机场的出租车问题

摘 要

送客到机场的出租车司机都将会面临两个选择：前往到达区排队等待载客返回市区、直接空载返回市区拉客。本文主要研究出租车司机选择决策模型，并根据深圳宝安国际机场和深圳出租车的情况分析模型的合理性；同时综合考虑乘车效率和出租车收益，合理安排“上车点”和短途载客“优先权”。

针对问题一，建立一个模拟出租车司机决策的模型。司机决策的最终目的是使收益率（单位时间内的收益）最大。影响司机决策的因素有司机可观测到的确定信息——机场航班的抵达时间和“蓄车池”里已有的车辆数。司机可根据二者预测排队的等待时间，用于决定是否排队载客。由于机场的出租车载客收益与载客的行驶里程有关，机场排队载客还需分为一定比例的短途载客和长途载客，最终用所设符号刻画出排队载客和回市区拉客的不同收益率期望，从而做出决策。

针对问题二，在深圳宝安国际机场官网中提取每天的航班信息，模拟司机查看最近一段时间飞机抵达深圳的情况；同时搜集关于深圳出租车、交通路况的数据，为所设符号赋值，并代入问题一所建立的模型中，用 Matlab 绘制出排队载客和回市区拉客的收益率随排队等待时间的变化情况。二者图象交点所对应的排队时间即为临界值：若司机预估时间超出临界值，司机会选择返回市区拉客，若未超出，则选择排队等候载客。为验证模型的合理性，我们将从模型得出的结论与现实生活中的数据或常识进行比较；由临界值变化的灵敏程度分析对出租车起步价、里程价、市区平均行驶速度等变量的依赖性。

针对问题三，本模型综合分析了三种常见的出租车上客区构造，致力于在两车道狭窄情况下对机场上出租车上客进行合理规划，最终根据狭窄道路的局限性选出了最适宜的单车道上客模型。

针对问题四，现给予短途载客再次返回的出租车一定的“优先权”，使它们与长途载客的出租车的单位时间的收益尽量均衡。“优先”体现在这些出租车能直接返回上客区，减少第二次排队载客的时间，降低再次载客的时间成本。以正态分布模拟接客行驶时间的分布，沿用问题一的时间轴，用所设符号表达出短途车相对于长途车的平均收益，相对收益为 0 时所对应的行驶时间，即为区分短途载客和长途载客的标准。利用问题二的部分数据和上海浦东机场的出租车数据，得到划分时间标准并判断是否符合实际，以验证该模型的合理性。

关键词：机场；决策模型；出租车蓄车场；出租车上客区

一、问题重述

1.1 问题背景

大多数乘客下飞机后要去市区或周边的目的地，出租车是主要的交通工具之一。国内多数机场都是将送客（出发）与接客（到达）通道分开的。当出租车司机送客到机场后，他/她将会面临以下两个选择：

(A) 前往到达区排队等待载客返回市区：出租车必须到指定的“蓄车池”排队等候，依“先来后到”排队进场载客。

(B) 直接放空返回市区拉客。

两种选择的优缺点如下表所示：

表 1 两种选择的优缺点一览

方案	(A) 机场排队载客	(B) 空载回市区拉客
优点	有可靠的载客收益保证。	无需等待，时间成本较少。
缺点	等待时间长短取决于排队出租车和乘客的数量多少，需要付出一定的时间成本。	出租车司机会付出空载费用和可能损失潜在的载客收益。

通常有多年驾龄的司机做决策与其个人的经验判断有关，比如在某时间段抵达航班的多少和“蓄车池”里已有的车辆数等；在实际中，还有很多影响出租车司机决策的确定和不确定因素，其关联关系各异，影响效果也不尽相同。

如果乘客在下飞机后想打车，就要到指定的“乘车区”排队，按先后顺序在“上车点”乘车。机场出租车管理人员负责“分批定量”放行出租车进入“乘车区”，同时安排一定数量的乘客上车。

1.2 问题提出

根据以上背景，需要解决以下四个问题：

(1) 分析研究与出租车司机决策相关因素的影响机理，综合考虑机场抵达航班和出租车司机的收益，建立出租车司机选择决策模型，并给出司机的选择策略。

(2) 收集深圳宝安国际机场的航班信息及其深圳出租车的收费标准等数据，给出该机场出租车司机的选择方案，并分析模型合理性和对相关因素的依赖性。

(3) “乘车区”经常会出现出租车排队载客和乘客排队乘车的情况。现有两条并行车道，管理部门应如何设置“上车点”，并合理安排出租车和乘客，在保证车辆和乘客安全的条件下，使得总的乘车效率最高。

(4) 机场的出租车载客收益与载客的行驶里程有关，乘客的目的地有远有近，管理部门拟对某些短途载客再次返回的出租车给予一定的“优先权”，使这些出租车的收益尽量均衡，试给出一个可行的“优先”安排方案。

二、问题分析

2.1 问题一分析

对问题一，我们的任务是建立可模拟司机决策的模型：模型的输入包括机场航班的抵达时间和“蓄车池”里已有的车辆数，二者是司机已知的确定信息，也是影响司机作出选择的重要因素；模型的输出为司机的选择。

在研究机场排队载客的情况时，由于载客收益与行驶里程有关，需考虑长途载客和短途载客的差异和比例。此外，经验丰富的司机可依据“蓄车池”已有的车辆数，预判大致需要等待多少趟航班，从而根据航班抵达时间预估等待时间。

在研究空载回市区拉客的情况时，已知从机场附近到市区的时间，并需考虑机场附近和市区的速度差异、空载所损耗的油电费。

司机决定机场排队载客或是空载回市区拉客，取决于单位时间内预期收益的多少。

2.2 问题二分析

问题二实质上是对问题一所建立模型的分析与验证。经过深圳宝安国际机场网页数据提取和分析、互联网搜索深圳出租车的相关信息，我们可将数据代入上题模型中，用 Matlab 绘出单位时间内预期收益与预估的排队等待时间的关系，判断结论是否合乎实际；此外，由模型变化的灵敏程度分析对各变量的依赖性。

模型可视为输入与输出之间的一种转化。由于我们暂时收集不到司机在不同情况下作出的决策情况，故无法通过黑盒测试(Black-Box Testing)对模型反复迭代修改和验证本模型的合理性，但可通过表面效度证明从模型得到的一些结论符合实际[1]，详细阐明本模型的结构和数据如何反映现实情况。

2.3 问题三分析

问题三聚焦于机场乘车区双车道出租车上客模型，提出较高上客率的规划。由于双车道本身在机场中较为狭窄，故应先从可行模型中分析拥堵影响，挑出容易发生拥堵混乱的模型进行排除，以保证安全和管理。而后在安全的前提下，设计上车点和相应的调度规则，尽可能提高乘车效率。

对于上车点的设计，可以从上车点间距、车辆在上车点的排列形态、上客方式及道路占用大小等因素入手分析；对于调度规则，则可以从乘客放行策略、出租车放行策略及其他为追求提高效率而尽可能遵守某些行为约定等方向入手分析[2]。在一定假设的基础上，对不同的设计和策略规则组合，建立相应参数和规则简化模型，寻找使得乘车效率最高的作为方案参考。

2.4 问题四分析

问题四所提出的可行安排方案，要求为使得短途接客出租车和长途接客出租车的收益尽量均衡，从可行性的角度分析，对于每名出租车乘客都能保证利益相同是不太可能的。故对于出租车收益的均衡应该分为两部分，即长途载客和短途载客。本方案的最终目的为保证长途车与短途车的平均收益从总体上相同。

题干中已经确定方案需要基于“优先权”操作：“优先”体现在出租车经历一次短途载客后返回机场，能减少第二次排队载客的时间，即让短途车能以较低时间成本再次来到机场载客。由此可知，给予“优先权”的具体任务应为“判定短途车与长途车的行驶时间分界值”，总体上使收益相对较差的短途车由于“优先权”也能获得不错的收益，从而弥补短途车的损失，满足方案的最终目的。

三、模型假设与符号说明

3.1 模型的假设

(1) 本模型旨在模拟司机的选择策略，由于人在日常生活中仅使用少量信息对未来进行简单的预估，因此模型只研究当下情况的一次决策，而不进行长远推算和全局优化。

(2) 司机均富有经验，能对机场排队载客策略中的等待时间以及空载回市区拉客策略中的返回时间有一定的衡量标准。

(3) 模型中只考虑机场和市区两个地点，不考虑属于同一部分的出租车之间行驶情况的差异，只考虑司机在机场和市区的营运差异。

(4) 并联式流线模式“蓄车池”可以有效地应用在大量用车的场所[3]，各运行流线是单行道，不允许进入“蓄车池”后中途折返。

(5) “乘车区”一定有车进行排队，不存在出租车车辆数不足的现象。

3.2 符号的说明

表 2 符号的说明

符号	说明	单位
v	市区平均行驶速度	km/h
v'	机场附近道路平均行驶速度	km/h
M_0	里程价：出租车每公里的价格	元/km
Q_0	出租车起步价	元
s	出租车起步里程	km
L	里程利用率：载客里程占总里程的百分比	%
n	平均每小时接单的数量	个/h

t	机场排队载客时司机所预估的等待时间	h
c	平均每公里的消耗费用（燃油、电等）	元/km
$\boldsymbol{F}$	观测到的飞机即将抵达的时间组成的列向量	
$N_{车}$	观测到的“蓄车池”里已有的车辆数	辆
η	机场载客短途率	%
T	司机返回市区需要的时间	h
T_i	$i=1$，长途载客的时间 $i=2$，短途载客的时间	h

