**摘要**

最后再写（约2页）

[问题背景与问题重述 1](#_Toc1225478369)

[1.1 问题背景 1](#_Toc99289406)

[1.2 问题重述 2](#_Toc162252923)

[基本假设与符号说明 2](#_Toc1828128818)

[问题一的分析与建模 2](#_Toc1312506497)

[问题二的分析建模 2](#_Toc344673095)

[问题三的分析建模 3](#_Toc1157311706)

[3.1 特征参数的选取 3](#_Toc1178451863)

[3.2 运动学片段的参数计算 3](#_Toc2146268807)

[3.3 聚类分析 3](#_Toc1057020590)

[3.4 工况的合成 4](#_Toc1360328146)

[模型评估 4](#_Toc924243860)

[4.1 指标计算 4](#_Toc1017336269)

[4.2 合理性分析 4](#_Toc105875669)

[参考文献 4](#_Toc1335909167)

[附录 4](#_Toc683840384)

# 1 问题背景与问题重述

## 1.1 问题背景

汽车行驶工况又称为车辆测试循环，是描述典型车辆行驶的速度-时间曲线。它可以用于确定车辆污染物排放量及燃油消耗量，体现汽车在道路行驶中的燃料经济性、排放等运动学特征，是汽车工业一项核心技术。目前，我国乘用车行驶工况采用的是欧洲的NEDC，但由于我国的交通状况、行驶状况与道路状况等因素都与欧盟国家有很大不同，因此NEDC工况与我国城市汽车的实际行驶数据偏差较大。另一方面，我国地域辽阔，不同城市之间的交通状况、发展程度与气候条件等都存在着差异，这也导致各个城市的汽车行驶工况也不尽相同。因此，基于各个城市自身的汽车行驶数据制定反映我国实际道路行驶状况的汽车行驶工况，显得越来越重要。

## 1.2 问题重述

在上述背景下，题目给出了某城市同一辆测试汽车在不同时间段采集的实际行驶数据，要求利用该数据解决以下几个问题：

问题一：由于数据采集过程中存在着不可避免的实际问题，原始数据中会包含一些不良值。要求通过合理的方法对异常数据进行预处理，并给出处理后的数据记录数。

问题二：我们将车辆从一个怠速开始到下一个怠速开始之间的运动定义为运动学片段。题目要求设计合理的方法，将处理后的数据划分为多个运动学片段，并给出每个数据文件得到的运动学片段数。

问题三：基于上面处理好的数据，采用科学有效的方法构建一条能体现采集数据源行驶特征的汽车行驶工况曲线。同时使用合理的评估体系对行驶工况曲线进行评估，说明其合理性。

# 2 基本假设与符号说明

## 2.1 基本假设

1. 假设普通轿车在一般情况下最大的加速度为4m/s2，紧急刹车时的最大减速度为8m/s2。
2. 假设当汽车车速小于10km/h时，就认为是怠速状态。当怠速状态时间超过180秒就被认为是异常情况，怠速时间最长可按180秒处理。
3. 我们定义汽车加速度大于0.1m/s2的连续过程为加速阶段，汽车加速度小于-0.1m/s2的连续过程为减速阶段。
4. 假设采样频率为1Hz。

2.2 符号说明

|  |  |
| --- | --- |
| 符号 | 说明 |
| Vm | 平均速度。即一段时间周期内，汽车速度的算术平均值 |
| Vmr | 平均行驶速度。即汽车在行驶状态下汽车速度的算术平均值，即不包含汽车怠速状态。 |
| Vsd | 速度标准差。即一段时间周期内，汽车速度的标准差，包括怠速状态。 |
| Ti | 怠速时间比。即一段时间周期内，怠速状态的累计时间长度占该时间周期总时间长度的百分比。 |
| Ta | 加速时间比。即一段时间周期内，处在加速状态的累计时间长度占该时间周期总时间长度的百分比。 |
| Td | 减速时间比。即一段时间周期内，处在减速状态的累计时间长度占该时间周期总时间长度的百分比。 |
| Am | 平均加速度。即汽车在加速状态下各单位时间（秒）加速度的算术平均值。 |
| Ad | 平均减速度。即汽车在减速状态下各单位时间（秒）减速度的算术平均值。 |
| Asd | 加速度标准差。即一段时间周期内，处在加速状态的汽车加速度的标准差。 |

# 3 问题一的分析与建模

题目给出了某城市同一辆测试汽车在不同时间段采集的实际行驶数据，覆盖了工作日与节假日、高峰期与非高峰期等多时段的数据，保证了测试数据的可靠性。但由于多种主客观因素影响，原始测试数据中可能包含多种不良数据。问题一就是要求通过多种合理的处理方法对这些异常数据进行识别与预处理。

原始的测试数据包含3个数据文件，其中数据文件1的原始数据记录数为**185725**条，数据文件2的原始数据记录数为**145825**条，数据文件3的原始数据记录数为**164914**条。

## 3.1 加减速度异常的数据处理

从上文的基本假设中我们可知，在一般情况下普通轿车从0加速至100km/h的时间应大于7秒，即最大加速度约为4m/s2，而轿车紧急刹车的最大减速度约为8m/s2。在实际道路行驶过程中可能会出现车辆急加速与急刹车的情况，这种情况下往往会导致车辆加速度过大。而超出最大加减速度阈值的不良数据显然会对我们构建行驶工况曲线造成影响，因此需要对这些不良数据进行处理。

我们通过对题目给出的原始采集数据的各条记录进行加速度计算后，发现数据集中也存在这种加减速度异常的数据值。数据文件1的异常数据的记录数为**28**条，数据文件2的异常数据的记录数为**401**条，数据文件3中**无**加减速度异常的记录。图1展示了数据文件2中的一段异常记录，从图中我们可以明显看出测量到的GPS速度变化异常，几乎每秒都在进行急加速或急减速，加减速度均超出了正常范围。使用这样的异常数据来构建行驶工况曲线无疑会出现较大的偏差。

