中国研究生创新实践系列大赛
“华为杯”第十六届中国研究生数学建模竞赛

学　　校	湖南大学
参赛队号	19105320001
队员姓名	1.蒋新超
	2.余小龙
	3.刘习洲

中国研究生创新实践系列大赛
“华为杯”第十六届中国研究生数学建模竞赛

题 目　　汽车行驶工况构建

摘　　要：

汽车行驶工况是描述汽车行驶的速度-时间曲线，体现的是汽车道路行驶的运动学特征。本文对已知数据进行了充分的预处理，并设计了包含片段有效性检测的切片方法，采用了随机短行程法将得到的数据构建成汽车代表性行驶工况曲线，最后提出了 Mini-Batch Optimization(MBO)算法优化工况的构建方式。具体方法和结果如下：

针对问题 1，我们需要对设备采集到的不良数据进行清洗。找到文件 1、2、3 中不连续的断点数据，个数分别为 724、2375、1097。针对不同类型断点，采用不同的插值处理得到连续且完备的数据，个数分别为 518137、518279、431784。对于加、减速度异常的数据利用数据示踪法处理，即将其对应点的速度赋为一个极大的值，如 300km/h；对于因长期停车采集到的异常数据进行抹平“尖刺”处理；对于长时间堵车(以 180s 为判断依据)、断断续续低速行驶情况，将其速度赋值为 0；对于包含长怠速区间的运动学片段只截取后 180s 作为其怠速区间。为保持数据连续性,怠速时间过长的情况在第 2 问中处理会更佳，至此所有不良数据均清洗完成。

针对问题 2，我们采取怠速检测机制将上述经处理后的数据分别切分为 1203、834、867 段。针对性地设计了阈值过滤机制，将上述利用数据示踪法标记的加、减速度异常的数据（$a<-8\text{m/s}^2$或$a>3.97\text{m/s}^2$）所对应的片段剔除；设计了多 0 检测机制，有效处理了怠速超过 180s 的异常数据，截取长怠速区间后 180s 作为运动学片段开始的怠速区间；设计了匀速检测机制，剔除了异常运动学片段。最终得到文件 1、2、3 的有效运动学片段个数分别为 705、414、511，共计 1630 个。

针对问题 3，我们共选取了 14 个评估指标。由于指标的维度较高，我们先利用主成分分析进行降维处理，从而保留原数据的大部分信息，然后用 K-means 聚类分析将实验所得运动学片段分为“拥堵”和“畅通”两类工况。最后采取随机短行程法，分别对两类工况进行构建，与实验数据相比，得到特征参数的平均误差为 8.47%和 10.88%。由于误差较大，我们对构建方式进行了创新，提出 MBO 算法，优化后的误差为 4.39%和 5.18%，相较于之前分别提高了 48.2%和 52.4%。最终得到了与车辆行驶工况更为接近的代表性行驶工况曲线。

关键词：行驶工况，数据示踪法，主成分分析，K-means 聚类分析，MBO 算法；

1. 问题背景

车辆行驶工况是用来描述特定交通环境下车辆行驶特征的速度—时间历程，又称汽车运转循环（Driving Cycle）[1][2]。行驶工况一般通过某种车辆在特定交通环境中的行驶实验，经过数据分析，运用多元统计理论建立，能够反映特定交通环境下车辆的典型行驶特征。行驶工况主要用于确定车辆在特定交通环境下的燃料消耗和污染物排放、新车型的研发与评估以及交通控制效果评价分析等，是交通领域的一项共性核心技术。由于不同国家、城市和地区的交通环境与特征不同，因此车辆实际运行工况必然有很大的差异，这种差异会使同一型号汽车在不同地区实际运行过程中表现出不同的性能，特别是汽车的燃料经济性和排放性差别较大。因此，各个国家和地区都相继开发适合本地区交通特征的行驶工况。有代表性的行驶工况有美国工况、欧洲工况和日本工况，其中美国 FTP 瞬态工况和 ECE 模态工况最有代表性[3]，并且广为其他国家借鉴采用。

2000 年后，我国沿用欧洲的 NEDC 行驶工况对汽车的能耗、排放进行认证，一定程度上推动了我国节能减排技术的发展，贯彻落实了可持续发展的要求。但近年来，国民经济飞速发展，汽车持有量不断增长，道路状况沧海桑田。政府、企业和民众日渐发现原始 NEDC 工况基准下标定的汽车油耗指标相差甚远。作为车辆开发、评价最为重要的基础依据，开展深入研究，制定反映我国实际道路行驶状况的测试工况显得至关重要[4]。

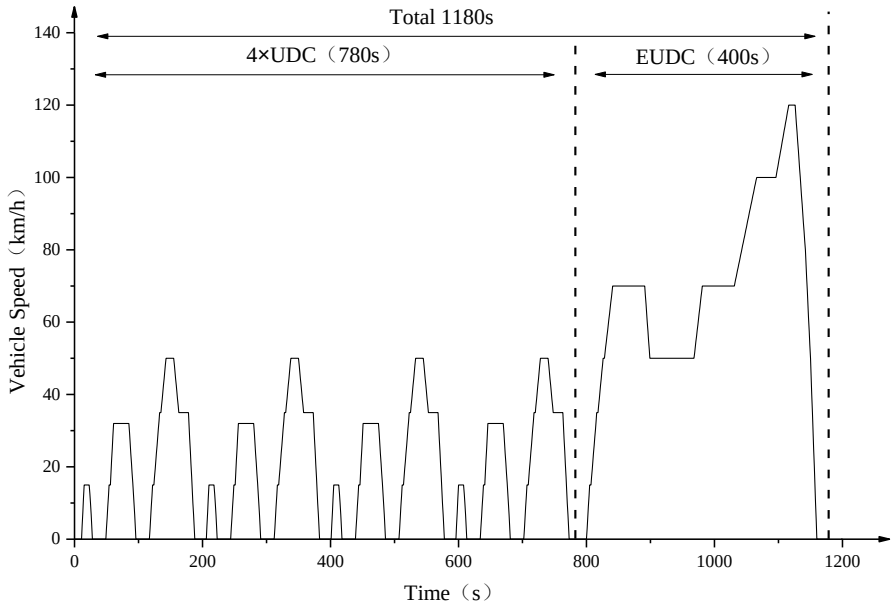


图 1.1 欧洲 NEDC 工况

2. 模型假设

(1) 车辆可进行正常稳定行驶，不会产生故障；

(2) 车辆在行驶过程中不会发生交通事故；

(3) 附件中所提供数据均为真实行驶实验数据；

(4) 传感器安装正确且保持正常工作；

(5) 假设采样频率 1Hz 下，相邻采样速度之差可以作为前一个速度的加速度。

3. 符号说明

表 3.1 符号系统

符号	意义
t_i	断点 i 的时间
v_{di}	断点 i 的速度
v_{inter}	断点之间的差值速度
$\mathbb{N}$	速度为 0 的索引集合
λ	0 索引位置
Ω	运动学片段集合
φ	运动学片段
T	运行时间
T_a	加速时间
T_d	减速时间
T_i	怠速时间
T_y	匀速时间
$V_{\max}$	最大速度
V_{mean}	平均速度
V_d	正常行驶平均速度（不包含怠速）
V_{std}	速度标准差
$a_{\max}$	最大加速度
$a_{\min}$	最大减速度
a_{std}	加速度标准差
a_a	加速阶段加速度均值
a_d	减速阶段加速度均值
E	特征值向量集合
ξ_i	某个运动学片段特征值向量
θ_j	某类工况实验数据特征值向量

4. 问题 1

4.1 问题 1 重述与分析

(1)**问题重述**：由汽车行驶数据的采集设备直接记录的原始采集数据往往会包含一些不良数据值，不良数据主要包括几个类型：

1) 由于高层建筑覆盖或过隧道等，GPS 信号丢失，造成所提供数据中的时间不连续；
2) 汽车加、减速度异常的数据（普通轿车一般情况下：0 至 100km/h 的加速时间大于 7 秒，紧急刹车最大减速度在 7.5~8m/s^2）；
3) 长期停车（如停车不熄火等候人、停车熄火了但采集设备仍在运行等）所采集的异常数据。
4) 长时间堵车、断断续续低速行驶情况（最高车速小于 10km/h）,通常可按怠速情况处理。
5) 一般认为怠速时间超过 180 秒为异常情况，怠速最长时间可按 180 秒处理。

因此，我们需要设计合理的方法将上述不良数据进行预处理，并给出各文件数据经处理后的记录数。

(2)**问题分析**：由题目信息可知，由汽车行驶数据的采集设备直接记录的原始采集数据往往会包含一些不良数据值，这些原始数据的可靠性和准确性直接关系到后续运动学片段的划分以及最后行驶工况的构建，因此我们首先利用 Matlab 软件编写程序，针对第一种不良数据，判断原始采集数据中是否存在不连续的断点，然后依据不同断点的时间跨度，有选择的采用相应的方法对三个附件中的速度及加速度进行插值处理，使得原始数据连续且完备，然后按照问题需求，根据需求编程处理后续各类型的不良数据，如毛刺数据和长时间的慢行数据等，便可得到预处理后的数据。

4.2 模型建立与求解

4.2.1 数据示踪法

在化学中常用同位素原子示踪法[5]追踪物质的运行和变化规律。同位素示踪法大量应用于生命科学、医学等领域。这种方法利用的是同位素具有相同的生物学性质和化学性质，只是具有不同的核物理性质这一特点。它被广泛用于细胞、微生物等各类标记。

本文受该方法启发，提出使用一个标记性的数据对原始数据进行标记的处理方法。对于加速度异常数据（过大），在连续性的运动学片段中，直接删除单个数据的办法是不可取的，而是存在此类数据的整个运动学片段都应该被处理掉。

这保证了运动学片段在严格意义上能代表一般的行驶工况。该方法可表述为：

$$v_i = 300, \quad \begin{cases} a > a_{\max} \\ a < a_{\min} \end{cases} \tag{4-1}$$

这种方式处理并不会改变整个数据的顺序，也不会破坏数据的连续性，且可以高效的追踪异常数据所在的运动学片段。它只是对加速度异常的进行识别，并将相应的速度处理标记，这里 300 也可以是其他的“大数”，正常车速不能达到的情况均是可以的。

4.2.2 数据预处理方式

题中给出了 5 种主要的数据类型，针对不同的情况本文分别做了以下处理：

a. 由于高层建筑覆盖或过隧道等，GPS 信号丢失，造成所提供数据中的时间不连续问题。

上述时间不连续可以认为是寻找如下目标值，并构造如下插值模型：

$$\text{find:} \quad t_i + 1 \neq t_{i+1} \tag{4-2}$$

$$v_{\text{inter}} = \begin{cases} 0 & , \quad v_{di} = 0 \text{ or } v_{di+1} = 0 \\ \dfrac{v_{di+1} - v_{di}}{|v_{\text{inter}}|} x + v_{di} & , \quad \text{otherwise} \\ 0 & , \quad t_{i+1} - t_i > 180 \text{ and } \dfrac{\Delta v_{di}}{v_{di}} > 0.2 \end{cases} \tag{4-3}$$

其中 t_i 为汽车行驶上一时刻，t_{i+1} 为汽车行驶下一时刻其中 v_{di} 为汽车行驶上一断点时刻速度，$v_{\text{di+1}}$ 为汽车行驶下一断点时刻速度。通过在 Matlab 软件中编写程序（所有程序详细见附录）查找原始数据中的断点。断点的存在破坏了数据的连续性，运动片段的划分存在问题。因此本文认为需要对数据进行相应的差值处理。若断点两端任一速度为 0，则在两断点间插值得到速度 0。这是考虑到数据在此段时间内可能会有相对来说比较复杂的变化，可能包括各种汽车行驶的状态，即使是插值处理也不能保证此段形成的运动学片段具有实际的意义。若断点的时间太长（超过 180s），且断点速度两端速度的变化量超过 20%，则进行 0 插值处理。这是考虑到长时间的速度的波动太大，简单的插值是不可能给出具有参考意义的速度插值的，会较大的影响后续统计指标的构建。若断点两端速度均不为 0，且时间较短则采用一元函数对速度进行线性插值，从而得到连续的数据，这样处理是考虑到短隧道行驶可能会有数据丢失，且在较短时间的速度变化在一定可接受范围的。

b．汽车加、减速度异常的数据问题（减速度认为是加速度反向）。

根据提供的条件分为两种情况：加速过程的加速度、紧急刹车过程的减速度。这里：

$$v = v_0 + at \tag{4-4}$$

在加速过程我们取加速度为 $\frac{100\text{km/h}}{7\text{s}} \approx 3.97\text{m/s}^2$，刹车过程取减速度为 $8\,\text{m/s}^2$，因此正常数据应该保证加速度在 $-8 \le v_{i+1} - v_i \le 3.97$ 之间，其中 v_i 为汽车行驶上一时刻速度，v_{i+1} 为汽车行驶下一时刻速度，我们运用程序判断加速度是否介于 -8m/s^2 和 3.97m/s^2 之间，若介于此范围之内，我们认为这是有效数据，若介于此范围之外，我们认为这是不良数据，对此我们采用数据示踪法将加速度异常的采集点所对应的速度设置为 300km/h，以“大速度”作为标记，利于后续处理。

c．长期停车（如停车不熄火等候人、停车熄火了但采集设备仍在运行等）所采集的异常数据问题。

我们认为这部分数据为尖刺数据，制定判定依据为在尖刺处的非零速度值数量小于 3 且尖刺两端的速度为零的数据点数目不少于 20。以此为依据，我们筛选去除了尖刺数据。

d．中长时间堵车、断断续续低速行驶（最高车速小于 10km/h）。

我们对此情况发生的时长进行了判断，将这种情况连续出现 180 秒以上的数据利用程序进行了磨平处理（赋值为 0）。

e. 怠速时间超过 180 秒的异常问题。

我们认为这个数据在数据保持完备连续性的情况下，在问题 2 处理更佳，因此会在问题 2 运动学片段的分割和求解中给出相应的处理办法。

4.2.3 数据预处理结果分析

在经过上述的方法将不良数据进行预处理后，我们得到处理结果如下：

（1）各文件均存在时间不连续的数据，我们统计每个文件的断点数目，并通过插值得到在实际时间区间下，传感器采样频率为 1Hz 时的完备数据。

表 4.1 数据的断点数和理论数据大小

-	文件 1	文件 2	文件 3
断点数目	724	2375	1097
补全数据的个数	518137	518279	431784

（2）通过对加减速度的判断筛选出共 202 组异常加减速度数据，并以“大速度”标记以示踪，以异常点编号 1706 及其附近 10 个数据点的速度为例，其效果如图 4.1 所示，由于数据被强行放大，所在的运动学片段将很容易通过滤波处理掉。

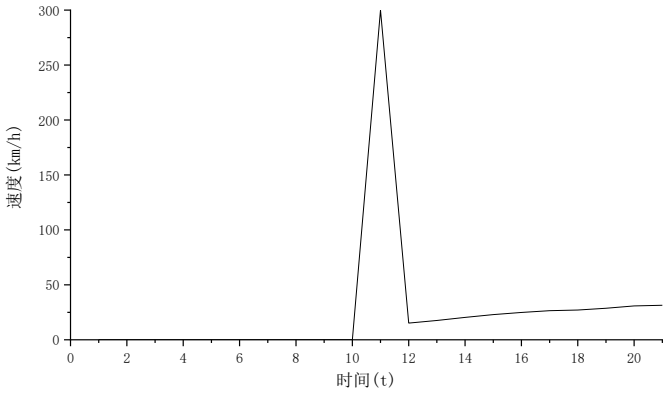


图 4.1 数据示踪法示意图

（3）针对类型三“尖刺”数据的清理，共得到 72 组不良数据，其代表性的处理结果如图 4.2 所示。

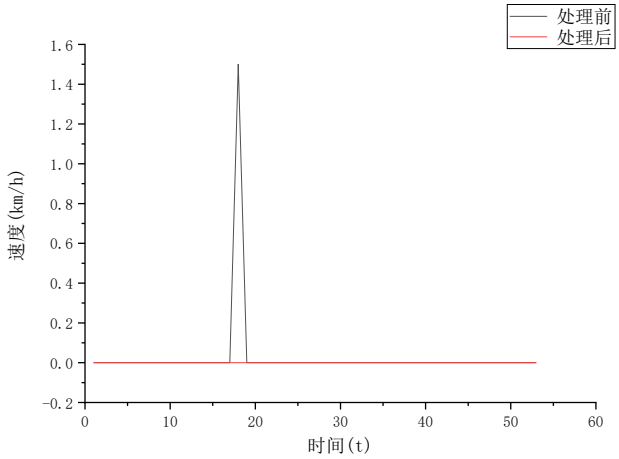


图 4.2 “尖刺”数据清理示意图

(4)我们处理了类型四的不良数据共 319314 组，其中包含了堵车时车速为 0 的数据以及断断续续低速行驶时车速小于 10km/h 的数据。以第一组处理的共 312 组数据为例，截取其中一段，可得其处理结果对比图。

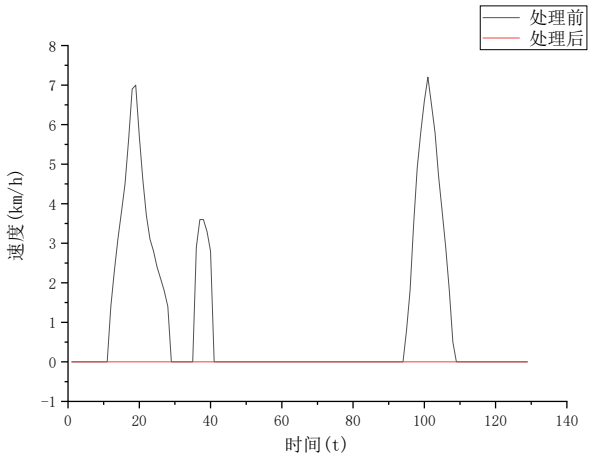


图 4.3 长期怠速数据处理示意图

5. 问题 2

5.1 问题 2 重述与分析

(1)**问题重述**：运动学片段是指汽车从怠速状态开始至下一个怠速状态开始之间的车速区间，（基于运动学片段构建汽车行驶工况曲线是目前最常用的方法之一，但并不是必须的步骤，有些构建汽车行驶工况曲线的方法并不需要进行运动学片段划分和提取）。我们需要设计合理的方法，将上述经处理后的数据划分为多个运动学片段，并给出各数据文件最终得到的运动学片段数量。