四、模型建立与求解

4.1 问题一

本题出租车司机决策相关因素包括航班抵达信息 $\boldsymbol{F}=[f_1,f_2,\cdots f_n]^T$（其中 $f_i<f_{i+1}$，即 f_i 比 f_{i+1} 晚，$i=1,2,\cdots n-1$）和“蓄车池”里已有的车辆数 $N_{车}$ 等。根据司机的经验，一趟航班的抵达将让 $N_{航}$ 辆车载客，故司机预估若排队等候载客，将至少需要等 $m=\lceil N_{车}/N_{航}\rceil$ 趟飞机抵达机场，再根据即将抵达的航班信息找到 f_m 对应的时间，对照此时 t_{now} 即可预估需要的等待时间 t：

$$t=f_{\lceil N_{车}/N_{航}\rceil}-t_{now}$$

设从在时间轴上看，机场排队载客的两种情况如下图(a)(b)所示，空载回市区拉客的情况如下图(c)所示：

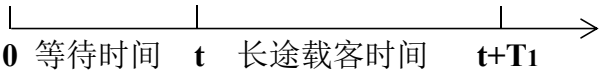


图 1(a) 机场排队载客之长途载客

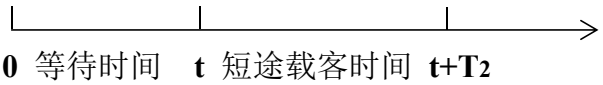


图 1(b) 机场排队载客之短途载客（其中 $T_2<T_1$）

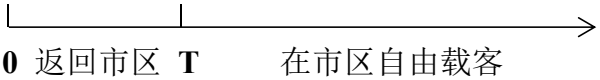


图 1(c) 空载回市区拉客

假设机场到市区的行驶里程大于起步里程，定义补偿起步价

$$Q_0'=Q_0-M_0\cdot s$$

从而可将乘客车费简化为$Q_0'+M_0\times$行驶里程。

若选择机场排队载客，则在$t+T_i$这段时间内的收益=乘客付的车费-油电消耗费（当$i=1$时为长途载客，当$i=2$时为短途载客），用所设符号描述为：

$$Q_{等i}=T_i\cdot v'(M_0-c)+Q_0'$$

若选择空载回市区拉客，在相同时间段$t+T_i$内的收益=市区载客的车费收入-全程的油电消耗费，用所设符号描述为：

$$Q_{回i}=-v'Tc+[v(LM_0-c)+nQ_0'](t+T_i-T)$$

司机决策目的为最大收益率，即单位时间内获得收益最大。收益率定义如下：

$$W_{等i}=Q_{等i}/(t+T_i)\qquad W_{回i}=Q_{回i}/(t+T_i)$$

现加入机场载客短途率，计算收益率的期望：

$$\begin{bmatrix}W_{等1} & W_{等2}\\ W_{回1} & W_{回2}\end{bmatrix}\begin{bmatrix}1-\eta\\ \eta\end{bmatrix}=\begin{bmatrix}W_{等}\\ W_{回}\end{bmatrix}$$

其中$W_{等}$是选择在机场排队载客的收益率期望，$W_{回}$是选择回市区拉客的收益率期望。司机通过二者比较择优进行决策，在下一部分中将代入深圳宝安国际机场和深圳出租车的相关数据验证本模型。

4.2 问题二

4.2.1 信息收集与代入模型

通过在深圳宝安国际机场官网上对航班信息的抓取，我们搜集了 2019 年 9 月 12 日-13 日的航班抵达时间，一天中时间分布直方图如下：

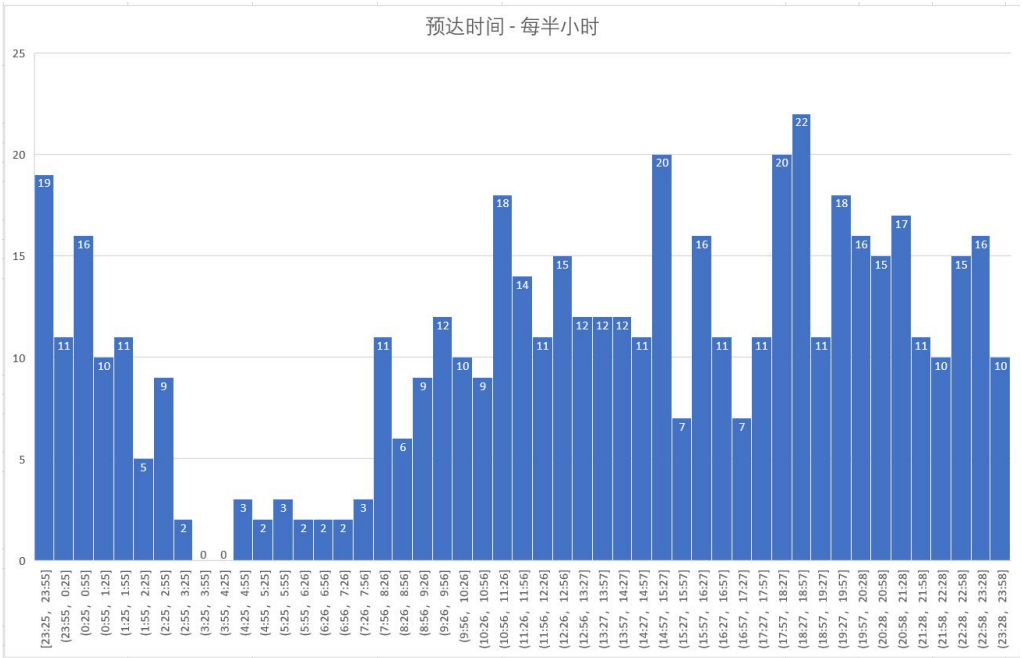


图 2(a) 深圳宝安国际机场 2019 年 9 月 12 日的航班预计抵达时间直方图

图 2(b) 深圳宝安国际机场 2019 年 9 月 13 日的航班预计抵达时间直方图

现实生活中司机也很容易在飞猪、携程等 APP 上获取这些信息，再结合“蓄车池”里已有的车辆数 $N_{车}$，从而根据在 4.1 节推导的公式

$$t = f_{\lceil N_{车}/N_{航} \rceil} - t_{now}$$

预测排队等待时间 $t_{估}$。

现根据搜集到的深圳出租车等信息，对模型中的符号进行赋值：

根据 2019 年深圳出租车最新收费标准[4]，深圳出租车起步里程为 $s = 2km$，起步价为 $Q_0 = 10.0$元，里程价为 $M_0 = 2.6$元$/km$；

在市区行驶时里程利用率[5]（即跑一公里有多少公里产生效益）$L = 69\%$；

高德地图显示白天时段的市区平均行驶速度 $v = 25km/h$，机场附近道路平均行驶速度 $v' = 50km/h$，并假设司机返回市区所需的时间 $T = 0.4h$（以从深圳宝安国际机场至华侨城为例）；

假设在深圳市区每个小时的接单数量 $n = 2$个$/h$；

据深圳交委官方宣布，截至 12 月底，落地运营的纯电动出租车占比高达 94.21%，深圳出租车基本实现纯电动化[6]。由粤易充中对耗电费用的收取和电动车每公里耗电量，可估计每公里的汽车损耗费用（电费等）$c = 0.15$元$/km$；

假设长途载客的时间 $T_1 = 0.8h$，短途载客的时间 $T_2 = 0.3h$；

据报道[7]合理推测深圳机场载客短途率 η 在 15%左右，取区间[5%,30%].

故由 4.1 的推导可绘出机场排队载客的收益率 $W_{等i}$ 和空载回市区拉客的收益率 $W_{回i}$ 随排队等待时间 t 的关系（当 $i = 1$ 时为长途载客，当 $i = 2$ 时为短途载客），见下页图 3：

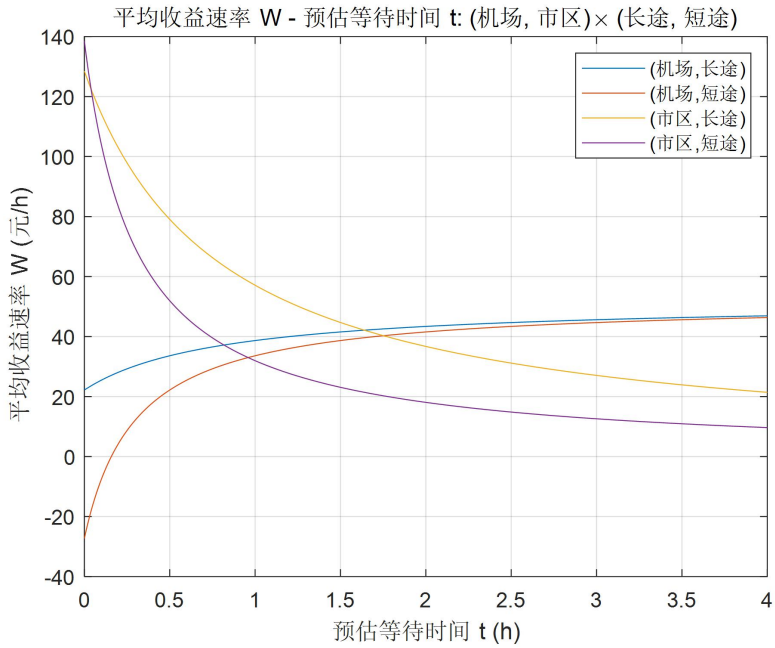


图 3 两种选择和长途/短途载客情况下的平均收益率

$W_{等i}$ 和 $W_{回i}$ 图象的交点对应的排队时间设为 $t_{tolerance\,i}$，即（机场，长途）和（机场，短途）两条曲线的交点为 $t_{tolerance\,i}$，（机场，长途）和（机场，短途）两条曲线的交点为 $t_{tolerance\,i}$ 分别表示在长途载客和短途载客情况下不同的临界值。

若再加入机场排队载客决策中短途载客的比例因素 η，则可得到在机场排队载客的收益率期望 $W_{等}$ 和选择回市区拉客的收益率期望 $W_{回}$：

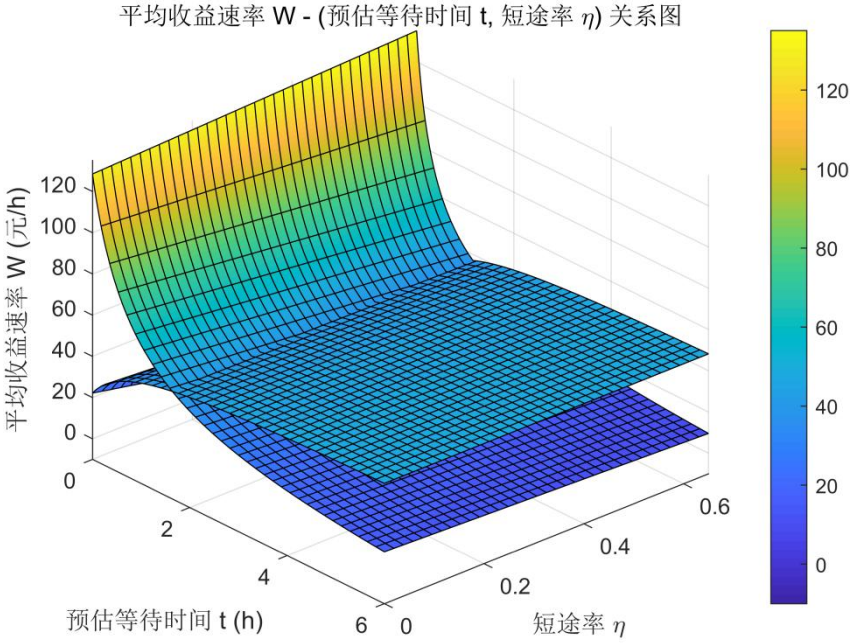


图 4 两种决策的收益率期望与预估等待时间、短途率的关系

4.2.2 模型的合理性

由上图可见，预估等待时间越少，机场排队载客的平均收益率就越高，而回市区需要一定时间，其平均收益率就越低；相反，预估等待时间越长，机场排队载客所付出的时间成本就越高，因而收益率迅速降低。随着短途率的升高，机场排队载客的收益速率越来越慢，这也是问题四中所要解决的。故上图曲面中的变化趋势符合我们通过常识得出的预判，从表面效度方面验证了该模型的合理性。