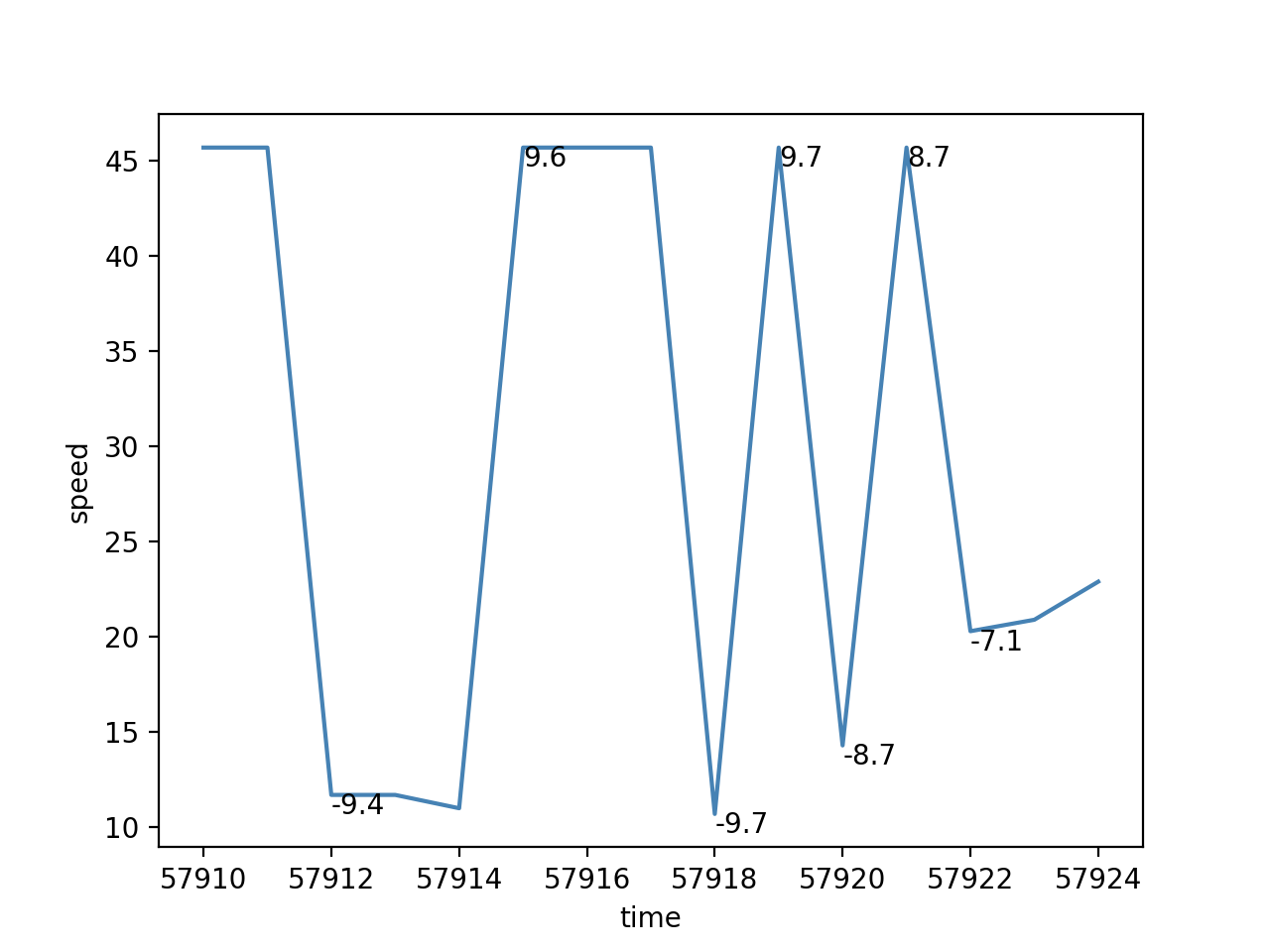


图1 加减速度异常的速度曲线图

对于这类加减速度异常的原始数据，我们的处理方法是对异常数据记录进行剔除，并使用**线性插值法**对原本的异常数据点进行修正，从而消除离群点数据对于构建行驶工况曲线带来的不利影响。图2展示了修正后的数据记录，可以看出通过线性插值法修正后加减速度均已在正常范围内，数据异常问题得以解决。

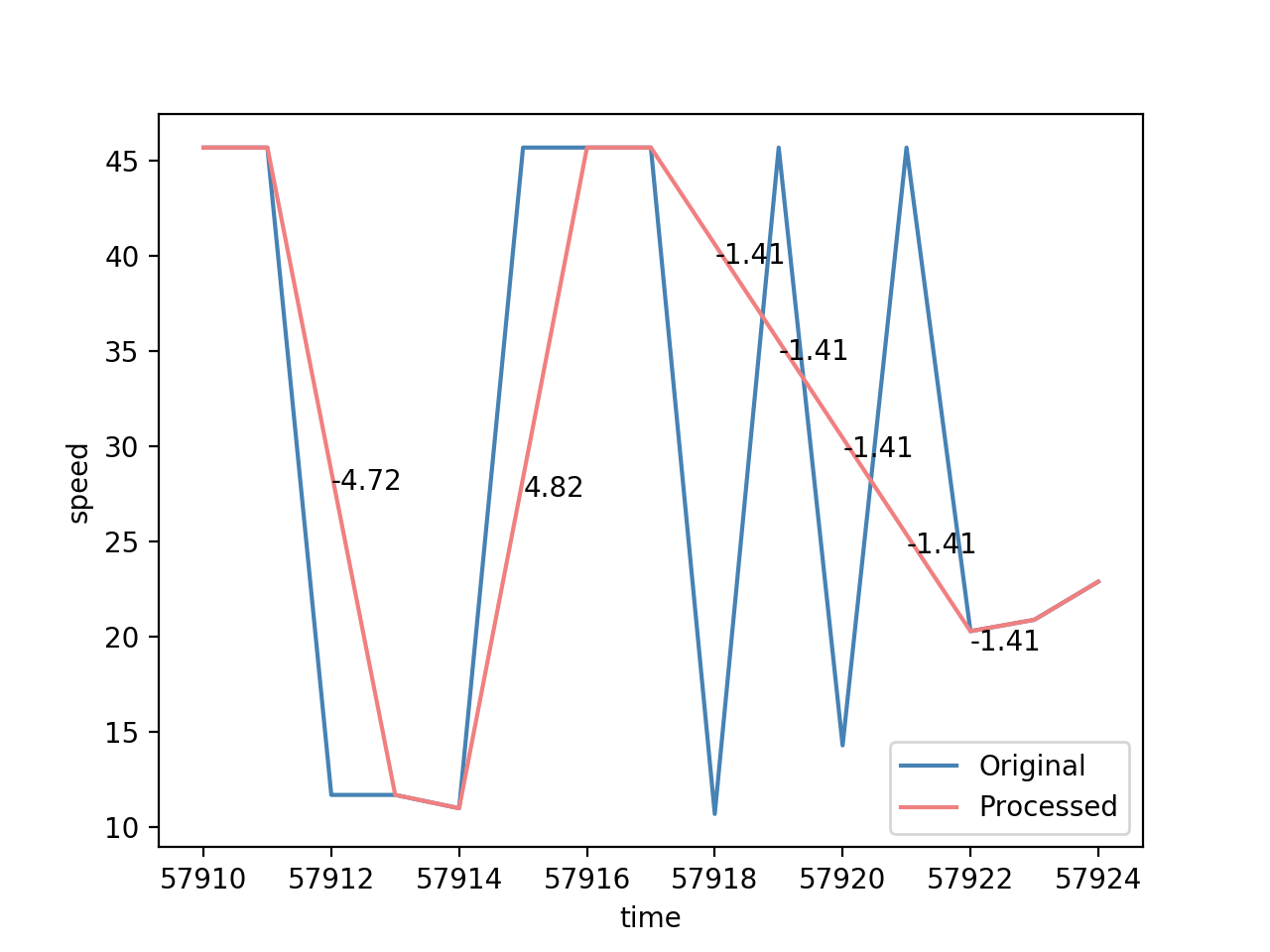


图2 修正后的速度曲线图

## 3.2 时间不连续的缺失数据处理

汽车行驶数据通过车载的终端采集设备直接采集，采集过程中可能会出现采集设备异常等问题，导致数据丢失。此外，现实路况也可能给数据采集带来一定的影响，例如在汽车过隧道时，会导致GPS信号丢失，无法进行正常的数据采集。这些问题造成了提供的数据中时间不连续，部分数据丢失的问题。

在对缺失数据段进行观察后，我们发现不同的数据段数据缺失的严重程度不同。图3展示了两种不同的缺失程度，图3 a中仅缺失了3秒的数据，而图3 b中缺失了超过60秒的数据。结合原始数据质量与现实生活的考量，我们设置允许最大不连续时间段的阈值为**30**秒。连续缺失数据未超过阈值的数据段（如图x a仅缺失3秒的数据段）我们通过**线性插值法**对缺失数据进行填充修正处理，而超过阈值的数据我们则进行删除。在构建行驶工况曲线的过程中，我们选择基于运动学片段进行构建。考虑到缺失数据过多会对整个运动学片段带来影响，因此对于那些超过阈值需要被删除的异常数据，我们选择删除该段缺失数据所在的**整个运动学片段**的数据记录。

