173	-.976	-6.396	.024
	（常量）	6.168	.217		28.380	.001
	因变量为 ln(日均订单)。					

基于上述两个经验公式，我们可以得到日均订单量的预测值，见表格 3-1。对用户量为因变量的模型分析表明，S 曲线拟合效果不显著，故在接下来的分析中，我们只考虑以日均订单量为因变量的随机模拟。

我们把日均订单量对应到问题(1)中的乘客需求量。但是这里的单位是万，而问题(1)中的单位为个,为此，在进行模拟时，我们需要考虑日均订单量除以用户量的百分比，采用前面的 S 曲线拟合，拟合结果见表 3-4 和表 3-5。

表 3-4 日均订单量/用户数的百分比的 S 曲线拟合的拟合度

	模型摘要				ANOVA					
自变量为乘客累计补贴	R	R 平方	调整后的 R 平方	标准估算的错误		平方和	自由度	均方	F	显著性
	.955	.912	.868	.194	回归(R)	0.785	1	0.785	20.763	.045
					残差	.076	2	.038		
					总计	0.861	3			
自变量为司机累计补贴	R	R 平方	调整后的 R 平方	标准估算的错误		平方和	自由度	均方	F	显著性
	.964	.929	.894	.174	回归(R)	0.800	1	0.800	26.325	.036
					残差	.061	2	.030		
					总计	0.861	3			

表 3-5 日均订单量/用户数的百分比的 S 曲线拟合的参数估计

	系数				
		非标准化系数		标准系数	t	显著性
		B	标准错误	贝塔		
自变量为乘客累计补贴	1 / 乘客累计补贴	-11.532	2.531	-.955	-4.557	.045
	（常量）	1.645	.131		12.553	.006
自变量为司机累计补贴	1 / 司机累计补贴	-12.026	2.344	-.964	-5.131	.036
	（常量）	1.683	.122		13.788	.005
	因变量为 ln(日均订单/用户数)					

从表 3-4 可以看出，以乘客累计补贴和司机累计补贴为自变量，以日均订单量/用户数的百分比为因变量的 S 曲线拟合效果非常显著，R 平方非常接近于 1，方差分析的显著性水平也小于 0.05，由 S 曲线拟合的系数估计表 3-5，记 D_1为日均订单量，D_2为乘客累计补贴，D_3为司机累计补贴，D_4为用户数；可以得到如下经验公式：

$$D_1 / D_4 = \exp(1.645 - \frac{11.532}{D_2})$$

或者

$$D_1 / D_4 = \exp(1.683 - \frac{12.026}{D_3})$$

表 3-6　日均订单/用户数百分比 S 曲线预测结果

时间	间隔	乘客补贴	司机补贴	乘客累计补贴	司机累计补贴	用户数	日均订单	百分比=日均订单/用户数	基于乘客累计补贴的百分比预测	基于司机累计补贴的百分比预测
2014/1/10	0	10	10	10	10	2200	35	1.590909091	1.63547	1.61726
2014/1/20	10	10	10	20	20				2.91108	2.95074
2014/1/21	11	10	10	30	30				3.52796	3.60563
2014/2/9	30	10	10	40	40	4000	183	4.575	3.88382	3.98572
2014/2/11	32	5	10	45	50				4.01025	4.23274
2014/2/12	33	5	10	50	60				4.11434	4.40587
2014/2/17	38	10	5	60	65				4.27557	4.47433
2014/2/18	39	20	5	80	70				4.48603	4.53386
2014/2/19	40	20	5	100	75				4.61724	4.58608
2014/2/24	45	20	5	120	80	8260	316	3.82566586	4.70684	4.63228
2014/3/3	52	20	5	140	85				4.7719	4.67342
2014/3/4	53	20	10	160	95				4.82129	4.74355
2014/3/18	67	5	10	165	105				4.83183	4.80108
2014/3/22	71	5	10	170	115				4.84178	4.84914
2014/3/23	72	5	2	175	117				4.85117	4.85782
2014/3/27	76	5	2	180	119	10000	521.83	5.2183	4.86006	4.86622

表 3-6 为日均订单/用户量百分比预测结果，可以看出后期稳定后，其影响的百分比在 5%左右。

在模拟时，我们首先需要设定乘客的补贴标准，比如我们按每天补贴 1-5 元不等，补贴 1 个月，按 30 天计算，可以得到模拟的日均订单量/用户量的百分比。假定问题(1)所获得的出租车分布数据不变，然后结合补贴方案 1 以及百分比的 S 曲线拟合值，乘以问题(1)中所获得乘客的需求数据，即可得到在方案 1 下，乘客需求的变化情况，从而可进行方案 1 在缓解打车难问题上的评价。

6.3.3．方案 1 的模拟评价

方案 1：低峰时段出行并成功打上车的给予一定的现金或积分奖励。

问题(1)的求解中，我们已经指出早高峰时段出现在早上 8 点-9 点，晚小高峰时段是傍晚 19 点。为了说明补贴方案 1 的是否合理。我们假定预计在 8 点钟出行的乘客，在补贴方案 1 的鼓励下，提前至早上 7 点出行，看看能否显著的缓解打车难问题。

经过测算，相应的 Matlab 代码见附录 C。我们发现，若按 S 曲线模型，最后稳定

在 5%左右的影响率的话，基本上不会太影响问题(1)中所定义的难易度。也就是，如果仅是乘客避让高峰期，而且是在补贴方案影响不大的情况下，很难缓解打车难问题。

究其原因，通过对附录 A 的出租车分布图进行比较分析，我们发现，无论哪个时段，出租车的分布规律基本上处于一个不变的趋势，大致都是那些区域。

所以，为了缓解打车难，我们得从出租车入手。

6.3.4. 方案 2 的模拟评价

基于方案 1 的评价，我们提出了让出租车跑起来的方案 2，希望方案 2 能有效缓解打车难问题。

方案 2：鼓励出租车前往人流密集处，打车软件可以通过大数据的实时分析，告诉司机哪个区域打车需求大，通过一定的现金或者积分奖励，鼓励司机抢人流密集处的定单。

我们依然以高峰时段的早上 8 点为例进行评价。

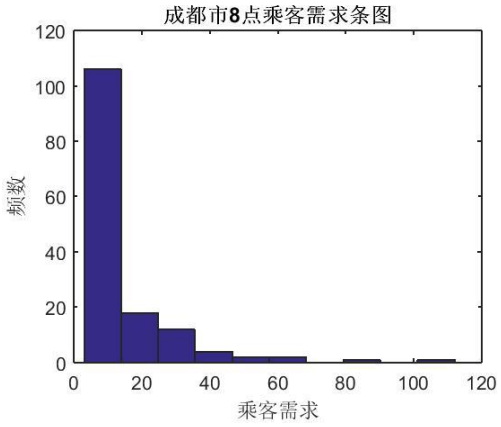


图 3-1 成都市 8 点乘客需求条型图

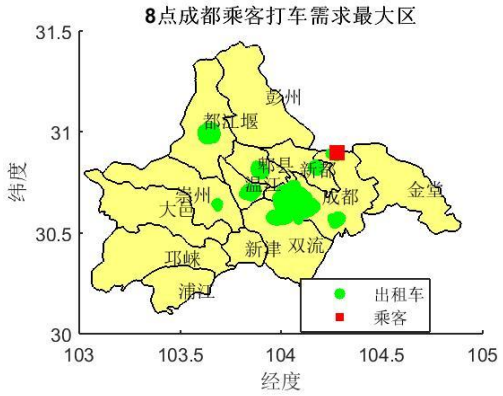


图 3-2 成都市 8 点打车需求最密集点

早上 8 点，成都市的乘客需求的条形图见图 3-1，从图 3-1 可知，有 72.6%的乘客数据点的打车需求 8 辆左右，12.33%的乘客数据点的打车需求在 20 辆左右，只有极少数数据点的打车需求超过 30 辆。