取$\eta=15\%$的横截面进行分析，如下图所示：

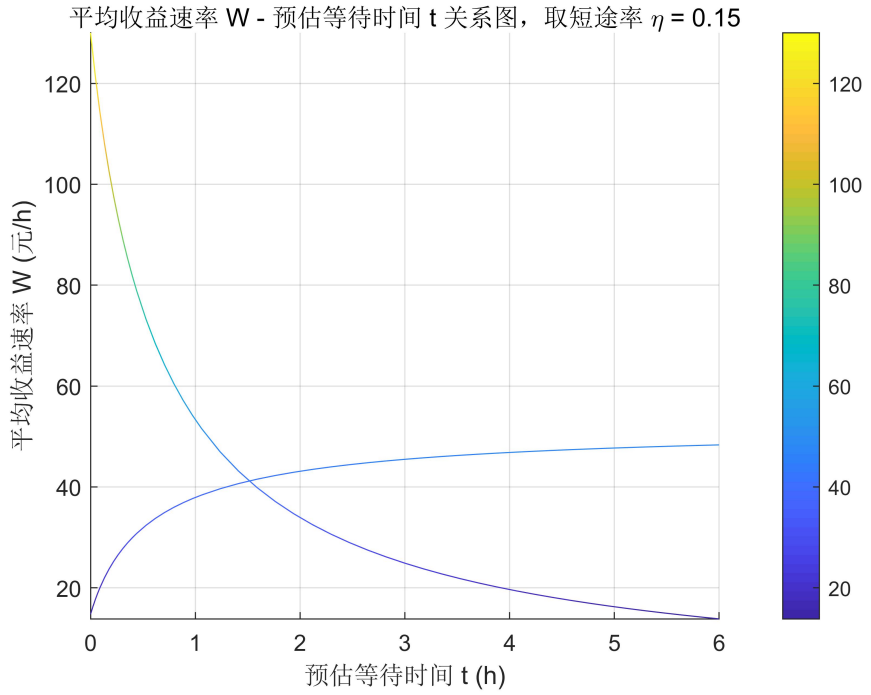


图 5 对图 4 中取一横截面找到交点并进行决策

可见$W_{等}$和$W_{回}$随排队时间的变化而产生交点。设该交点所对应的排队时间值为$t_{tolerance}$，则司机之前的预估$t_{估}>t_{tolerance}$时会选择返回市区拉客，若$t_{估}\le t_{tolerance}$则在机场排队等待载客。本图中$t_{tolerance}$=1.5h，参考报道[7]中提供的机场出租车平均排队等待时间约 2 小时，而司机对两种选择的临界值应略低于平均等待时间，故本模型所得结论符合实际。

4.2.3 相关因素的依赖性

下面考虑出租车起步里程s、起步价Q_0、每公里的价格M_0、里程利用率L、市区平均行驶速度v、机场附近道路平均行驶速度v'、在市区每小时的接单数量n、平均每公里的消耗费用（燃油费/电费）c等这些相关因素对临界值$t_{tolerance_i}$的影响：（括号内前者为长途载客的临界值$t_{tolerance_1}$的解析式，后者为短途载客的临界值$t_{tolerance_2}$的解析式）

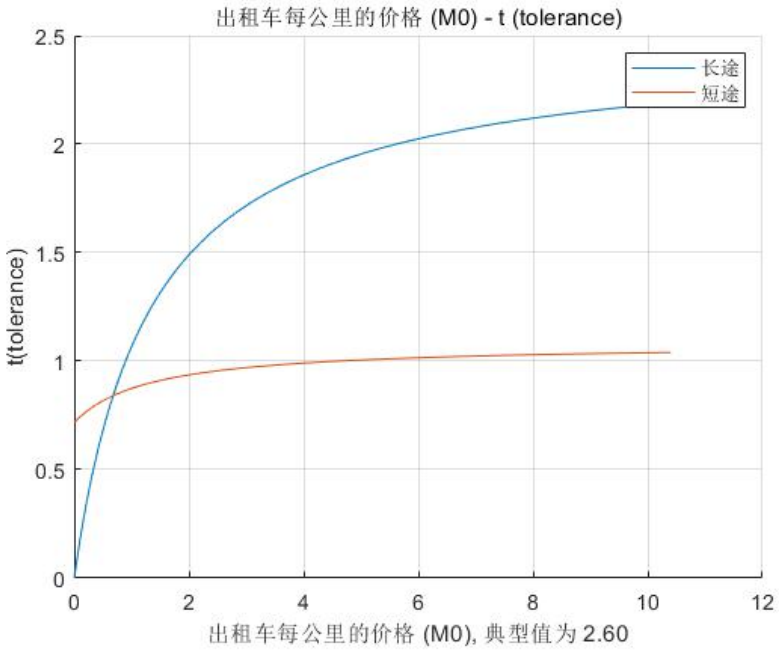


图 6 出租车每公里的价格变化对临界值的影响

其解析式为 $\left(\frac{13080\,M_0+14}{5300\,M_0+6965}\quad\frac{11460\,M_0+9993}{10600\,M_0+13930}\right)$

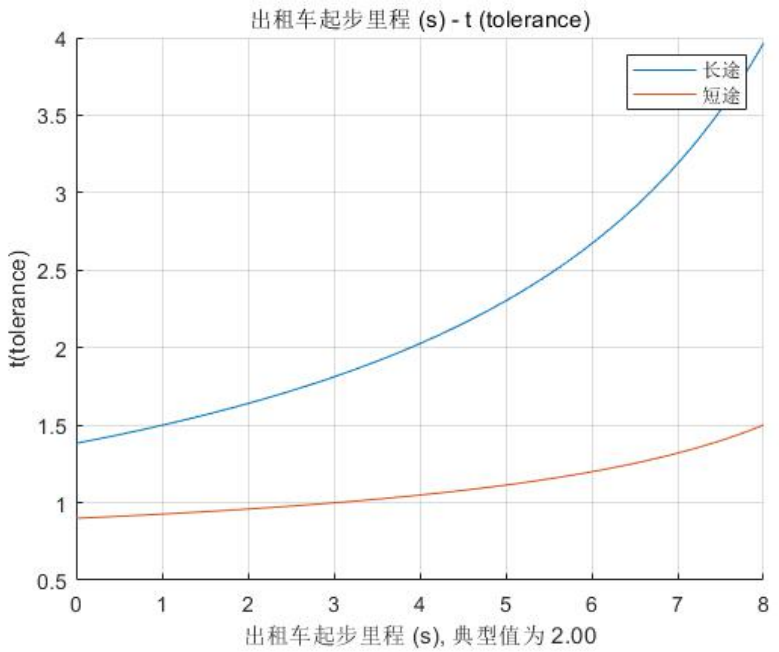


图 7 出租车起步里程对临界值的影响

其解析式为 $\left(\frac{208\,s-34438}{2080\,s-24905}\quad\frac{2496\,s-44781}{4160\,s-49810}\right)$

由于篇幅有限，其他因素的影响情况将在附录 3 中呈现。由这些图象横向比较，这些因素的变化一般对长途载客影响较为剧烈一些；纵向比较其解析式和部分图象斜率，模型决策的临界值 $t_{tolerance_i}$ 对每公里里程价格 M_0、里程利用率 L 和

市区平均行驶速度 v 的依赖性较高，而对每公里的汽车损耗费用（燃油费/电费）c 依赖性较低。

4.3 问题三

结合自身经验调研，得知目前机场乘车区设计普遍具有以下特点：

1. 整体呈条带形；
2. 来自蓄车区的空载出租车沿路排队载客，待乘车乘客在路边站台排队等待上车。
3. 站台通常设置排队护栏，站台与道路隔开，乘客只能从站台指定的上车点进入道路上车。
4. 此外，往往还设有现场指挥引导人员，负责指挥人员和车辆出入。

以上设计在实践中能够被普遍采纳，说明其安全性和效率基础都相对较好，因此接下来的模型将在如上设计特点的基础上进行构建。

因此根据现实中的常用模型，提出三种上客模型：单车道上客模型、双车道上客模型，综合斜列式上客模型。

图 8 单车道上客模型[8]

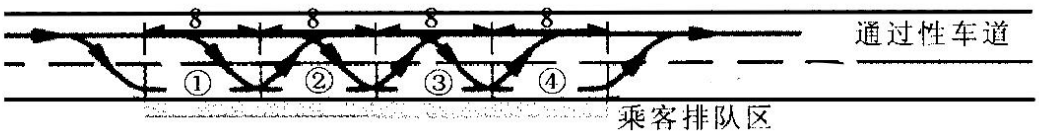


图 9 双车道模型[9]

其中双车道模型由于前端车辆上客会影响整个道路的同行，在只有两条车道的情况下会造成整体同行出租车速率的大幅降低，造成严重拥堵。就其上客本身，后端出租车的上客会在一定程度上使得前端乘客无法搭车。从简单的定性分析来看，此模型不能够适用于狭窄的双车道模型，只能试用于三车道和四车道的模型，这恰恰符合使用该模型的现实情况（双流机场等）。

综合斜列式模型由于其上客的规则（倒车进入上客泊车区），会严重堵塞主同行道，而且同样，同行道的堵塞会反过来影响泊车位上的车辆进入通行道。目前使用该模型的机场（浦东机场等）也是在多车道的条件下应用的。