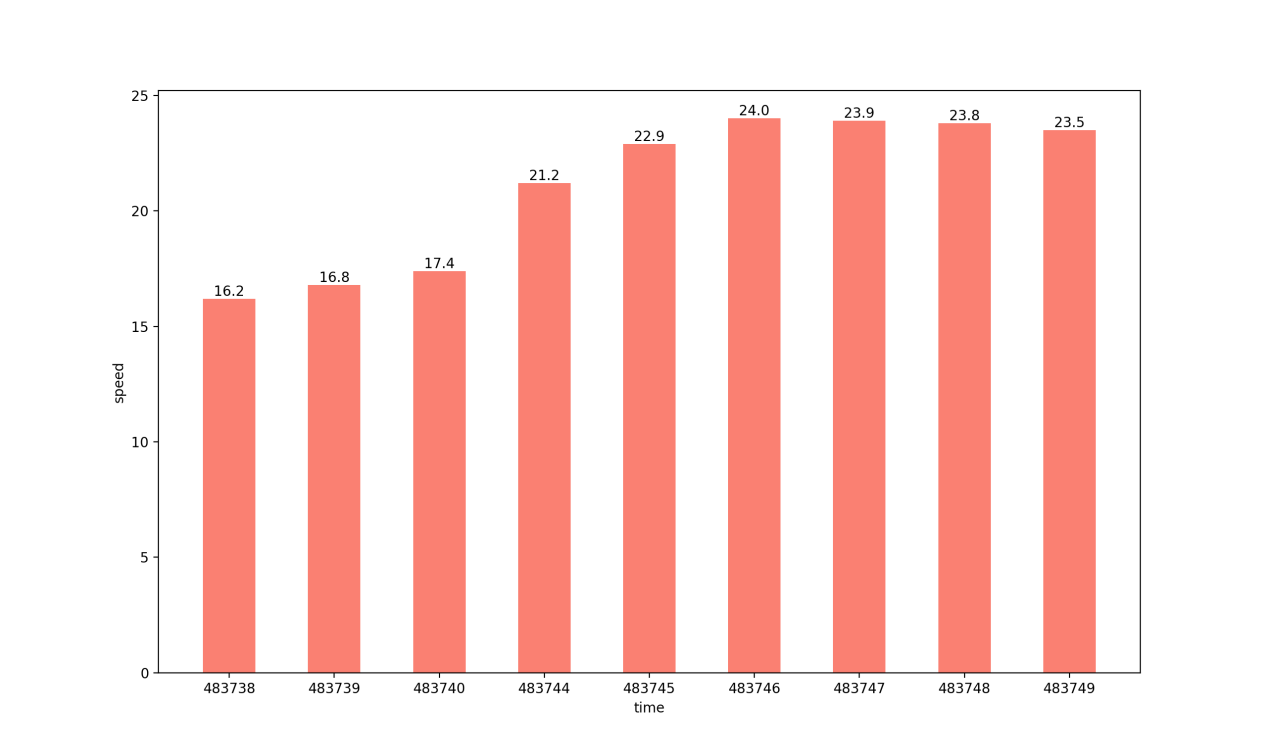


图3 a 缺失较少的异常数据段

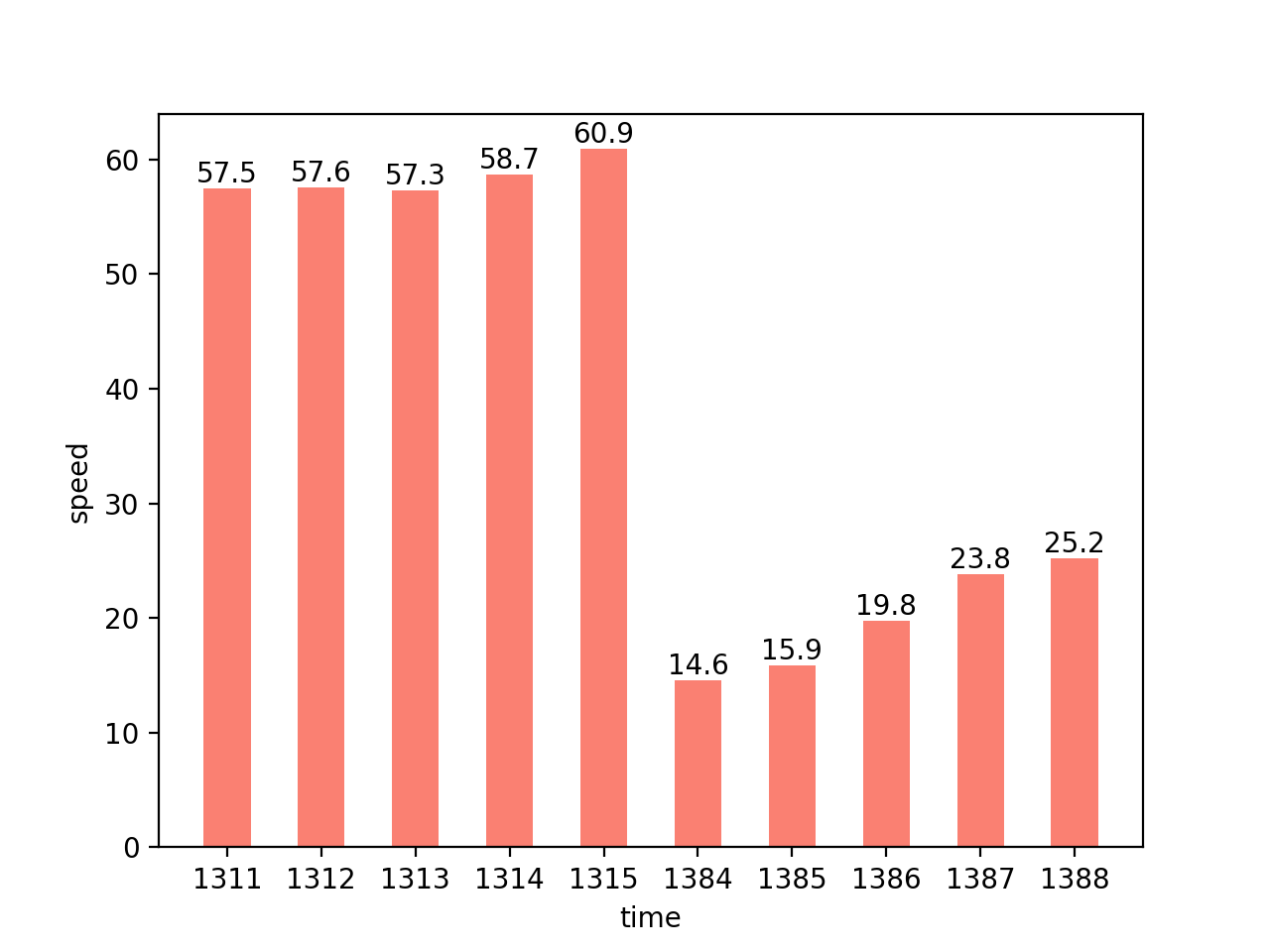


图3 b 缺失较多的异常数据段

经过对缺失数据进行处理后，数据文件1剩余总记录数为**136810**条。数据文件2剩余总记录数为**110395**条。数据文件3剩余总记录数为**108808**条。

## 3.3 怠速异常的数据处理

由于汽车行驶工况受交通状况、道路状况、行驶状况等多种主客观因素影响，在数据采集的过程中往往会出现多种类型的行驶数据。我们的原始数据覆盖了工作日与节假日、高峰期与非高峰期等多时段的数据，这其中也包含了几种怠速异常的不良数据。

一种情况如图4所示，数据中存在着一段长时间内行驶速度均为0km/h的怠速异常情况，在现实中长期停车熄火等人，或者停车熄火了但仍在采集数据等都会造成这种异常情况的发生。从上文的基本假设中我们可知，怠速时间超过180秒被认为是异常情况，怠速时间最长可按180秒进行处理。因此对于这类异常我们的处理方法是识别出整段怠速片段，删除多余的异常数据，只保留180秒（即最长怠速时间）内的数据。