(2)**问题分析**：根据问题 2 所提供结合相关文献资料可知，一个基本的运动学片段包括了怠速、加速、减速及匀速运动等车辆工况。因此在问题 1 的经处理数据上切分，但确定运动学片段时需要对是否包含此四类基本工况进行判定，并需要对预处理中的数据进行相应的剔除工作，最终确定各文件中所包含的有效运动学片段数量。

5.2 模型建立与求解

5.2.1 运动学片段

汽车的实际行驶工况一般是由一定数量的运动学片段组成的，而一个运动学片段通常包含怠速、加速、减速和巡航/匀速四种工况[6][7]，如图 5.1 所示。根据题目要求，按照以下原则定义怠速工况、加速工况、减速工况和巡航/匀速工况：

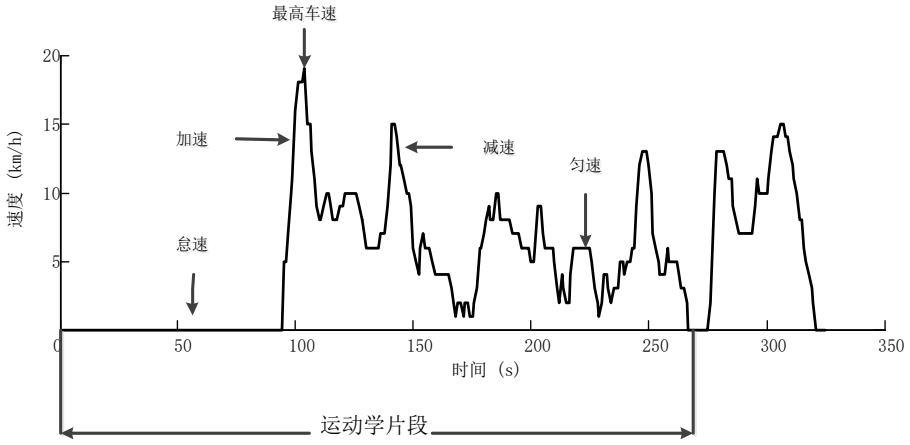


图 5.1 运动学片段示意图

（1）怠速工况：汽车停止运动，但发动机保持最低转速运转的连续过程；

（2）加速工况：汽车加速度大于 0.1m/s^2 的连续过程；

（3）减速工况：汽车加速度小于-0.1m/s^2 的连续过程；

（4）巡航/匀速工况：汽车加速度的绝对值小于 0.1m/s^2 非怠速的连续过程。

本文根据以上定义对预处理后的数据进行运动学片段的划分。

5.2.2 运动学片段切分方式

通过对问题 1 的处理，数据已经被清洗得相对干净。在此基础上根据我们对数据的处理方式设计了如下怠速检测机制，以切分数据。对 3 个文件的数据 D_1, D_2, D_3,筛选出其中表征汽车速度的数据V_1, V_2, V_3进行处理。

$$\begin{aligned} &\text{find：} \quad v_i = 0 \\ &\text{return：} \ \mathbb{N}=\{\lambda_1, \ \lambda_2, \cdots, \lambda_n\} \end{aligned} \tag{5-1}$$

利用 Matlab 寻找速度向量中的 0 元素，可以获得一个关于速度的v_i 的位置索引序列向量$\mathbb{N}$， λ_i 为汽车速度为 0 的时刻序号。将此$\mathbb{N}$ 向量作为切分运动学判断的基本数学依据。令Ω 为运动学片段φ_1， φ_2,…， φ_3 的集合，则有：

$$\pi=\{\lambda_i - 1 \mid \lambda_i \in \mathbb{N} \ \ \lambda_i - \lambda_{i-1} \neq 1\}, \ \ \lambda_0=0, \ \ i=1,2\cdots \tag{5-2}$$

$$\varphi_i = \{v_{\pi_{i-1}+1}, \cdots, v_{\pi_i - 1}, 0\}, \ \ \pi_0=0, \ \ i=1,2,\cdots, \tag{5-3}$$

例如存在序列v_i=$\{0,0,0,2,5,8,5,0,0,2,4,0\}$，则 $\mathbb{N}$=$\{1,2,3,8,9,12\}$， π=$\{7,11\}$，$\varphi_1 = \{v_0, \cdots, v_7, 0\}$=$\{0,0,0,2,5,8,5,0\}$， $\varphi_2 = \{v_8, \cdots, v_{11}, 0\} = \{0,0,2,4,0\}$。

这样，数据便可以切分为若干段运动学片段。

5.2.3 片段有效性检测方式

(1) 阈值过滤

上述经处理的数据中，包含示踪异常速度。针对加减速度异常已标记的示踪速度进行阈值检测，当运动学片段中的$V_{\max}$ 存在等于 300km/h 的数据则剔除此段运动学片段。

(2) 匀速检测

对任意的$\varphi_i \in \Omega$，φ_i 均存在由 0 增加速度最后减速到 0 这样的过程，而匀速的过程却不一定有。由于数据清洗到位，不存在怠速区间的数据也不可能存在。因此我们选择具有代表性的匀速阶段，作为工况是否可靠的一种检测指标。由于实际情况下传感器检测的速度多少都有一定的波动性，速度不变的情况是极少的，所以我们采用加速度作为判断是否存在匀速行驶阶段的指标。这里按运动学片段定义中的加速度绝对值小于阈值 0.1m/s^2，进行筛选。考虑到某些短行程有断断续续的慢行存在，我们将 3 秒内$|a|<0.1$m/s^2 的阶段作为匀速阶段。若没有此类行驶过程，则剔除整段运动学片段。

(3) 多 0 检测

在第 1 问的数据预处理环节中，我们保留了怠速超过 180s 的数据，因此在切分完整个数据之后必须对此类数据进行处理。利用 Matlab 对每个运动学片段的怠速时间进行检测并统计长度，将大于 180s 的怠速时间截取为原怠速时间的后 180s 作为此运动学片段的怠速区间。但此类运动学片段不进行剔除整段剔除，而仅仅剔除此片段中高于 180s 的多余怠速时间。

5.2.4 运动学片段切分结果

如表5.1所示，每个文件按照我们设计的切分方式分别切分为1203,834,867个。对其进行有效性判断最后分别得到了705,414,511，共计1630个有效运动学片段。

表5.1 运动学片段切分结果

-	文件1	文件2	文件3
粗切分运动学片段	1203	834	867
有效运动学片段	705	414	511

6. 问题3

6.1 问题3重述与分析

(1)**问题重述**：根据处理后的数据，构建一条能体现参与数据采集汽车行驶特征的汽车行驶工况曲线（1200-1300秒），要求该曲线的汽车运动特征能代表所采集数据源的相应特征，两者间的误差越小，说明所构建的汽车行驶工况的代表性越好。具体要求如下：

1）科学、有效的构建方法（数学模型或算法，若采用已有的方法，必须注明来源）；

2）合理的汽车运动特征评估体系（至少包含但不限于以下指标：平均速度（km/h）、平均行驶速度（km/h）、平均加速度（m/s^2）、平均减速度（m/s^2）、怠速时间比（%）、加速时间比（%）、减速时间比（%）、速度标准差（km/h）、加速度标准差（m/s^2）等）；

3）按照所构建的汽车行驶工况及汽车运动特征评估体系，分别计算出汽车行驶工况与该城市所采集数据源（经处理后的数据）的各指标（运动特征）值，并说明所构建的汽车行驶工况的合理性。

(2)**问题分析**：此题在前两问的基础上上升了一个层面，相当于挖掘上述经处理过后的数据的隐藏信息。我们可以建立了一套包含题目所给指标在内但不限于此的评价指标，用这套指标对各个片段进行评价，得到各片段特征值向量。因为实验数据包含各种实际交通状况的数据，为了更好的挖掘不同类型数据信息，因此考虑对片段进行区分，考虑用聚类分析进行分类。由于数据维度较大，为了提高计算效率，可以利用降维的方法进行预处理。在分类处理完成后，需要在各类别中分别选取运动学片段进行拼接。在运动片段的选取方面，我们首先尝试用随机短行程法进行选取，完成传统方式的构建。然后可以自主思考提出创新算法，改善传统算法的不足，增加数据信息挖掘的可靠性，使得构建的汽车行驶工况的代表性更好。

6.2 模型建立与求解

6.2.1 随机短行程法

短行程(运动学片段)是指相邻两个停车点之间的汽车循环过程，由一个怠速部分和一个行驶部分构成。对车辆实际道路数据进行统计分析，并定义相关判定准则，然后按照短行程定义进行运动学片段划分，并定义每个运动学片段为候选工况，统计每个运动学片段特征参数，对比总体特征参数，根据相关性大小排列所有候选工况，确定所要建立工况长度，随机选择工况进行组合，当选择的组合工况长度符合时间长度和误差要求即为最终构建的行驶工况[8][9]，其具体流程如图6.1所示。

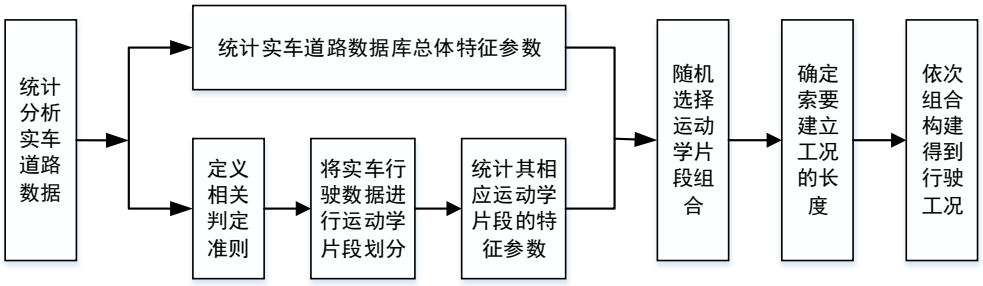


图 6.1 随机短行程法构建行驶工况流程图

短行程工况合成方法的思路是：在运动学片段库中对所有运动学片段进行随机组合，随机组合后的运动学片段构成候选工况，当候选工况的特征参数与实测数据的特征参数的相对误差小于一定范围，且候选工况时间长度达到规定要求，该候选工况即为具有一定代表性的行驶工况[10]。题目要求构建的行驶工况的时间长度定为 1200s 到 1300s，由于随机选取的短行程具有过高的随机性，因此本文将各项特征指标的平均相对误差控制在 15%以内，认为构建的工况指标误差在此之内能被接受。

该方法为已有方法，主要参考来源于姜平博士论文《城市道路混合工况的构建研究》。

6.2.2 批优化策略（Mini-Batch Optimization）

短行程法虽然简单可行，但由于存在极大偶然性，短行程法在合成工况时，较难获得较好的结果，构建的工况没有足够的代表性。为了获得较小误差，有学者提出了最佳增量法，即遍历所有片段，从中挑选出较优的片段组合在一起。但是这种算法在效率上相较于随机方法又有较大差距。为了综合两者的优势，我们提出了 Mini-batch Optimization(MBO)，即批优化策略，MBO 的基本流程如图 6.2 所示。首先先随机挑取一个有效运动学片段作为初始状态，然后每次取有限数量的片段作为候选片段，然后遍历这些片段以取出较优的一个。这里我们采用加入新的候选片段后平均相对误差作为目标函数，挑选出与初始片段融合拼接后离实验数据误差最低的片段一起作为新的初始状态。在满足给定的长度约束下，如此

往复得到最终的片段组合。该方法可抽象为一下数学模型，式(6-1)为算法初始化，式(6-2)为达到算法结束条件前每个循环的数学模型表达。φ_0 为随机选择的初始片段，Ω_n 为 Mini-Batch 抽样得到的 n(本文 n=50)个运动学片段的集合，E 为每个运动学片段与初始状态融合后的平均特征值向量的集合，$f(\varphi_i)$ 表示某个运动学片段映射得到特征值向量，θ_j 表示某一类交通状态下的实验数据总体平均特征值水平。“⊕”表示两段运动学片段拼接在一起。

$$\text{random}: \quad \varphi_0 \tag{6-1}$$

$$\begin{cases} \Omega_n = \{\varphi_1, \varphi_2, \cdots, \varphi_n\} \\ \Omega_n = \{\varphi_0 \oplus \varphi_1, \cdots, \varphi_0 \oplus \varphi_n\} \\ E = f(\Omega_n) \\ \{\xi_i \mid \xi_i \in E \quad \min\{\xi_i - \theta_j, \ i=1,2,\cdots\}\} \\ \varphi_0 = \varphi_0 \oplus \varphi_i \end{cases} \tag{6-2}$$

相较于随机的短行程法，MBO 方法因为对片段具有的选择性，因此在保持高计算效率的基础上还大幅度提高了构建所得特征参数的精度。相比与最佳增量法，但数据量较大的情况下，最佳增量法计算效率和算法灵活性则显得不足。批优化策略（MBO）在兼顾两种方法的优势情况下，在本题所给的数据下表现出优秀的效果。

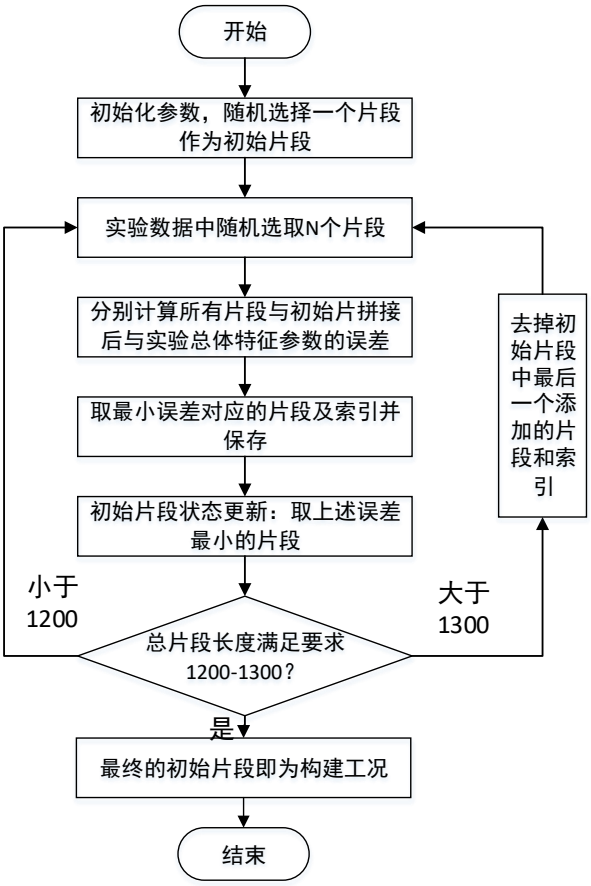


图 6.2 MBO 方法流程图

6.2.3 运动学片段特征参数指标体系

汽车实际行驶工况是由一定数量的运动学片段组成的，代表性行驶工况就是从大量的运动学片段中选择合适的运动学片段来构建的。我们选用如表 6.1 中的 14 个特征值用于评估行驶工况。

表 6. 1 用于分类的运动学片段特征参数

序号	符号	意义
1	$T(\mathrm{s})$	运行时间
2	$T_a/T(\mathrm{s})$	加速时间比
3	$T_d/T(\mathrm{s})$	减速时间比
4	$T_i/T(\mathrm{s})$	怠速时间比
5	$T_y/T(\mathrm{s})$	匀速时间比
6	$V_{\max}(\mathrm{km/h})$	最大速度
7	$V_{mean}(\mathrm{km/h})$	平均速度
8	$V_d(\mathrm{km/h})$	正常行驶平均速度（不包含怠速）
9	$V_{std}(\mathrm{km/h})$	速度标准差
10	$a_{\max}(\mathrm{m/s^2})$	最大加速度
11	$a_{\min}(\mathrm{m/s^2})$	最大减速度
12	$a_{std}(\mathrm{m/s^2})$	加速度标准差
13	$a_a(\mathrm{m/s^2})$	加速阶段加速度均值
14	$a_d(\mathrm{m/s^2})$	减速阶段加速度均值

每个运动学片段的特征参数可以由以下公式计算：

（1）时间类特征参数：

$$T=\text{运动学片段的总长度} \tag{6-3}$$

$$T_a=\text{加速度大于}0.1\mathrm{m/s^2}\text{的总个数} \tag{6-4}$$

$$T_d=\text{加速度小于}-0.1\mathrm{m/s^2}\text{的总个数} \tag{6-5}$$

$$T_i=\text{怠速区间的总长度} \tag{6-6}$$

$$T_y=T-T_a-T_d-T_i \tag{6-7}$$

（2）速度、加速度类特征参数：

$$V_{\max}=\max\{\mathrm{v_i},i=1,2,\cdots\} \tag{6-8}$$

$$V_{\mathrm{mean}}=\sum_{\mathrm{i}=1}^{k}V_i\Big/T \tag{6-9}$$

$$V_d=\sum_{i=1}^{k}V_i\Big/(T-T_i) \tag{6-10}$$

$$V_{std}=\sqrt{\frac{1}{k-1}\sum_{i=1}^{k}(V_i-V_{mean})^2} \tag{6-11}$$

$$a_i = \frac{v_{i+1} - v_i}{t_{i+1} - t_i} \times \frac{1000}{3600} = \frac{v_{i+1} - v_i}{3.6}, i = 1, 2, \cdots \quad (6\text{-}12)$$

$$a_{\max} = \max\{a_i, i = 1, 2, \cdots, k-1\} \quad (6\text{-}13)$$

$$a_{\min} = \min\{a_i, i = 1, 2, \cdots, k-1\} \quad (6\text{-}14)$$

$$a_a = \frac{sum\{a_i \mid a_i > 0.1, i = 1, 2, \cdots, k-1\}}{T_a} \quad (6\text{-}15)$$

$$a_{\mathrm{d}} = \frac{sum\{a_i \mid a_i < -0.1, i = 1, 2, \cdots, k-1\}}{T_d} \quad (6\text{-}16)$$

$$a_{std} = \sqrt{\frac{1}{k-1}\sum_{i=1}^{k} a_i^{\,2}} \quad (6\text{-}17)$$

为了将短行程片段与整个实验数据特征参数相比较，可以按类似单个运动学片段的方式计算整个实验数据的特征参数。即计算拼接三个文件所得到的所有的1630 个有效运动学片段的整体的特征参数作为实验标准数据。