不难发现 8 点的乘客需求中，有一个是极端点，该点的经度为 104.277，纬度为 30.8972，乘客的打车需求达到了 112 辆，远远大于其他数据点。见图 3-2 中的红色方框所在地。从地图对应关系来看，大致可以判断该点是西南石油大学所在地，而我们的数据采集是 2015 年 9 月 9 号，恰逢新生入学报到，故而出租车的需求会明显高于其他地方。

假如该点的乘客发出订单，表明有打车需求，如果打车软件鼓励较远的出租车司机进行抢单，抢单并搭载成功，按距离远近给予不同程度的奖励，采用 S 曲线的初始值 1%的影响率，假如所有的出租车分布点都有 1%的出租车迁移到该点，我们看看情况会如何呢？

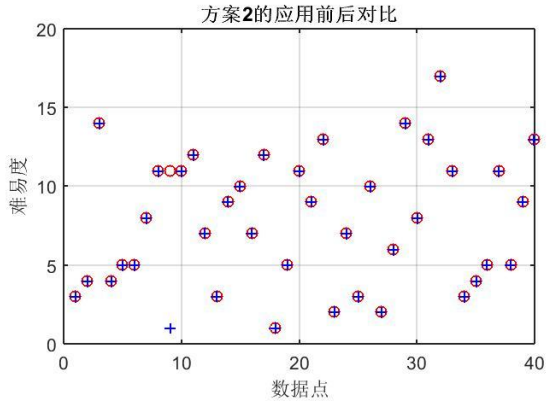


图 3-3 方案 2 的应用前后对比图

图3-3绘制了乘客部分数据点使用方案2前后的打车难易度对比结果，相应的Matlab代码见附录 C。不难看出，方案 2 使用后，第 9 个数据点（也就是前面的打车需求最大的数据点）的打车难度显著下降，从原来的难易度为 11 下降到现在的 1，而其他数据点的变化几乎为 0。这表明针对乘客打车密集点的补贴方案 2 能较好的缓解打车难问题。

七．模型评价与展望

本文针对“互联网+”时代的出租车资源配置的评价以及优化问题进行了数据分析和数学建模。搜集了成都市 2015 年 9 月 9 日的出租车分布以及乘客打车需求数据。基于该数据，通过统计分析以及数学建模手段，给出了评判出租车资源“供求匹配”程度大小，即难易度度量指标。借助于该指标，我们得出了时间上，成都市早上 8-9 点是出租车打车的高峰时段，傍晚 18 点是另一个小高峰时段。这两个时段较易出现打车难问题。而通过区域比较分析，空间上，我们发现成都东火车站，以及双流北，大约在双流机场等位置，较易出现打车难问题。

对现行的最流行的滴滴打车和快的打车软件的补贴措施的比较分析，我们发现，现行的补贴措施主要是为了吸引用户，扩大软件影响力。当然，这也大大提高了用户使用软件打车率，在一定程度上缓解了打车难问题。但是，应该看到，这样的补贴措施，不分时段，不分地点，没有侧重点，没有体现时空差异，以及大数据时代的数据时效性，以及实时监控策略，存在着一定的不足之处。

基于问题(1)的求解，以及问题(2)中各大软件补贴存在的问题，在问题(3)中，我们提出了 2 种设计软件的补贴方案。经过建模分析，我们发现方案 1，基于乘客避让高峰的补贴措施效果并不明显；而方案 2，鼓励出租车司机接纳打车密集地乘客订单的方案，在缓解打车难问题上，效果显著。

未来的研究方向，我们可以搜集更多一些的数据，比如北京，上海，广州等一线城市的，以及其他二、三线城市的出租车运行数据，利用本文所建立的模型来评价各个地方的出租车“供求匹配”程度大小。此外，我们还可以搜集一周，或者一个月，或者一季度，一年的数据，进一步探讨出租车“供求匹配”随时间的变化趋势。除此之外，我们的老年人由于使用智能手机的人数较少，所以没办法通过打车软件进行打车，所以打车软件公司可以专门设置搭乘老年人额外奖励的方案，缓解老年人打车难的问题。总之，还可以提出更好的补贴方案，优化出租车资源配置，缓解打车难问题。

参考文献

[1] 姜启源，数学模型，北京：高等教育出版社，2007.

[2] 张圣勤，MATLAB 7.0 实用教程，北京：机械工业出版社，2006.

[3] 罗聪坤，关于出租车打车软件问题的研究，兰州教育学院学报，第 31 卷第 3 期：1-3，2015.

[4] 随心，滴滴打车用户数正式破亿，http://news.mydrivers.com/1/298/298626.htm，2015-09-12.

[5] 汪光焘等，关于加强打车软件综合管理的建议，城市交通，2014 年第 5 期，1-3，2014.

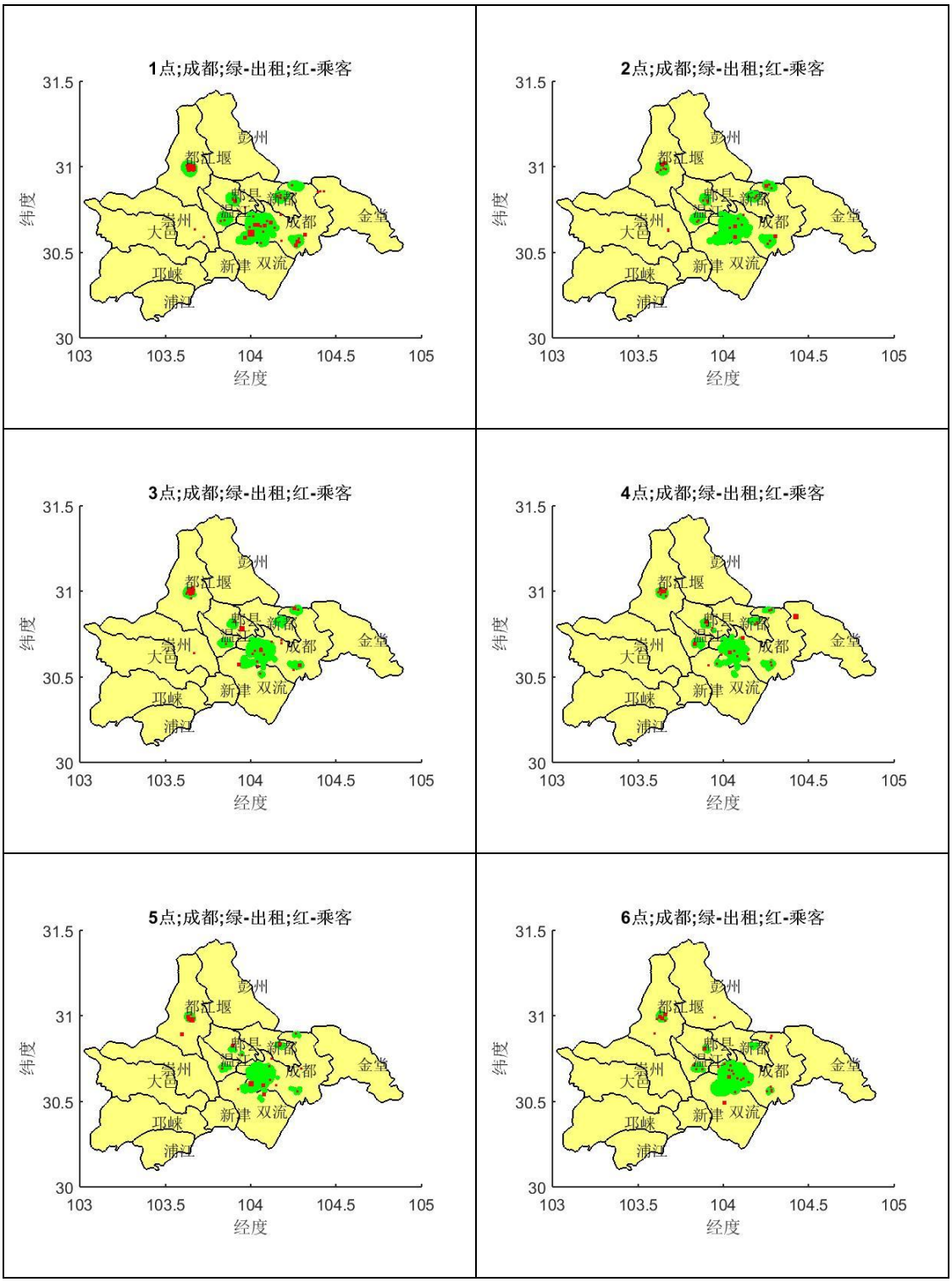
[6] 胡建军等，基于最近邻优先的高校聚类算法，四川大学学报（工程科学版），第 36 卷第 6 期，3-6，2004.