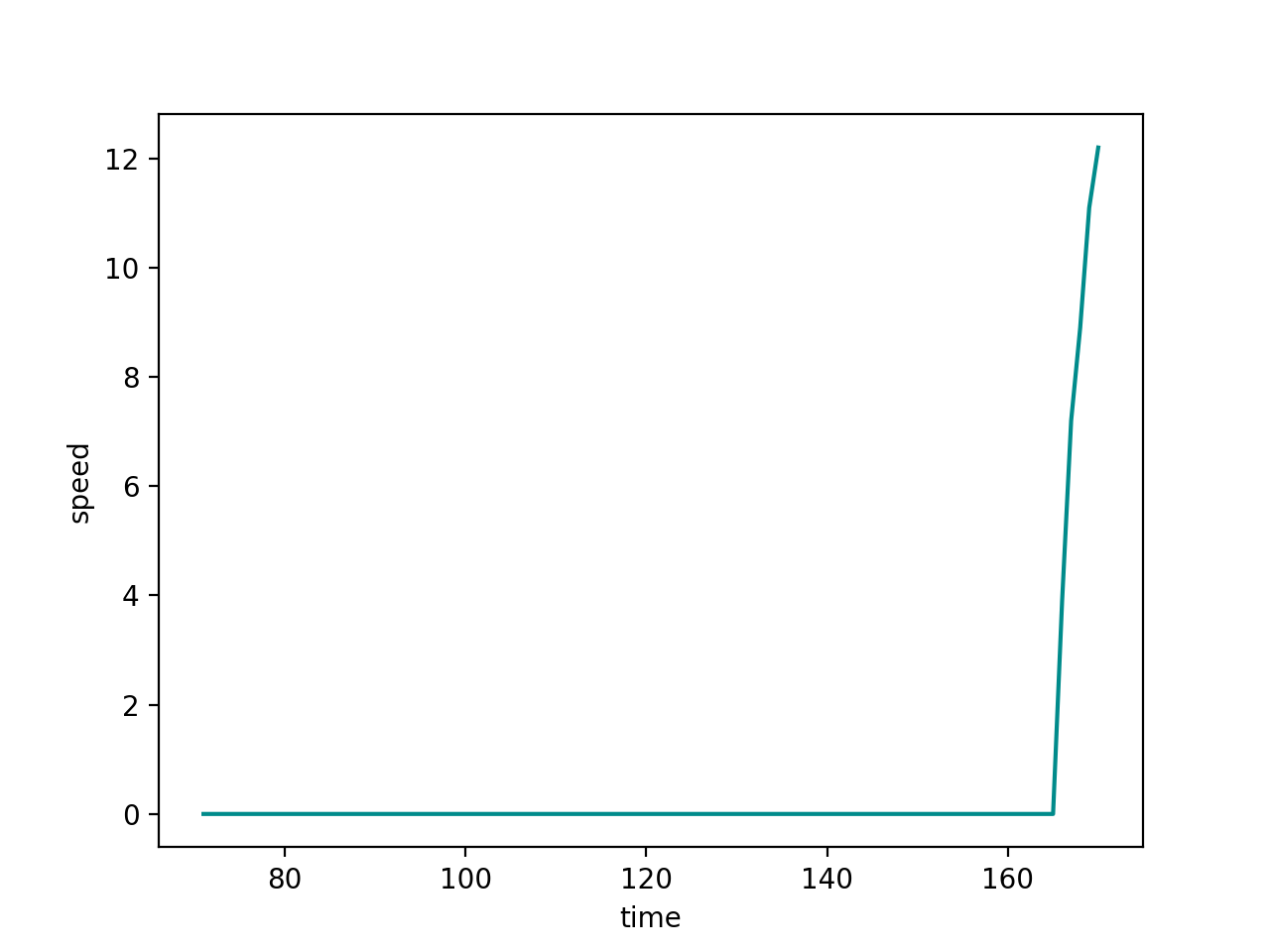


图4 长期停车的异常数据

另外一种异常情况与实际更加相关。在现实生活中，高峰期或节假日时期的路况一般较为拥堵，极易出现长时间堵车的情况。这时汽车一般处于断断续续行驶且只能低速行驶的情况。这对应于数据中存在着一段长时间行驶速度不为0但均小于10km/h的异常情况，如图5所示。对于这类不良数据，我们的处理方法是将这类数据按照怠速情况进行处理。若连续低速的数据记录超过60秒（说明是**长时间**堵车）则将其速度记录全部置为0km/h，认为这是怠速状况，并按照上述怠速异常的方法进行处理。若连续低速的数据记录未超过60秒，则认为是正常情况，保留原始的速度记录。

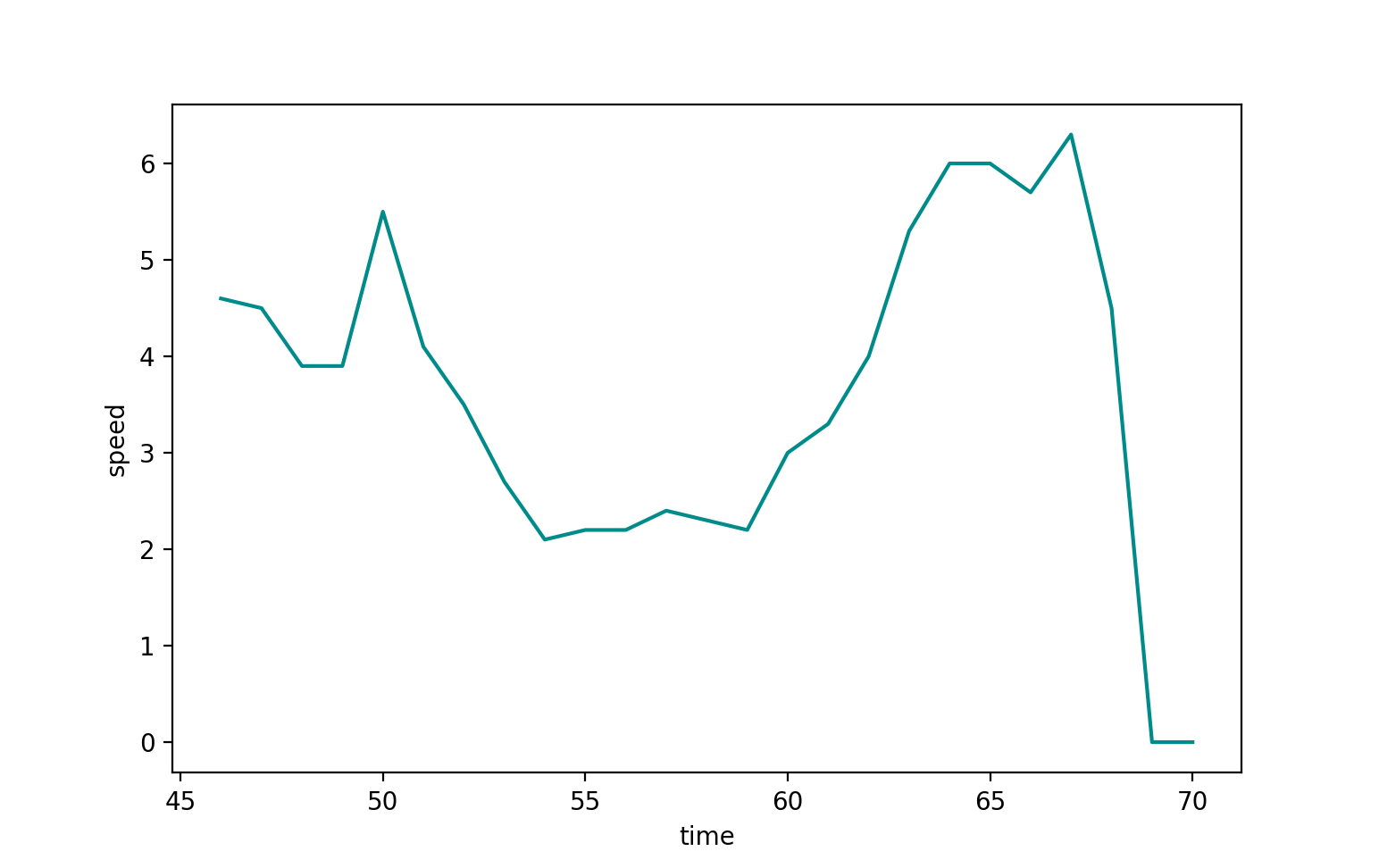


图5 长时间低速行驶的异常数据

经过上面3部分全部预处理之后，数据文件1剔除的异常记录数为54551条，总记录数为**131174**条。数据文件2剔除的异常记录数为37748条，总记录数为**108077**条。数据文件3剔除的异常记录数为58177条，总记录数为**106737**条。

# 4 问题二的分析建模

在现有的构建行驶工况曲线的方法中，基于运动学片段进行构建是最常用的方法之一。在车辆行驶的过程中，会出现频繁的启动、加速、减速等过程，运动学片段就是指车辆从一个怠速状态开始至下一个怠速状态开始之间的运动过程。如下图所示便是一个运动学片段的定义说明。



图6 运动学片段的定义

在经过问题一的预处理后我们得到了一份处理后的数据源，图7展示了处理后的部分数据，图中可以清晰地看出在这段数据中存在着多个运动学片段。我们基于上述运动学片段定义对数据进行划分。自一个怠速状态开始至下一个怠速状态开始被认为是一个运动学片段，图中a-b段即是一个运动学片段。