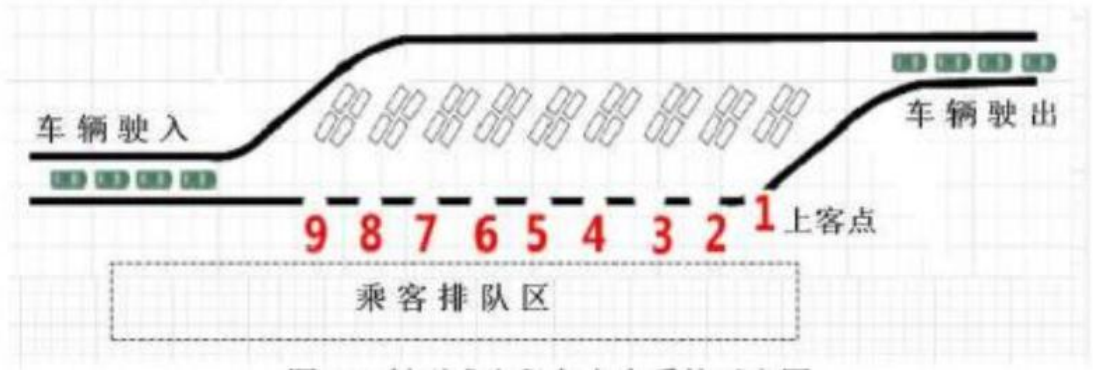


图 10 多车道模型[8]

根据现实数据和定性分析，只能采用单车道模型进行出租车上客。根据题意可将每个上车点的构造表示如下：

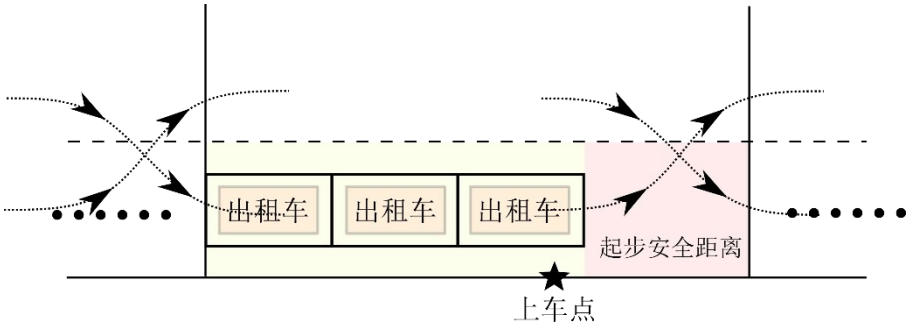


图 11 每个上车点的构造

上图主要描绘的是一个上车点所配套的一段基本上车区，该区包括 N 个（图示为 3 个）出租车车位和一段起步安全距离。设计多个出租车车位的目的在于，后续入场的出租车可在乘客上车期间及时“补车”就绪，从后方驶入上车区车位排队等候，起到小型缓冲池的作用。

记上图出租车车位的长度为 l，且 l 大于出租车的车长，使得车位长度 l 可视为安全停车距离与车长之和。起步安全距离（红色底色）长度为 d，提供出租车在驶出和驶入上车区并道时所需要的空间。由此，记一个上车点所配套的基本上车区的长度为 $L_{区} = N \cdot l + d$。

不同乘客从踏入上车区开始上车到最终乘车离开（车辆驶离基本上车区）所需要的时间不同，具有一定的分布特点，称该时间为乘客上客时间 $T_{客}$。如下图所示为某机场所统计的出租车乘客上客时间累计频率曲线。

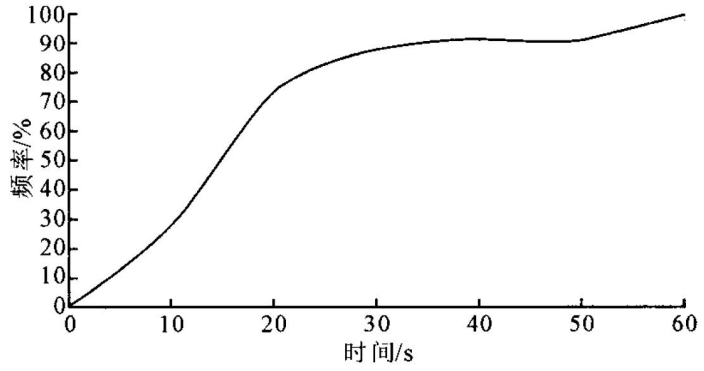


图 12　出租车乘客上客时间累计频率曲线[9]

在实践中根据大量观测测算出一个典型的乘客上客时间，记该值为 $T_{客\text{-}典型}$。

从排在上车区最前面的出租车开始驶离上车区，至后一辆出租车前跟补位停稳，也需要花费一定时间，并且在这一段时间内，为保证人车安全，乘客不能踏入上车区，记这段时间为 $T_{跟}$。

至此，我们可以得到一辆出租车接一次乘客的平摊时间为 $T_{客}+T_{跟}$，其中 $T_{客}=T_{客\text{-}典型}$。据此建立单个上车区运行效率评价指标 $\theta=1/L_{区}\cdot(T_{客}+T_{跟})$，该参数的意义是考量平均每单位时间每单位上车区长度能通行的车辆数。

现在考虑上客区旁的行车道的繁忙状况。行车道的繁忙情况与一段时间内，与由数个基本上车区组成的整个上客区发车数有关。实际上，发车还伴随着补车。若车道过度繁忙，则已载客出租车驶出基本上车区所需要的时间就会相应增加，这是因为司机在驶离并道时需要避让等待的情形会出现得更频繁。所以，此时乘客上客时间 $T_{客}$（计算到乘客离开基本上车区为止）应当加以修正，遂引入修正 $T_{客}=T_{客\text{-}典型}+k\theta$，其中 $k>=0$，为修正系数。

显然，修正后的系统构建了一个负反馈规则：若要提高上客效率 θ，则考虑减小 $L_{区}$ 或 $T_{客}$，但是 θ 增加会导致会导致 $T_{客}$ 增加。具体地，可以将符号带入，得到目前 θ 所需要满足的解析式：

$\theta=1/L_{区}\cdot(T_{客}+T_{跟})=1/[(N\cdot l+d)(T_{客\text{-}典型}+k\theta+T_{跟})]$由上述式子，可以探究 θ 与 N 的关系，以合理设置基本上车区的车位数量。

4.4 问题四

沿用问题一中建立的模型，机场排队载长途客情况如下：

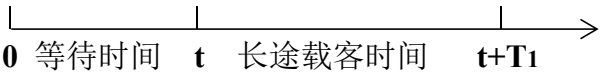


图 13 问题 4 中机场排队载客之长途载客

在“优先权”方案已经实行的情况下，基于模型的最终目的，对于短途载客，优先权给予了司机可观的再接客收益。若司机不选择返回机场，将会直接承受机场短途载客的损失，因此，司机在经历一次短途载客后必定会返回机场载客，直至长途载客为止，又因为短途率η较低（约 15%），进行二次短途载客的几率非常小，故本模型只对典型的一次短途接客的情况分析如下：

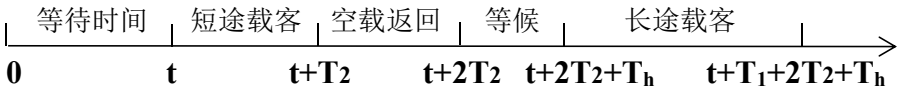


图 14 问题 4 中短途载客+长途载客

在上图短途载客模型中，司机在短途接客后应直接返回机场再次进行接客，该时间内司机处于空载状态，之后直接进入机场出租车上客区，由于机场设计原因，在上客区仍需要排队等待一定时间，等待时间设为T_h。

根据模型的目的，应该直接对两种情况收益进行比较，由于等待时间和长途接客均存在于短途接客和长途接客模型中，因此在这里为方便比较，先对短途载客进行等效比较转换如下：

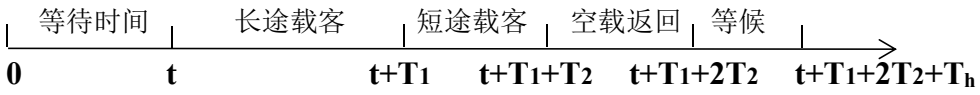


图 15 问题 4 中短途载客+长途载客等效结构

由此可计算出，短途接客相对于长途接客的收益如下：

$$\Delta Q=\left[T_2v'\left(M_0-c\right)+Q_0-T_2v'c\right]-\left(2T_2+T_h\right)\left[v\left(M_0L-c\right)+nQ_0\right]$$

该处应该要考虑整个载客路途的分布，才能合理地判断长短途行驶时间分界值，对于机场对短途车的判定也同样可行。

假设机场的接客行驶时间概率密度分布函数已知，横坐标 *x* 表示短途载客行驶时间，而 *f(x)*表示对应行驶时间载客的概率密度。设定 *m* 为表示长短途行驶时间分界值。

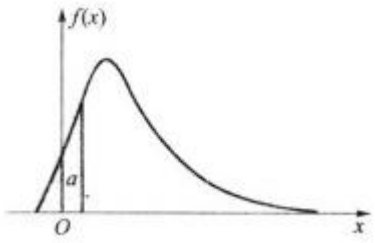


图 16 概率密度分布函数示意图

现将具体的 T_2 用正态函数连续化表示，则短途载客行驶时间 x 对应相对收益可化为：

$$\Delta Q(x)=\left[xv'(M_0-c)+Q_0-T_2v'c\right]-(2x+T_h)\left[v(M_0L-c)+nQ_0\right]$$

其中 T_h 可以假设与短途接客出租车数量有关，因为所有的短途车必定会返回机场接客，所以 T_h 与短途率有关，可以假设如下：

$$T_h=\eta t$$

其中 η 为机场载客短途率，t 为机场排队载客时司机所预估的等待时间。

短途率可以通过概率密度分布函数进行计算：

$$\eta = \int_0^m f(x)dx$$

为了描述所有短途车的相对平均收益，可通过概率的加权平均表示如下：

$$\Delta q=[\int_0^m \Delta Q(x)f(x)dx]/\ \eta$$

当 Δq 为零时，即所有短途车的相对平均收益为零时，达到问题叙述的“出租车的收益尽量均衡”，根据机场的接客行驶时间概率密度分布函数可以计算出长短途行驶时间的分界值，对于短程出租车实行“优先权”政策，即可保证收益相对均衡。