6.2.4 工况曲线的构建

构建工况是否能够以少量的行驶工况数据代表道路采集行驶数据，通常由实际道路行驶中采集的数据特征参数进行比较来判定。本文在构建行驶工况时，分别采用随机短行程法和提出的 MBO 算法进行符合题目所要求时间长度的工况曲线的构建。

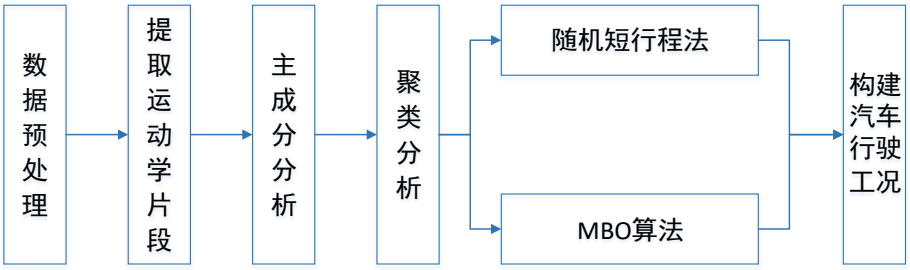


图 6.3 工况曲线构建流程图

由于选取的特征参数维度较高，数据量较为庞大，如果仅仅随机选择一两个行驶片段特征值来分析评估车辆运动学规律和水平，则可能会丢失一些关键信息。每个评价指标都包含有一定的信息，但有些指标是彼此相关联的，从而使得这些变量提供的信息有一定范围的重叠。例如：运行时间、运行距离和平均速度这三个特征值一定是紧密相关的。因此，引入主成分分析（Principal Component Analysis，PCA）方法来解决这一问题。主成分分析又称主分量分析，是将多个变量通过线性变换以选出较少个数重要变量的一种多元统计分析方法[7]。即构造原变量的一系列线性组合，使各线性组合在彼此不相关的前提下，尽可能多地反映原变量的信息，即使其方差最大。

主成分分析法是一种降维的统计方法[11]。它利用正交变换将相关的原随机向量转化成不相关的新随机向量。这在几何上表现为将原坐标系变换成新的正交坐

标系，使之指向距离最远的 p 个正交方向，然后通过构造适当的价值函数对多维变量进行降维处理，转换成低维变量系统并保持较高的精度。

通过主成分分析，可以得到各成分的贡献率如表 6.2 所示。

表 6.2 主成分分析贡献率

主成分序号	特征值	贡献率	累计贡献率
1	5.121	36.6%	36.6%
2	2.683	19.2%	55.7%
3	2.115	15.1%	70.8%
4	1.229	8.8%	79.6%
5	1.010	7.2%	86.8%
6	0.662	4.7%	91.6%
7	0.487	3.5%	95.0%
8	0.327	2.3%	97.4%
9	0.139	1.0%	98.4%
10	0.105	0.8%	99.1%
11	0.078	0.6%	99.7%
12	0.035	0.2%	99.9%
13	0.009	0.1%	100.0%
14	7.28E-32	0.0%	100.0%

由上表可知，前五个主成分的累计贡献率已达到 85%以上，已经能携带原有数据地绝大部分信息，因此取前五个主成分作为所有维度的表征。

表 6.3 给出了前五个主成分中各指标的成分系数。

表 6.3 主成分中各指标的成分系数

指标	主成分				
	1	2	3	4	5
T	0.258	-0.330	-0.115	0.201	0.025
T_a / T	0.229	0.368	0.286	-0.299	-0.024
T_d / T	0.150	0.285	0.474	0.066	-0.299
T_i / T	-0.314	-0.127	-0.402	-0.168	-0.250
T_y / T	0.244	-0.180	0.180	0.397	0.524
$V_{\max}$	0.407	-0.083	-0.169	-0.078	-0.172
V_{mean}	0.422	-0.082	-0.001	0.058	0.011
V_d	0.398	-0.126	-0.188	-0.060	-0.138
V_{std}	0.341	-0.020	-0.250	-0.243	-0.296
$a_{\max}$	0.094	0.186	-0.174	0.584	-0.400

$a_{\min}$	-0.192	-0.209	0.298	0.059	-0.406
a_{std}	0.119	0.566	-0.053	0.020	0.042
a_a	-0.150	0.263	-0.233	0.497	-0.081
a_d	-0.005	-0.358	0.429	0.125	-0.320

由表可知，主成分 1 主要包含了最大速度、平均速度及正常行驶平均速度的信息；主成分 2 主要包含了加速时间、加速度标准差、减速阶段加速度均值的信息；主成分 3 主要包含了减速时间、怠速时间、减速阶段加速度均值的信息；主成分 4 主要包含了匀速时间、最大加速度、加速阶段加速度均值的信息；主成分 5 主要包含了匀速时间、最大加速度、最大减速度的信息。

基于特征参数的前五个主成分，我们需要对所有运动学片段进行分类[12][13]，此处考虑运用 K-means 聚类分析方法[14]对所有的运动学片段进行分析。

K-means 聚类算法是一种广泛应用的经典聚类方法，具有收敛速度快、算法简单和处理数据效率高的特点[15]。K-means 聚类算法是基于距离计算的聚类算法，距离是相似程度的评价指标，该算法认为距离靠近的对象构成了簇，把在整体数据中得到的若干独立且紧凑的族作为算法的最终目标。算法的数学模型如下：

表 6. 4 K-means 聚类算法

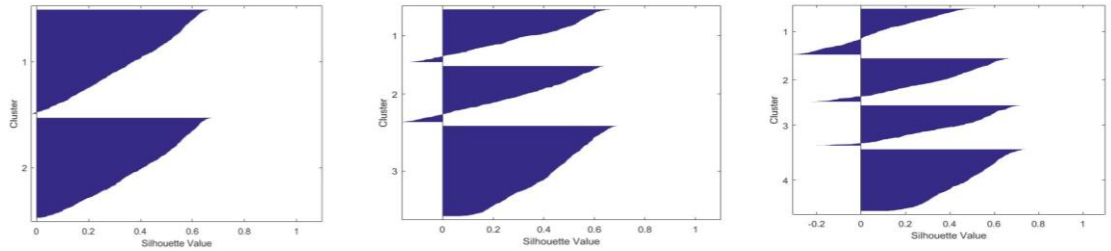
算法伪代码
（1）确定聚类数目 K 和数据集$\{x_1, x_2, ..., x_n\}$，并挑选 K 个数据作为初始聚类中心$\{c_1, c_2, ..., c_k\}$；
（2）计算其他数据与聚类中心c_k的距离$D(x_i, c_j), \{i = 1, 2, ..., n; j = 1, 2, ..., k\}$，距离计算方法采用欧式距离，计算公式为: $$D(x_i, c_j)=\sqrt{(x_{i1}-c_{j1})^2+(x_{i2}-c_{j2})^2+...+(x_{im}-c_{jm})^2}$$ 如果$D(x_i, c_j)$满足 $$D(x_i, c_j)=\min\{D(x_i, c_j)\}$$ 则将数据x_i分类到c_j中；
（3）数据得到分类后，需对聚类中心进行更新，计算公式为： $$c_j^* = \frac{1}{n}\sum_{i=1}^{n_j} x_i^j$$
（4）计算数据的误差平方和准则函数 J: $$J^* = \sum_{k=1}^{n}\sum_{j=1}^{k}\left\|x_k^j - c_j^*\right\|^2$$
（5）若对于任意J^*都满足$\left\|J^* - J\right\| < \xi$，则说明聚类准则得到收敛，聚类算法完成。否则回到第（2）步继续迭代。

对运动学片段聚类后，得到分为 2 类、3 类和 4 类的聚类分析结果。根据 K-means 聚类分析结果，运用 Silhouette 函数绘制轮廓图，从轮廓图上判断每个运动学片段的分类是否合理。其中 Silhouette 函数的定义如下：

$$L(i) = \frac{\min(b) - a}{\max[a, \min(b)]} \tag{6-18}$$

其中，a 为第 i 个运动学片段与同类运动学片段之间的平均距离；b 为一个向量，其元素是第 i 个运动学片段与不同类的各运动学片段间的平均距离。

$L(i)$ 的取值范围为[-1，1]，$L(i)$ 值越大，说明第 i 个运动学片段分类越合理；当 $L(i)$<0 时，说明第 i 个运动学片段的分类不合理，应该还有比目前更好的分类。K-means 聚类各分类结果的 Silhouette 函数值的轮廓见图 6.4。由图 6.4（a）可看出，当分 2 类时，各类的 Silhouette 函数值均大于 0，说明分类时各类型已经被很好地区分；由图 6.4（b）可看出，分 3 类时，第 1、2 种类型的 Silhouette 函数值出现少量负值；由图 6.4（c）可看出，分 4 类时，第 1、2 和第 3 种类型的 Silhouette 函数值出现少量负值的情况，说明聚类分析分为 3 类和 4 类时存在未被很好区分的运动学片段。根据分析结果，选取分为 2 类的结果作为 K-means 聚类分析的最终结果。



(a) 2 类　　(b) 3 类　　(c). 4 类

图 6.4 不同分类时 Silhouette 函数值的轮廓

聚类完成后，根据上述运动学片段聚类结果，分两类数据分别采用两种方式构建[16][17]的代表性工况曲线如下所示：

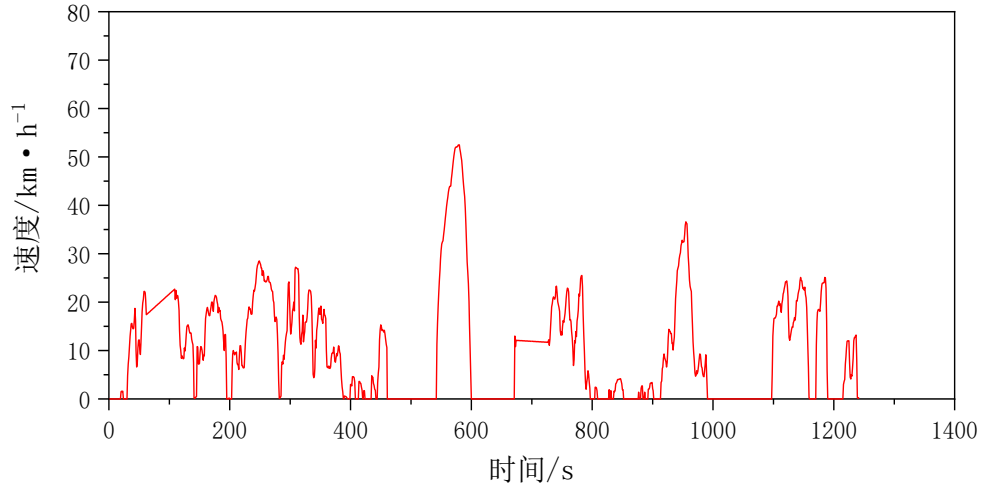


图 6.5 随机短行程法拥堵交通代表性行驶工况

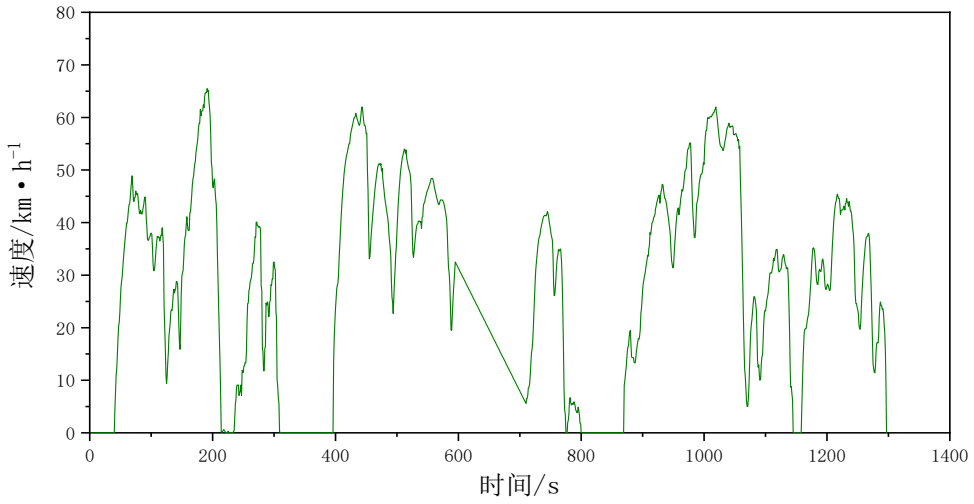


图 6.6 随机短行程法畅通交通代表性行驶工况

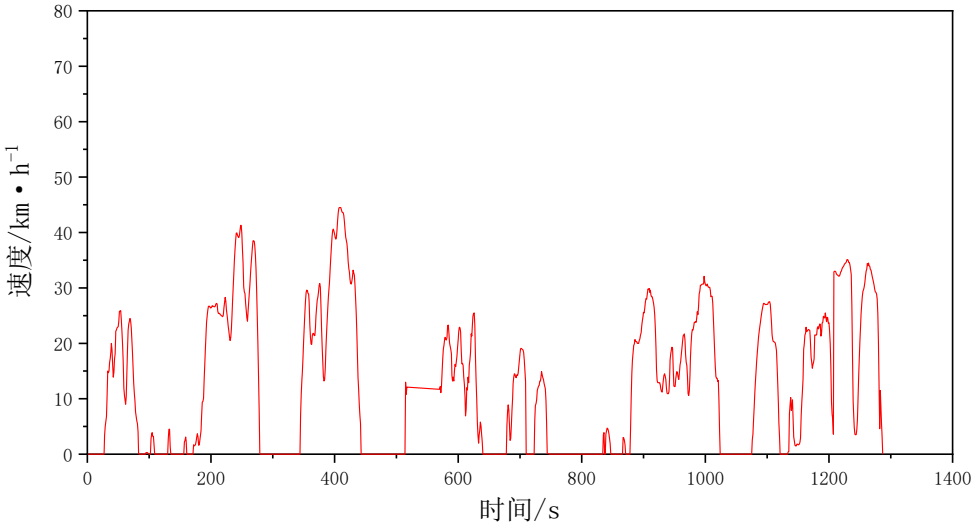


图 6.7 MBO 算法拥挤交通代表性行驶工况

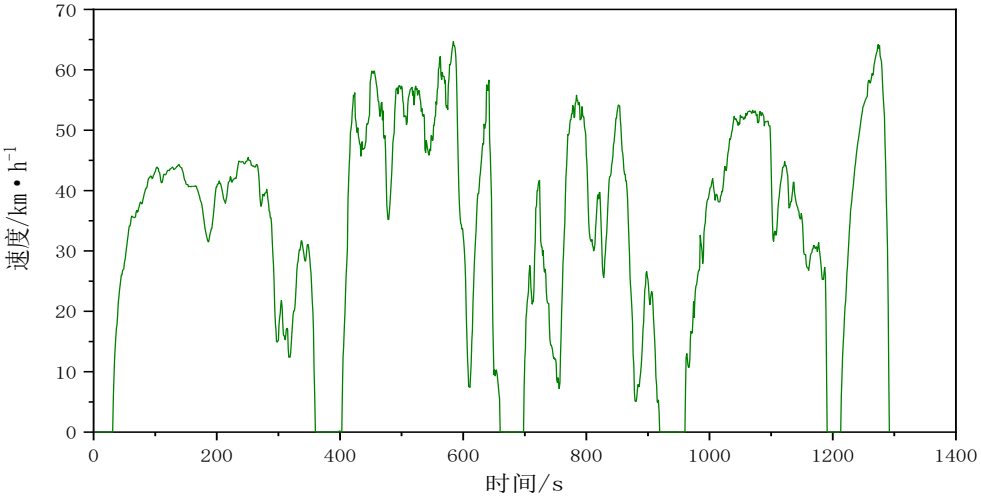


图 6.8 MBO 算法畅通交通代表性行驶工况

6.2.5 结果分析

（1）拥堵交通行驶工况

随机短行程法生成的拥堵交通代表性行驶工况的速度分布参数[16][17]如表 6.5，MBO 生成的拥堵交通代表性行驶工况的速度分布参数如表 6.6。在随机法生成的代表性行驶工况中，速度分布在 0-10 km/h 区间的比例最高，达到 54.26%，平均速度仅有 1.82 km/h，分布在 10-20km/h 区间的比例次之，速度大于 50 km/h 的比例最低。而在 MBO 法生成的代表性行驶工况中，速度分布在 0-10 km/h 区间的比例仍最高，速度大于 50 km/h 的比例为 0。

表 6.5 拥堵交通代表性行驶工况(随机短行程法)的速度分布参数

速度分布（km/h）	平均速度（km/h）	最大速度（km/h）	最小速度（km/h）	比例分布（%）
[0,10]	1.82	10.00	0	54.26
(10,20]	14.87	20.00	10.10	27.49
(20,30]	23.27	29.70	20.10	13.75
(30,40]	34.33	39.80	30.20	2.09
(40,50]	44.78	49.30	40.80	1.37
>50	51.65	52.50	50.10	1.05