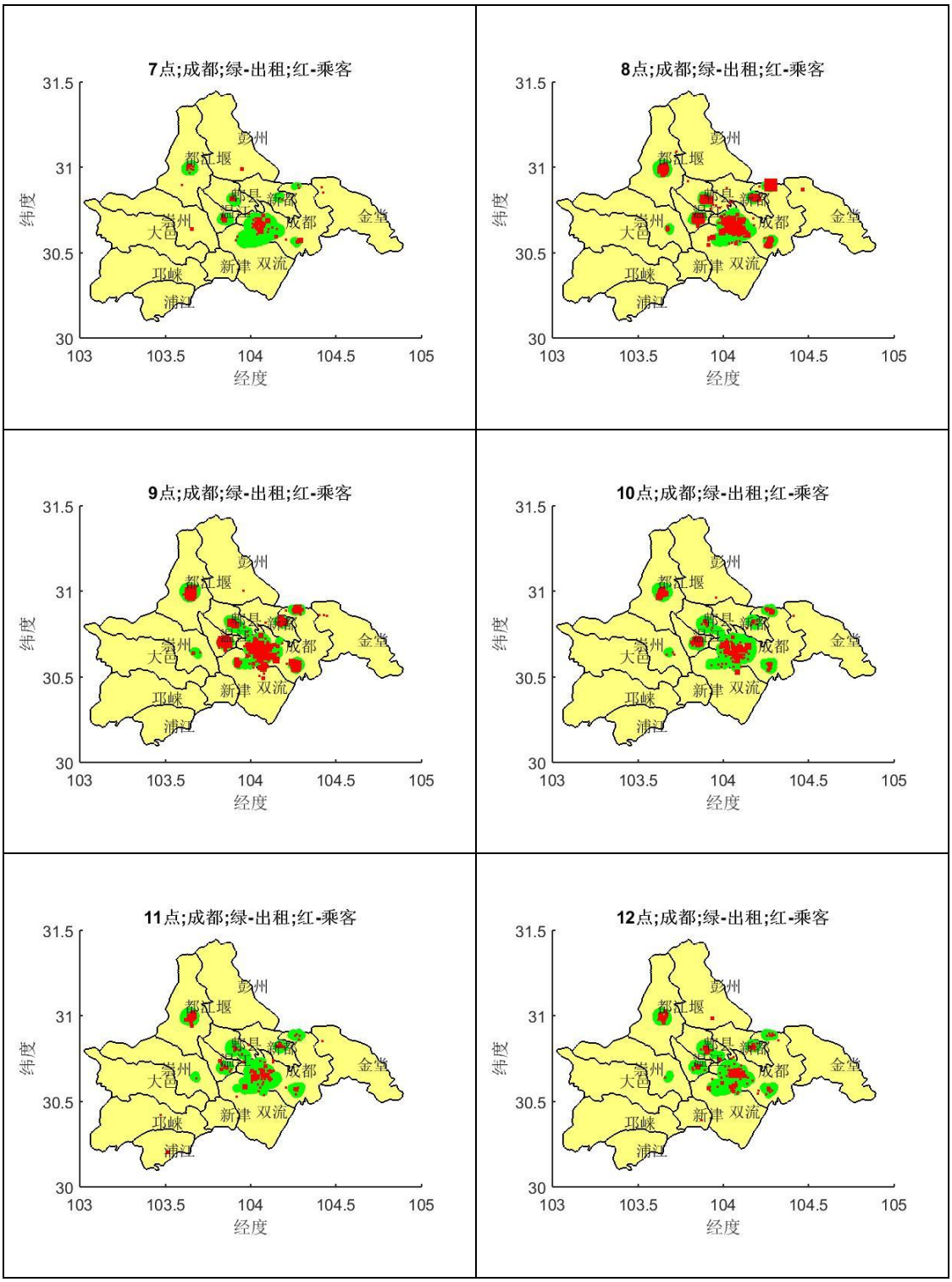
[7] 张文彤，SPSS 统计分析基础教程（第二版），北京：高等教育出版社，2011.

