# 基于多模型对比的空气质量数据校准研究

## 摘要

本文根据自建点和国控点的监测数据，建立合理的数学模型，采用计算机程序计算方法，得到了自建点空气质量校准模型，并进行了验证分析。

通过数据的统计观察，发现原始数据存在指标异位、数值异常和重复问题。其中，指标异位指的是 2018 年国控点的 NO2 和 SO2 出现对调现象。利用 python 和 matlab 软件进行数据问题处理，并建立时间序列方法检验数据处理的合理性。

**针对问题一：**从自建点国控点的数据描述和两点数据对比分析这两个方面入手，采用统计量计算和统计图绘制的方法，对题目数据进行探索分析。通过自建点和国控点数据探索可以发现各个指标的数值范围、均值情况和波动情况等。而通过时间频次统一化和匹配后的两点数据对比分析发现，PM2.5、PM10、CO、NO2、SO2 和 O3 的绝对浓度差均值为 17、45、0.51、21.98、5.34 和 33.04，说明国控点和自建点浓度数据存在明显差异，且除 CO 自建点数据较国控点偏低，其余污染物自建点数据皆偏高。

**针对问题二：**国控点和自建点数据差异的形成原因有：零点漂移、量程漂移、非常规气态污染物对传感器的交叉干扰和天气因素影响。采用分阶段校准方法，以零点漂移、量程漂移为系统内部干扰因素，其影响很小，基于公式进行自主校准即可；以非常规气态污染物浓度和天气因素为外部干扰因素，并作为因变量，而两点浓度差作为自变量，建立回归模型进行因素分析。根据结果，对各个污染物模型中影响较大的前三项因素做以下罗列。固体颗粒物 PM2.5 和 PM10 模型皆为湿度、对应自建点浓度和气压；CO 模型为气压、SO2 浓度和对应自建点浓度；NO2 模型为对应自建点浓度、风速和湿度；SO2 模型为湿度、O3 浓度和 NO2 浓度；O3 模型为对应自建点浓度、温度和 CO 浓度。

回归模型参数皆显著，但拟合优度较低。因此构建调整后回归模型、支持向量机模型和梯度提升回归树模型进行分析，对比择优，如下表。

| 污染物种类 | 拟合优度 | | | | 最终选择模型 |
|---|---|---|---|---|---|
| | 回归模型 | 调整后回归模型 | 支持向量机模型 | 梯度提升树模型 | |
| PM2.5 | 0.45 | **0.90** | 0.54 | 0.59 | 调整后的回归模型 |
| PM10 | 0.77 | 0.77 | 0.77 | **0.78** | 梯度提升树 |
| CO | 0.36 | 0.38 | 0.68 | **0.73** | 梯度提升树 |
| NO2 | 0.61 | 0.63 | **0.80** | 0.78 | 支持向量机 |
| SO2 | 0.61 | 0.52 | 0.63 | **0.74** | 梯度提升树 |
| O3 | 0.81 | 0.69 | **0.90** | 0.86 | 支持向量机 |

**针对问题三：**数据按比例 0.75：0.25 分割，前者为为训练集，后者为测试集。基于训练集数据，对问题二模型进行转换，将浓度差中的自建点指标进行移项，得到空气质量校准模型。利用测试集数据，对比分析空气质量校准数据和国控点数据的差异，发现误差值小，模型效果较好。进一步地，结合全文构建得出空气质量校准机制仪并绘制了其说明书。

**关键词：**异常值区间检验；梯度提升回归树模型；多模型对比分析和检验

# 一、问题重述

空气污染对生态环境和人类健康危害巨大，通过对“两尘四气”（PM2.5、PM10、CO、$NO_2$、$SO_2$、$O_3$）浓度的实时监测可以及时掌握空气质量，对污染源采取相应措施。虽然国家监测控制站点（国控点）对“两尘四气”有监测数据，且较为准确，但因为国控点的布控较少，数据发布时间滞后较长且花费较大，无法给出实时空气质量的监测和预报。某公司自主研发的微型空气质量检测仪（如图所示）花费小，可对某一地区空气质量进行实时网格化监控，并同时监测温度、湿度、风速、气压、降水等气象参数。