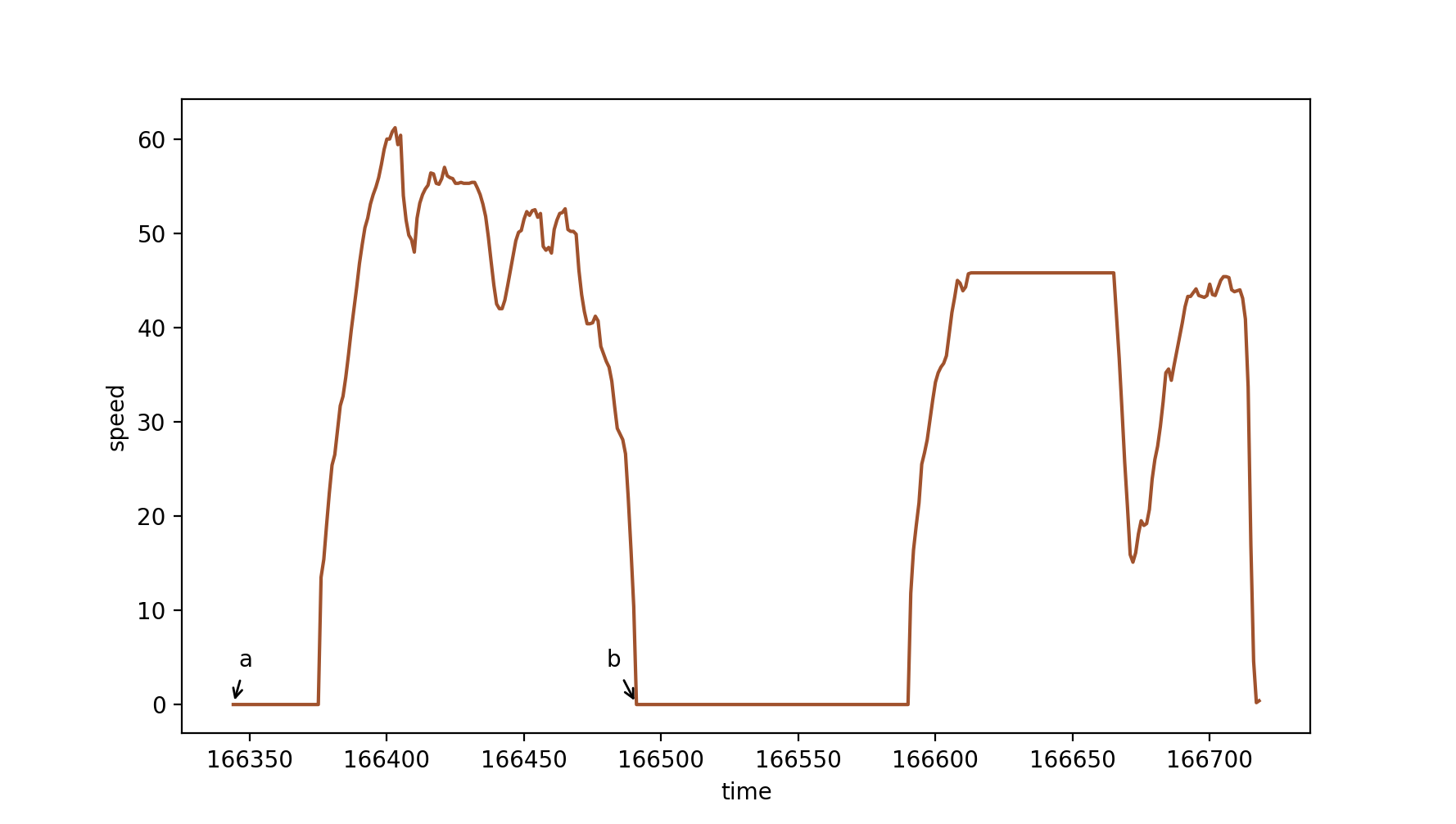


图7 运动学片段示例

按上述方法划分后，最终数据文件1我们得到了**1181**个运动学片段，数据文件2我们得到了**725**个运动学片段，数据文件3我们得到了**649**个运动学片段。

# 5 问题三的分析建模

## 5.1 特征参数的选取

对于每一个运动学片段，我们从其中提取9种运动学特征参数，并在之后对其进行计算和筛选。详细特征参数见表1。

表1 特征参数

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **特征参数** | **编号** | **特征参数** |
| **1** | **平均速度** | 6 | **加速时间比** |
| **2** | **平均行驶速度** | 7 | **减速时间比** |
| **3** | **平均加速度** | 8 | **速度标准差** |
| **4** | **平均减速度** | 9 | **加速度标准差** |
| **5** | **怠速时间比** |  |  |

其中，平均速度为在一个运动学片段内，行驶的距离与时间的比值，而平均行驶速度是不包括怠速阶段在内的平均速度，显然平均速度小于等于平均行驶速度。平均加速度、平均减速度为该运动学片段内正负加速度各自的平均值。怠速时间比，加速时间比，减速时间比则是各行驶状态在整个运动学片段内所占比例，三者之和小于等于1。最后，速度标准差，加速度标准差则是用来衡量速度、加速度的离散程度。

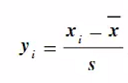
## 5.2 归一化处理与主成分分析

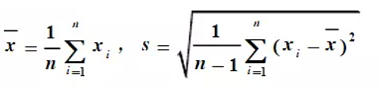
在对每一个运动学片段提取了特征参数后，我们得到了一个2555 \* 9的特征参数矩阵。

为了避免使用所有特征参数作为分类指标，导致信息重叠，我们对这2555 \* 9 的特征参数矩阵进行了主成分分析。

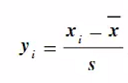
在进行主成分分析之前，为了消除特征参数间不同量纲的影响，我们对特征参数进行了归一化处理。在进行归一化的算法选择上，我们使用了两种归一化算法。

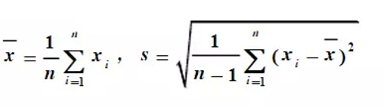
我们选取的第一种归一化方法是Min-Max Normalization，将各个特征参数落到了[0,1]区间内，得到的新的特征值序列均值为0，方差为1，且无量纲。具体公式如下图所示：





我们选择的第二种归一化方法是Zero-mean Normalization，即标准差标准化，基于原特征序列的均值和标准差来进行序列的归一化，具体公式如下：



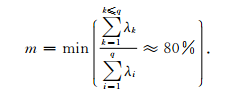


我们对每一张的结果都进行了主成分分析。

第一种方法的分析结果如下表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **贡献率** | **编号** | **贡献率** |
| **1** | **0.55971444** | **6** | **0.01751926** |
| **2** | **0.1947529** | **7** | **0.01433069** |
| **3** | **0.09489261** | **8** | **0.00540254** |
| **4** | **0.07485658** | **9** | **0.00159128** |
| **5** | **0.0369397** |  |  |

鉴于第二种方法区分度较低，我们最终选择了Min-max标准化后的结果。在其累计贡献率约为80%时应保留的主成分为：



基于此公式，本次保留前三个主成分进行聚类。

## 5.3 聚类分析

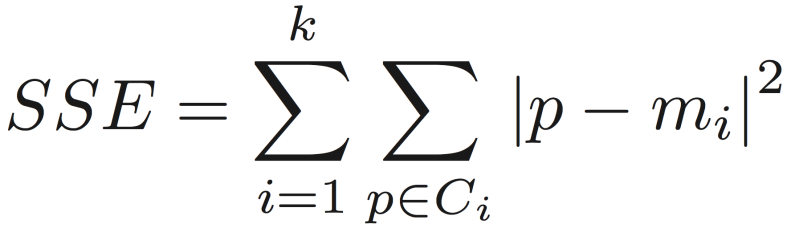
### 5.3.1 K均值聚类法

我们使用K均值聚类算法对之前所获取到的2555个运动学片段进行聚类。

k均值聚类是最著名的划分聚类算法，由于简洁和效率使得他成为所有聚类算法中最广泛使用的。给定一个数据点集合和需要的聚类数目k， k均值算法根据某个距离函数反复把数据分入k个聚类中。它是一种迭代求解的聚类分析算法，其步骤是随机选取K个对象作为初始的聚类中心，然后计算每个对象与各个种子聚类中心之间的距离，把每个对象分配给距离它最近的聚类中心。聚类中心以及分配给它们的对象就代表一个聚类。每分配一个样本，聚类的聚类中心会根据聚类中现有的对象被重新计算。这个过程将不断重复直到满足某个终止条件。终止条件可以是没有（或最小数目）对象被重新分配给不同的聚类，没有（或最小数目）聚类中心再发生变化，误差平方和局部最小。

由于K均值聚类算法需要指定K的值，因此我们首先需要决策出合理的K值。在决策K值时我们所采取的方法是“手肘法”。

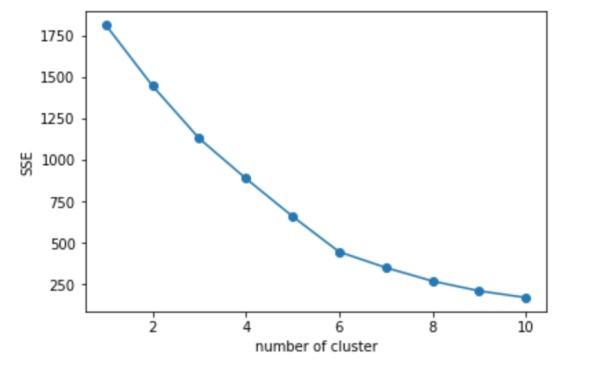
手肘法的核心思想是：随着聚类数k的增大，样本划分会更加精细，每个簇的聚合程度会逐渐提高，那么误差平方和SSE（sum of the squared errors，误差平方和）自然会逐渐变小。并且，当k小于真实聚类数时，由于k的增大会大幅增加每个簇的聚合程度，故SSE的下降幅度会很大，而当k到达真实聚类数时，再增加k所得到的聚合程度回报会迅速变小，所以SSE的下降幅度会骤减，然后随着k值的继续增大而趋于平缓，也就是说SSE和k的关系图是一个手肘的形状，而这个肘部对应的k值就是数据的真实聚类数。这也是该方法被称为手肘法的原因。计算SSE的具体公式如下：