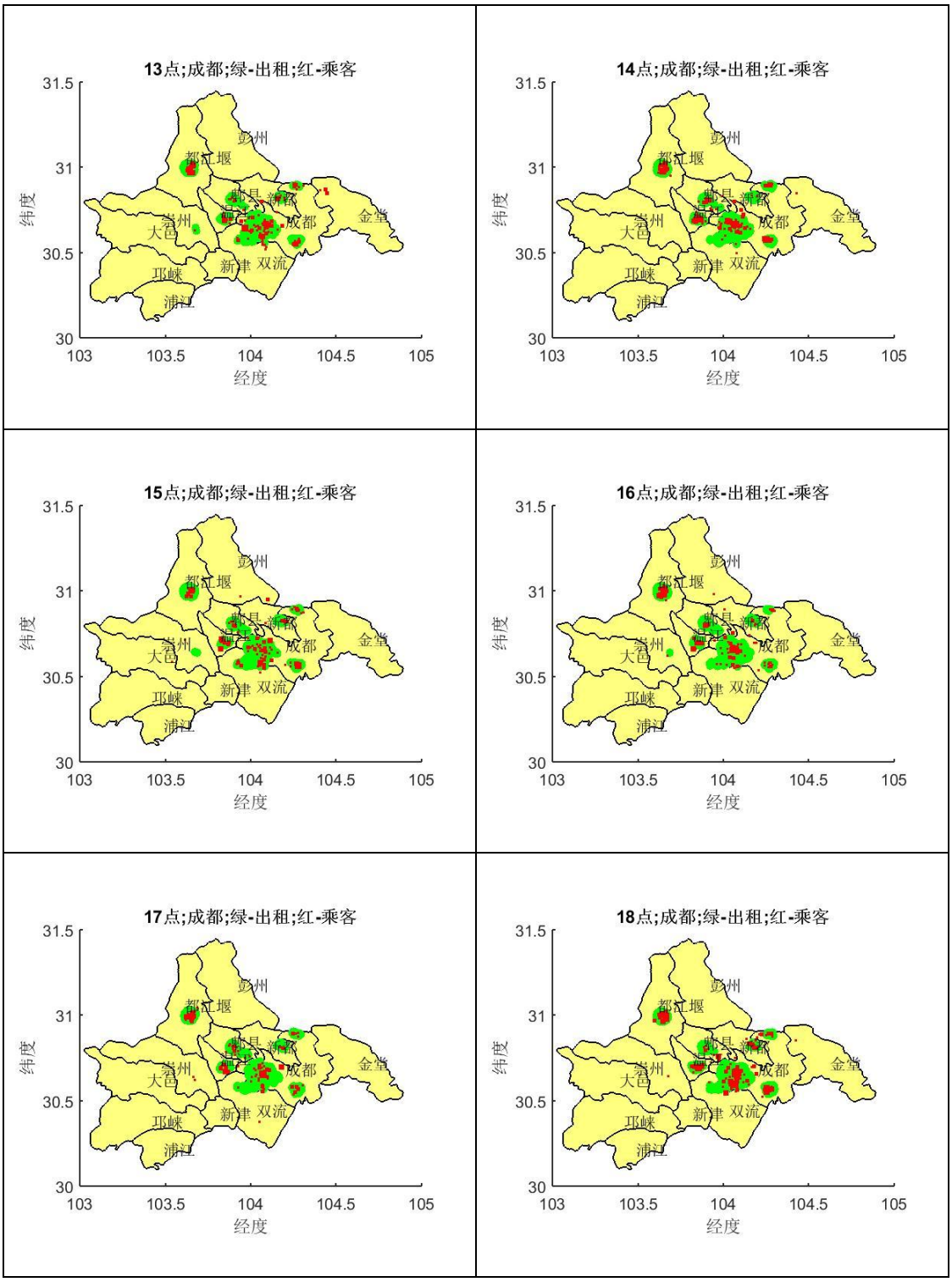
[8] 宋晓霞，基于 MATLAB 的通用数据拟合办法，山东大同大学学报（自然科学版），第 30 卷第 4 期，2-4，2014.