由于所使用的电化学气体传感器在长时间使用后会产生一定的零点漂移和量程漂移，非常规气态污染物（气）浓度变化对传感器存在交叉干扰，以及天气因素对传感器的影响，在国控点近邻所布控的自建点上，同一时间微型空气质量检测仪所采集的数据与该国控点的数据值存在一定的差异，因此，需要利用国控点每小时的数据对国控点近邻的自建点数据进行校准。

附件 1.CSV 和附件 2.CSV 分别提供了一段时间内某个国控点每小时的数据和该国控点近邻的一个自建点数据，各变量单位见附件 3。请建立数学模型研究下列问题：1. 对自建点数据与国控点数据进行探索性数据分析。

2. 对导致自建点数据与国控点数据造成差异的因素进行分析。
3. 利用国控点数据，建立数学模型对自建点数据进行校准。

# 二、问题分析

## 2.1 名词准备

**零点漂移：**

零点漂移是指没有交流电输入时，静态工作点因受温度变化、电源电压不稳等因素影响发生变化，被逐级放大和传输，从而出现电路输出端电压围绕原固定值上下漂动的现象[1]。

**量程漂移：**

量程是度量工具的测量范围，由度量工具的最小、最大测量值决定；漂移是指测量仪器在规定条件下随时间的慢变化。其产生原因大多是压力、温度、湿度的变化，或测量仪器本身性能的不稳定，从而使得数值不能准确反映测试值[2]。

**气体交叉干扰：**

气体交叉干扰是指传感器会对除目标气体以外的其他气体产生一定的响应，从而对观测数据的准确性产生严重影响。在定电位电解法原理或非色散红外光谱法原理中都存在一定程度的气体交叉干扰，其产生原因以及影响程度随其原理而异 [3]。

## 2.2 数据情况分析

根据附件 1 和附件 2 给的数据情况，以下几点需要进行调整：

1. 异常数据的处理，及其合理性检验；
2. 数据时间频次不同，需进行统一化处理；

3.训练集和测试集的区分。本文随机选择 75%的数据作为训练集，25%的数据做为测试集。训练集用来估算模型参数，而测试集用来评估模型效果。

### 2.3 问题一的分析

自建点数据包括高频的大气污染物浓度的数据和对应时间的天气因素数据，国控点数据包括不同于自建点时间频次的大气污染物数据。为了了解两组数据各自的情况及其交互情况，进行两方面的探索分析：自建点和国控点的数据统计量计算和图表说明、国控点和自建点数据对比分析。在进行国控点和自建点数据对比分析时，首先进行时间频次的统一，再进行时间点的匹配，最后实现对国控点和自建点数据差异的统计描述分析。

### 2.4 问题二的分析

为了分析两点浓度差的影响因素情况，并优化得到准确的误差预测模型，该问题可分为两步：第一步，基于零点量程计算和多元回归模型的影响程度分析；第二步是多预测模型对比分析，按拟合效果择优。

**第一步：**电化学气体传感器在长时间使用后，会受到零点漂移、量程漂移、非常规气态污染物浓度和天气因素的影响，其中零点漂移和量程漂移属于传感器系统内部干扰因素，剩余两项为环境外部干扰因素。对于系统内部干扰因素，查阅相关文献，根据一定指标对自建点的污染物浓度数据进行影响程度分析和调整。而对于系统外部干扰因素，以国标点和自建点的数据差值为因变量，非常规气态污染物浓度和天气因素为自变量建立回归模型进行预测分析，以确定数据差异的形成原理，具体思路见图 1。

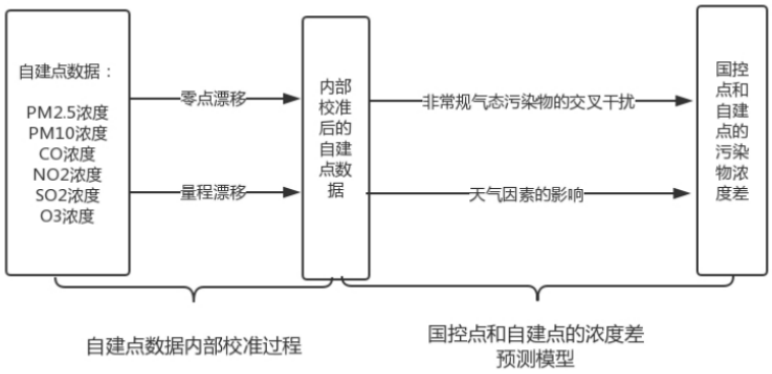



**图 1　问题二思路图**

**第二步：**根据参考文献[4],从方法角度进行分类，大气污染的定量预测模型有统计预测、智能预测和数值预测三大类。其中常用的统计预测模型主要包括灰色预测和回归预测；智能预测主要包括神经网络和支持向量机等；数值预测则是基于大气动力学相关数据进行分析。