其中，Ci是第i个簇，p是Ci中的样本点，mi是Ci的质心（Ci中所有样本的均值），SSE是所有样本的聚类误差，代表了聚类效果的好坏。

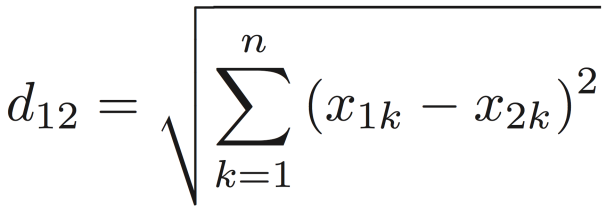
我们假定最合理的聚类结果产生的类别数量不会太多，因此我们计算了K从1到8时的SSE分布曲线，如图X所示。并最终选取了4作为本次K均值聚类算法的K值。

图X SSE-K分布曲线

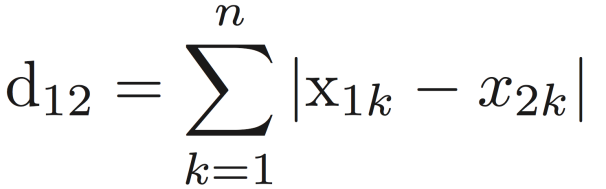


在进行聚类之前，我们还需要确定用于评估“两个样本间距离”的指标。我们选取了最常用的四种计算距离的公式进行调研，分别是欧氏距离、曼哈顿距离、夹角余弦距离和相关距离。

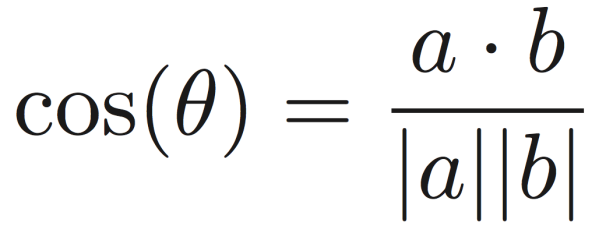
欧氏距离是这4种方法中最简单直观，也是最常使用的一种方法，其具体公式如下：



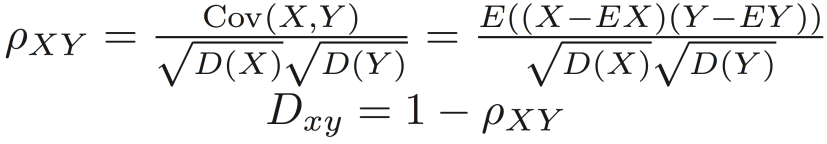
曼哈顿距离为标准坐标系上的绝对轴距综合，其结果是两个坐标差的绝对值，是一种“城市区块距离”，其计算公式如下：



夹角余弦距离是通过从原点出发，并指向两个位置坐标所产生的两条向量间的夹角来测距，其计算公式如下：



相关距离依据相关系数来计算距离。相关系数是衡量给定随机变量X与Y的相关性的一种方法，取值范围是[-1,1]。相关系数和相关距离的计算公式分别如下所示：



经过调研，我们认为相关距离和夹角余弦距离更适合用于文本数据聚类的计算。而曼哈顿距离与欧式距离较为接近，但通常使用欧氏距离聚类质量更高，且各类聚类算法实现中均默认实现了欧氏距离。因此我们选择了使用欧式距离作为评估样本间相似性的指标。

### 5.3.2 基于K均值的聚类分析

最后，我们为K均值聚类算法编写了程序进行聚类分析。聚类结果如图X所示。

如果不好弄图，就再改改这句。

对于每一类的运动学片段，我们计算了其特征参数，结果如表x所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 特征值(单位) | 簇1 | 簇2 | 簇3 | 簇4 |
| Vm（km/h） | 19.80 | 11.22 | 39.66 | 1.66 |
| Vmr（km/h） | 25.80 | 25.34 | 45.95 | 7.62 |
| Vsd（km/h） | 12.03 | 13.05 | 19.34 | 3.03 |
| Ti（%） | 16 | 50 | 12 | 72 |
| Ta（%） | 43 | 27 | 47 | 17 |
| Td（%） | 38 | 24 | 35 | 18 |
| Am（m/s2） | 2.01 | 4.16 | 1.59 | 3.04 |
| Ad（m/s2） | -2.19 | -4.49 | -2.33 | -3.02 |
| Asd（m/s2） | 2.89 | 4.29 | 3.20 | 1.69 |

从表中数据我们可以发现每一类数据的特征不尽相同，可以代表不同类型的行驶状况。

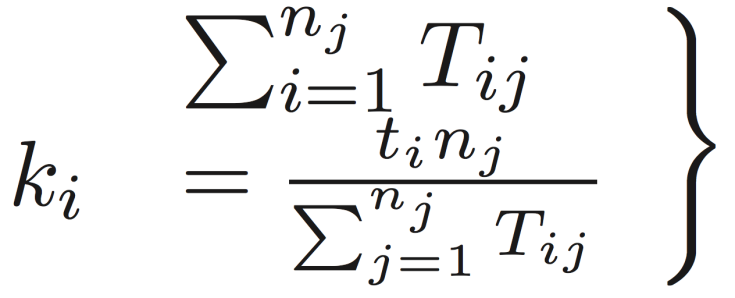
第一，二类数据主要反映了交通状况适中的路段。其中第一类数据车辆平均行驶速度适中且怠速时段较少，可能对应了一般的正常路段。而第一类数据车辆平均行驶速度适中但怠速时段占比偏高，可能对应于经常遇到红绿灯等存在停车时间的路段。

第三类数据主要反映了交通状况较好的路段，其平均行驶速度很高，怠速时间比占比较低且加速时段占比较高。

第四类数据主要反映了较为拥堵的交通状况，车辆平均速度较低，怠速时间比占比很高。

## 5.4 工况的合成

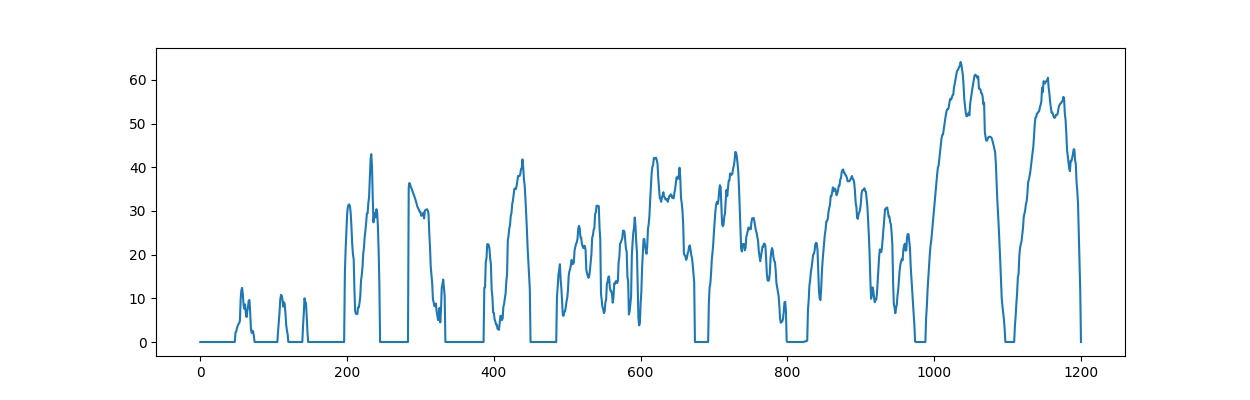
本次我们需要拟合出时间长度在1200~1500s的运动学片段。我们根据各类在整个数据集中所占的时间比例，来确定各类片段在最终工况合成中所占的时间和个数。具体公式如下：