下面给出模型的应用示例：本模型优点在于直观且可操作性强，可以从生活中获取简单数据并作出合理假设，即可通过模型计算出一个较为准确的参考值。

为了验证问题四中模型的合理性，现用上海浦东机场进行验证。假设机场的接客行驶时间概率密度分布函数呈如下正态分布，其它数据与问题二保持一致：

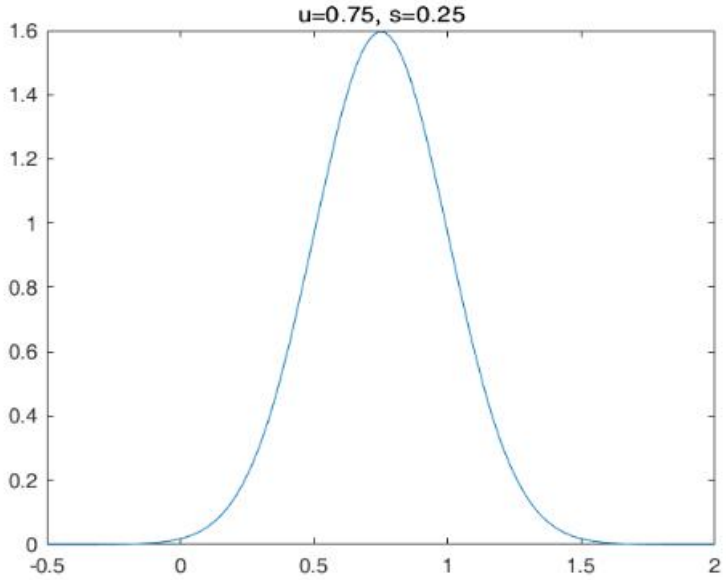


图 17(a) 接客行驶时间的概率密度分布函数（简化为正态分布）

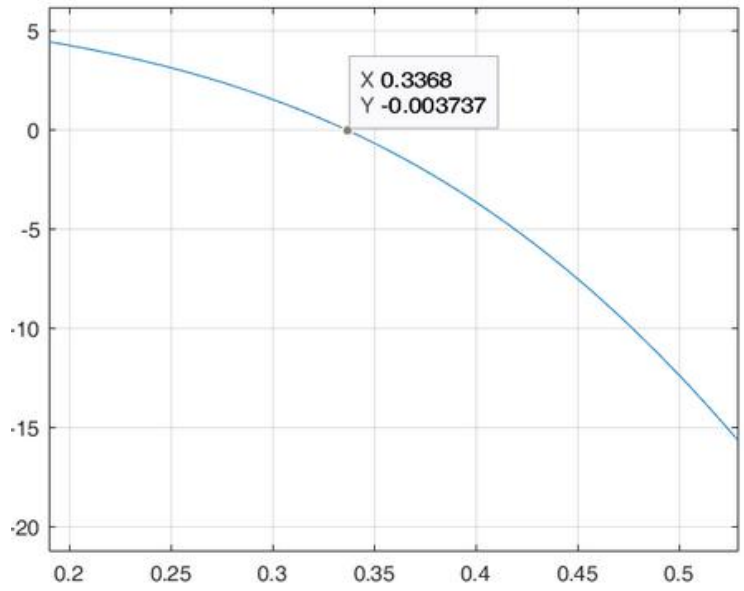


图 17(b) 短途车的相对平均收益函数

上图中纵轴表示短途车的相对平均收益，横轴表示行驶时间。

令 $\Delta q(m)=0$，可见 x 约为 $0.34h$，即行驶时间在 20 分钟内的短途车应该获得“优先权”，考虑到返程时间和候车误差，模型得到的结论为：在 50 分钟内返回机场上客区的出租车可以享有“优先权”。根据新闻[10]显示，浦东机场为了使出租车收益尽量均衡，制定了机制为“持有短途票、出租汽车行驶里程在 22 公里以内且在一小时内返回的，方可进入缓冲区”。模型所得结论较符合事实，从表面效度看有一定的可行性。

五、模型评价与改进

5.1 模型的优点

(1) 在问题 1 和问题 2 线性决策模型中，优点在于其稳定性和可移植性强。判定标准是司机较为关心的单位时间内的收益，模型输出会给出明确的选择方案。若给定某城市机场的抵达时间安排和出租车的起步价、起步里程等数据，可得到选择两种方案之一的预估排队时间临界值，从而可移植到不同城市送客到机场的出租车的模拟。

(2) 在问题 4 中，优点在于直观且可操作性强，可以从生活中获取简单数据并作出正态分布等合理假设，即可通过模型计算出一个较为准确的参考值，在 4.4 节利用上海浦东机场的数据估算，在现实生活中较为可行。

5.2 模型的缺点与改进方向

在问题 1 和问题 2 的决策模型中，通过表面效度可部分验证其合理性，但由

于建立模型过程中没有司机真实决策作为参考，进行迭代修改、比较和再修改，从而难以用黑箱测试来验证。若可收集到真实情景下司机作出的选择，该模型将进一步优化，更加贴合司机所做的决策。

问题 4 的模型的局限性在于仅建立了简单的负反馈机制，并且表示方法略有瑕疵，并没有深挖一个非常具体而贴切的模型。

参考文献

[1] 宋承龄. 关于仿真模型验证[J]. 计算机仿真, 2000, 17(4):8-11.

[2] 黄岩, 王光裕. 虹桥机场 T2 航站楼出租车上客系统组织管理优化探讨[J]. 城市道桥与防洪, 2014(12):7-9.

[3] 张明宇,陈妙洁.出租车蓄车楼及其运行模式分析[J]. 建筑学报, 2012(2):84-88.

[4] 2019 年深圳出租车最新收费标准一览. http://jt.sz.bendibao.com/news/2018531/804711.htm

[5] 各城市出租车相关数据统计表. https://wenku.baidu.com/view/6267d84cb42acfc789eb172ded630b1c59ee9bbc.html

[6] 深圳出租车基本纯电动. http://sz.people.com.cn/n2/2019/0104/c202846-32490435.html

[7] 蓄车场排队时间缩至 2 小时 大数据让浦东机场出租车“车等人”. http://sh.eastday.com/m/20170929/u1ai10894685.html

[8] 吴娇蓉, 李铭, 梁丽娟. 综合客运枢纽出租车上客点管理模式和效率分析[J]. 交通信息与安全, 2012, 30(4):18-23.

[9] 孙健. 基于排队论的航空枢纽陆侧旅客服务资源建模与仿真[D].中国矿业大学(北京),2017.

[10] 浦东机场终结短途票潜规则. http://news.ifeng.com/a/20141226/42807341_0.Shtml

附录

附录 1：完成深圳宝安国际机场官网上航班信息提取的源代码

(1) fetch.py

```
import logging
import requests
from pyquery import PyQuery as pq
import re
from datetime import datetime
import pickle

logging.basicConfig(level=logging.DEBUG,

format="[%(asctime)s] %(name)s %(levelname)s %(message)s",
                   datefmt='%Y-%m-%d %H:%M:%S'
                   )


# 抓取网页内容
def fetch(limit_num=''):
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0)
Gecko/20100101 Firefox/69.0'
    }
    data = {
        'flag': 'WHERE',
        'hbxx_hidmdd': '',
        'hbxx_hidhkgs': '',
        'istime': 'WHERE',
        'hbxx_jcbz_ds': 'A',
        'hbxx_mdd': '中文/拼音',
        'hbxx_hkgs': '中文/拼音',
        'hbxx_hbh': '输入航班号',
        'start': 0,
        'limit': limit_num
    }
    r  =  requests.post(r'http://www.szairport.com/frontapp/HbxxServlet',
data=data, headers=headers)
    print(r.request.headers)
    print(r.request.body)
    r.encoding = 'utf-8'
    rawhtml = r.text
    print(r.encoding, rawhtml)
    return rawhtml
```

```
def get_html_from_file(suffix: str):
    with open("rawhtml" + suffix + ".html", "r", encoding='utf-8') as f:
        return f.read()

def put_html_to_file(suffix: str, rawhtml: str):
    with open("rawhtml" + suffix + ".html", "w", encoding='utf-8') as f:
        return f.write(rawhtml)

# 从全部 html 数据中提取 tbody 子 html
def get_tbody(html: str):
    tbody_html = re.search(r'<tbody id=\"trs\">[\S\s]*?</tbody>', html)
    if tbody_html is None:
        raise ValueError
    return tbody_html.group(0)

# 从 tbody 源代码中解析，返回到港详细数据数组
# [!!!] ['2', 'CZ3912', '南方', 'A320', '乌鲁木齐-> 武汉', '09/11 16:40',
'09/11 21:10', '--:--', '--:--', '到达', '']
# [!!!] [None, '武汉->深圳', '09/11 22:10', '09/12 00:10', '09/11 22:39',
'09/12 00:14']
# [!!!] ['2', 'Y87522', '金鹏', 'B738', '海拉尔-> 呼和浩特', '09/11 16:55',
'09/11 19:20', '--:--', '--:--', '到达 ， 延误原因：流量控制', '']
def parse_from_tbody(tbody_html: str):
    dom = pq(tbody_html)
    nontrivial_data = [] # ret
    # rowspan = 0
    tr_list = dom('tr')
    for tr in tr_list.items():
        tds = tr('td')
        if tds.length == 0:
            continue
        temp = tds.eq(0).attr("rowspan")
        # rowspan = temp if temp is not None else rowspan

        row = []
        row.append(temp)
        for td in tds.items():
            row.append(td.text().strip())
        nontrivial_data += [row]
        # print("[!!!]", row)

    return nontrivial_data
```

```
# 从 Raw_html 中解析数据
def parse(raw_html: str):
    logging.debug("parse")
    tbody_html = get_tbody(raw_html)
    return parse_from_tbody(tbody_html)


def pickle_load(filename: str):
    with open(filename, "rb") as f:
        return pickle.load(f)


def pickle_dump(obj, filename: str):
    with open(filename, "wb") as f:
        pickle.dump(obj, f)


# 处理数据
suffix = '20190913-212208'

if __name__ == '__main__':
    logging.info("已启动")
    unique_timemark = datetime.now().strftime("%Y%m%d-%H%M%S")

    # 正常步骤：
    suffix = unique_timemark
    # For Dev:
    # suffix = '20190913-212208'

    # 正常步骤：
    rawhtml = fetch(511)
    put_html_to_file(suffix, rawhtml)
    # For Dev:
    # rawhtml = get_html_from_file(suffix)

    # 正常步骤：
    records = parse(rawhtml)
    pickle_dump(records, "first_data" + suffix + ".bin")
    # For Dev:
    # records = pickle_load("first_data" + suffix + ".bin")

    for x in records:
```

```
        print(x)
```