表 6.6 拥堵交通代表性行驶工况(MBO)的速度分布参数

速度分布（km/h）	平均速度（km/h）	最大速度（km/h）	最小速度（km/h）	比例分布（%）
[0,10]	1.07	10.00	0.00	47.14
(10,20]	14.74	20.00	10.10	24.39
(20,30]	24.53	30.00	20.10	21.78
(30,40]	34.26	39.90	30.10	5.20
(40,50]	42.45	44.50	40.20	1.49
>50	0.00	0.00	0.00	0.00

拥堵交通实验行驶工况的速度-加速度联合分布表[18][19]如表 6.7，随机法生成的拥堵交通代表性行驶工况的速度-加速度联合分布表如表 6.8, MBO 生成的拥堵交通代表性行驶工况的速度-加速度联合分布表如表 6.9。拥堵交通实验行驶工况和随机法生成的拥堵交通代表性行驶工况最大分布的区间在 0-10 km/h，加速度区间在-1-0 m/s^2，最大分布分别为 0.4873 和 0.4598。用 MBO 生成代表性行驶工况后，最大分布的区间没有改变，但是最大分布减小为 0.4149。

表 6.7 拥堵交通实验行驶工况的速度(km/h)-加速度(m/s^2)联合分布表

加速度 / 速度	<=-3	(-3,-2]	(-2,-1]	(-1,0]	(0,1]	(1,2]	(2,3]	(3,4]
[0,10]	0.0000	0.0005	0.0046	0.4873	0.0481	0.0073	0.0020	0.0005
(10,20]	0.0002	0.0008	0.0074	0.0829	0.0842	0.0036	0.0002	0.0001
(20,30]	0.0001	0.0007	0.0057	0.0760	0.0786	0.0011	0.0001	0.0000
(30,40]	0.0000	0.0002	0.0027	0.0411	0.0403	0.0003	0.0000	0.0000
(40,50]	0.0000	0.0000	0.0005	0.0104	0.0102	0.0001	0.0000	0.0000
>50	0.0000	0.0000	0.0001	0.0012	0.0009	0.0000	0.0000	0.0000

表 6.8 随机法生成的拥堵交通代表性行驶工况的速度(km/h)-加速度(m/s^2)联合分布表

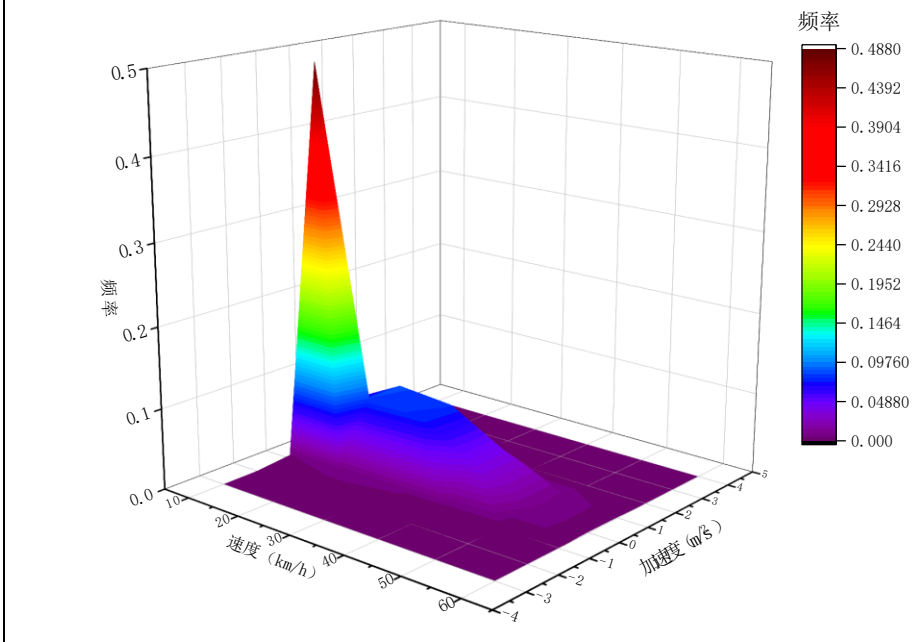
加速度 速度	<=-3	(-3,-2]	(-2,-1]	(-1,0]	(0,1]	(1,2]	(2,3]	(3,4]
[0,10]	0.0000	0.0024	0.0048	0.4598	0.0651	0.0064	0.0016	0.0024
(10,20]	0.0000	0.0008	0.0161	0.1326	0.1190	0.0056	0.0008	0.0000
(20,30]	0.0000	0.0008	0.0064	0.0619	0.0683	0.0000	0.0000	0.0000
(30,40]	0.0000	0.0000	0.0024	0.0048	0.0137	0.0000	0.0000	0.0000
(40,50]	0.0000	0.0000	0.0000	0.0056	0.0080	0.0000	0.0000	0.0000
>50	0.0000	0.0000	0.0000	0.0056	0.0048	0.0000	0.0000	0.0000

表 6.9 MBO 生成的拥堵交通代表性行驶工况的速度(km/h)-加速度(m/s^2)联合分布表

加速度 速度	<=-3	(-3,-2]	(-2,-1]	(-1,0]	(0,1]	(1,2]	(2,3]	(3,4]
[0,10]	0.0000	0.0007	0.0067	0.4149	0.0394	0.0074	0.0000	0.0022
(10,20]	0.0000	0.0000	0.0089	0.1331	0.0967	0.0052	0.0000	0.0000
(20,30]	0.0000	0.0000	0.0045	0.1160	0.0974	0.0000	0.0000	0.0000
(30,40]	0.0000	0.0000	0.0022	0.0297	0.0201	0.0000	0.0000	0.0000
(40,50]	0.0000	0.0000	0.0000	0.0104	0.0045	0.0000	0.0000	0.0000
>50	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

拥堵交通实验行驶工况的速度-加速度联合分布图[20]、随机法生成的拥堵交通代表性行驶工况的速度-加速度联合分布图和 MBO 生成的拥堵交通代表性行驶工况的速度-加速度联合分布图如图 6.9(a)、图 6.9(b)和图 6.9(c)。实验行驶工况、随机法生成的行驶工况[21]和 MBO 生成的行驶工况形状相似，频率均为先升高，后降低，但是 MBO 生成的行驶工况在图边界的拟合上更为接近。

(a)拥堵交通实验行驶工况



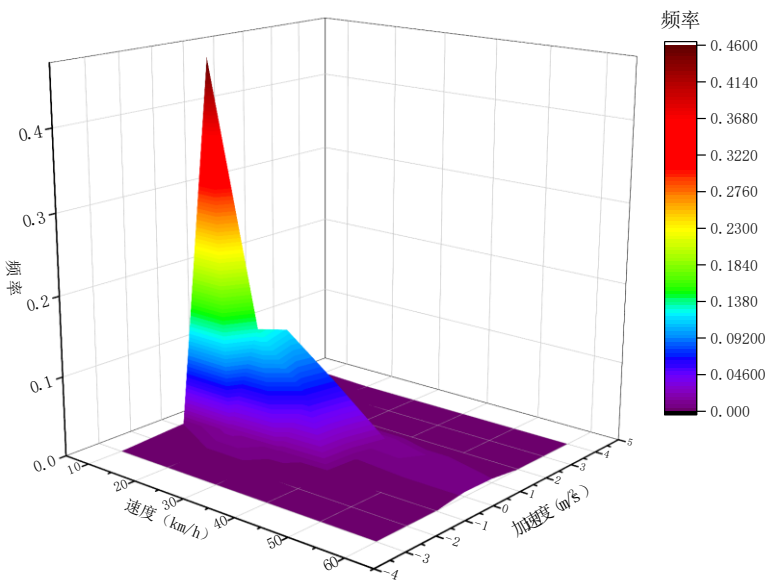
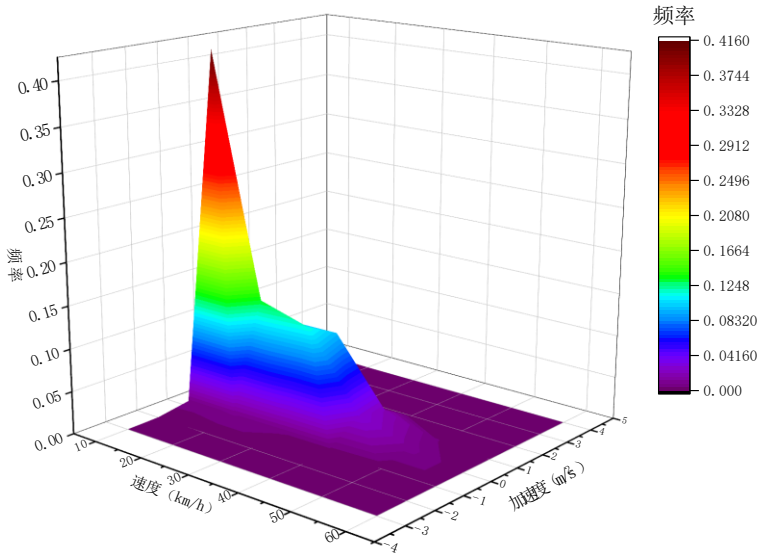
(b)随机法生成的拥堵交通代表性行驶工况	
(c)MBO 生成的拥堵交通代表性行驶工况	

图 6.9 速度-加速度联合分布图

我们还针对随机法、MBO 得到的拥挤交通代表性行驶工况与实验数据生成的行驶工况进行特征参数的对比，如表 6.10。

表 6.10 拥挤交通代表性行驶工况特征参数对比

指标	实验数据	随机法	相对误差	MBO	相对误差
T(s)	296.64	259.20	12.62%	258.60	12.83%
T_a(s)/T(s)	0.28	0.31	8.33%	0.30	4.47%
T_d(s)/T(s)	0.21	0.21	0.23%	0.21	1.23%
T_i(s)/T(s)	0.15	0.17	16.32%	0.15	1.96%
T_y(s)	0.36	0.31	13.33%	0.35	3.60%
$V_{\max}$(km/h)	56.72	55.00	3.04%	56.70	0.04%
V_{mean}(km/h)	31.92	26.10	18.23%	33.04	3.52%
V_d(km/h)	37.06	31.44	15.17%	38.75	4.56%

V_{std}(km/h)	17.17	17.09	0.50%	18.77	9.31%
$a_{\max}$(m/s^2)	2.09	2.27	8.19%	1.89	9.84%
$a_{\min}$(m/s^2)	-2.44	-2.47	1.16%	-2.44	0.02%
a_{std}(m/s^2)	0.50	0.55	9.39%	0.49	1.03%
a_a(m/s^2)	0.48	0.49	3.47%	0.44	7.62%
a_d(m/s^2)	-0.67	-0.72	8.62%	-0.68	1.51%
相对误差均值	-	-	8.47%	-	4.39%

由表可知，随机法的相对误差为 8.47%，而 MBO 相对误差为 4.39%，相较于随机法提高了 48.17%。且在单项特征参数方面，随机法存在相对误差大于 15%的差数据，而 MBO 的单项最大相对误差为 12.83%。

（2）畅通交通行驶工况

随机法生成的畅通交通代表性行驶工况的速度分布参数如表 6.11，MBO 生成的畅通交通代表性行驶工况的速度分布参数如表 6.12。在随机法生成的代表性行驶工况中，速度分布在 0-10km/h 区间的比例最高，平均速度为 1.79km/h，速度大于 60km/h 的比例最低，为 2.78%，在 10-50km/h 的速度分布中，各区间分布较为均匀。在 MBO 生成的代表性行驶工况中，速度分布的更加合理，其中在 40-50km/h 区间的比例最高，30-40km/h 和 40-50km/h 的比例均超过了 20%，符合日常交通行驶情况。

与拥堵交通代表性行驶工况的速度分布参数相比，二者的最大速度不同，在低速区和高速区所占的比例明显不同，拥堵交通代表性行驶工况在低速区的分布比例较高，包含了许多怠速的情况，畅通交通代表性行驶工况则在高速区的分布比例高。

表 6.11 随机法生成的畅通交通代表性行驶工况的速度分布参数

速度分布（km/h）	平均速度（km/h）	最大速度（km/h）	最小速度（km/h）	比例分布（%）
[0,10]	1.79	10.00	0	24.85
(10,20]	15.18	19.87	10.04	11.34
(20,30]	25.36	29.93	20.10	15.43
(30,40]	34.84	39.90	30.10	18.21
(40,50]	44.33	50.00	40.10	18.06
(50,60]	55.36	60.00	50.20	9.34
>60	61.80	65.50	60.10	2.78

表 6.12 MBO 生成的畅通交通代表性行驶工况的速度分布参数

速度分布（km/h）	平均速度（km/h）	最大速度（km/h）	最小速度（km/h）	比例分布（%）
[0,10]	1.49	10.00	0.00	17.88
(10,20]	15.06	19.90	10.20	7.58
(20,30]	25.55	30.00	20.10	12.90
(30,40]	35.64	40.00	30.10	20.96
(40,50]	44.04	50.00	40.10	22.05
(50,60]	54.15	60.00	50.10	16.72

>60	62.46	64.70	60.20	1.91

畅通交通实验行驶工况的速度-加速度联合分布表如表 6.13，随机法生成的畅通交通代表性行驶工况的速度-加速度联合分布表如表 6.14,MBO 生成的畅通交通代表性行驶工况的速度-加速度联合分布表如表 6.15。畅通交通实验行驶工况和随机法生成的畅通交通代表性行驶工况最大分布的区间在 0-10km/h，加速度区间在 -1-0 m/s^2，最大分布分别为 0.1398 和 0.2168。在 MBO 生成的代表性行驶工况中，最大分布的区间没有改变，最大分布为 0.1536，相较于随机法有所减小，更接近实验行驶工况。

表 6.13 畅通交通实验行驶工况的速度（km/h）-加速度（m/s^2）联合分布表

加速度 / 速度	<=-3	(-3,-2]	(-2,-1]	(-1,0]	(0,1]	(1,2]	(2,3]	(3,4]
[0,10]	0.0000	0.0004	0.0026	0.1398	0.0167	0.0041	0.0013	0.0003
(10,20]	0.0002	0.0006	0.0054	0.0289	0.0396	0.0037	0.0001	0.0000
(20,30]	0.0003	0.0011	0.0065	0.0463	0.0640	0.0020	0.0001	0.0000
(30,40]	0.0002	0.0009	0.0061	0.0724	0.0874	0.0012	0.0000	0.0000
(40,50]	0.0001	0.0004	0.0040	0.0872	0.0893	0.0005	0.0000	0.0000
(50,60]	0.0001	0.0003	0.0021	0.0592	0.0587	0.0005	0.0000	0.0000
>60	0.0001	0.0002	0.0014	0.0850	0.0787	0.0003	0.0000	0.0000

表 6.14 随机法生成的畅通交通代表性行驶工况的速度（km/h）-加速度（m/s^2）联合分布表

加速度 / 速度	<=-3	(-3,-2]	(-2,-1]	(-1,0]	(0,1]	(1,2]	(2,3]	(3,4]
[0,10]	0.0000	0.0008	0.0023	0.2168	0.0224	0.0046	0.0008	0.0008
(10,20]	0.0000	0.0008	0.0077	0.0563	0.0448	0.0031	0.0008	0.0000
(20,30]	0.0015	0.0000	0.0131	0.0664	0.0718	0.0015	0.0000	0.0000
(30,40]	0.0008	0.0015	0.0100	0.0710	0.0988	0.0000	0.0000	0.0000
(40,50]	0.0000	0.0008	0.0054	0.0787	0.0957	0.0000	0.0000	0.0000
(50,60]	0.0000	0.0000	0.0031	0.0394	0.0509	0.0000	0.0000	0.0000
>60	0.0000	0.0000	0.0008	0.0162	0.0108	0.0000	0.0000	0.0000

表 6.15 MBO 生成的畅通交通代表性行驶工况的速度（km/h）-加速度（m/s^2）联合分布表

加速度 / 速度	<=-3	(-3,-2]	(-2,-1]	(-1,0]	(0,1]	(1,2]	(2,3]	(3,4]
[0,10]	0.0000	0.0014	0.0020	0.1536	0.0137	0.0068	0.0014	0.0000
(10,20]	0.0000	0.0014	0.0055	0.0266	0.0396	0.0027	0.0000	0.0000
(20,30]	0.0007	0.0007	0.0048	0.0560	0.0635	0.0034	0.0000	0.0000
(30,40]	0.0000	0.0014	0.0027	0.1085	0.0949	0.0014	0.0007	0.0000
(40,50]	0.0000	0.0000	0.0068	0.1106	0.1031	0.0000	0.0000	0.0000
(50,60]	0.0000	0.0007	0.0048	0.0805	0.0792	0.0020	0.0000	0.0000
>60	0.0000	0.0000	0.0000	0.0082	0.0109	0.0000	0.0000	0.0000

畅通交通实验行驶工况的速度-加速度联合分布图、随机法生成的畅通交通代表性行驶工况的速度-加速度联合分布图和 MBO 生成的畅通交通代表性行驶工况的速度-加速度联合分布图如图 6.10(a)、图 6.10(b)和图 6.10(c)。实验行驶工况、随机法行驶工况和 MBO 法行驶工况形状相似，频率均为先升高，后降低，再升高，最后降低。在次高峰的描述上，MBO 生成的代表工况对应的联合分布图更为准确。从分布图整体的轮廓上可以看出 MBO 方法构建的工况与实际行驶工况更为接近。