其中：Ti为第i类在工况合成中所占的时间，Ni为第i类片段中片段的个数，Tij为第i类中第j个片段所持续的时间，Tall为样本中所有片段持续的总时间，Tcycle为工况合成所需的时间，Ki为第i类片段在拟合的工况中的片段个数。

依据前一阶段得到的聚类结果，我们拟合出了时间长度为1201s的用车行驶工况，如图X所示。

图X 行驶工况



# 6 模型评估

## 6.1 指标计算

对于5.4中合成的工况，我们计算其各项特征参数值，其结果如表x所示。

|  |  |
| --- | --- |
| 特征值(单位) | 拟合的工况曲线的特征参数值 |
| Vm（km/h） | 19.05 |
| Vmr（km/h） | 26.83 |
| Vsd（km/h） | 12.80 |
| Ti（%） | 29 |
| Ta（%） | 38 |
| Td（%） | 30 |
| Am（m/s2） | 2.59 |
| Ad（m/s2） | -2.99 |
| Asd（m/s2） | 2.44 |

## 6.2 合理性分析

### 6.2.1 特征参数值对比

为了对拟合的行驶工况曲线进行评估，我们计算了处理后的数据源的各项特征参数，并于拟合的行驶工况曲线进行对比，结果如表x所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 特征值(单位) | 拟合的工况 | 原始数据源 | 误差/% |
| Vm（km/h） | 19.05 | 17.21 | 10.7 |
| Vmr（km/h） | 26.83 | 28.10 | 4.5 |
| Vsd（km/h） | 12.80 | 11.04 | 15.9 |
| Ti（%） | 32 | 39 | 17.9 |
| Ta（%） | 35 | 32 | 9.3 |
| Td（%） | 30 | 28 | 7.1 |
| Am（m/s2） | 2.59 | 2.75 | 5.8 |
| Ad（m/s2） | -2.99 | -3.01 | 0.6 |
| Asd（m/s2） | 2.44 | 2.94 | 17.0 |

由表x的数据我们可以发现，拟合的行驶工况曲线各项特征值与原始数据源的平均误差约为9.8%，误差值小于10%，说明我们构建的汽车行驶工况代表性较好，与原始数据吻合度较高。

### 6.2.2 卡方检验

# 参考文献

林滨. K\_Means聚类的多种距离计算方法的文本实验比较[J]. 《福建工程学院学报》, 2016 ,14 (1) :80-85

高建平，孙中博，丁伟，郗建国. 车辆行驶工况的开发和精度研究[J]. 《浙江大学学报:工学版》 , 2017 , 51 (10) :171-179

石琴, 郑与波, 姜平. 基于运动学片段的城市道路行驶工况的研究[J]. 汽车工程, 2011, 33(3):256-261.

# 附录

问题一部分代码

%% 第一部分  
*# read csv*d = pd.read\_csv(**'original\_data/file1.csv'**, usecols=[**'时间'**, **'GPS车速'**])  
print(**"Before data processing,data shape is:"**)  
print(d.shape)  
  
*# data processing*print(**"Data processing..."**)  
*# 1 data transform --replace datatime to time series*d[**'时间'**] = d[**'时间'**].apply(**lambda** x: int(  
 (str\_to\_datetime(x.replace(**".000."**, **""**)) - datetime.datetime(2017, 12, 1, 19, 43, 56)).total\_seconds()))  
%% 第二部分  
*# 2 data processing  
#* ***重复值清洗****# 2.0长期低速度行驶(超过20秒)*i = 0  
**while** i < d.shape[0] - 1:  
 **if** d[**'GPS车速'**][i] < 10:  
 forward = i  
 **while** d[**'GPS车速'**][forward] < 10 **and** forward < d.shape[0] - 1:  
 forward += 1  
 **if** (forward - i) > 20:  
 d[**'GPS车速'**][i:forward] = 0  
 i = forward + 1  
 **continue** i += 1  
  
i = 0  
**while** i < d.shape[0] - 1:  
 *# 2.1 时间不连续删除整个运动片段* **if** d[**'时间'**][i + 1] - d[**'时间'**][i] > 3:forward = i + 1 *# 运动片段尾指针* backward = i *# 运动片段头指针* **while** d[**'GPS车速'**][forward] >= 10 **and** forward < d.shape[0] - 1:  
 *# 非怠速* forward += 1  
 **while** (**not** (d[**'GPS车速'**][backward - 1] >= 10 **and** d[**'GPS车速'**][backward] <= 10)) **and** backward > 1:  
 *# 非怠速* backward -= 1  
 to\_drop.append([backward, forward])  
 i = forward + 1  
 *# print("#1 " + str(i))* **continue** *# 2.2 加速度异常删除整个运动片段* **if** d[**'GPS车速'**][i] > 100:  
 len = i  
 **while** d[**'GPS车速'**][len] > 10 **and** len > 1:  
 len -= 1  
 **if** i - len < 7:  
 forward = i + 1 *# 运动片段尾指针* backward = i *# 运动片段头指针* **while** d[**'GPS车速'**][forward] >= 10 **and** forward < d.shape[0] - 1:  
 forward += 1  
 **while** (**not** (d[**'GPS车速'**][backward - 1] >= 10 **and** d[**'GPS车速'**][backward] <= 10)) **and** backward > 1:  
 backward -= 1  
 to\_drop.append([backward, forward])  
 i = forward + 1  
 *# print("#2 " + str(i))* **continue** *# 2.3长期怠速(停车超过180秒)* **if** d[**'GPS车速'**][i] < 10:  
 forward = i  
 **while** d[**'GPS车速'**][forward] < 10 **and** forward < d.shape[0] - 1:  
 forward += 1  
 **if** (forward - i) > 180:  
 *# print("长期停车/怠速异常" + str(i))  
 # 删剩180s* to\_drop.append([i, forward - 180])  
 i = forward - 180 + 1  
 *# print("#3 " + str(i))* **continue** i += 1  
  
**for** x **in** to\_drop:  
 d = d.drop(d.index[x[0]:x[1]])

%% 第三部分  
*# write csv*print(**"Writing processed data..."**)d.to\_csv(**'processed\_data/file1.csv'**, index=**False**)

问题二部分代码

%% 第一部分  
*# read csv*d = pd.read\_csv(**'processed\_data/file1.csv'**)  
print(d.shape)  
%% 第二部分  
i = 0  
spl = []  
**while** i < d.shape[0] - 1:  
 **if** d[**'GPS车速'**][i] != 0 **and** d[**'GPS车速'**][i + 1] == 0:  
 *# end of a fragment* spl.append(i)  
 i += 1  
  
print(spl)  
label = 0  
**for** k **in** range(len(spl)):  
 new = d[label:spl[k] + 2]  
 new = new.reset\_index(drop=**True**) *# 重新进行index生成* new.to\_csv(**'processed\_data/file1\_fragments/fragment'** + str(k + 1) + **'.csv'**, index=**False**)  
 label = spl[k] + 1

问题三部分代码

%% 第一部分 汽车运动特征评估体系

**def** calculate(filePath):  
 d = pd.read\_csv(filePath) *# 运动时长* total\_time = d.shape[0]  
  
 *# 平均速度* ave\_speed = format(np.mean(d[**'GPS车速'**]), **'.2f'**) *# 平均行驶速度* ave\_runtime\_speed = format(np.mean(d[d[**'GPS车速'**] >= 10][**'GPS车速'**]), **'.2f'**)acceleration\_time = 0  
 deceleration\_time = 0  
 acceleration\_sum = 0  
 deceleration\_sum = 0  
 **for** i **in** range(d.shape[0] - 1):  
 **if** d[**'GPS车速'**][i] < d[**'GPS车速'**][i + 1]:  
 acceleration\_time += 1  
 acceleration\_sum += d[**'GPS车速'**][i + 1] - d[**'GPS车速'**][i]  
 **elif** d[**'GPS车速'**][i] > d[**'GPS车速'**][i + 1]:  
 deceleration\_time += 1  
 deceleration\_sum += d[**'GPS车速'**][i + 1] - d[**'GPS车速'**][i]  
  