本文基于问题的数据情况和分析目的，以国控点和自建点的浓度差为因变量，相关外部环境因素为自变量，建立回归模型、支持向量机模型和基于交叉验证的梯度提升回归树模型对六种污染物分别进行准确的差异预测，如图 2。


图 2　预测模型说明图

### 2.5 问题三的分析

国控点数据监测虽然准确，但是存在布控少、时间滞后长和成本高的问题。

而自建点数据监测虽然误差较国控点大，但是有成本低和实时监测无滞后的优点。因此，通过相关影响因素的分析，建立合理的数学模型，采取程序化计算方式，从而实现了自建点数据的校准。

问题二的模型对国控点和自建点的浓度差进行了预测，本小题将问题二的模型进行变换得到，以国控点数据、国控点与自建点浓度差值为因变量的自建点空气质量校准模型。将 75%的训练数据集代入该校准模型，得到自建点的校准数据，选择 25%测试集数据，对比分析自建点的校准数据和国控点的数据，进行模型评价。进一步，基于全文的分析，构建空气质量校准机制仪并绘制说明书。

# 三、模型假设

为便于下文模型的建立与求解，在此做出以下假设：

1.假设仪器数据值不受人为因素的干扰；

2.假设国控点检测数据准确，预处理完毕后可作为自建点数据校准的标准；

3.假设数据记录期间，仪器未发生损坏。

# 四、符号说明及变量单位

**4.1 符号说明**

| 符号 | 说明 |
|---|---|
| $T_0$ | 自建点某指标在 2018/11/14 10:02 时的值 |
| $T_{0_i}$ | 每隔一个小时后的自建点数值记为 |
| $\delta_{0_i}$ | 零点漂移 |
| $pop(i)$ $i = CO, NO_2 SO_2, O_3$ | 自建点（private observation point）污染物浓度 |
| $nop(i)$ $i = CO, NO_2 SO_2, O_3$ | 国控点（national observation point）污染物浓度 |
| $dc(i)$ $i = CO, NO_2 SO_2, O_3$ | 污染物两点浓度差（difference in concentrations） |
| $spop(i)$ $i = CO, NO_2 SO_2, O_3$ | 自建点校准后的污染物浓度 |
| $T$ | 温度（temperature） |
| $H$ | 湿度（humidity） |
| $V$ | 风速（velocity） |
| $Ap$ | 气压（air pressure） |
| $P$ | 降水量（precipitation） |

**4.2 变量单位**

| 变量 | 单位 | 变量 | 单位 |
|---|---|---|---|
| PM2.5 | $\mu g/m^3$ | 风速 | m/s |
| PM10 | $\mu g/m^3$ | 压强 | Pa |
| CO | $mg/m^3$ | 降水量 | $mm/m^2$ |
| $NO_2$ | $\mu g/m^3$ | 温度 | ℃ |
| $SO_2$ | $\mu g/m^3$ | 湿度 | rh% |
| $O_3$ | $\mu g/m^3$ | 时间 | YY/MM/DD H:M |

# 五、数据预处理

**5.1 数据清洗**

题目提供了 3 个附件：附件 1.CSV 提供了一段时间内某个国控点每小时的数据，共 4200 条；附件 2.CSV 提供了一段时间内该国控点近邻的一个自建点数据（相应于国控点时间且间隔在 5 分钟内），共 234717 条；附件 3 给出了各变量的单位。