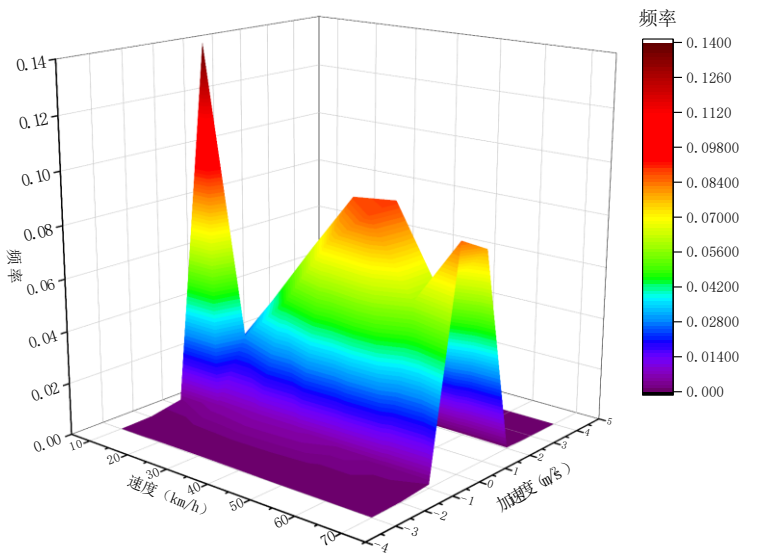
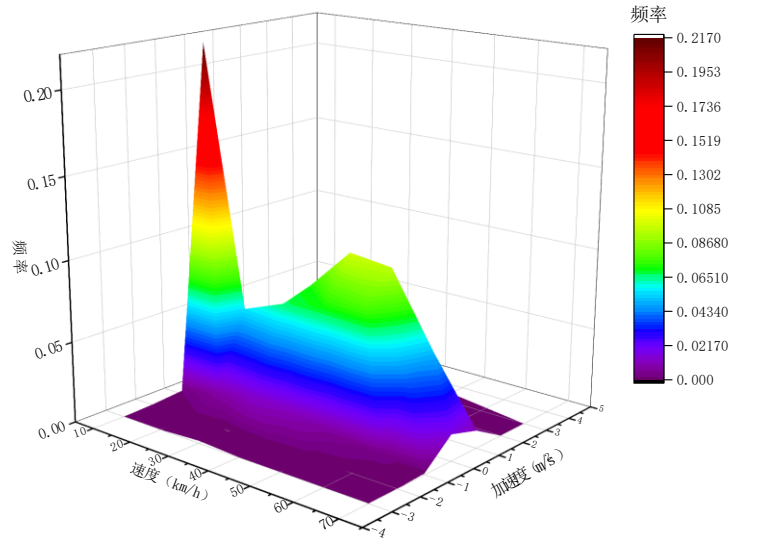
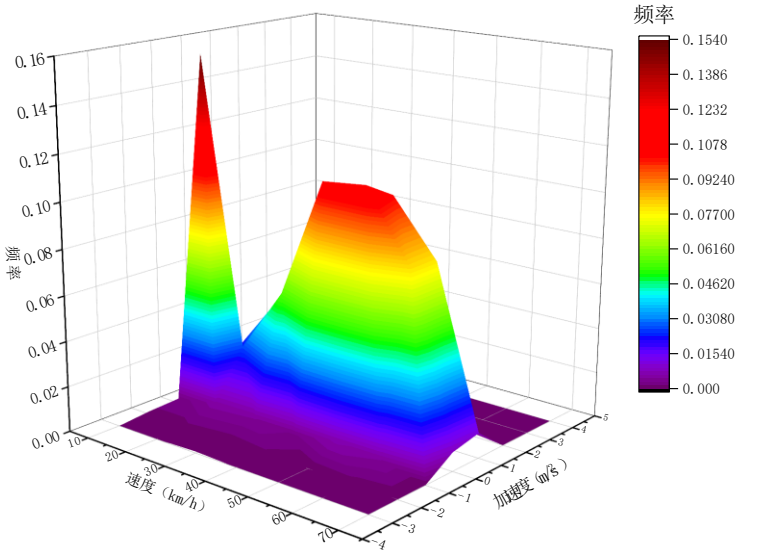
(a)畅通交通实验行驶工况	
(b)随机法生成的畅通交通代表性行驶工况	
(c)MBO 生成的畅通交通代表性行驶工况	

图 6.10 速度-加速度联合分布图

针对随机法、MBO 得到的畅通交通代表性行驶工况与实验数据生成的行驶工况，我们进行了特征参数的对比，如表 6.16。

表 6.16 畅通交通代表性行驶工况特征参数对比

指标	实验数据	随机法	相对误差	MBO	相对误差
$T(\mathrm{s})$	146.06	138.22	5.37%	134.11	8.18%
$T_a(\mathrm{s})/T(\mathrm{s})$	0.20	0.19	4.83%	0.20	2.38%
$T_d(\mathrm{s})/T(\mathrm{s})$	0.17	0.14	16.42%	0.17	0.33%
$T_i(\mathrm{s})/T(\mathrm{s})$	0.40	0.44	10.89%	0.40	0.68%
$T_y(\mathrm{s})$	0.23	0.23	2.69%	0.23	2.93%
$V_{\max}(\mathrm{km/h})$	29.91	27.17	9.16%	28.48	4.77%
$V_{mean}(\mathrm{km/h})$	11.05	9.48	14.17%	10.59	4.12%
$V_d(\mathrm{km/h})$	18.19	16.85	7.37%	17.61	3.20%
$V_{std}(\mathrm{km/h})$	10.11	9.26	8.39%	10.38	2.66%
$a_{\max}(\mathrm{m/s^2})$	1.86	2.33	24.96%	1.68	10.01%
$a_{\min}(\mathrm{m/s^2})$	-1.71	-1.81	5.89%	-1.47	13.80%
$a_{std}(\mathrm{m/s^2})$	0.43	0.48	12.46%	0.41	4.63%
$a_a(\mathrm{m/s^2})$	0.56	0.59	5.08%	0.52	7.96%
$a_d(\mathrm{m/s^2})$	-0.65	-0.80	24.66%	-0.60	6.85%
相对误差均值	-	-	10.88%	-	5.18%

由上表可知，MBO 得到的畅通交通代表性行驶工况与实验数据生成的行驶工况的相对误差均值为 5.18%，相较于随机法的 10.88%提高了 52.39%，提升十分明显。且不难看出，随机法在单项特征参数的相对误差上存在极差的数据，达到了 24.96%，而 MBO 的单项指标最高相对误差为 13.80%，提高了 44.71%。

通过对上述拥挤和畅通两种交通状况的分析，结合加速度-速度联合分布的分析以及构建工况与实验工况的特征参数对比分析可以得出，随机法具有一定的可靠度，但不是很好。我们提出的 MBO 算法相对于随机法有更高的可靠性以及合理性，各项指标基本上均优于随机法。

7. 模型评价及推广

我们在问题 1 中建立了数据处理的数学模型，对于类型一的部分异常数据进行了线性插值处理，但是实际车辆的行驶工况中可能存在速度非线性变化的情况，而线性插值无法很好地还原这些情况，因此会造成部分运动学片段的可靠性降低。对于类型二的异常数据进行了“大速度”示踪。对于类型三的异常数据，由于长期停车的定义并没有具体量化的定义，因此筛选的数据会因人而异，而模型中对于此筛选标准并没有进行评价，即无法确定最好的筛选标准是怎样的，因此存在需要改进的空间。但其余类型的不良数据均依据问题 1 的模型进行了准确的筛选。

问题 2 模型中，切分片段的方式简单明了，此问主要考虑的应该是片段的有

效性判断准则。其中匀速检测机制中我们对匀速的判定指标为加速度绝对值波动在 0.1m/s^2，时间跨度超过 3s。此处时间跨度没有严格的理论依据，缺乏有效的数据实验验证。设计的切分方式虽然可靠，但依然存在改进的地方。

在问题 3 的数学模型中，我们建立了以 14 个指标为基础的汽车运动特征评价体系，但还有一小部分与汽车运动特征相关的指标，因其关联程度不高并没有被纳入评价体系，因此在描述汽车运动特征的完整度上可能还存在一定程度的缺失。在指标的降维处理上，我们采用了主成分分析，得到了降维后的数据，这部分数据对于原数据的信息携带方面存在部分丢失的情况，因此会影响后续聚类分析的结果。在利用运动学片段构建最终的代表性汽车工况时，我们在模型中先考虑了随机法，但效果不尽如人意，于是提出了新算法，即 (MBO)算法，最终得到了较为满意的结果。值得一提的是，在第三问中我们并没有追求构建的曲线与实验数据的特征参数的平均相对误差取到一个极小的值，事实上增加代码运行的次数是可以得到更好的结果的，我们更加关注算法在正常少量试验次数下的普遍效果。

我们所建立的模型总体来说良好。两处创新点在于：

1.针对数据筛选的问题，特别提出了数据示踪法，用大速度(300km/h)对异常加、减速度数据进行标记并巧妙地利用此标记在第 2 问中对这部分数据进行了剔除。

2.提出了 Mini-batch Optimization(MBO) 算法，在随机短行程法的基础上结合了最佳增量法的思想，因此在保留随机法较高计算效率的前提下较大幅度地减小了其计算所得的特征参数与实验所得特征参数的误差。

在问题 1 及问题 3 模型中的此两点创新分别可以可推广到各类数据追踪情况和其他的数据的挑选方式过程当中去。

8. 参考文献

[1]王楠楠. 城市道路行驶工况构建及油耗研究[D].合肥工业大学,2012.

[2]张宏,姚延钢,杨晓勤.城市道路轻型汽车行驶工况构建[J/OL].西南交通大学学报:1-9[2019-09-20].

[3] Karande, S., Olson, M., and Saha, B. Development of Representative Vehicle Drive Cycles for Hybrid Applications[J]. SAE Technical Paper 2014-01-1900, 2014, doi:10.4271/2014-01-1900.

[4]李阿午. 太原市轻型车行驶工况构建与排放特性的研究[D].太原理工大学,2018.

[5]周浩泽,于彭城,徐寒梅,沈子龙.同位素示踪技术在药学研究领域的应用进展[J].药学进展,2019,43(06):459-467.

[6]马洪龙. 汽车行驶工况的构建及传动系参数优化匹配研究[D].合肥工业大学,2014.

[7]郑殿宇,吴晓刚,陈汉,杜玖玉.哈尔滨城区乘用车行驶工况的构建[J].公路交通科技,2017,34(04):101-107.

[8]高建平. 基于 PCA 和 FCM 的汽车行驶工况研究与构建[A]. 河南省汽车工程学会.第十三届河南省汽车工程科技学术研讨会论文集[C].河南省汽车工程学会:河南省汽车工程学会,2016:4.
[9]王川.车辆道路行驶工况构建典型方法研究[J].赤峰学院学报(自然科学版),2016,32(17):35-36.
[10]姜平.城市混合道路行驶工况的构建研究[D].合肥工业大学,2011.
[11]曾小荣,孔令文,杨学易,李朋,汪勇.主成分分析在车辆行驶工况中的应用[J].汽车实用技术,2014(05):5-9.
[12]董恩源,颜文胜,申江卫,李江波.聚类分析法在城市公交行驶工况开发中的应用[J].昆明理工大学学报(自然科学版),2013,38(05):41-44+51.
[13]李洋. 基于聚类算法的汽车行驶工况研究[D].北京理工大学,2016.
[14]杜勇,相臣,马洪龙.基于 FCM 聚类法对行驶工况的构建[J].农业装备与车辆工程,2014,52(09):42-45.
[15]石琴,马洪龙,丁建勋,龙建成,凌翔.改进的 FCM 聚类法及其在行驶工况构建中的应用[J].中国机械工程,2014,25(10):1381-1387.
[16]李宁. 城市道路车辆行驶工况的构建与研究[D].河北农业大学,2013.
[17]张家顺. 高原地区混合动力公交客车城市道路行驶工况研究[D].昆明理工大学,2013.
[18]朱俊虎,石琴,周洁瑜.城市公交车行驶工况的构建[J].交通科技与经济,2011,13(03):108-112.
[19]Lin J, Niemeier D A. Exploratory analysis comparing a stochastic driving cycle to California's regulatory cycle[J]. Atmospheric Environment, 2002, 36(38):5759-5770.
[20]姜平，石琴，陈无畏，黄志鹏. 基于小波分析的城市道路行驶工况构建的研究[J]. 汽车工程, 2011(1):70-73.
[21]Ho, Sze-Hwee, Wong, Yiik-Diew, Chang, Victor Wei-Chung. Developing Singapore Driving Cycle for passenger cars to estimate fuel consumption and vehicular emissions [J]. Atmospheric Environment,2014,97:353-362.

9. 附录与代码

声明：相关代码已压缩上传为附件，代码使用说明如下。

文件夹 Q1code：

Q11duandian:分别取三个文件的断点并保存在 EXCEL 文件“时间不连续断点.xlsx”当中，文件的第 1 到 3 列分别对应文件 1 到 3。

输入：“文件 X(X=1,2,3). xlsx”
输出：数据断点

Q11interpolation:对各个文件的断点按模型需求进行线性插值，得到插值后的数据保存在矩阵“WA”中，对应的插值点及原有数据点在矩阵“WA”中的索引保存在“WAindex”当中，“WA”与“WAindex”均保存在“Inter.mat”文件中。运行代码时需要在注释更改的部分(5.11.72 行)对数字进行相应改动，1 对应文件 1，以此类推。

输入：“时间不连续断点.xlsx”“文件 X(X=1,2,3) - 副本.xlsx”“数值时间.xlsx”

输出：“InterX(X=1,2,3).mat”

Q1234abnormal:对插值得到的各个“Inter.mat”文件进行异常数据类型 2-4 的处理，处理后的数据保存在“WA”中，“WA”保存于“abnormal.mat”文件中。

输入:“InterX(X=1,2,3).mat”
输出：“abnormalX(X=1,2,3).mat”

文件夹 Q2code：
Q2：对各个“abnormal.mat”按模型求取运动学片段，并将各片段特征参数及片段中的原数据分别保存于“feature_value.mat”与“Segement_.mat”中。运行代码时需要在注释更改的部分(1,112,113 行)进行相应对数字进行相应改动，1 对应文件 1，以此类推。

输入：“abnormalX(X=1,2,3).mat”
输出：“feature_valueX(X=1,2,3).mat”“Segement_X(X=1,2,3).mat”

文件夹 Q3code：
Q3_random:将获得的三个文件中的运动学片段拼接在一起，进行主成分分析后再聚类分析，保存拼接后的所有运动学片段“s”及聚类分析后各类中的片段在总片段中的索引，这里聚类共得到两类。在两类中分别随机选取片段拼凑成所需长度，得到其在总片段中的索引“C1”与“C2”。“s”、“C1”与“C2”均放置于“sum_s1.mat”中。同时在“jvlei_result.mat”保存了所有运动学片段“s”及其特征参数值，以及两类片段在总片段中的索引“D1”与“D2”。

输入：“feature_value1.mat”，“feature_value2.mat”，“feature_value2.mat”“Segement_1.mat”，“Segement_2.mat”，“Segement_3.mat”
输出：“sum_s1.mat”，“jvlei_result.mat”

Q3_MBO:用 MBO 算法实现运动学片段的选取。运行代码时需要在注释更改的部分(13,14,15 行)进行对相应数字进行更改。

输入：“jvlei_result.mat”
输出：“result_第 X(X=1,2)类_改进”

Q3_post:计算所选两类运动学片段以及对应所挑选的代表性片段的速度加速度联合分布，分别保存 ivA 与 ivA2，iva 与 iva2 当中。计算所挑选的代表性片段的平均速度、最大速度、最小速度及比例分布分别保存在矩阵 lianhe_s1 与 lianhe_s2 的第 1 至 4 列中。

输入(随机法)：“sum_s1.mat”，“jvlei_result.mat”
输出(随机法)：ivA，iva，ivA2，iva2，lianhe_s1，lianhe_s2

输入(MBO)：“result_第一类_改进”，“result_第一类_改进”，“sum_s1.mat”，“jvlei_result.mat”
输出(MBO)：ivA，iva，ivA2，iva2，lianhe_s1，lianhe_s2

代码：

Q11duandian:

```
clear all;
[~,~,A1] = xlsread('文件 1.xlsx');
[~,~,A2] = xlsread('文件 2.xlsx');
[~,~,A3] = xlsread('文件 3.xlsx');
t1 = A1(2:end,1);
t2 = A2(2:end,1);
t3 = A3(2:end,1);
cellnum1 = cellfun(@(x) str2double(regexp(x,'\d*\.?\d*','match')),...
    t1,'UniformOutput',false);
cellnum2 = cellfun(@(x) str2double(regexp(x,'\d*\.?\d*','match')),...
    t2,'UniformOutput',false);
cellnum3 = cellfun(@(x) str2double(regexp(x,'\d*\.?\d*','match')),...
    t3,'UniformOutput',false);
t_hms1 = [];
t_hms2 = [];
t_hms3 = [];
for i = 1:size(t1,1)
%    y1 = cellnum1{i,1}(1,1);
%    m1 = cellnum1{i,1}(1,2);
   d1 = cellnum1{i,1}(1,3);
   d1 = d1-18;
   hh1 = cellnum1{i,1}(1,4);
   mm1 = cellnum1{i,1}(1,5);
   ss1 = cellnum1{i,1}(1,6);
   t_hms1(i,1) = d1*3600*24+hh1*3600+mm1*60+ss1;
end
for i = 1:size(t2,1)
%    y2 = cellnum2{i,1}(1,1);
%    m2 = cellnum2{i,1}(1,2);
   d2 = cellnum2{i,1}(1,3);
   d2 = d2-1;
   hh2 = cellnum2{i,1}(1,4);
   mm2 = cellnum2{i,1}(1,5);
   ss2 = cellnum2{i,1}(1,6);
   t_hms2(i,1) = d2*3600*24+hh2*3600+mm2*60+ss2;
end
for i = 1:size(t3,1)
%    y3 = cellnum3{i,1}(1,1);
%    m3 = cellnum3{i,1}(1,2);
   d3 = cellnum3{i,1}(1,3);
   d3 = d3-1;
   hh3 = cellnum3{i,1}(1,4);
   mm3 = cellnum3{i,1}(1,5);
   ss3 = cellnum3{i,1}(1,6);
   t_hms3(i,1) = d3*3600*24+hh3*3600+mm3*60+ss3;
end
cd1=[];
cd2=[];
cd3=[];
indexcd=0;
for i = 1:size(t1,1)-1
    if t_hms1(i+1,1)~=t_hms1(i,1)+1
        indexcd = indexcd+1;
        cd1(indexcd,1)=i+1;
    end
end
indexcd=0;
for i = 1:size(t2,1)-1
    if t_hms2(i+1,1)~=t_hms2(i,1)+1
        indexcd = indexcd+1;
        cd2(indexcd,1)=i+1;
    end
```

```
end
indexcd=0;
for i = 1:size(t3,1)-1
    if t_hms3(i+1,1)~=t_hms3(i,1)+1
        indexcd = indexcd+1;
        cd3(indexcd,1)=i+1;
    end
end
```