 *# 平均加速度* **if** acceleration\_time == 0:  
 ave\_acceleration = 0.00  
 **else**:  
 ave\_acceleration = format(acceleration\_sum / acceleration\_time, **'.2f'**) *# 平均减速度* **if** deceleration\_time == 0:  
 ave\_deceleration = 0.00  
 **else**:  
 ave\_deceleration = format(deceleration\_sum / deceleration\_time, **'.2f'**) *# 怠速时间比* slow\_rate = format(d[d[**'GPS车速'**] == 0].shape[0] / d.shape[0], **'.2f'**) *# 加速时间比* acceleration\_rate = format(acceleration\_time / (d.shape[0] - 1), **'.2f'**) *# 减速时间比* deceleration\_rate = format(deceleration\_time / (d.shape[0] - 1), **'.2f'**) *# 速度标准差* std\_speed = format(np.std(d[**'GPS车速'**], ddof=1), **'.2f'**)acceleration = []  
 **for** i **in** range(d.shape[0] - 1):  
 acceleration.append(d[**'GPS车速'**][i + 1] - d[**'GPS车速'**][i])  
 *# 加速度标准差* std\_acceleration = format(np.std(acceleration, ddof=1), **'.2f'**)line = [filePath, ave\_speed, ave\_runtime\_speed, ave\_acceleration, ave\_deceleration, slow\_rate,  
 acceleration\_rate, deceleration\_rate, std\_speed, std\_acceleration]out = open(**'processed\_data/fragments\_set/all\_file.csv'**, **'a'**, newline=**''**)  
 csv\_write = csv.writer(out, dialect=**'excel'**)  
 csv\_write.writerow(line)

%% 第二部分 PCA主成分分析

data = pd.read\_csv(inputfile) *# 读入数据  
# 数据标准化  
# 1、min-max标准化（Min-Max Normalization）***def** min\_max(data):  
 **return** (data - data.min()) / (data.max() - data.min())  
*# 2、Z-score标准化方法(主成分太多了，没有用这种方法)***def** Z\_score(data):  
 **return** (data - data.mean()) / (data.std())  
  
index, cal = data[**"filePath"**], data  
data\_norm = min\_max(cal)  
*# data\_norm["filePath"] = index*print(data\_norm.head())**from** sklearn.decomposition **import** PCA  
pca = PCA() *# 保留所有成分*pca.fit(data\_norm)  
print(pca.components\_) *# 返回模型的各个特征向量*print(**'每个成分各自方差百分比 :'**)  
print(pca.explained\_variance\_ratio\_) *# 返回各个成分各自的方差百分比(也称贡献率）*pca = PCA(3) *# 选取累计贡献率大于80%的主成分（1个主成分）*pca.fit(data\_norm)  
low\_d = pca.transform(data\_norm) *# 降低维度*pca = pd.DataFrame(low\_d, index)  
pd.concat([pca, cal], axis=1).to\_csv(outputfile)

%% 第三部分 K-Means聚类

loan\_data = pd.read\_csv(**'processed\_data/fragments\_set/all\_file\_pca.csv'**) *# 设置要进行聚类的字段*loan = np.array(loan\_data[[**'0'**, **'1'**, **'2'**]]) *# 设置类别为3*clf = KMeans(n\_clusters=4)  
*# 将数据代入到聚类模型中*clf = clf.fit(loan)  
  
loan\_data[**'label'**] = clf.labels\_ *# 提取不同类别的数据*loan\_data0 = loan\_data.loc[loan\_data[**"label"**] == 0]  
loan\_data1 = loan\_data.loc[loan\_data[**"label"**] == 1]  
loan\_data2 = loan\_data.loc[loan\_data[**"label"**] == 2]  
loan\_data3 = loan\_data.loc[loan\_data[**"label"**] == 3]

%% 第四部分 选取每个簇的代表样本

*# 计算最接近簇心的科学运动片段  
# 欧几里得距离***def** get\_sample(filePath, cluster\_center):  
 loan\_data = pd.read\_csv(filePath)  
  
 loan\_data[**'distant'**] = ((loan\_data[**'0'**] - cluster\_center[0]) \*\* 2 + (  
 loan\_data[**'1'**] - cluster\_center[1]) \*\* 2 + (  
 loan\_data[**'2'**] - cluster\_center[2]) \*\* 2) \*\* 0.5  
 **return** loan\_data[loan\_data[**'distant'**] == loan\_data[**'distant'**].min()]  
  
simple0 = get\_sample(filePath0, cluster\_center0)  
simple1 = get\_sample(filePath1, cluster\_center1)  
simple2 = get\_sample(filePath2, cluster\_center2)  
simple3 = get\_sample(filePath3, cluster\_center3)  
  
  
print(simple0) *# processed\_data/file1\_fragments/fragment594.csv*print(simple1) *# processed\_data/file1\_fragments/fragment1167.csv*print(simple2) *# processed\_data/file1\_fragments/fragment793.csv*print(simple3) *# processed\_data/file1\_fragments/fragment101.csv*

*# matplotlib作图***def** draw\_simple(filePath, cluster):  
 loan\_data = pd.read\_csv(filePath)  
 X = loan\_data[**"时间"**]-min(loan\_data[**"时间"**])  
 Y = loan\_data[**"GPS车速"**]  
 plt.plot(X, Y)  
 plt.plot(max(X+1),0)  
 plt.savefig(**'picture/k=4/cluster'** + str(cluster) + **'.jpg'**)  
 plt.show()

%% 第五部分 计算每个簇的评估体系指标

**def** cal(data, list0):  
 ave\_speed = []  
 ave\_runtime\_speed = []  
 ave\_acceleration = []  
 ave\_deceleration = []  
 slow\_rate = []  
 acceleration\_rate = []  
 deceleration\_rate = []  
 std\_speed = []  
 std\_acceleration = []  
 **for** i **in** range(data.shape[0]):  
 filePath = data[**"filePath"**][i]  
 **if** filePath **in** list0:  
 ave\_speed.append(data[**"平均速度"**][i])  
 ave\_runtime\_speed.append(data[**"平均行驶速度"**][i])  
 ave\_acceleration.append(data[**"平均加速度"**][i])  
 ave\_deceleration.append(data[**"平均减速度"**][i])  
 slow\_rate.append(data[**"怠速时间比"**][i])  
 acceleration\_rate.append(data[**"加速时间比"**][i])  
 deceleration\_rate.append(data[**"减速时间比"**][i])  
 std\_speed.append(data[**"速度标准差"**][i])  
 std\_acceleration.append(data[**"加速度标准差"**][i])

%% 第六部分 构建汽车行驶工况曲线

*# 按照样本比例选取的每个簇代表样本*

cluster1\_1 = pd.read\_csv(**"processed\_data/file2\_fragments/fragment123.csv"**)  
cluster1\_2 = pd.read\_csv(**"processed\_data/file1\_fragments/fragment747.csv"**)  
cluster1\_3 = pd.read\_csv(**"processed\_data/file1\_fragments/fragment444.csv"**)  
cluster2\_1 = pd.read\_csv(**"processed\_data/file3\_fragments/fragment4.csv"**)  
cluster2\_2 = pd.read\_csv(**"processed\_data/file3\_fragments/fragment32.csv"**)  
cluster2\_3 = pd.read\_csv(**"processed\_data/file2\_fragments/fragment366.csv"**)  
cluster3\_1 = pd.read\_csv(**"processed\_data/file1\_fragments/fragment923.csv"**)  
cluster3\_2 = pd.read\_csv(**"processed\_data/file2\_fragments/fragment370.csv"**)  
cluster4\_1 = pd.read\_csv(**"processed\_data/file1\_fragments/fragment118.csv"**)  
cluster4\_2 = pd.read\_csv(**"processed\_data/file1\_fragments/fragment119.csv"**)  
cluster4\_3 = pd.read\_csv(**"processed\_data/file1\_fragments/fragment412.csv"**)  
cluster1 = pd.concat([cluster1\_1, cluster1\_2, cluster1\_3], axis=0)  
cluster2 = pd.concat([cluster2\_1, cluster2\_2, cluster2\_3], axis=0)  
cluster3 = pd.concat([cluster3\_1, cluster3\_2], axis=0)  
cluster4 = pd.concat([cluster4\_1, cluster4\_2, cluster4\_3], axis=0)  
total = pd.concat([cluster4, cluster2, cluster1, cluster3], axis=0)  
print(total)  
total = total.reset\_index()  
total.to\_csv(**"clusterALL/final.csv"**)  
plt.figure(figsize=(12.5, 4))  
Y = total[**"GPS车速"**]  
plt.plot(Y)  
  
plt.savefig(**'picture/k=4/fileALL/combined.jpg'**)  
plt.show()

%% 第七部分 评估验证模型  
kf\_data = np.array([cluster1, cluster2, cluster3, cluster4, total, final])  
kf = chi2\_contingency(kf\_datad)  
print(**'chisq-statistic=%.4f, p-value=%.4f, df=%i expected\_frep=%s'**%kf)