(2) process.py

```
import logging
from functools import cmp_to_key
from fetch import pickle_load, pickle_dump
from datetime import datetime, timedelta

# 返回：航班号，航空公司，机型，始发->经停->到达，预飞时间，预达时间，实际起飞，实际到达，状态
# 接受：合并指示，[航班号，航空公司，机型，]始发->经停->到达，预飞时间，预达时间，实际起飞，[实际到达，状态]
# [3 列隐式继承]
def conbine_rowspan(data: list):
    ret = []
    length = len(data)
    i = 0
    while i < length:
        row = data[i][:]
        if row[0] is not None and row[0] == '2':
            next_row = data[i+1][:]
            # row len = 11, '2', data....
            # next_row len = 6, none, [3 列隐式继承], data... [2 列隐式继承]
            # 合并经停信息
            trans: str = row[4] + "->" + next_row[1]
            trans_list = trans.split("->")
            trans_list.pop(2)
            # 去除经停信息多余空格
            trans = "->".join(map(lambda x: x.strip(), trans_list))
            temp = row[1:4] + [trans] + next_row[2:] + [row[-2]]
            i += 2
        else:
            temp = row[1:-1]
            trans = temp[3][:]
            # 去除经停信息多余空格
            trans = "->".join(map(lambda x: x.strip(), trans.split("->")))
            temp[3] = trans
            i += 1
        ret.append(temp)
        # print(temp)
    return ret


def strtime2datetime(text: str, default_date: datetime=datetime(2019, 1,
```

```
1)):
    if text.find(" ") > -1:
        temp = datetime.strptime(text, "%m/%d %H:%M")
        temp = temp.replace(year=default_date.year)
    elif text.find("-") == -1:
        temp = datetime.strptime(text, "%H:%M")
        temp            =            temp.replace(year=default_date.year,
month=default_date.month, day=default_date.day)
    else:
        return datetime.max
    return temp


# def cmp_func_datetime(data1: str, data2: str):
#     sort_index = 5
#
#         date1,  date2  =  map(lambda  x:  convert2datetime(x[sort_index],
default_date), (data1, data2))
#     date1: timedelta
#     date2: timedelta
#     print(date1-date2, (date1-date2).seconds)
#     return (date1-date2).total_seconds()
#     # if data


def datetime2uniformstrtime(param_datetime: datetime):
    if param_datetime == datetime.max:
        return ""
    else:
        return param_datetime.strftime("%m/%d %H:%M")


def uniformstrtime(text: str, default_date: datetime=datetime(2019, 1, 1)):
    return datetime2uniformstrtime(strtime2datetime(text, default_date))


default_date = None
suffix = "20190913-233349"
default_date = datetime(2019, 9, 13)


if __name__ == '__main__':
    unique_timemark = datetime.now().strftime("%Y%m%d-%H%M%S")
    data = pickle_load("first_data" + suffix + ".bin")
```

```
    # for row in data[0:40]:
    #     print(row)

    data = conbine_rowspan(data)
    # print(data)
    # for row in data[0:80]:
    #     print(row)

    # 按预达时间排序数据
    # sorted_data = sorted(data, key=cmp_to_key(cmp_func_datetime))
    sorted_data = sorted(data, key=lambda x: strtime2datetime(x[5], default_date))
    # for row in sorted_data:
    #     print(row)

    # 规范化日期
    for row in sorted_data:
        temp = row
        temp[5], temp[7] = list(map(lambda x: uniformstrtime(x, default_date), (temp[5], temp[7])))
        row[:] = temp

    # 统计航空公司集合
    airline = set()
    for row in sorted_data:
        airline.update(row[1].split("\n"))
    # {'重庆航空', '成都航空', '海南', '菲律宾宿务', '东方', '德国汉莎', '吉祥', '台湾中华', '国航', '西藏航空', '澜湄', '维珍', '澳门亚太', '春秋', '联邦快递',
    #  '菲律宾亚洲航空', '韩国大韩', '联合包裹服务', '祥鹏航空', '阿联酋联合', '河北', '泰国亚洲', '泰国狮航', '福州航空', '新加坡胜安', '深圳', '圆通', '厦门', '金鹏',
    #  '韩国韩亚', '四川', '博立航空', '东海航空', '南方', '长龙航空', '空桥货运', '马来西亚亚洲', '新加坡', '西伯利亚', '上海', '台湾立荣', '天津航空', '西部', '狮航',
    #  '乌鲁木齐航空', '奥凯', '昆航', '中国联合航空', '顺丰', '山东'}

    # 过滤掉货运航空
    cargo_airline = {'空桥货运', '圆通', '顺丰', '联邦快递', '联合包裹服务'}
    sorted_data[:] = [row for row in sorted_data if row[1] not in cargo_airline]



    # 统计机型
```

```
    airplane_type_stat = {}
    for row in sorted_data:
        type = row[2]
        airplane_type_stat[type] = airplane_type_stat.get(type, 0) + 1
    # 机型数量排序
    airplane_type_stat = [[k, v] for k, v in airplane_type_stat.items()]
    airplane_type_stat = sorted(airplane_type_stat, key=lambda x: x[1],
reverse=True)
    for k in airplane_type_stat:
        print(k)

    for row in sorted_data:
        print(row)
    # print(airline)
    # print(airplane_type_stat)

    # 导出 csv
    # import csv
    # suffix = unique_timemark
    # with open('airplane_data' + suffix + '.csv', 'w', newline='') as f:
    #     writer = csv.writer(f)
    #     writer.writerows(sorted_data)
```

附录 2：对问题一所建模型进行分析验证的 Matlab 源代码

(1) W3DPlot.m

```
clear;
syms t;
global M0 s Q0 L v v_ n c T2d Ta;
% 里程价：出租车每公里的价格
M0 = 2.6;
% 出租车起步里程
s = 2.0;
% 出租车起步价
Q0 = 10.0;
% 因为机场到其他地方，行驶里程必定大于起步里程，所以定义补偿起步价 Q0_
% 使得车费可简化为 Q0_ + M0 * 里程
Q0_ = Q0 - M0 * s;
% 历程利用率：跑一公里有多少公里载客(产生效益)
L = 0.69;
% 市区平均行驶速度
v = 25;
% 机场附近道路平均行驶速度
v_ = 50;
```

```
% 在市区每个小时接单数量
n = 2;
% 每公里的燃油/电费
c = 0.15;
% 机场排队载客所预估的等待时间
% t = 0;
% 司机返回市区需要的典型时间
T2d = 0.4;
% [长程行驶时间, 短程行驶时间]
Ta = [0.8, 0.3];
% 机场拉客车费 = 补偿起步价 + 机场拉客里程 * (历程价 - 油电费)
% 机场拉客里程 = 机场行驶速度 * 机场拉客行驶时间
% 短程车费 = 补偿起步价 + (机场行驶速度 * 短程行驶时间) * (里程价 - 油电费)
% Q(2) = Q0_ + v_*Ta(2) * (M0 - c)
% 长程车费 = 补偿起步价 + (机场行驶速度 * 长程行驶时间) * (里程价 - 油电费)
% Q(1) = Q0_ + v_*Ta(1) * (M0 - c)
Qa = Q0_ + v_*Ta * (M0 - c);
% 假设机场拉客未发生，考虑机场拉客时间用于返回市区拉客。
% 市区行驶时间 =  (机场预估等待时间 + 机场拉客行驶时间 - 返回市区时间)
Td = t + Ta - T2d;
% 市区接客车费 = - 返回市区空载成本 + 市区拉客收益
% 市区接客车费 =  - 机场速度 * 返回市区时间 * 油电费
%  + (市区每小时接单量*补偿起步价+市区行驶速度*里程利用率*(里程价-油电费)) *
市区行驶时间
Qd = -v_*T2d * c + (n*Q0_ + v*L*(M0 - c)) * Td
% 短途率
% ita = 0.15;
deltaQ = Qd - Qa
% 令收益差为 0，求解临界时间 t tolerance.
t_result = [solve(deltaQ(1)==0, t), solve(deltaQ(2)==0, t)]
eval(t_result)
% 探究平均收益速率 W - (预估等待时间 t, 短途率 ita) 关系图
[Qa; Qd]
Wa_ori = vpa(Qa ./ (t+Ta));
Wd_ori = vpa(Qd ./ (t+Ta));
W_ori = [Wa_ori; Wd_ori]
syms ita
k = [1-ita; ita]
W2dim = (W_ori*k).'
vpa(W2dim)
range = [0 0.65 0 6];
grid on
obj_surf(1) = fsurf(sym('t'), sym('ita'), W2dim(1), range);
hold on
```

```
obj_surf(2) = fsurf(sym('t'), sym('ita'), W2dim(2), range);
% hold on
% fsurf(sym('t'), sym('ita'), sym('t'), [0.15 0.30 0 6])
hold off
xlabel('预估等待时间 t (h)')
ylabel('短途率 \eta')
zlabel('平均收益速率 W (元/h)')
title("平均收益速率 W - (预估等待时间 t, 短途率 \eta) 关系图");
colorbar
% print(gcf, '三维图', '-dpng', '-r600')
% obj_surf(1).ShowContours = 'on'
% obj_surf(2).ShowContours = 'on'
view([48,33])
% obj_surf(1).EdgeColor = 'none';
% obj_surf(2).EdgeColor = 'none';
obj_surf(1).FaceAlpha = 0.8;
obj_surf(2).FaceAlpha = 0.8;
% print(gcf, '三维图-透视方向', '-dpng', '-r600')
% 绘制 ita = 0.15 的截面图
range = [0.15 0.1501 0 6];
obj_surf(1) = fsurf(sym('t'), sym('ita'), W2dim(1), range);
hold on
obj_surf(2) = fsurf(sym('t'), sym('ita'), W2dim(2), range);
hold off
view([0 0])
obj_surf(1).EdgeColor = 'interp';
obj_surf(2).EdgeColor = 'interp';
colorbar
xlabel('预估等待时间 t (h)')
ylabel('短途率 \eta')
zlabel('平均收益速率 W (元/h)')
title(['平均收益速率 W - 预估等待时间 t 关系图，取短途率 \eta = ',
num2str(0.15)]);
% print(gcf, '三维图-ita = 0.15', '-dpng', '-r600')
% [X,Y] = meshgrid(0:0.25:6, 0.05:0.01:0.50);
% [M,N] = size(X);
% Z = zeros(M,N);
%
% for i = 1:M
%     for j = 1:N
%         Z(i,j) = subs(W2dim(1), sym(["t", "ita"]), [X(i,j), Y(i,j)]);
%     end
% end
% surf(X,Y,Z)
```

```
[Qa; Qd]
func = Qd ./ (t+Ta)
func_2 = Qa ./ (t+Ta)
vpa(func)
vpa(func_2)
x = [0:0.01:4];
plot(x, subs(func(1), t, x), x, subs(func(2), t, x), x, subs(func_2(1), t,
x), x, subs(func_2(2), t, x))
title("平均收益速率 W - 预估等待时间 t: (机场, 市区) \times (长途, 短途)");
ylabel("平均收益速率 W (元/h)");
xlabel("预估等待时间 t (h)");
grid on
legend('(机场,长途)', '(机场,短途)', '(市区,长途)','(市区,短途)')
print(gcf, '平均收益速率 W - 预估等待时间 t, (机场, 市区) x (长途, 短途)',
'-dmeta', '-r600')
% 已使用截面表达
% [Qa; Qd]
% a = 1./(t+Ta').*[0.85; 0.15]
%
% func = [Qa; Qd] * a
% vpa(func)
%
% x = [0:0.01:4]
% plot(x, subs(func(1), t, x), x, subs(func(2), t, x))
% title("机场、市区载客，长短途复合收益速率 - 排队等待时间");
% ylabel("收益速率 ");
% xlabel("排队等待时间 t");
% grid on
% legend('机场', '市区')
[Qa; Qd]
func = (Qa - Qd)
vpa(func)
x = [0:0.01:4]
plot(x, subs(func(1), t, x), x, subs(func(2), t, x))
title("机场与市区载客收益差 \DeltaW - 预估等待时间 t");
ylabel("收益差 (机场收益 - 市区收益) \DeltaW (元/h)");
xlabel("预估等待时间 t (h)");
grid on
legend('长途', '短途')
% 单位时间相对收益：（机场收益 - 市区收益） / 运营时间
% 不具有研究意义，舍弃。
% func = (Qa - Qd)./(t+Ta)
% vpa(func)
%
```

```
% x = [0:0.01:4]
% plot(x, subs(func(1), t, x), x, subs(func(2), t, x))
%
% ylabel("单位时间相对收益");
% xlabel("排队等待时间");
% grid on
% hold on
% 研究参量对 t tolerance 的影响
% 转由其他文件负责研究。
% c = 0.7;
% Q0_ = Q0 - M0 * s;
% Qa = Q0_ + v_*Ta * (M0 - c);
% Td = t + Ta - T2d;
% Qd = -v_*T2d * c + (n*Q0_ + v*L*(M0 - c)) * Td;
% deltaQ = Qd - Qa;
%
% results = [];
% n = 3;
%
% i = 1;
% for c = [0.5:0.05:1]
%     c;
% %     L
%     deltaQ = calc_deltaQ();
%
%     t_result = [solve(deltaQ(1)==0, t), solve(deltaQ(2)==0, t)];
%     t_tolerance = eval(t_result);
%     results(i,:) = [c, t_tolerance];
%     i = i + 1;
%     sprintf("\n");
% end
%
% results
% plot(results(:,1), results(:,2), results(:,1), results(:,3));
% title('L Line Plot')
% xlabel('c')
% ylabel('t(tolerance)')
function [deltaQ] = calc_deltaQ()
    syms t;
    global M0 s Q0 L v v_ n c T2d Ta;
    Q0_ = Q0 - M0 * s;
    Qa = Q0_ + v_*Ta * (M0 - c);
    Td = t + Ta - T2d;
    Qd = -v_*T2d * c + (n*Q0_ + v*L*(M0 - c)) * Td;
```

```
    deltaQ = Qd - Qa;
end
```