Q11interpolation：

```
clear all;
cd=xlsread('时间不连续断点.xlsx');
cd11 = cd(1:end,3);  %更改
cd1=cd11(find(~isnan(cd11)));
[~,~,A] = xlsread('文件 3 - 副本.xlsx');%更改
A1 = A(2:end,2:end);
A1=cell2mat(A1);
title = A(1,1:end);
% time = A(2:end,1);
% time=cell2mat(time);
T1 = xlsread('数值时间.xlsx','t3'); %更改
WA=[];
index = cd1-1;
index = [0;index];
WAindex = [1];
for i = 1:size(cd1,1)
    dindex = index(i+1)-index(i);
    WAindex = [WAindex;WAindex(4*i-3)+dindex-1;WAindex(4*i-3)+dindex];%加 dindex
    dt = T1(index(i+1)+1)-T1(index(i+1));     %加 dt-1
    WAindex = [WAindex;dt-2+WAindex(4*i-1);dt-1+WAindex(4*i-1)];
    if i == size(cd1,1)
        a = size(A1,1)-index(i+1);
        WAindex = [WAindex;WAindex(end)+a-1];
    end
end
index(1,1)=[1];
for i = 1:size(cd1,1)
   P1=[];
   if i==1
       WA(WAindex(4*i-3):WAindex(4*i-2),2:14)=A1(index(i):index(i+1),:);
       WA(WAindex(4*i-3):WAindex(4*i-2),1)=T1(index(i):index(i+1),:);
   else
       WA(WAindex(4*i-3):WAindex(4*i-2),2:14)=A1(index(i)+1:index(i+1),:);
       WA(WAindex(4*i-3):WAindex(4*i-2),1)=T1(index(i)+1:index(i+1),:);
   end
   v1 = A1(index(i+1),1);
   a11 = A1(index(i+1),2);
   a12 = A1(index(i+1),3);
   a13 = A1(index(i+1),4);
   v2 = A1(index(i+1)+1,1);
   a21 = A1(index(i+1)+1,2);
   a22 = A1(index(i+1)+1,3);
   a23 = A1(index(i+1)+1,4);
   dt = T1(index(i+1)+1,1)-T1(index(i+1),1);
   dv = v2-v1;
   da1 = a21-a11;
   da2 = a22-a12;
   da3 = a23-a13;
   t1 = T1(index(i+1),1);
   for j = 1:(dt-1)
       t1 = t1+1;
       part = (t1-T1(index(i+1),1))/dt;
       a_1 = a11+part*da1;
       a_2 = a12+part*da2;
       a_3 = a13+part*da3;
       v_1 = v1+part*dv;
       if v1==0||v2==0
```

```
        v_1 = 0;
      end
      P1 = [P1;v_1,a_1,a_2,a_3];
    end
    WA(WAindex(4*i-1):WAindex(4*i),2:5)=P1;
    tt1 = WA(WAindex(4*i-1)-1,1)+1;
    tt2 = tt1+WAindex(4*i)-WAindex(4*i-1);
    ttt = linspace(tt1,tt2,tt2-tt1+1);
    WA(WAindex(4*i-1):WAindex(4*i),1)=ttt';
    if i == size(cd1,1)
      WA(WAindex(end-1):WAindex(end),2:14)=A1(index(i+1)+1:size(A1,1),:);
      WA(WAindex(end-1):WAindex(end),1)=T1(index(i+1)+1:size(A1,1),:);
    end
  end
  save('Inter3.mat','WA','WAindex'); %更改
```

Q1234abnormal:

```
  clear all;
  for i=1:3
    stemp = ['Inter' num2str(i) '.mat'];
    load(stemp,'WA','WAindex');
    WAa = zeros(size(WA,1),1);%在最后一列增加加速度
    WA = [WA WAa];
    for j=2:size(WA,1)      %Q12(加速度异常)
      dv=WA(j,2)-WA(j-1,2);
      WA(j,end)=dv;
      if dv<-28.8||dv>14.2
        WA(j-1,2)=300;
      end
    end
    %%
    sn=[];
    for n = 1:size(WA,1)  %Q14 怠速大于 3 分钟
      if WA(n,2)<10
        sn=[sn;n];
      else
        if size(sn,1)>=180
          WA(sn,2)=0;
        end
        sn=[];
      end
    end
    %% 由于要判断中间突然出现的异常数据，需要把插在异常数据间的数据清理掉
    zong = (size(WAindex,1)-2)/4;
    for q=1:zong
      if abs(WA(WAindex(4*q-1)-1))<=3&&abs(WA(WAindex(4*q)+1))<=3
        WA(WAindex(4*q-1):WAindex(4*q),:)=[];
      end
    end
    index0 = find(WA(:,2)==0);
    B = [0 find(diff(index0)~=1)' length(index0)];
    for m = 1:length(B)-2    %Q13(长期停车)
      tmp1(m,:)= index0((B(m)+1):B(m+1));
      if size(tmp1,1)>=20
        n = m+1;
        tmp2(n,:) = index0((B(n)+1):B(n+1));
        if size(tmp2,1)>=20
          num_0 = tmp2(1,1)-tmp1(1,end)-1;
          if num_0<=3
            WA(tmp1(1,end)+1:tmp2(1,1)-1)=0;
          end
        end
      end
      tmp1=[];
      tmp2=[];
    end
    stemp = ['abnormal' num2str(i) '.mat'];
```

```
    save(stemp,'WA');
end
```

Q2：

```
number = load('abnormal3.mat');%更改
velocity = number.WA(:,2);
feature = number.WA(:,2:5);
feature = [feature number.WA(:,end)];
index = find(velocity == 0);%找到速度中的 0 元素索引
Segement = {};
anchor = 1;

for i=2:size(index,1)
   if index(i)-index(i-1)~=1
      seg = feature(anchor:index(i)-1,:);
      b=seg(:,1);
      seg_0_ind = find(seg(:,1) == 0);
      %判断片段是否存在异常长时间怠速
      if seg_0_ind(end)>180
         j = seg_0_ind(end)-180;
         seg = seg(j+1:end,:);  %只取最后的 180 秒作为怠速阶段
      end
      %判断是否存在堵车等慢速运行
      k = size(find(seg(:,1)<=10),1);
      m = size(seg,1);
      if k == m
         anchor = index(i)- size(seg,1);
      else
          anchor = index(i);
          Segement = [Segement,seg];
      end
   end
end
seg = feature(anchor:size(velocity,1),:);
Segement = [Segement,seg];
disp(['运动片段的个数:' sprintf('%4i\t',size(Segement,2))])

%筛选有用片段%
Segement_ = {};
for i=1:size(Segement,2)
   seg_ = Segement(i);
   seg_ = cell2mat(seg_);
   %判断片段的可靠性%
   %1~是否存在示踪的大速度异常值
   num_0 = size(find(seg_(:,1) == 300 ),1);
   if num_0 == 0

      %2~是否存在匀速阶段等四个过程
      a = seg_(:,5);
      V = seg_(:,1);
      V_0_index = find(V == 0);
      num2zero = V_0_index(end)-1;
      %V(1:num2zero) = 0;
      a(1:num2zero) = 0;
      seg_(:,5) = a;
      %seg_(:,1) = V;
      a_index = find(a<=0.54 & a>=-0.54);
      a_lianxu =  [0 find(diff(a_index)~=1)' length(a_index)];

      for i=3:size(a_lianxu,2) %跳过怠速区
         ii = a_lianxu(i)-a_lianxu(i-1);
         if ii>3          %超过 3 秒就当做匀速阶段
            Segement_ = [Segement_, seg_];
            break;
         end

```

```
        end
    end
end
disp(['有效运动片段的个数:' sprintf('%4i\t',size(Segement_,2))])

%筛选特征值
feature_value = [];
for i=1:size(Segement_,2)
    seg_feature = Segement_(i);
    seg_feature = cell2mat(seg_feature);
    V = seg_feature(:,1);
    %怠速时间
    seg_0_index = find(V == 0);
    Ti = seg_0_index(end);

    A = seg_feature(:,5);
    T = size(seg_feature,1); %运行时间
    Ta = 0; %加速时间初始值
    Ta_index = find(A>0.54);
    Ta_lianxu =  [0 find(diff(Ta_index)~=1)' length(Ta_index)];
    for i=2:size(Ta_lianxu,2)
        ii = Ta_lianxu(i)-Ta_lianxu(i-1);
        Ta = Ta+ii;
    end
    Td = 0; %减速时间初始值
    Td_index = find(A<-0.54);
    Td_lianxu =  [0 find(diff(Td_index)~=1)' length(Td_index)];
    for i=2:size(Td_lianxu,2)
        ii = Td_lianxu(i)-Td_lianxu(i-1);
        Td = Td+ii;
    end

    Ty = T-(Ta + Td + Ti);%匀速时间

    Vmax = max(V);
    Vmean = mean(V);
    Vd = mean(V(Ti:end));
    Vstd = std(V);
    amax = max(A);
    amin = min(A);
    astd = std(A);
    aa = mean(A(Ta_index));
    ad = mean(A(Td_index));


    feature_value = [feature_value;T Ta/T Td/T Ti/T Ty/T Vmax Vmean...
      Vd  Vstd amax/3.6 amin/3.6 astd/3.6 aa/3.6 ad/3.6] ;
end
%save ('feature_value1.mat','feature_value')%更改
%save ('Segement_1.mat','Segement_')%更改
```

Q3_random:

```
clear all
load('feature_value1.mat');
x1 = feature_value;
load('feature_value2.mat');
x2 = feature_value;
load('feature_value3.mat');
x3 = feature_value;
load('Segement_1.mat');
s1 = Segement_;
load('Segement_2.mat');
s2 = Segement_;
load('Segement_3.mat');
s3 = Segement_;
s = [s1 s2 s3];
```

```
x0 = [x1;x2;x3];
x = zscore(x0);
[p,princ,egenvalue]=princomp(x); %调用主成分
p1=p(:,1:5);          %输出前 3 主成分系数
y = x*p1;
[Idx,Ctrs,SumD,D] = kmeans(y,2);%聚类
silhouette(y,Idx);
D1=[];
D2=[];
for i = 1:size(D,1)
   if D(i,1)<= D(i,2)
      D(i,3) = 1;
      D1 = [D1;i D(i,1) D(i,2)];
   else
      D(i,3) = 2;
      D2 = [D2;i D(i,1) D(i,2)];
   end
end
sum1=0;
sum2=0;
index_1 = [];
index_2 = [];
for i = 1:1000
   index1 = D1(randperm(size(D1,1),1),1);
   index_1 = [index_1;index1];
   L1 = size(s{1,index1},1);
   sum1 = sum1 + L1;
   if sum1 >= 1200
      if sum1 <= 1300
         break
      else
         sum1 = sum1-L1;
         index_1(end,1)=0;
      end
   end
end

for i = 1:1000
   index2 = D2(randperm(size(D2,1),1),1);
   index_2 = [index_2;index2];
   L2 = size(s{1,index2},1);
   sum2 = sum2 + L2;
   if sum2 >= 1200
      if sum2 <= 1300
         break
      else
         sum2 = sum2-L2;
         index_2(end,1)=0;
      end
   end
end

C1 = index_1(index_1~=0);
C2 = index_2(index_2~=0);
% save('sum_s1.mat','s','C1','C2')
% save('jvlei_result.mat','s','x0','D1','D2')
```

Q3_MBO:

```
% MBO
clear all
Data = load('jvlei_result.mat');
Data_1 = Data.x0(Data.D1(:,1),:); %类别 1 的特征值数据
Data_2 = Data.x0(Data.D2(:,1),:); %类别 2 的特征值数据
D1 = Data.D1(:,1);%原索引
D2 = Data.D2(:,1);
feature_mean_1 = mean(Data_1);   %类别 1 的特征均值
feature_std_1 = std (Data_1,0,1); %类别 1 的特征标准差
```

```
feature_mean_2 = mean(Data_2);
feature_std_2 = std (Data_2,0,1);
%-----------计算第二类的指标只需要将一下的 Data_1，feature_mean_1，替换成第二类即可----------%
feature_mean_c = feature_mean_1;%特征均值
Data_c = Data_1; %特征值
D_c = D1; %索引
%-------------------------------------------------------------------------------------------%
curve = [];%切片组成
s_index = [];
%随机选择一个片段作为工况初始运动片段
initial_seg_index = randperm(size(Data_c,1),1);  %随机
s_index = [s_index,D_c(initial_seg_index)];%原片段中的索引
initial_seg = cell2mat(Data.s(s_index));
initial_feature_value = Data_c(initial_seg_index,:);
initial_feature = initial_feature_value;
curve = [curve, initial_seg];

%bagging 随机 N 个片段
num_seg = 50;
key = 1;
optimal_index_ = [];%候选最优索引
optimal_feature_ = [];%候选最优片段特征值集合
while key
    bagging_seg_index = randperm(size(Data_c,1),num_seg); %获得 num_seg 个候选片段索引
    err = [];%每个候选片段的各特征偏差
    err_m=[];%每个片段的平均偏差

    %时间检验是否小于 1200 秒
    if length(curve)<1200
        for i=1:num_seg  %遍历候选区
            r_feature_value = Data_c(bagging_seg_index(i),:);%获得该片段的特征值
            r_ = (r_feature_value+initial_feature_value)/2; %挑选的片段与已有片段特征值融合
            bagging_seg = cell2mat(Data.s(D_c(bagging_seg_index(i))));%genggai
            for j=1:14
%                 bagging_seg = cell2mat(Data.s(Data_c(bagging_seg_index(j))));
                if length(bagging_seg)+length(curve)<1300
                    e = abs((feature_mean_c(j)-r_feature_value(j))/feature_mean_c(j));
                    err = [err,e];
                    err_mean = mean(err);
                else
                    e =1;
                    err = [err,e];
                    err_mean = mean(err);
                end
            end
            err_m = [err_m,err_mean];
        end
        [err_min, min_index] = min(err_m);
        optimal_index = bagging_seg_index(min_index);%对比偏差，得到候选片段最优的个体
        optimal_index_ = [optimal_index_,optimal_index];
        s_optimal_index = D_c(optimal_index);%bagging 的原 S 索引
        s_index = [s_index,s_optimal_index];
        optimal_seg_size = size(cell2mat(Data.s(s_optimal_index)),1); %挑取添加片段的大小
        curve = [curve ;cell2mat(Data.s(s_optimal_index))]; %片段合并
        %索引出随机最佳片段组合的各项特征值

        optimal_feature = Data_c(optimal_index_(end),:);% 当前循环下的候选区最优片段的特征值
        optimal_feature_ =[optimal_feature_ ;optimal_feature];% 保存每个循环的最优片段特征值

        initial_feature_value = (initial_feature_value+optimal_feature)/2; %更新初始状态，当前以构建工况特征值
    else
        if length(curve)>1200 && length(curve)<1300  %满足时间条件结束工况构建
```

```
        key = 0;
        break;
      else %去掉最后一个添加的片段
        curve(end-optimal_seg_size+1:end,:)=[]; %工况添加不合理去除 genggai
        initial_feature_value = 2*initial_feature_value - optimal_feature;%还原到上一个状态
        optimal_feature_ (end,:)=[];
        optimal_index_ (end)=[];
        s_index(end) = [];

      end

    end

end

optimal_feature_ = [initial_feature; optimal_feature_ ];
optimal_feature_mean = mean(optimal_feature_); %得到最佳片段组的各项特征值

e_optimal_ = [];

for i=1:14
   e_optimal = abs((feature_mean_c(i)-optimal_feature_mean(i))/feature_mean_c(i));
   e_optimal_ = [e_optimal_,e_optimal];
end
e_optiaml_end = mean(e_optimal_);%最终工况的特征值平均相对误差
result = [feature_mean_c;optimal_feature_mean;e_optimal_];

% xlswrite('result_第一类_2.xlsx',result,'Sheet1');
% xlswrite('result_第一类_2.xlsx',s_index,'Sheet2');
% xlswrite('result_第一类_2.xlsx',curve,'Sheet3');
```