为了便于本题的数据分析以及求解，本文在参考国家标准[5]的基础上运用 Excel 以及 Python 软件对数据进行清洗整理，具体思路见图 3。

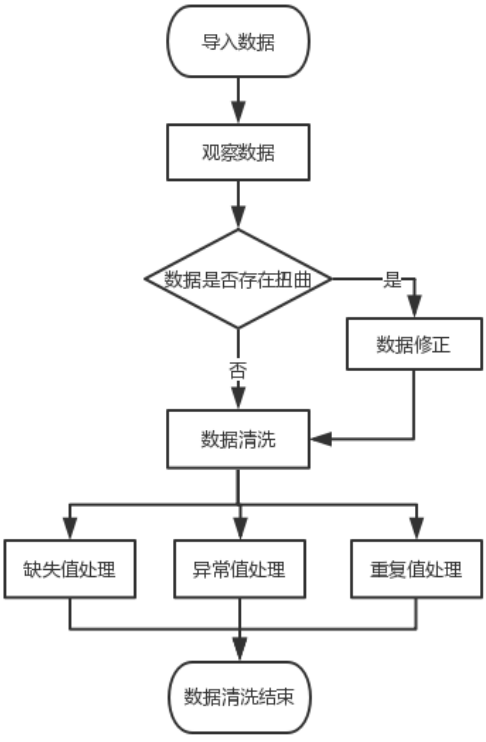



图 3　数据清洗流程图

**1.数据扭曲的发现及修正：**

首先，对附件 1、2 中国控点、自控点的数据进行观察，将数据依次导入 MATLAB。

然后，在考虑时间序列影响的基础上，通过 MATLAB 编程绘制指标趋势图并检测异常值。

最后，对比发现附件 1 中国控点的 N02 以及 SO2 两个指标存在明显断层，趋势图见图 4、图 5，断层扩大图见图 6。

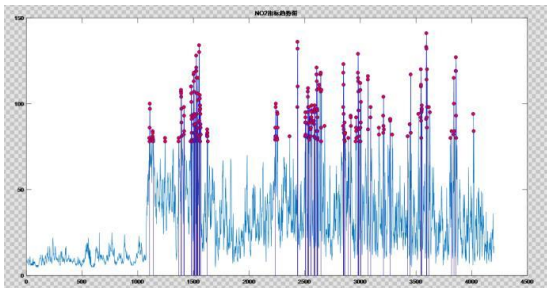


图 4　NO2 指标趋势图

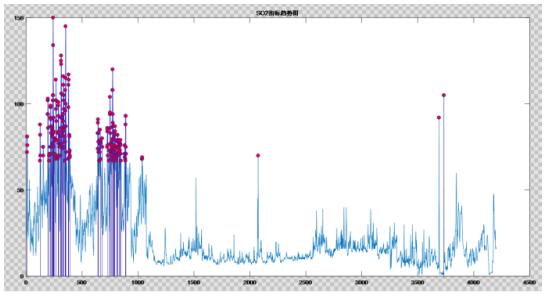


图 5　SO2 指标趋势图

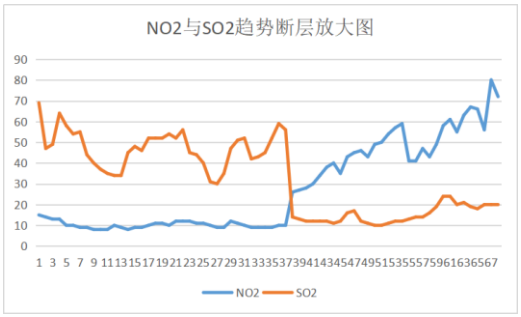



图 6　断层放大图

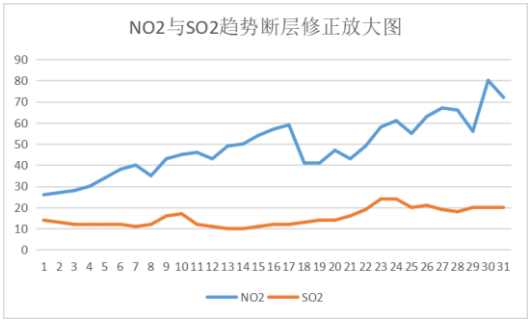



图 7　趋势修正放大图

由此，本文根据附件 2 中自建点相对应两指标的数据趋势对附件 1 中的 NO2 以及 SO2 指标进行修正，修正完毕后的趋势图放大图见图 7。

观察得出，修正后的趋势较吻合，说明修正成功，所修正数据合理。

**2.缺失值处理：**

将修正后的附件 1、附件 2 的数据依次导入 Python，用 Python 语言编写代码检测是否存在缺失值，通过语句 isna( )判断得出：不存在缺失值。

**3.异常值处理：**

异常值即指样本中出现的明显偏离大多数观测值的个别值，异常值检验有很多种方法，在此本文采用箱形图（不受异常值的影响，能够准确稳定地描绘出数据的离散分布情况，同时也利于数据的清洗）。