(2) TtoleranceVarPlot.m

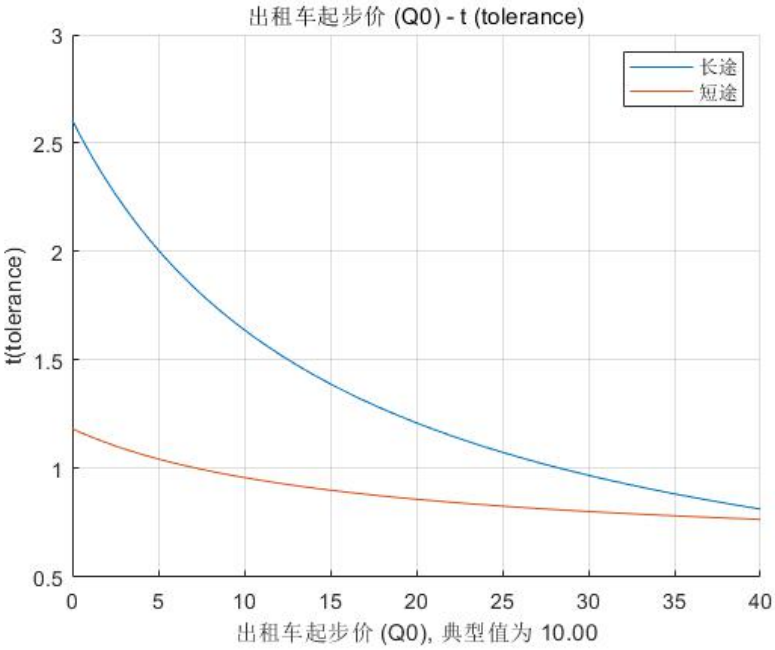
```
clear;
syms t;
global M0 s Q0 L v v_ n c T2d Ta;
% 里程价：出租车每公里的价格
M0 = 2.6;
% 出租车起步里程
s = 2.0;
% 出租车起步价
Q0 = 10.0;
% 因为机场到其他地方，行驶里程必定大于起步里程，所以定义补偿起步价 Q0_
% 使得车费可简化为 Q0_ + M0 * 里程
% Q0_ = Q0 - M0 * s;
% 历程利用率：跑一公里有多少公里载客(产生效益)
L = 0.69;
% 市区平均行驶速度
v = 25;
% 机场附近道路平均行驶速度
v_ = 50;
% 在市区每个小时接单数量
n = 2;
% 每公里的燃油/电费
c = 0.15;
% 机场排队载客所预估的等待时间
% t = 0;
% 司机返回市区需要的典型时间
T2d = 0.4;
% [长程行驶时间, 短程行驶时间]
Ta = [0.8, 0.3];
% 机场拉客车费 = 补偿起步价 + 机场拉客里程 * (历程价 - 油电费)
% 机场拉客里程 = 机场行驶速度 * 机场拉客行驶时间
% 短程车费 = 补偿起步价 + (机场行驶速度 * 短程行驶时间) * (里程价 - 油电费)
% Q(2) = Q0_ + v_*Ta(2) * (M0 - c)
% 长程车费 = 补偿起步价 + (机场行驶速度 * 长程行驶时间) * (里程价 - 油电费)
% Q(1) = Q0_ + v_*Ta(1) * (M0 - c)
% Qa = Q0_ + v_*Ta * (M0 - c);
% 假设机场拉客未发生，考虑机场拉客时间用于返回市区拉客。
% 市区行驶时间 =  (机场预估等待时间 + 机场拉客行驶时间 - 返回市区时间)
% Td = t + Ta - T2d;
% 市区接客车费 = - 返回市区空载成本 + 市区拉客收益
% 市区接客车费 =  - 机场速度 * 返回市区时间 * 油电费
```

```
%  + (市区每小时接单量*补偿起步价+市区行驶速度*里程利用率*(里程价-油电费)) *
市区行驶时间
% Qd = -v_*T2d * c + (n*Q0_ + v*L*(M0 - c)) * Td
% deltaQ = Qd - Qa
[delta_Q, var_list_syms] = deltaQ_func()
var_list_values = [M0 s Q0 L v v_ n c T2d];
var_list_name = ["出租车每公里的价格"; "出租车起步里程"; "出租车起步价";
    "里程利用率"; "市区平均行驶速度"; "机场附近道路平均行驶速度";
    "在市区每个小时接单数量"; "每公里电费燃油费"; "司机返回市区的典型时间"];
var_list_units = [""]
N = size(var_list_syms, 2);
for i = 1:N
    % 当前保留符号
    disp(['用 ' char(var_list_syms(i)) ' 来表达'])
    disp(var_list_syms(i))

    % 剩余符号
    tmp_var_list_syms = var_list_syms([1:i-1, i+1:N]);
    tmp_var_list_values = var_list_values([1:i-1, i+1:N]);

    % 带入其他符号
    simple_delta_Q = subs(delta_Q, tmp_var_list_syms,
tmp_var_list_values);
    % 带入 Ta
    simple_delta_Q = subs(simple_delta_Q, sym('Ta'), Ta)
    disp("令 simple_delta_Q = 0，解得：")
    t_result = [solve(simple_delta_Q(1)==0, t), solve(simple_delta_Q(2)==0,
t)]

    hold on
    grid on
    x = linspace(0.0,var_list_values(i)*4);
    y = [subs(t_result(1), x); subs(t_result(2), x)];
    plot(x, y(1,:), x, y(2,:));
    legend('长途', '短途')
    var_name = sprintf("%s (%s)", var_list_name(i),
char(var_list_syms(i)));
    fig_title = sprintf('%s - t (tolerance)', var_name);
    title(fig_title);
    xlabel(sprintf("%s, 典型值为 %.2f", var_name, var_list_values(i)));
    ylabel(sprintf('t(tolerance)'));
    hold off

    % 打印
```

```
    flag_need_print = 0;
    if flag_need_print
        print(gcf, fig_title, '-dmeta', '-r600')
    end

    figure
%     break;
end
simple_delta_Q = subs(delta_Q, var_list_syms, var_list_values);
disp("令 simple_delta_Q = 0，解得：")
t_result = solve(simple_delta_Q==0, t)
hold on
grid on
x = linspace(0.0,4);
y = eval(subs(t_result, x));
plot(x, y);
str=['(' num2str(x(50)) ', ' num2str(y(50)) ')'];
text(x(50), y(50), cellstr(str));
syms Ta
var_name = sprintf("%s (%s)", '机场途平均运行时间', char(Ta));
fig_title = sprintf('%s - t (tolerance)', var_name);
title(fig_title);
xlabel(sprintf("%s", var_name));
ylabel(('t(tolerance)'));
hold off
print(gcf, fig_title, '-dmeta', '-r600')
%     saveas(gcf, [fig_title, '.emf'])
function [deltaQ, alphabet, Qa, Qd] = deltaQ_func()
    syms t M0 s Q0 L v v_ n c T2d Ta;
    alphabet = [M0 s Q0 L v v_ n c T2d];

    % 补偿起步价 Q0_，使得车费可简化为 Q0_ + M0 * 里程
    Q0_ = Q0 - M0 * s;
    % 机场拉客车费 = 补偿起步价 + 机场拉客里程 * (历程价 - 油电费)
    % 机场拉客里程 = 机场行驶速度 * 机场拉客行驶时间
    % 机场拉客车费 = 补偿起步价 + (机场行驶速度 * 机场拉客行驶时间) * (里程价 -
油电费)
    Qa = Q0_ + v_*Ta * (M0 - c);
    % 市区行驶时间 = 机场预估等待时间 + 机场拉客行驶时间 - 返回市区时间
    Td = t + Ta - T2d;
    % 市区接客车费 = - 返回市区空载成本 + 市区拉客收益
    % 市区接客车费 =  - 机场速度 * 返回市区时间 * 油电费
    %  + (市区每小时接单量*补偿起步价+市区行驶速度*里程利用率*(里程价-油电费))
* 市区行驶时间
```