Q3_post:

```
clear all
load('jvlei_result','D1','D2')
% load('sum_s1.mat');%改进前
% s1 = s(1,C1)';%改进前
% s2 = s(1,C2)';%改进前
load('sum_s1.mat','s');%改进后
C1 = xlsread('result_第一类_改进','sheet2');%改进后
C2 = xlsread('result_第二类_改进','sheet2');%改进后
s1 = s(1,C1)';
s2 = s(1,C2)';
s1 = cell2mat(s1);
s2 = cell2mat(s2);
S1 = s(1,D1(:,1))';%实验数据聚类 1
S2 = s(1,D2(:,1))';%实验数据聚类 2
S1 = cell2mat(S1);
S2 = cell2mat(S2);
%% 代表数据特征值
Data = load('jvlei_result.mat');
Data_1 = Data.x0(Data.D1(:,1),:);  %类别 1 的特征值数据
Data_2 = Data.x0(Data.D2(:,1),:);  %类别 2 的特征值数据
feature_mean_1 = mean(Data_1);     %类别 1 的特征均值
feature_std_1 = std (Data_1,0,1);  %类别 1 的特征标准差
feature_mean_2 = mean(Data_2);
feature_std_2 = std (Data_2,0,1);
%% 挑选数据特征值
type_1 = Data.x0(C1,:);  %类别 1 随机选取的特征值数据
type_2 = Data.x0(C2,:);  %类别 1 随机选取的特征值数据
type_mean_1 = mean(type_1);     %类别 1 随机选取的特征均值
type_std_1 = std (type_1,0,1);  %类别 1 随机选取的特征标准差
type_mean_2 = mean(type_2);
type_std_2 = std (type_2,0,1);
%% 联合分布及速度分布参数 类别 1
lianhe_s1=zeros(7,4);
```

```
34.
35. s1(:,5) = s1(:,5)/3.6; %加速度转换
36. s2(:,5) = s2(:,5)/3.6; %加速度转换
37.
38. S1(:,5) = S1(:,5)/3.6; %加速度转换
39. S2(:,5) = S2(:,5)/3.6; %加速度转换
40. %实验路况
41. ivA = zeros(7,8);
42. ivA(1,1)=size(find(S1(:,1)>=0&S1(:,1)<=10&S1(:,5)<=-3),1);
43. ivA(1,2)=size(find(S1(:,1)>=0&S1(:,1)<=10&S1(:,5)>-3&S1(:,5)<=-2),1);
44. ivA(1,3)=size(find(S1(:,1)>=0&S1(:,1)<=10&S1(:,5)>-2&S1(:,5)<=-1),1);
45. ivA(1,4)=size(find(S1(:,1)>=0&S1(:,1)<=10&S1(:,5)>-1&S1(:,5)<=0),1);
46. ivA(1,5)=size(find(S1(:,1)>=0&S1(:,1)<=10&S1(:,5)>0&S1(:,5)<=1),1);
47. ivA(1,6)=size(find(S1(:,1)>=0&S1(:,1)<=10&S1(:,5)>1&S1(:,5)<=2),1);
48. ivA(1,7)=size(find(S1(:,1)>=0&S1(:,1)<=10&S1(:,5)>2&S1(:,5)<=3),1);
49. ivA(1,8)=size(find(S1(:,1)>=0&S1(:,1)<=10&S1(:,5)>3&S1(:,5)<=4),1);
50.
51. ivA(2,1)=size(find(S1(:,1)>10&S1(:,1)<=20&S1(:,5)<=-3),1);
52. ivA(2,2)=size(find(S1(:,1)>10&S1(:,1)<=20&S1(:,5)>-3&S1(:,5)<=-2),1);
53. ivA(2,3)=size(find(S1(:,1)>10&S1(:,1)<=20&S1(:,5)>-2&S1(:,5)<=-1),1);
54. ivA(2,4)=size(find(S1(:,1)>10&S1(:,1)<=20&S1(:,5)>-1&S1(:,5)<=0),1);
55. ivA(2,5)=size(find(S1(:,1)>10&S1(:,1)<=20&S1(:,5)>0&S1(:,5)<=1),1);
56. ivA(2,6)=size(find(S1(:,1)>10&S1(:,1)<=20&S1(:,5)>1&S1(:,5)<=2),1);
57. ivA(2,7)=size(find(S1(:,1)>10&S1(:,1)<=20&S1(:,5)>2&S1(:,5)<=3),1);
58. ivA(2,8)=size(find(S1(:,1)>10&S1(:,1)<=20&S1(:,5)>3&S1(:,5)<=4),1);
59.
60. ivA(3,1)=size(find(S1(:,1)>20&S1(:,1)<=30&S1(:,5)<=-3),1);
61. ivA(3,2)=size(find(S1(:,1)>20&S1(:,1)<=30&S1(:,5)>-3&S1(:,5)<=-2),1);
62. ivA(3,3)=size(find(S1(:,1)>20&S1(:,1)<=30&S1(:,5)>-2&S1(:,5)<=-1),1);
63. ivA(3,4)=size(find(S1(:,1)>20&S1(:,1)<=30&S1(:,5)>-1&S1(:,5)<=0),1);
64. ivA(3,5)=size(find(S1(:,1)>20&S1(:,1)<=30&S1(:,5)>0&S1(:,5)<=1),1);
65. ivA(3,6)=size(find(S1(:,1)>20&S1(:,1)<=30&S1(:,5)>1&S1(:,5)<=2),1);
66. ivA(3,7)=size(find(S1(:,1)>20&S1(:,1)<=30&S1(:,5)>2&S1(:,5)<=3),1);
67. ivA(3,8)=size(find(S1(:,1)>20&S1(:,1)<=30&S1(:,5)>3&S1(:,5)<=4),1);
68.
69. ivA(4,1)=size(find(S1(:,1)>30&S1(:,1)<=40&S1(:,5)<=-3),1);
70. ivA(4,2)=size(find(S1(:,1)>30&S1(:,1)<=40&S1(:,5)>-3&S1(:,5)<=-2),1);
71. ivA(4,3)=size(find(S1(:,1)>30&S1(:,1)<=40&S1(:,5)>-2&S1(:,5)<=-1),1);
72. ivA(4,4)=size(find(S1(:,1)>30&S1(:,1)<=40&S1(:,5)>-1&S1(:,5)<=0),1);
73. ivA(4,5)=size(find(S1(:,1)>30&S1(:,1)<=40&S1(:,5)>0&S1(:,5)<=1),1);
74. ivA(4,6)=size(find(S1(:,1)>30&S1(:,1)<=40&S1(:,5)>1&S1(:,5)<=2),1);
75. ivA(4,7)=size(find(S1(:,1)>30&S1(:,1)<=40&S1(:,5)>2&S1(:,5)<=3),1);
76. ivA(4,8)=size(find(S1(:,1)>30&S1(:,1)<=40&S1(:,5)>3&S1(:,5)<=4),1);
77.
78. ivA(5,1)=size(find(S1(:,1)>40&S1(:,1)<=50&S1(:,5)<=-3),1);
79. ivA(5,2)=size(find(S1(:,1)>40&S1(:,1)<=50&S1(:,5)>-3&S1(:,5)<=-2),1);
80. ivA(5,3)=size(find(S1(:,1)>40&S1(:,1)<=50&S1(:,5)>-2&S1(:,5)<=-1),1);
81. ivA(5,4)=size(find(S1(:,1)>40&S1(:,1)<=50&S1(:,5)>-1&S1(:,5)<=0),1);
82. ivA(5,5)=size(find(S1(:,1)>40&S1(:,1)<=50&S1(:,5)>0&S1(:,5)<=1),1);
83. ivA(5,6)=size(find(S1(:,1)>40&S1(:,1)<=50&S1(:,5)>1&S1(:,5)<=2),1);
84. ivA(5,7)=size(find(S1(:,1)>40&S1(:,1)<=50&S1(:,5)>2&S1(:,5)<=3),1);
85. ivA(5,8)=size(find(S1(:,1)>40&S1(:,1)<=50&S1(:,5)>3&S1(:,5)<=4),1);
86.
87. ivA(6,1)=size(find(S1(:,1)>50&S1(:,1)<=60&S1(:,5)<=-3),1);
88. ivA(6,2)=size(find(S1(:,1)>50&S1(:,1)<=60&S1(:,5)>-3&S1(:,5)<=-2),1);
89. ivA(6,3)=size(find(S1(:,1)>50&S1(:,1)<=60&S1(:,5)>-2&S1(:,5)<=-1),1);
90. ivA(6,4)=size(find(S1(:,1)>50&S1(:,1)<=60&S1(:,5)>-1&S1(:,5)<=0),1);
91. ivA(6,5)=size(find(S1(:,1)>50&S1(:,1)<=60&S1(:,5)>0&S1(:,5)<=1),1);
92. ivA(6,6)=size(find(S1(:,1)>50&S1(:,1)<=60&S1(:,5)>1&S1(:,5)<=2),1);
93. ivA(6,7)=size(find(S1(:,1)>50&S1(:,1)<=60&S1(:,5)>2&S1(:,5)<=3),1);
94. ivA(6,8)=size(find(S1(:,1)>50&S1(:,1)<=60&S1(:,5)>3&S1(:,5)<=4),1);
95.
96. ivA(7,1)=size(find(S1(:,1)>60&S1(:,5)<=-3),1);
97. ivA(7,2)=size(find(S1(:,1)>60&S1(:,5)>-3&S1(:,5)<=-2),1);
98. ivA(7,3)=size(find(S1(:,1)>60&S1(:,5)>-2&S1(:,5)<=-1),1);
99. ivA(7,4)=size(find(S1(:,1)>60&S1(:,5)>-1&S1(:,5)<=0),1);
```

```
ivA(7,5)=size(find(S1(:,1)>60&S1(:,5)>0&S1(:,5)<=1),1);
ivA(7,6)=size(find(S1(:,1)>60&S1(:,5)>1&S1(:,5)<=2),1);
ivA(7,7)=size(find(S1(:,1)>60&S1(:,5)>2&S1(:,5)<=3),1);
ivA(7,8)=size(find(S1(:,1)>60&S1(:,5)>3&S1(:,5)<=4),1);

ivA = ivA/size(S1,1);
%代表路况
iva = zeros(7,8);
iva(1,1)=size(find(s1(:,1)>=0&s1(:,1)<=10&s1(:,5)<=-3),1);
iva(1,2)=size(find(s1(:,1)>=0&s1(:,1)<=10&s1(:,5)>-3&s1(:,5)<=-2),1);
iva(1,3)=size(find(s1(:,1)>=0&s1(:,1)<=10&s1(:,5)>-2&s1(:,5)<=-1),1);
iva(1,4)=size(find(s1(:,1)>=0&s1(:,1)<=10&s1(:,5)>-1&s1(:,5)<=0),1);
iva(1,5)=size(find(s1(:,1)>=0&s1(:,1)<=10&s1(:,5)>0&s1(:,5)<=1),1);
iva(1,6)=size(find(s1(:,1)>=0&s1(:,1)<=10&s1(:,5)>1&s1(:,5)<=2),1);
iva(1,7)=size(find(s1(:,1)>=0&s1(:,1)<=10&s1(:,5)>2&s1(:,5)<=3),1);
iva(1,8)=size(find(s1(:,1)>=0&s1(:,1)<=10&s1(:,5)>3&s1(:,5)<=4),1);

iva(2,1)=size(find(s1(:,1)>10&s1(:,1)<=20&s1(:,5)<=-3),1);
iva(2,2)=size(find(s1(:,1)>10&s1(:,1)<=20&s1(:,5)>-3&s1(:,5)<=-2),1);
iva(2,3)=size(find(s1(:,1)>10&s1(:,1)<=20&s1(:,5)>-2&s1(:,5)<=-1),1);
iva(2,4)=size(find(s1(:,1)>10&s1(:,1)<=20&s1(:,5)>-1&s1(:,5)<=0),1);
iva(2,5)=size(find(s1(:,1)>10&s1(:,1)<=20&s1(:,5)>0&s1(:,5)<=1),1);
iva(2,6)=size(find(s1(:,1)>10&s1(:,1)<=20&s1(:,5)>1&s1(:,5)<=2),1);
iva(2,7)=size(find(s1(:,1)>10&s1(:,1)<=20&s1(:,5)>2&s1(:,5)<=3),1);
iva(2,8)=size(find(s1(:,1)>10&s1(:,1)<=20&s1(:,5)>3&s1(:,5)<=4),1);

iva(3,1)=size(find(s1(:,1)>20&s1(:,1)<=30&s1(:,5)<=-3),1);
iva(3,2)=size(find(s1(:,1)>20&s1(:,1)<=30&s1(:,5)>-3&s1(:,5)<=-2),1);
iva(3,3)=size(find(s1(:,1)>20&s1(:,1)<=30&s1(:,5)>-2&s1(:,5)<=-1),1);
iva(3,4)=size(find(s1(:,1)>20&s1(:,1)<=30&s1(:,5)>-1&s1(:,5)<=0),1);
iva(3,5)=size(find(s1(:,1)>20&s1(:,1)<=30&s1(:,5)>0&s1(:,5)<=1),1);
iva(3,6)=size(find(s1(:,1)>20&s1(:,1)<=30&s1(:,5)>1&s1(:,5)<=2),1);
iva(3,7)=size(find(s1(:,1)>20&s1(:,1)<=30&s1(:,5)>2&s1(:,5)<=3),1);
iva(3,8)=size(find(s1(:,1)>20&s1(:,1)<=30&s1(:,5)>3&s1(:,5)<=4),1);

iva(4,1)=size(find(s1(:,1)>30&s1(:,1)<=40&s1(:,5)<=-3),1);
iva(4,2)=size(find(s1(:,1)>30&s1(:,1)<=40&s1(:,5)>-3&s1(:,5)<=-2),1);
iva(4,3)=size(find(s1(:,1)>30&s1(:,1)<=40&s1(:,5)>-2&s1(:,5)<=-1),1);
iva(4,4)=size(find(s1(:,1)>30&s1(:,1)<=40&s1(:,5)>-1&s1(:,5)<=0),1);
iva(4,5)=size(find(s1(:,1)>30&s1(:,1)<=40&s1(:,5)>0&s1(:,5)<=1),1);
iva(4,6)=size(find(s1(:,1)>30&s1(:,1)<=40&s1(:,5)>1&s1(:,5)<=2),1);
iva(4,7)=size(find(s1(:,1)>30&s1(:,1)<=40&s1(:,5)>2&s1(:,5)<=3),1);
iva(4,8)=size(find(s1(:,1)>30&s1(:,1)<=40&s1(:,5)>3&s1(:,5)<=4),1);

iva(5,1)=size(find(s1(:,1)>40&s1(:,1)<=50&s1(:,5)<=-3),1);
iva(5,2)=size(find(s1(:,1)>40&s1(:,1)<=50&s1(:,5)>-3&s1(:,5)<=-2),1);
iva(5,3)=size(find(s1(:,1)>40&s1(:,1)<=50&s1(:,5)>-2&s1(:,5)<=-1),1);
iva(5,4)=size(find(s1(:,1)>40&s1(:,1)<=50&s1(:,5)>-1&s1(:,5)<=0),1);
iva(5,5)=size(find(s1(:,1)>40&s1(:,1)<=50&s1(:,5)>0&s1(:,5)<=1),1);
iva(5,6)=size(find(s1(:,1)>40&s1(:,1)<=50&s1(:,5)>1&s1(:,5)<=2),1);
iva(5,7)=size(find(s1(:,1)>40&s1(:,1)<=50&s1(:,5)>2&s1(:,5)<=3),1);
iva(5,8)=size(find(s1(:,1)>40&s1(:,1)<=50&s1(:,5)>3&s1(:,5)<=4),1);

iva(6,1)=size(find(s1(:,1)>50&s1(:,1)<=60&s1(:,5)<=-3),1);
iva(6,2)=size(find(s1(:,1)>50&s1(:,1)<=60&s1(:,5)>-3&s1(:,5)<=-2),1);
iva(6,3)=size(find(s1(:,1)>50&s1(:,1)<=60&s1(:,5)>-2&s1(:,5)<=-1),1);
iva(6,4)=size(find(s1(:,1)>50&s1(:,1)<=60&s1(:,5)>-1&s1(:,5)<=0),1);
iva(6,5)=size(find(s1(:,1)>50&s1(:,1)<=60&s1(:,5)>0&s1(:,5)<=1),1);
iva(6,6)=size(find(s1(:,1)>50&s1(:,1)<=60&s1(:,5)>1&s1(:,5)<=2),1);
iva(6,7)=size(find(s1(:,1)>50&s1(:,1)<=60&s1(:,5)>2&s1(:,5)<=3),1);
iva(6,8)=size(find(s1(:,1)>50&s1(:,1)<=60&s1(:,5)>3&s1(:,5)<=4),1);

iva(7,1)=size(find(s1(:,1)>60&s1(:,5)<=-3),1);
iva(7,2)=size(find(s1(:,1)>60&s1(:,5)>-3&s1(:,5)<=-2),1);
iva(7,3)=size(find(s1(:,1)>60&s1(:,5)>-2&s1(:,5)<=-1),1);
iva(7,4)=size(find(s1(:,1)>60&s1(:,5)>-1&s1(:,5)<=0),1);
iva(7,5)=size(find(s1(:,1)>60&s1(:,5)>0&s1(:,5)<=1),1);
```

```
iva(7,6)=size(find(s1(:,1)>60&s1(:,5)>1&s1(:,5)<=2),1);
iva(7,7)=size(find(s1(:,1)>60&s1(:,5)>2&s1(:,5)<=3),1);
iva(7,8)=size(find(s1(:,1)>60&s1(:,5)>3&s1(:,5)<=4),1);

iva = iva/size(s1,1);

iv1=find(s1(:,1)>=0&s1(:,1)<=10);
iv2=find(s1(:,1)>10&s1(:,1)<=20);
iv3=find(s1(:,1)>20&s1(:,1)<=30);
iv4=find(s1(:,1)>30&s1(:,1)<=40);
iv5=find(s1(:,1)>40&s1(:,1)<=50);
iv6=find(s1(:,1)>50&s1(:,1)<=60);
iv7=find(s1(:,1)>60);
v1 = size(iv1,1);
v2 = size(iv2,1);
v3 = size(iv3,1);