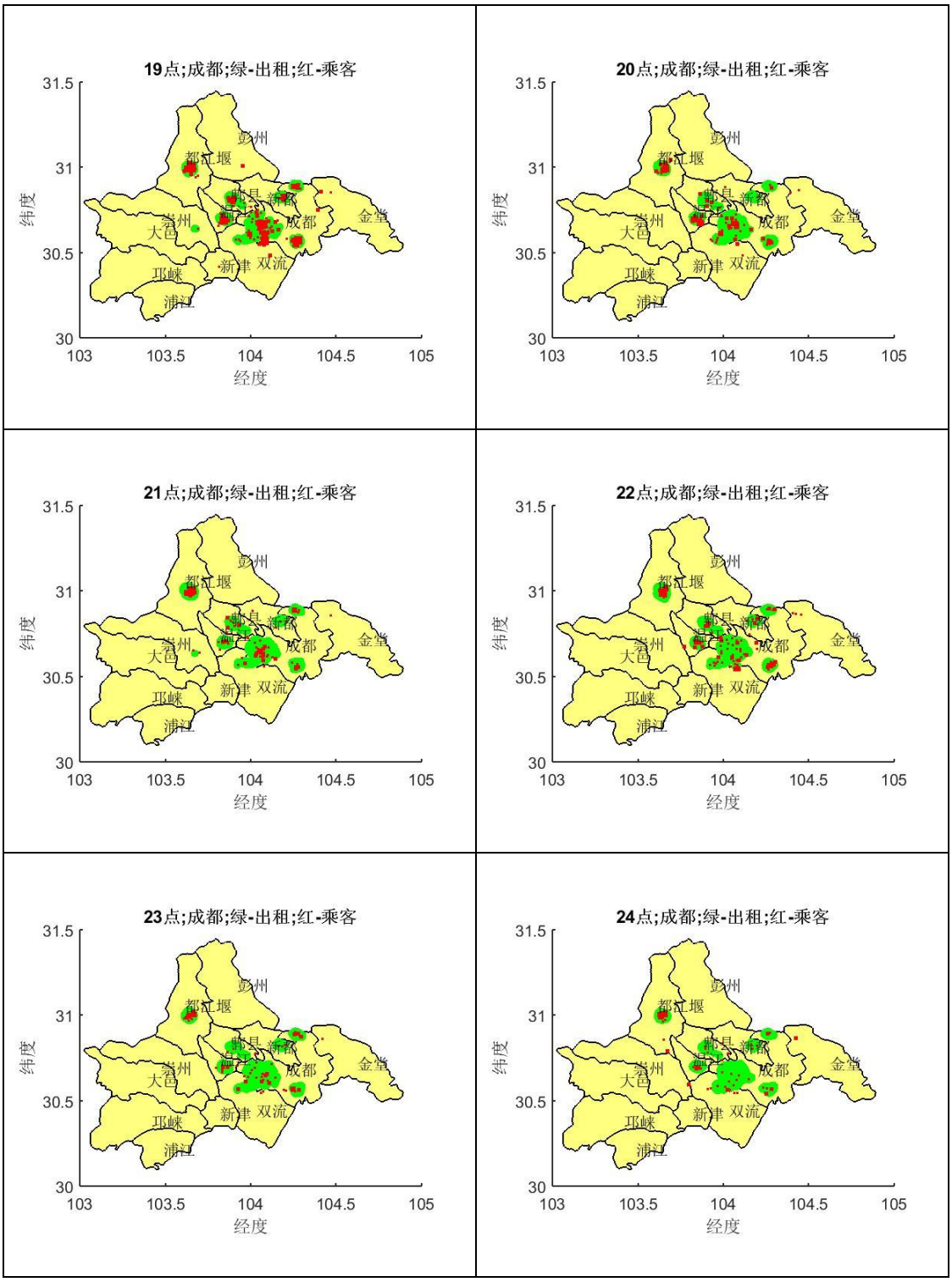
附录：

附录 A.成都市各个时点出租车和乘客的分布散点图

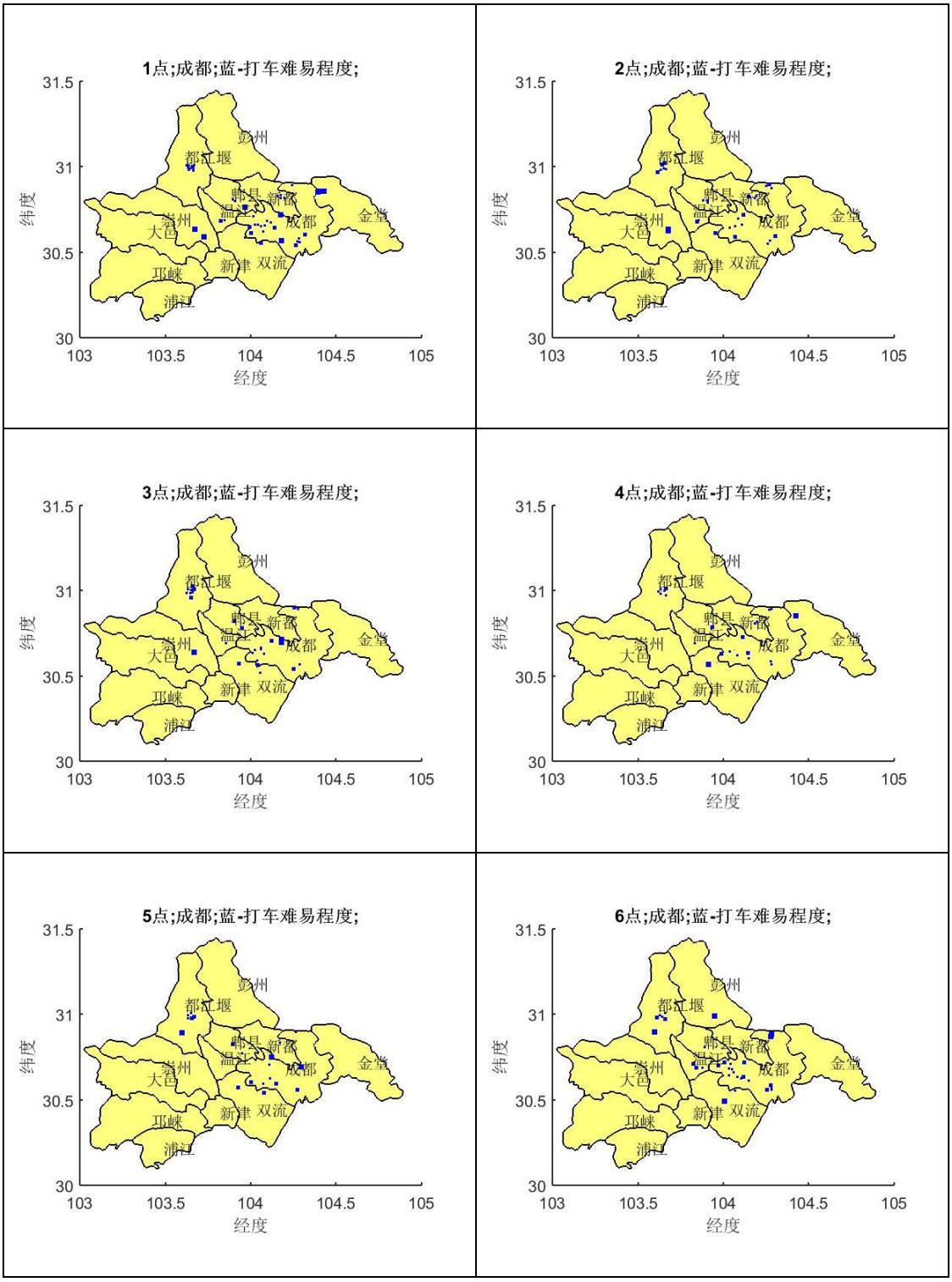


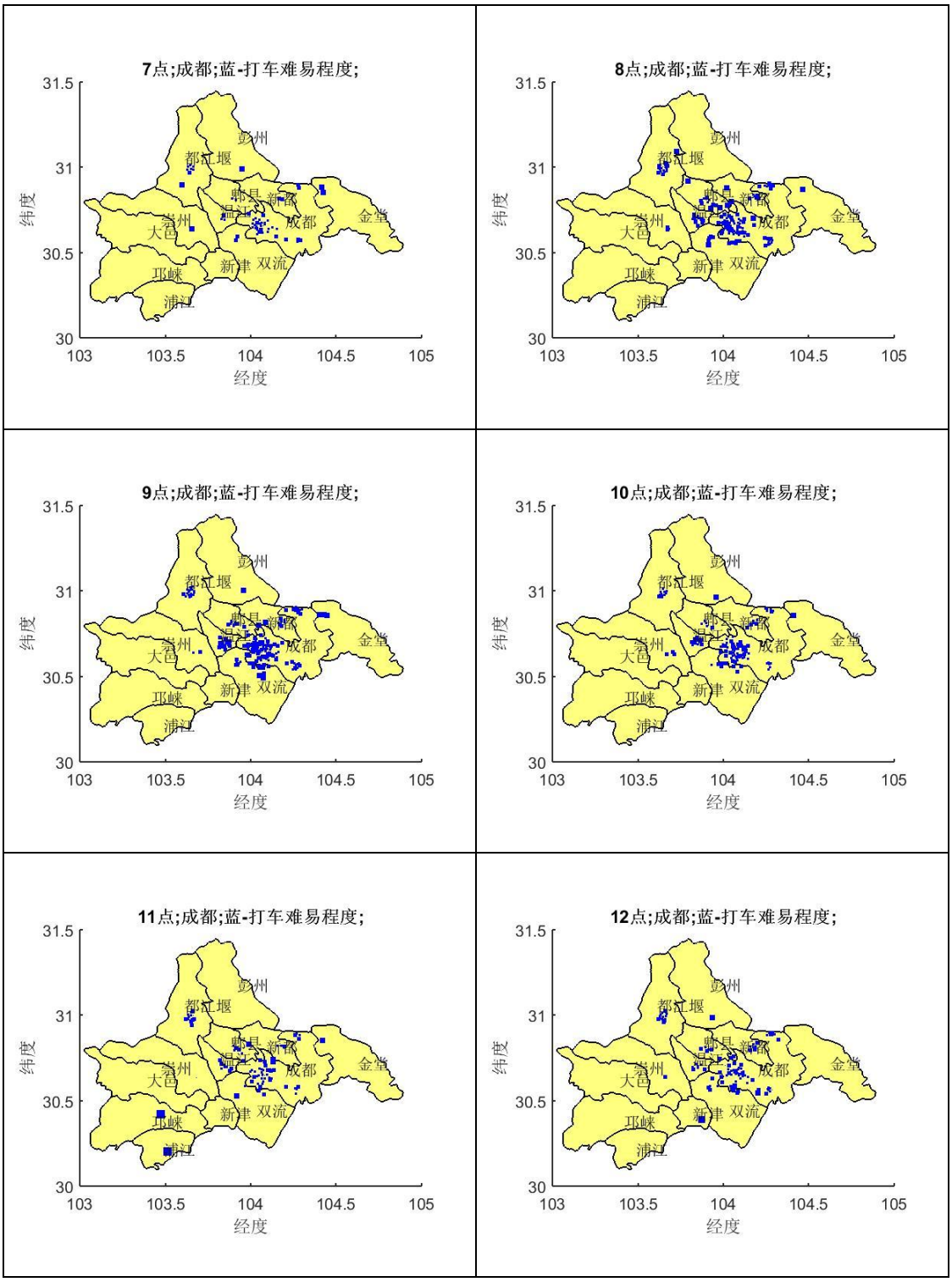


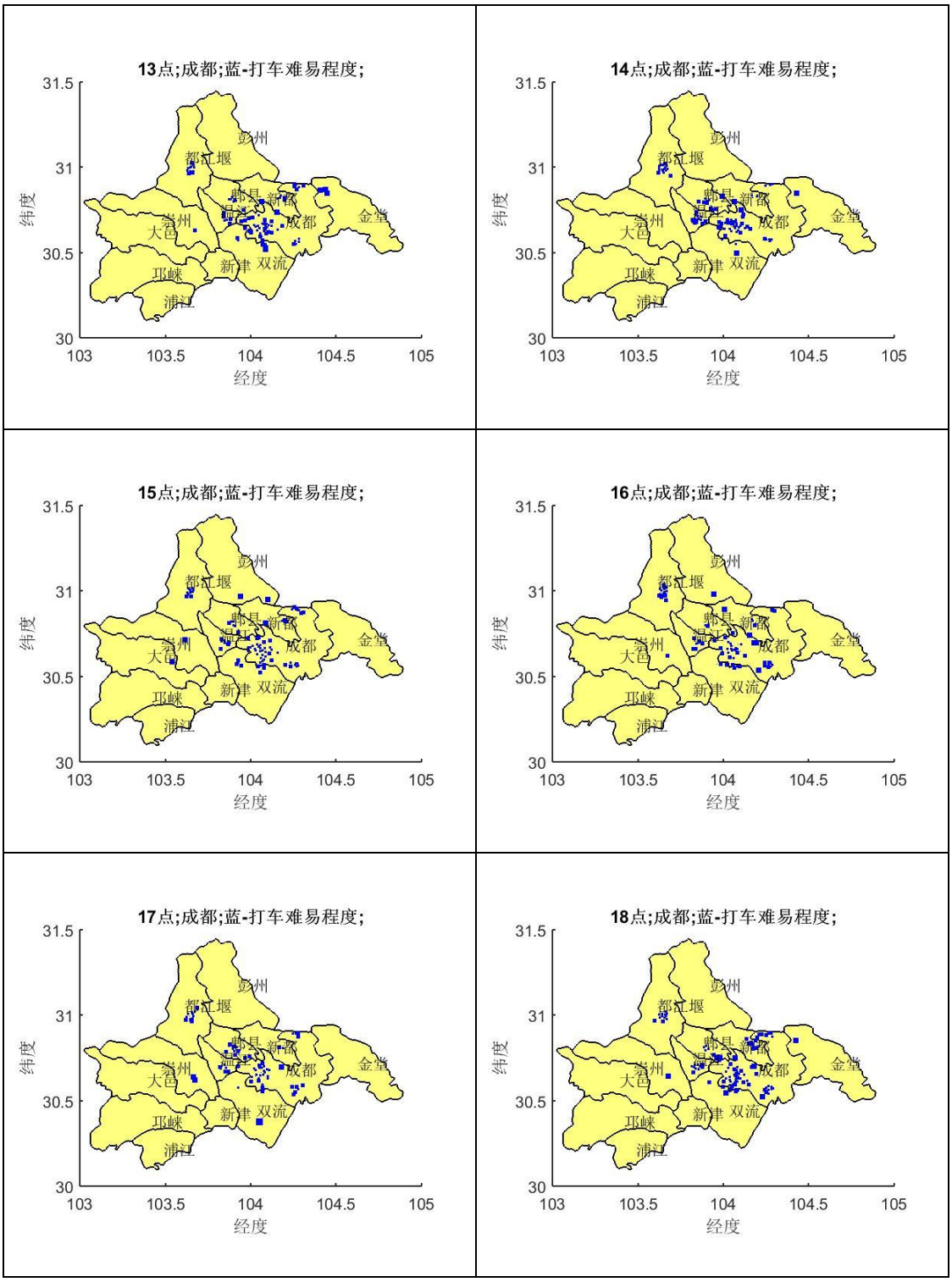

13点;成都;绿-出租;红-乘客
14点;成都;绿-出租;红-乘客
15点;成都;绿-出租;红-乘客
16点;成都;绿-出租;红-乘客
17点;成都;绿-出租;红-乘客
18点;成都;绿-出租;红-乘客
纬度
经度
31.5
31