以附件 2 中 SO2、CO 两个指标为例，其箱形图见图 8，其余指标箱形图见附录一。

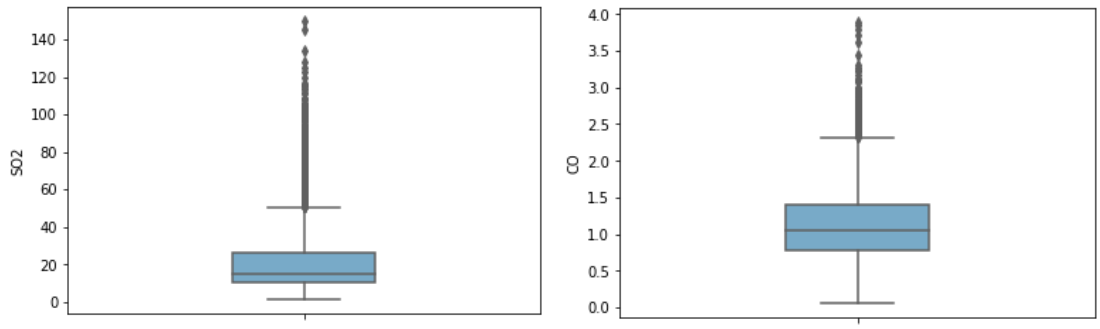



图 8　附件 2 中 SO2、CO 两个指标的箱形图

在此，异常值被定义为小于$Q_1 - 1.51QR$或大于$Q_3 + IQR$的值，对附件 1、附件 2 中的数据进行如上操作绘制箱形图，分别得出了各附件中个类别异常值的区间。针对异常值的区间，运用 Excel 对原始数据进行筛选与剔除，具体见表 1、表 2。

表 1　附件 1 删除数据

| 类别 | 删除条件 | 删除条数 | 占比 |
|---|---|---|---|
| PM2.5 | ＞143.5 | 94 | 2.24% |
| PM10 | ＞201.5 | 80 | 1.90% |
| CO | ＞2.329 | 93 | 2.21% |
| NO2 | ＞101.25 | 68 | 1.62% |
| SO2 | ＞26.5 | 110 | 2.62% |
| O3 | ＞160.5 | 195 | 4.64% |

表 2　附件 2 删除数据

| 类别 | 删除条件 | 删除条数 | 占比 |
|---|---|---|---|
| PM2.5 | ＞168.5 | 4482 | 1.91% |
| PM10 | ＞261 | 11420 | 4.87% |
| CO | ＞1.15 | 5419 | 2.31% |
| NO2 | ＞148 | 68 | 0.03% |
| SO2 | ＞21.5 或＜9.5 | 2042 | 0.87% |
| O3 | ＞154 | 5014 | 2.14% |

**4.重复值处理：**

以完成异常值筛选的数据为蓝本，通过 Python 语句 duplicated( )判断得出：附件 2 中存在 2957 条重复数据，约占比 1.26%。对于此类数据，本文在此将其剔除，部分重复数据展示见表 3。

表 3　附件 2 部分重复数据展示

| PM2.5 | PM10 | CO | NO2 | SO2 | O3 | 风速 | 压强 | 降水量 | 温度 | 湿度 | 时间 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 38 | 76 | 1 | 60 | 16 | 74 | 1 | 1018 | 155 | 11 | 94 | 2018/12/4 3:20 |
| 38 | 76 | 1 | 60 | 16 | 74 | 1 | 1018 | 155 | 11 | 94 | 2018/12/4 3:20 |

数据清洗的代码见附录一，清洗完毕的数据中，附件 1 的数据条数为 3679 条，数据保留度为 87.60%；附件 2 的数据条数为 213390 条，数据保留度为 90.91%。数据保留度较高，可进行下文的分析与求解。

**5.2 时间序列分析[6]——异常值区间检验（以自建点的 PM2.5 为例）**

**5.2.1 四分位法**

首先，将附件 2 中自建点 PM2.5 的数据定义为信号源 X，找到信号源 X 的 75%、25%分别位置记为 A1、A2。然后，求出信号源 X 中位数记为m0，得出四分位差距$\mathrm{A1} - \mathrm{A2}$,得出上四分位差距$\mathrm{b1} = \mathrm{m0} + \mathrm{A1} - \mathrm{A2}$，得到下四分位差距$\mathrm{b2} = \mathrm{mo} - \mathrm{A1} + \mathrm{A2}$。最后将信号源 X 中超出 b1 和 b2 的信号视为异常信号，并用红圈标明。