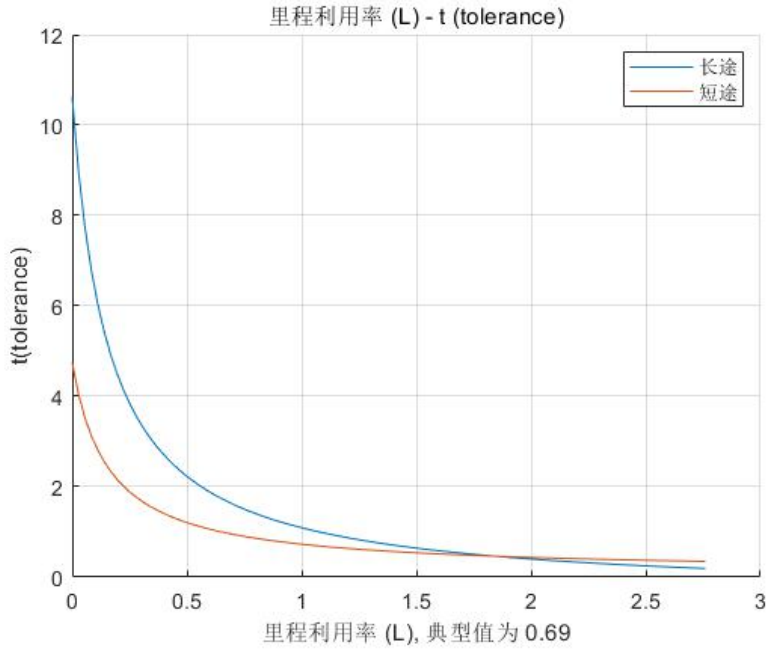
```
    Qd = -v_*T2d * c + (n*Q0_ + v*L*(M0 - c)) * Td;
    deltaQ = Qd - Qa;

end
```

附录 3：对 4.2.3 小节的补充



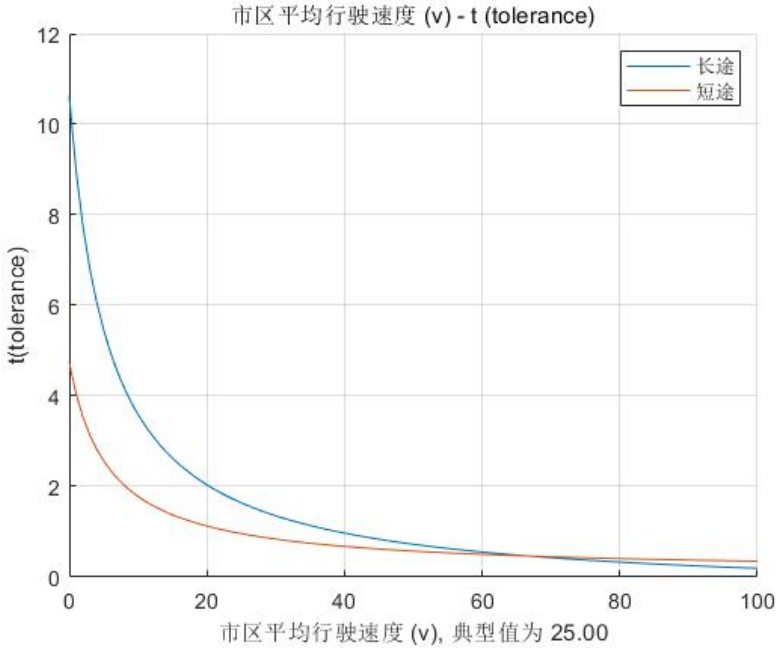
附录图 1 出租车起步价对临界值的影响

其解析式为 $\left(\frac{80\,Q_0+33222}{800\,Q_0+12745}\quad\frac{960\,Q_0+30189}{1600\,Q_0+25490}\right)$



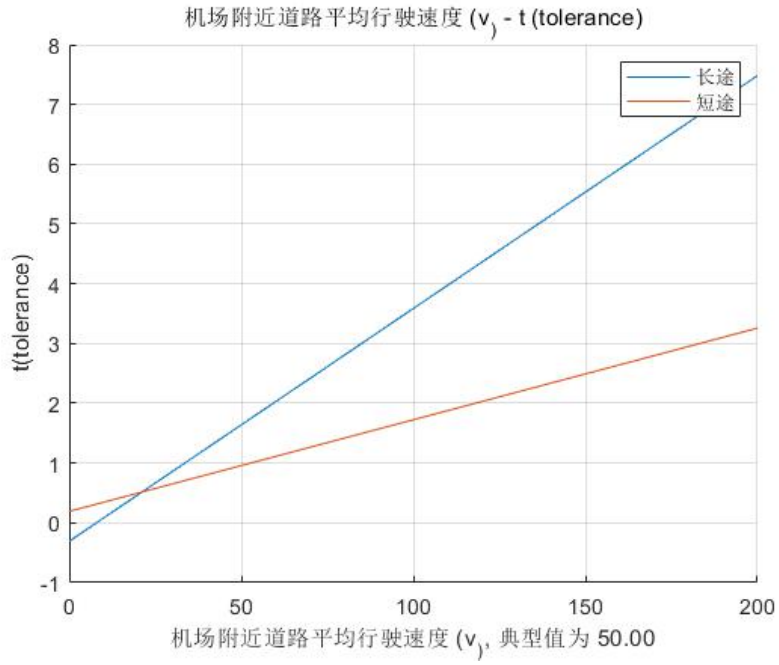
附录图 2 里程利用率对临界值的影响

其解析式为 $\left(-\frac{2450\,L-10196}{6125\,L+960}\quad\frac{1225\,L+9102}{12250\,L+1920}\right)$



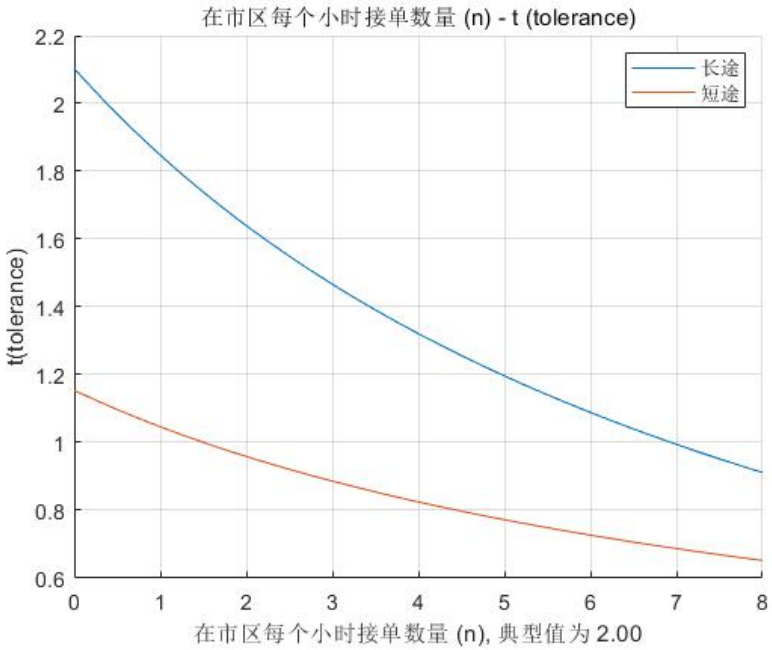
附录图 3 市区平均行驶速度对临界值的影响

其解析式为 $\left(-\frac{6762\,v-1019600}{16905\,v+96000}\quad\frac{1127\,v+303400}{11270\,v+64000}\right)$



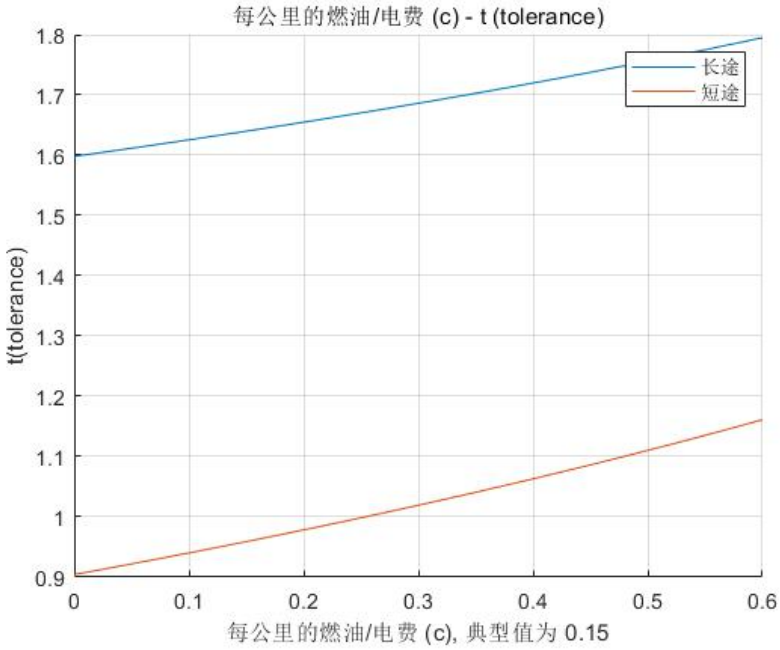
附录图 4 机场附近道路平均行驶速度对临界值的影响

其解析式为 $\left(\frac{808\,\bar{v}}{20745}-\frac{2126}{6915}\quad\frac{106\,\bar{v}}{6915}+\frac{2663}{13830}\right)$



附录图 5 在市区每个小时接单数量对临界值的影响

其解析式为 $\left(-\frac{768\,n-35558}{1920\,n+16905}\quad\frac{128\,n+13007}{1280\,n+11270}\right)$



附录图 6 每公里的燃油/电费对临界值的影响

其解析式为 $\left(\frac{1310\,c-8702}{1725\,c-5445}\quad-\frac{655\,c+9849}{3450\,c-10890}\right)$