30.5
30
103
103.5
104
104.5
105
彭州
都江堰
崇州
大邑
成都
金堂
新津
双流
邛崃
浦江

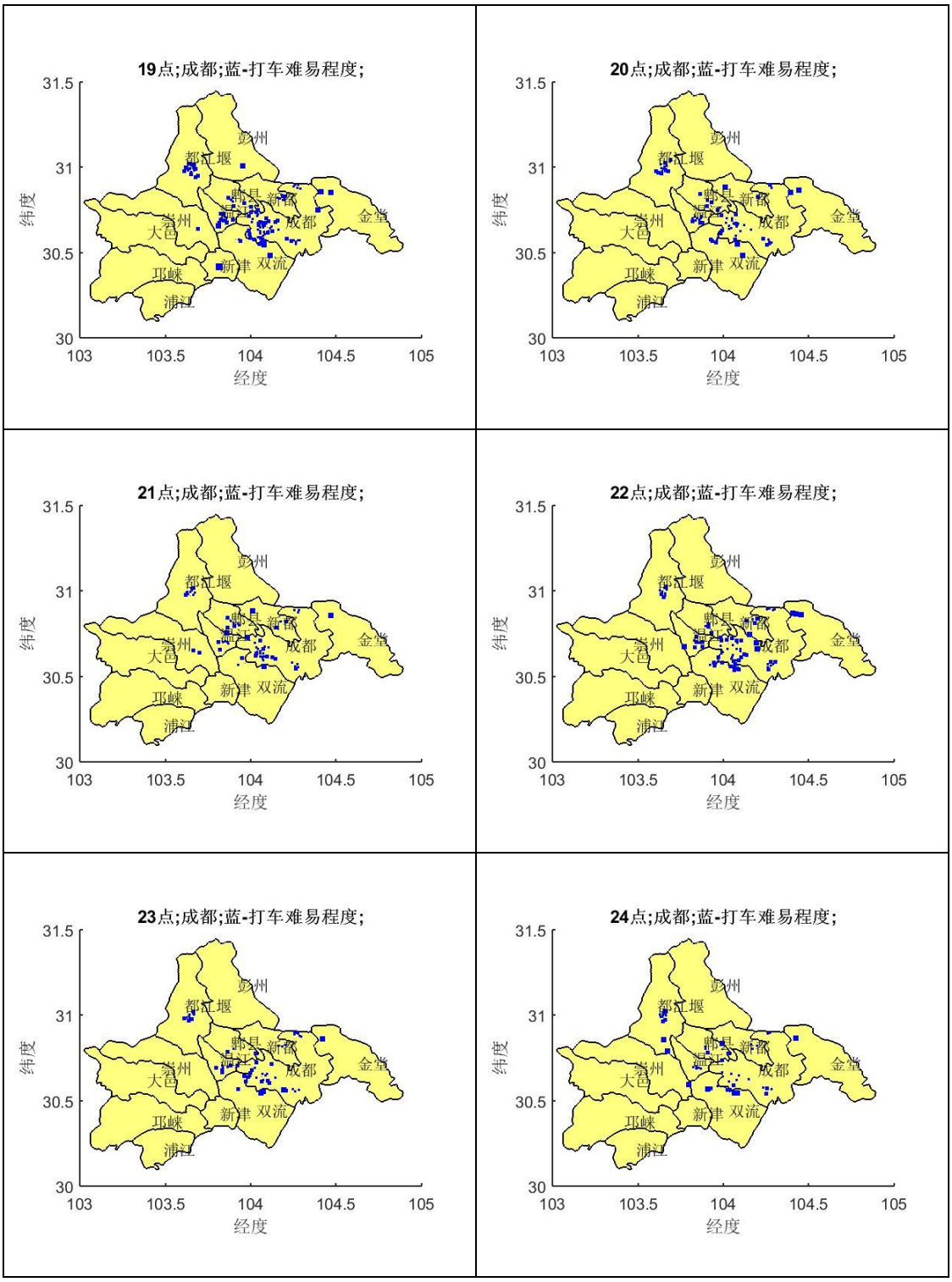


附录 B.成都市各个时点打车难易度分布图






13点;成都;蓝-打车难易程度;
14点;成都;蓝-打车难易程度;
15点;成都;蓝-打车难易程度;
16点;成都;蓝-打车难易程度;
17点;成都;蓝-打车难易程度;
18点;成都;蓝-打车难易程度;
纬度
经度
31.5
31
30.5
30
103
103.5
104
104.5
105
彭州
都江堰
崇州
大邑
邛崃
蒲江
新津
双流
成都
金堂