v4 = size(iv4,1);
v5 = size(iv5,1);
v6 = size(iv6,1);
v7 = size(iv7,1);
sumv = v1+v2+v3+v4+v5+v6+v7;
sv1 = s1(iv1,:);
sv2 = s1(iv2,:);
sv3 = s1(iv3,:);
sv4 = s1(iv4,:);
sv5 = s1(iv5,:);
sv6 = s1(iv6,:);
sv7 = s1(iv7,:);

lianhe_s1(1,1) = mean(sv1(:,1));%平均速度
lianhe_s1(1,2) = max(sv1(:,1));%最大速度
lianhe_s1(1,3) = min(sv1(:,1));%最小速度
lianhe_s1(1,4) = v1/sumv;%比例分布

lianhe_s1(2,1) = mean(sv2(:,1));%平均速度
lianhe_s1(2,2) = max(sv2(:,1));%最大速度
lianhe_s1(2,3) = min(sv2(:,1));%最小速度
lianhe_s1(2,4) = v2/sumv;%比例分布

lianhe_s1(3,1) = mean(sv3(:,1));%平均速度
lianhe_s1(3,2) = max(sv3(:,1));%最大速度
lianhe_s1(3,3) = min(sv3(:,1));%最小速度
lianhe_s1(3,4) = v3/sumv;%比例分布

lianhe_s1(4,1) = mean(sv4(:,1));%平均速度
lianhe_s1(4,2) = max(sv4(:,1));%最大速度
lianhe_s1(4,3) = min(sv4(:,1));%最小速度
lianhe_s1(4,4) = v4/sumv;%比例分布

lianhe_s1(5,1) = mean(sv5(:,1));%平均速度
lianhe_s1(5,2) = max(sv5(:,1));%最大速度
lianhe_s1(5,3) = min(sv5(:,1));%最小速度
lianhe_s1(5,4) = v5/sumv;%比例分布

lianhe_s1(6,1) = mean(sv6(:,1));%平均速度
lianhe_s1(6,2) = max(sv6(:,1));%最大速度
lianhe_s1(6,3) = min(sv6(:,1));%最小速度
lianhe_s1(6,4) = v6/sumv;%比例分布

lianhe_s1(7,1) = mean(sv7(:,1));%平均速度
lianhe_s1(7,2) = max(sv7(:,1));%最大速度
lianhe_s1(7,3) = min(sv7(:,1));%最小速度
lianhe_s1(7,4) = v7/sumv;%比例分布
%% 联合分布   类别 2
```

```
lianhe_s2=zeros(6,4);

%实验路况（拥堵）
ivA2 = zeros(6,8);
ivA2(1,1)=size(find(S2(:,1)>=0&S2(:,1)<=10&S2(:,5)<=-3),1);
ivA2(1,2)=size(find(S2(:,1)>=0&S2(:,1)<=10&S2(:,5)>-3&S2(:,5)<=-2),1);
ivA2(1,3)=size(find(S2(:,1)>=0&S2(:,1)<=10&S2(:,5)>-2&S2(:,5)<=-1),1);
ivA2(1,4)=size(find(S2(:,1)>=0&S2(:,1)<=10&S2(:,5)>-1&S2(:,5)<=0),1);
ivA2(1,5)=size(find(S2(:,1)>=0&S2(:,1)<=10&S2(:,5)>0&S2(:,5)<=1),1);
ivA2(1,6)=size(find(S2(:,1)>=0&S2(:,1)<=10&S2(:,5)>1&S2(:,5)<=2),1);
ivA2(1,7)=size(find(S2(:,1)>=0&S2(:,1)<=10&S2(:,5)>2&S2(:,5)<=3),1);
ivA2(1,8)=size(find(S2(:,1)>=0&S2(:,1)<=10&S2(:,5)>3&S2(:,5)<=4),1);

ivA2(2,1)=size(find(S2(:,1)>10&S2(:,1)<=20&S2(:,5)<=-3),1);
ivA2(2,2)=size(find(S2(:,1)>10&S2(:,1)<=20&S2(:,5)>-3&S2(:,5)<=-2),1);
ivA2(2,3)=size(find(S2(:,1)>10&S2(:,1)<=20&S2(:,5)>-2&S2(:,5)<=-1),1);
ivA2(2,4)=size(find(S2(:,1)>10&S2(:,1)<=20&S2(:,5)>-1&S2(:,5)<=0),1);
ivA2(2,5)=size(find(S2(:,1)>10&S2(:,1)<=20&S2(:,5)>0&S2(:,5)<=1),1);
ivA2(2,6)=size(find(S2(:,1)>10&S2(:,1)<=20&S2(:,5)>1&S2(:,5)<=2),1);
ivA2(2,7)=size(find(S2(:,1)>10&S2(:,1)<=20&S2(:,5)>2&S2(:,5)<=3),1);
ivA2(2,8)=size(find(S2(:,1)>10&S2(:,1)<=20&S2(:,5)>3&S2(:,5)<=4),1);

ivA2(3,1)=size(find(S2(:,1)>20&S2(:,1)<=30&S2(:,5)<=-3),1);
ivA2(3,2)=size(find(S2(:,1)>20&S2(:,1)<=30&S2(:,5)>-3&S2(:,5)<=-2),1);
ivA2(3,3)=size(find(S2(:,1)>20&S2(:,1)<=30&S2(:,5)>-2&S2(:,5)<=-1),1);
ivA2(3,4)=size(find(S2(:,1)>20&S2(:,1)<=30&S2(:,5)>-1&S2(:,5)<=0),1);
ivA2(3,5)=size(find(S2(:,1)>20&S2(:,1)<=30&S2(:,5)>0&S2(:,5)<=1),1);
ivA2(3,6)=size(find(S2(:,1)>20&S2(:,1)<=30&S2(:,5)>1&S2(:,5)<=2),1);
ivA2(3,7)=size(find(S2(:,1)>20&S2(:,1)<=30&S2(:,5)>2&S2(:,5)<=3),1);
ivA2(3,8)=size(find(S2(:,1)>20&S2(:,1)<=30&S2(:,5)>3&S2(:,5)<=4),1);

ivA2(4,1)=size(find(S2(:,1)>30&S2(:,1)<=40&S2(:,5)<=-3),1);
ivA2(4,2)=size(find(S2(:,1)>30&S2(:,1)<=40&S2(:,5)>-3&S2(:,5)<=-2),1);
ivA2(4,3)=size(find(S2(:,1)>30&S2(:,1)<=40&S2(:,5)>-2&S2(:,5)<=-1),1);
ivA2(4,4)=size(find(S2(:,1)>30&S2(:,1)<=40&S2(:,5)>-1&S2(:,5)<=0),1);
ivA2(4,5)=size(find(S2(:,1)>30&S2(:,1)<=40&S2(:,5)>0&S2(:,5)<=1),1);
ivA2(4,6)=size(find(S2(:,1)>30&S2(:,1)<=40&S2(:,5)>1&S2(:,5)<=2),1);
ivA2(4,7)=size(find(S2(:,1)>30&S2(:,1)<=40&S2(:,5)>2&S2(:,5)<=3),1);
ivA2(4,8)=size(find(S2(:,1)>30&S2(:,1)<=40&S2(:,5)>3&S2(:,5)<=4),1);

ivA2(5,1)=size(find(S2(:,1)>40&S2(:,1)<=50&S2(:,5)<=-3),1);
ivA2(5,2)=size(find(S2(:,1)>40&S2(:,1)<=50&S2(:,5)>-3&S2(:,5)<=-2),1);
ivA2(5,3)=size(find(S2(:,1)>40&S2(:,1)<=50&S2(:,5)>-2&S2(:,5)<=-1),1);
ivA2(5,4)=size(find(S2(:,1)>40&S2(:,1)<=50&S2(:,5)>-1&S2(:,5)<=0),1);
ivA2(5,5)=size(find(S2(:,1)>40&S2(:,1)<=50&S2(:,5)>0&S2(:,5)<=1),1);
ivA2(5,6)=size(find(S2(:,1)>40&S2(:,1)<=50&S2(:,5)>1&S2(:,5)<=2),1);
ivA2(5,7)=size(find(S2(:,1)>40&S2(:,1)<=50&S2(:,5)>2&S2(:,5)<=3),1);
ivA2(5,8)=size(find(S2(:,1)>40&S2(:,1)<=50&S2(:,5)>3&S2(:,5)<=4),1);

ivA2(6,1)=size(find(S2(:,1)>50&S2(:,5)<=-3),1);
ivA2(6,2)=size(find(S2(:,1)>50&S2(:,5)>-3&S2(:,5)<=-2),1);
ivA2(6,3)=size(find(S2(:,1)>50&S2(:,5)>-2&S2(:,5)<=-1),1);
ivA2(6,4)=size(find(S2(:,1)>50&S2(:,5)>-1&S2(:,5)<=0),1);
ivA2(6,5)=size(find(S2(:,1)>50&S2(:,5)>0&S2(:,5)<=1),1);
ivA2(6,6)=size(find(S2(:,1)>50&S2(:,5)>1&S2(:,5)<=2),1);
ivA2(6,7)=size(find(S2(:,1)>50&S2(:,5)>2&S2(:,5)<=3),1);
ivA2(6,8)=size(find(S2(:,1)>50&S2(:,5)>3&S2(:,5)<=4),1);

ivA2 = ivA2/size(S2,1);
%代表路况（拥堵）
iva2 = zeros(6,8);
iva2(1,1)=size(find(s2(:,1)>=0&s2(:,1)<=10&s2(:,5)<=-3),1);
iva2(1,2)=size(find(s2(:,1)>=0&s2(:,1)<=10&s2(:,5)>-3&s2(:,5)<=-2),1);
iva2(1,3)=size(find(s2(:,1)>=0&s2(:,1)<=10&s2(:,5)>-2&s2(:,5)<=-1),1);
iva2(1,4)=size(find(s2(:,1)>=0&s2(:,1)<=10&s2(:,5)>-1&s2(:,5)<=0),1);
iva2(1,5)=size(find(s2(:,1)>=0&s2(:,1)<=10&s2(:,5)>0&s2(:,5)<=1),1);
iva2(1,6)=size(find(s2(:,1)>=0&s2(:,1)<=10&s2(:,5)>1&s2(:,5)<=2),1);
```

```
iva2(1,7)=size(find(s2(:,1)>=0&s2(:,1)<=10&s2(:,5)>2&s2(:,5)<=3),1);
iva2(1,8)=size(find(s2(:,1)>=0&s2(:,1)<=10&s2(:,5)>3&s2(:,5)<=4),1);

iva2(2,1)=size(find(s2(:,1)>10&s2(:,1)<=20&s2(:,5)<=-3),1);
iva2(2,2)=size(find(s2(:,1)>10&s2(:,1)<=20&s2(:,5)>-3&s2(:,5)<=-2),1);
iva2(2,3)=size(find(s2(:,1)>10&s2(:,1)<=20&s2(:,5)>-2&s2(:,5)<=-1),1);
iva2(2,4)=size(find(s2(:,1)>10&s2(:,1)<=20&s2(:,5)>-1&s2(:,5)<=0),1);
iva2(2,5)=size(find(s2(:,1)>10&s2(:,1)<=20&s2(:,5)>0&s2(:,5)<=1),1);
iva2(2,6)=size(find(s2(:,1)>10&s2(:,1)<=20&s2(:,5)>1&s2(:,5)<=2),1);
iva2(2,7)=size(find(s2(:,1)>10&s2(:,1)<=20&s2(:,5)>2&s2(:,5)<=3),1);
iva2(2,8)=size(find(s2(:,1)>10&s2(:,1)<=20&s2(:,5)>3&s2(:,5)<=4),1);

iva2(3,1)=size(find(s2(:,1)>20&s2(:,1)<=30&s2(:,5)<=-3),1);
iva2(3,2)=size(find(s2(:,1)>20&s2(:,1)<=30&s2(:,5)>-3&s2(:,5)<=-2),1);
iva2(3,3)=size(find(s2(:,1)>20&s2(:,1)<=30&s2(:,5)>-2&s2(:,5)<=-1),1);
iva2(3,4)=size(find(s2(:,1)>20&s2(:,1)<=30&s2(:,5)>-1&s2(:,5)<=0),1);
iva2(3,5)=size(find(s2(:,1)>20&s2(:,1)<=30&s2(:,5)>0&s2(:,5)<=1),1);
iva2(3,6)=size(find(s2(:,1)>20&s2(:,1)<=30&s2(:,5)>1&s2(:,5)<=2),1);
iva2(3,7)=size(find(s2(:,1)>20&s2(:,1)<=30&s2(:,5)>2&s2(:,5)<=3),1);
iva2(3,8)=size(find(s2(:,1)>20&s2(:,1)<=30&s2(:,5)>3&s2(:,5)<=4),1);

iva2(4,1)=size(find(s2(:,1)>30&s2(:,1)<=40&s2(:,5)<=-3),1);
iva2(4,2)=size(find(s2(:,1)>30&s2(:,1)<=40&s2(:,5)>-3&s2(:,5)<=-2),1);
iva2(4,3)=size(find(s2(:,1)>30&s2(:,1)<=40&s2(:,5)>-2&s2(:,5)<=-1),1);
iva2(4,4)=size(find(s2(:,1)>30&s2(:,1)<=40&s2(:,5)>-1&s2(:,5)<=0),1);
iva2(4,5)=size(find(s2(:,1)>30&s2(:,1)<=40&s2(:,5)>0&s2(:,5)<=1),1);
iva2(4,6)=size(find(s2(:,1)>30&s2(:,1)<=40&s2(:,5)>1&s2(:,5)<=2),1);
iva2(4,7)=size(find(s2(:,1)>30&s2(:,1)<=40&s2(:,5)>2&s2(:,5)<=3),1);
iva2(4,8)=size(find(s2(:,1)>30&s2(:,1)<=40&s2(:,5)>3&s2(:,5)<=4),1);

iva2(5,1)=size(find(s2(:,1)>40&s2(:,1)<=50&s2(:,5)<=-3),1);
iva2(5,2)=size(find(s2(:,1)>40&s2(:,1)<=50&s2(:,5)>-3&s2(:,5)<=-2),1);
iva2(5,3)=size(find(s2(:,1)>40&s2(:,1)<=50&s2(:,5)>-2&s2(:,5)<=-1),1);
iva2(5,4)=size(find(s2(:,1)>40&s2(:,1)<=50&s2(:,5)>-1&s2(:,5)<=0),1);
iva2(5,5)=size(find(s2(:,1)>40&s2(:,1)<=50&s2(:,5)>0&s2(:,5)<=1),1);
iva2(5,6)=size(find(s2(:,1)>40&s2(:,1)<=50&s2(:,5)>1&s2(:,5)<=2),1);
iva2(5,7)=size(find(s2(:,1)>40&s2(:,1)<=50&s2(:,5)>2&s2(:,5)<=3),1);
iva2(5,8)=size(find(s2(:,1)>40&s2(:,1)<=50&s2(:,5)>3&s2(:,5)<=4),1);

iva2(6,1)=size(find(s2(:,1)>50&s2(:,5)<=-3),1);
iva2(6,2)=size(find(s2(:,1)>50&s2(:,5)>-3&s2(:,5)<=-2),1);
iva2(6,3)=size(find(s2(:,1)>50&s2(:,5)>-2&s2(:,5)<=-1),1);
iva2(6,4)=size(find(s2(:,1)>50&s2(:,5)>-1&s2(:,5)<=0),1);
iva2(6,5)=size(find(s2(:,1)>50&s2(:,5)>0&s2(:,5)<=1),1);
iva2(6,6)=size(find(s2(:,1)>50&s2(:,5)>1&s2(:,5)<=2),1);
iva2(6,7)=size(find(s2(:,1)>50&s2(:,5)>2&s2(:,5)<=3),1);
iva2(6,8)=size(find(s2(:,1)>50&s2(:,5)>3&s2(:,5)<=4),1);

iva2 = iva2/size(s2,1);

iv1=find(s2(:,1)>=0&s2(:,1)<=10);
iv2=find(s2(:,1)>10&s2(:,1)<=20);
iv3=find(s2(:,1)>20&s2(:,1)<=30);
iv4=find(s2(:,1)>30&s2(:,1)<=40);
iv5=find(s2(:,1)>40&s2(:,1)<=50);
iv6=find(s2(:,1)>50);
v1 = size(iv1,1);
v2 = size(iv2,1);
v3 = size(iv3,1);
v4 = size(iv4,1);
v5 = size(iv5,1);
v6 = size(iv6,1);
sumv = v1+v2+v3+v4+v5+v6;
sv1 = s2(iv1,:);
sv2 = s2(iv2,:);
sv3 = s2(iv3,:);
sv4 = s2(iv4,:);
```

```
sv5 = s2(iv5,:);
sv6 = s2(iv6,:);

lianhe_s2(1,1) = mean(sv1(:,1));%平均速度
lianhe_s2(1,2) = max(sv1(:,1));%最大速度
lianhe_s2(1,3) = min(sv1(:,1));%最小速度
lianhe_s2(1,4) = v1/sumv;%比例分布

lianhe_s2(2,1) = mean(sv2(:,1));%平均速度
lianhe_s2(2,2) = max(sv2(:,1));%最大速度
lianhe_s2(2,3) = min(sv2(:,1));%最小速度
lianhe_s2(2,4) = v2/sumv;%比例分布

lianhe_s2(3,1) = mean(sv3(:,1));%平均速度
lianhe_s2(3,2) = max(sv3(:,1));%最大速度
lianhe_s2(3,3) = min(sv3(:,1));%最小速度
lianhe_s2(3,4) = v3/sumv;%比例分布

lianhe_s2(4,1) = mean(sv4(:,1));%平均速度
lianhe_s2(4,2) = max(sv4(:,1));%最大速度
lianhe_s2(4,3) = min(sv4(:,1));%最小速度
lianhe_s2(4,4) = v4/sumv;%比例分布

lianhe_s2(5,1) = mean(sv5(:,1));%平均速度
lianhe_s2(5,2) = max(sv5(:,1));%最大速度
lianhe_s2(5,3) = min(sv5(:,1));%最小速度
lianhe_s2(5,4) = v5/sumv;%比例分布

if isempty(sv6)
    lianhe_s2(6,:)=0;
else
    lianhe_s2(6,1) = mean(sv6(:,1));%平均速度
    lianhe_s2(6,2) = max(sv6(:,1));%最大速度
    lianhe_s2(6,3) = min(sv6(:,1));%最小速度
    lianhe_s2(6,4) = v6/sumv;%比例分布
end
```