附录 C.Matlab 程序代码

1.问题(1)的代码

```
%问题(1)
sheng=shaperead('counties_china.shp', 'UseGeoCoords', true);
chengdu=sheng(2333:2345);
boundall=[];
name=cell(13,1);
for i=1:length(chengdu)
    boundall=[boundall;chengdu(i).BoundingBox];
    name{i}=chengdu(i).LAST_NAME9;
end
name{1}='成都';
name{2}='金堂';
name{3}='双流';
name{4}='温江';
name{6}='新都';
name{7}='大邑';
name{8}='浦江';
name{9}='新津';
name{10}='都江堰';
name{11}='彭州';
name{12}='邛崃';
name{13}='崇州';
minlon=min(boundall(:,1));%min(chengdu.Lon);
maxlon=max(boundall(:,1));%max(chengdu.Lon);
minlat=min(boundall(:,2));%min(chengdu.Lat);
maxlat=max(boundall(:,2));%max(chengdu.Lat);
dataall=[];
for i=1:24
    data=dlmread(['cityID510100realDate20150909_',num2str(i),'_t.txt']);
```

```
    tlen=length(data);
    nump=tlen/3;
    datat=zeros(nump,3);
    datat(:,1)=data(1:3:tlen);
    datat(:,2)=data(2:3:tlen);
    datat(:,3)=data(3:3:tlen);
    cancel=datat(:,1)>maxlon | datat(:,1)<minlon | datat(:,2)>maxlat | datat(:,2)<minlat;
    datat(cancel,:)=[];
    dataall(i).taxi=datat;
    data=dlmread(['cityID510100realDate20150909_',num2str(i),'_p.txt']);
    tlen=length(data);
    nump=tlen/3;
    datap=zeros(nump,3);
    datap(:,1)=data(1:3:tlen);
    datap(:,2)=data(2:3:tlen);
    datap(:,3)=data(3:3:tlen);
    cancel=datap(:,1)>maxlon | datap(:,1)<minlon | datap(:,2)>maxlat | datap(:,2)<minlat;
    datap(cancel,:)=[];
    dataall(i).passenger=datap;
end

%plot
fignum=[];
eachfignum=1;
for i=1:24
    datat=dataall(i).taxi;
    datap=dataall(i).passenger;
    if mod(i-1,eachfignum)==0
        k=ceil(i/eachfignum);
        hfig=figure(k);
```

```
        fignum=[fignum,hfig];
        if eachfignum==1
            set(gcf,'PaperType','A4', ...
                'paperOrientation', 'portrait', ...
                'paperunits','CENTIMETERS','PaperPosition',[.00, .00,10,8]);
        else
            set(gcf,'PaperType','A4', ...
                'paperOrientation', 'portrait', ...
                'paperunits','CENTIMETERS','PaperPosition',[.00, .00,21,29]);
        end
    end
    if eachfignum>1

subplot(ceil(eachfignum/2),2,mod(i,eachfignum)*(mod(i,eachfignum)~=0)+eachfignum*(mo
d(i,eachfignum)==0))
    end
    geoshow(chengdu);
    title([num2str(i),'点;黄-成都;绿-出租;红-乘客'])
    hold on
    scatter(datat(:,1),datat(:,2),datat(:,3),'filled','o','g')
    scatter(datap(:,1),datap(:,2),datap(:,3),'filled','s','r')
    title([num2str(i),'点;成都;绿-出租;红-乘客'])
    xlabel('经度')
    ylabel('纬度')
    for kk=1:length(chengdu)

text(mean(chengdu(kk).BoundingBox(:,1)),mean(chengdu(kk).BoundingBox(:,2)),name(kk));
    end
    hold off
    if mod(i-1,eachfignum)==0
        figname=strcat('chengdu_part',num2str(k),'.jpg');
```

```
            saveas(hfig, figname, 'jpg');
        end
end
savefig(fignum,'chengdu.fig')
close all
figs=openfig('chengdu.fig');
for k=1:length(fignum)
        figname=strcat('chengdu_part',num2str(k),'.jpg');
        saveas(figs(k), figname, 'jpg');
end
close all
%mean plot
meanall=[];
for i=1:24
        meanall=[meanall;[mean(dataall(i).taxi(:,3)),mean(dataall(i).passenger(:,3))]];
end
h=1:24;
set(gcf,'PaperType','A4', ...
        'paperOrientation', 'portrait', ...
        'paperunits','CENTIMETERS','PaperPosition',[.00, .00,16,10]);
[hAx,hLine1,hLine2]=plotyy(h,meanall(:,1),h,meanall(:,2));
ylabel(hAx(1),'出租车均值')
ylabel(hAx(2),'乘客均值')
set(hLine1,'LineStyle','--','Marker','*','MarkerSize',6,'LineWidth',1.5)
set(hLine2,'LineStyle',':','Marker','o','MarkerSize',6,'LineWidth',1.5)
xtlabel=num2cell(h);%cat(2,num2cell(h(1:4:end-2)),method);
set(hAx,'XTick',h,'XTickLabel',xtlabel,'xlim',[1,24])
grid on
xlabel('时间')
leglabel={'出租车均值','乘客需求均值'};
```

```
legend(leglabel,'Location','Best')
title('成都市 2015 年 9 月 9 日全天出租车分布和乘客需求均值图')
figname=strcat('chengdu_mean','.jpg');
saveas(gcf, figname, 'jpg');
%matching
C0=[0.1:0.1:0.9,1:50];
lc0=length(C0);
match=[];
matchall=[];
matchpoint=[];
for i=1:24
    p0=dataall(i).passenger;
    lp0=size(p0,1);
    t0=dataall(i).taxi;
    r=size(t0,1);
    disp=zeros(lp0,lc0);
    for j=1:lp0
        p00=ones(r,1)*p0(j,1:2);
        d=dis(p00(:,1),p00(:,2),t0(:,1),t0(:,2));
        for k=1:lc0
            ts=sum(t0(d<=C0(k),3));
            if ts<p0(j,3)
                disp(j,k)=1;
            elseif ts>p0(j,3)
                disp(j,k)=-1;
            else
                disp(j,k)=0;
            end
        end
    end
```

```
        idx=ones(lp0,1)*(1:lc0);
        idx(disp>0)=inf;
        idxsel=min(idx,[],2);
        idxsel(isinf(idxsel))=C0(end);
        match(i).loc=p0(:,1:2);
        match(i).matchp=disp;
        match(i).matchpstats=idxsel;
        matchall=[matchall;[i*ones(lp0,1),p0(:,1:2),disp,idxsel]];
        matchpoint=[matchpoint,lp0];
end
mathpointcum=cumsum(matchpoint);
newtime=zeros(size(matchall,1),1);
newtime(ismember(matchall(:,1),[8,9]))=1;
newtime(ismember(matchall(:,1),[18,19]))=2;
newtime(ismember(matchall(:,1),[7,10:17]))=3;
newtime(ismember(matchall(:,1),20:22))=4;
newtime(ismember(matchall(:,1),[1:6,23:24]))=5;
%plot
fignum=[];
eachfignum=1;
for i=1:24
        loc=match(i).loc;
        measure=match(i).matchpstats;
        if mod(i-1,eachfignum)==0
                k=ceil(i/eachfignum);
                hfig=figure(k);%figure(i);%figure(ceil(i/6));%
                fignum=[fignum,hfig];
                if eachfignum==1
                        set(gcf,'PaperType','A4', ...
                                'paperOrientation', 'portrait', ...
```

```
                    'paperunits','CENTIMETERS','PaperPosition',[.00, .00,10,8]);
            else
                set(gcf,'PaperType','A4', ...
                    'paperOrientation', 'portrait', ...
                    'paperunits','CENTIMETERS','PaperPosition',[.00, .00,21,29]);
            end
        end
        if eachfignum>1

subplot(ceil(eachfignum/2),2,mod(i,eachfignum)*(mod(i,eachfignum)~=0)+eachfignum*(mo
d(i,eachfignum)==0))
        end
        geoshow(chengdu); %  此处用 mapshow 投影会不正确
        hold on
        scatter(loc(:,1),loc(:,2),measure,'filled','s','b')
        title([num2str(i),'点;成都;蓝-打车难易程度;'])
        xlabel('经度')
        ylabel('纬度')
        for kk=1:length(chengdu)

text(mean(chengdu(kk).BoundingBox(:,1)),mean(chengdu(kk).BoundingBox(:,2)),name(kk));
        end
        hold off
        if mod(i-1,eachfignum)==0
            figname=strcat('chengdupipei_part',num2str(k),'.jpg');
            saveas(hfig, figname, 'jpg');
        end
end
savefig(fignum,'chengdupipei.fig')
close all
figs=openfig('chengdupipei.fig');
```

```
for k=1:length(fignum)
    figname=strcat('chengdupipei_part',num2str(k),'.jpg');
    saveas(figs(k), figname, 'jpg');
end
close all

%%%所有时点乘客打车难易度散点
set(gcf,'PaperType','A4', ...
    'paperOrientation', 'portrait', ...
    'paperunits','CENTIMETERS','PaperPosition',[.00, .00,16,14]);
plot(matchall(:,end),'*')
xlabel('乘客数据点')
ylabel('邻域半径(km)')
title('所有时点乘客打车难易度')
set(gca,'YTick',1:5:lc0)
xtlabel=cat(2,num2cell(C0(1:5:lc0)));
set(gca,'YTickLabel',xtlabel)
axis tight
grid on
figname=strcat('chengdupipei_scatter','.jpg');
saveas(gcf, figname, 'jpg');
%%%所有时点乘客打车难易度分布
set(gcf,'PaperType','A4', ...
    'paperOrientation', 'portrait', ...
    'paperunits','CENTIMETERS','PaperPosition',[.00, .00,16,14]);
geoshow(chengdu);
hold on
scatter(matchall(:,1),matchall(:,2),matchall(:,end),'filled','s','b')
title('所有时点;成都;蓝-打车难易程度;')
xlabel('经度')
```

```
ylabel('纬度')
for kk=1:length(chengdu)

text(mean(chengdu(kk).BoundingBox(:,1)),mean(chengdu(kk).BoundingBox(:,2)),name(kk));
end
hold off
figname=strcat('chengdupipei_all','.jpg');
saveas(gcf, figname, 'jpg');
```

2.问题(3)的代码

```
%问题(3)方案 1
time=8;
op=dataall(time).passenger;%选择 8 点的乘客数据
lp0=size(op,1);
ebt=10;%每天补贴 1 元
lbt=30;%补贴 30 天
bt=ebt*ones(1,lbt);
cbt=cumsum(bt);%累计补贴
b0=1.645;
baifenbi=@(butie)exp(b0-11.532./butie);%S curve
cper=baifenbi(cbt)/100;%预测订单数
t0=dataall(time-1).taxi;
r=size(t0,1);
matchn=[];
matchna=match(time).matchpstats;%8 点的难易度
for i=1:lbt
    p0=op;
    p0(:,3)=p0(:,3).*(1-cper(i));
    disp=zeros(lp0,lc0);
    for j=1:lp0
```

```
        p00=ones(r,1)*p0(j,1:2);
        d=dis(p00(:,1),p00(:,2),t0(:,1),t0(:,2));
        for k=1:lc0
            ts=sum(t0(d<=C0(k),3));
            if ts<p0(j,3)
                disp(j,k)=1;
            elseif ts>p0(j,3)
                disp(j,k)=-1;
            else
                disp(j,k)=0;
            end
        end
    end
    idx=ones(lp0,1)*(1:lc0);
    idx(disp>0)=inf;
    idxsel=min(idx,[],2);
    idxsel(isinf(idxsel))=C0(end);
    matchn(i).loc=p0(:,1:2);
    matchn(i).matchp=disp;
    matchn(i).matchpstats=idxsel;
    matchna=[matchna,idxsel];
end

%问题(3)方案 2
time=8;
op=dataall(time).passenger;%选择 8 点的乘客数据
[count,center]=hist(op(:,3));
count=count./sum(count);
set(gcf,'PaperType','A4', ...
    'paperOrientation', 'portrait', ...
```

```
    'paperunits','CENTIMETERS','PaperPosition',[.00, .00,10,8]);
hist(op(:,3))
ylabel('频数')
xlabel('乘客需求')
title('成都市 8 点乘客需求条图')
figname=strcat('chengdu8bar','.jpg');
saveas(gcf, figname, 'jpg');

%密集点
datat=dataall(8).taxi;
datap=dataall(8).passenger;
set(gcf,'PaperType','A4', ...
    'paperOrientation', 'portrait', ...
    'paperunits','CENTIMETERS','PaperPosition',[.00, .00,10,8]);
geoshow(chengdu); %  此处用 mapshow 投影会不正确
title('8 点成都乘客打车需求最大区')
hold on
scatter(datat(:,1),datat(:,2),datat(:,3),'filled','o','g')
[~,idx]=max(datap(:,3));
scatter(datap(idx,1),datap(idx,2),datap(idx,3),'filled','s','r')
legend('出租车','乘客','Location','Best')
xlabel('经度')
ylabel('纬度')
for kk=1:length(chengdu)

text(mean(chengdu(kk).BoundingBox(:,1)),mean(chengdu(kk).BoundingBox(:,2)),name(kk));
end
hold off
figname=strcat('chengdu8miji','.jpg');
saveas(gcf, figname, 'jpg');
```

```
%方案 2
datat=dataall(8).taxi;
datap=dataall(8).passenger;
sumt=sum(datat(:,3));
datat(:,3)=datat(:,3)*0.99;
datat(end+1,1:2)=datap(idx,1:2);
datat(end,3)=sumt-sum(datat(:,3));
p0=datap;
t0=datat;
r=size(t0,1);
disp=zeros(lp0,lc0);
matchn=[];
matchna=match(time).matchpstats;%8 点的难易度
for j=1:lp0
    p00=ones(r,1)*p0(j,1:2);
    d=dis(p00(:,1),p00(:,2),t0(:,1),t0(:,2));
    for k=1:lc0
        ts=sum(t0(d<=C0(k),3));
        if ts<p0(j,3)
            disp(j,k)=1;
        elseif ts>p0(j,3)
            disp(j,k)=-1;
        else
            disp(j,k)=0;
        end
    end
end
idxs=ones(lp0,1)*(1:lc0);
idxs(disp>0)=inf;
```

```
idxsel=min(idxs,[],2);
idxsel(isinf(idxsel))=C0(end);
matchn.loc=p0(:,1:2);
matchn.matchp=disp;
matchn.matchpstats=idxsel;
matchna=[matchna,idxsel];
set(gcf,'PaperType','A4', ...
    'paperOrientation', 'portrait', ...
    'paperunits','CENTIMETERS','PaperPosition',[.00, .00,12,8]);
plot(1:40,matchna(1:40,1),'ro',1:40,matchna(1:40,2),'b+')
title('方案 2 的应用前后对比')
xlabel('数据点')
ylabel('难易度')
grid on
figname=strcat('chengdu8mijiduibi','.jpg');
saveas(gcf, figname, 'jpg');
```