**5.2.2 方差法**

求信号源 X 的均值 W，然后求信号源 X 的方差 F，通过计算$W \pm F$得到$\lambda_1$、$\lambda_2$。用$[\lambda_1，\lambda_2]$作为正常值的上下区间，将不属于$[\lambda_1，\lambda_2]$的信号视为异常信号，并用红圈标明。

**5.2.3 滑动窗口分析法**

滑动窗口分析法是指随时提供 N 个时间段的数据信息，当数据信息更新时（“窗口”向前滑），将最新时间“窗口”内的数据包含进去，把“窗口”内旧时间“窗口”内的数据丢弃，使得窗口大小保持不变。当滑动时间窗口前进到下一时

间间隔时，活动数据库被更新，增加某一新窗口数据集，同时删除另一窗口数据集，具体原理图见图 9。

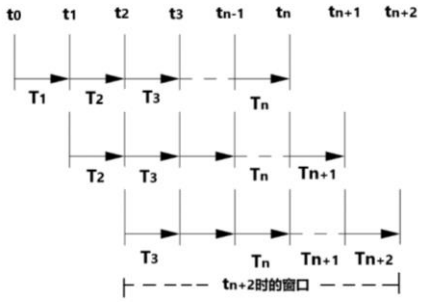



图 9　滑动窗口原理图

通过上述四个理论通过 MATLAB 编写代码（见附录一）对原始数据进行异常值判别，同时验证上文运用四分位法求出异常值范围并删去的异常的方法是否合理，四种方法处理异常值的结果、准确率以及错误数据覆盖率见表 4。

表 4　四种处理异常值方法的结果

| 方法 | 错误个数 | 准确率 | 错误数据覆盖率 |
|---|---|---|---|
| 四分位法 | 114899 | 51.05% | 48.95% |
| 四分位差距中位数法 | 36413 | 84.49% | 15.51% |
| 方差法检测 | 63198 | 73.07% | 26.93% |
| 滑动四分位法 | 227497 | 3.08% | 96.92% |

从表四得出，运用四分位差距中位数法筛选异常值较为合理且准确率较高，即证明上文中的数据清洗合理。

### 5.3 国控点和自建点数据集的统一化

利用 python 的 groupby 和 merge 函数进行分组和匹配，以匹配成功的数据作为支撑模型建立的数据集。随机选取其中的 75%作为训练数据集，其余 25%则为测试数据集，用于问题二与问题三的分析。

## 六、模型的建立与求解

### 6.1 问题一的求解

#### 6.1.1 探索性数据分析[7]含义解释

探索性数据分析所用到的技术与方法可以划分为两类，一类是基于定量方法的（计算 EDA），另一类是基于图像的（图形 EDA）。前者包括简单的统计计算以及高级的探索分析多变量数据的多元统计分析方法，后者即数据探索分析的可视化，常用图形有直方图、箱形图、QQ 图等。

#### 6.1.2 自建点、国控点数据统计量计算和图表说明

**1.计算 EDA：**针对自建点数据本题运用 Python 从均值、标准差、最大最小值、百分位数角度，运用 MATLAB 从偏度（skew）、峰度（kurt）角度汇总计算附件中的各指标，附件 1、附件 2 的输出结果分别见表 5、表 6。代码见附录二。

表 5　附件 1 各指标汇总统计

| | PM2.5 | PM10 | CO | NO2 | SO2 | O3 |
|---|---|---|---|---|---|---|
| Mean | 53.21 | 77.37 | 1.06 | 40.83 | 11.74 | 49.39 |
| Std | 30.13 | 40.97 | 0.42 | 20.42 | 4.65 | 36.81 |
| Min | 1 | 2 | 0.05 | 5 | 1 | 1 |
| 25% | 30 | 46.5 | 0.75 | 25 | 8 | 20 |
| 50% | 47 | 72 | 1.02 | 37 | 11 | 44 |
| 75% | 72 | 104 | 1.354 | 54 | 15 | 70.5 |
| max | 143 | 201 | 2.321 | 101 | 26 | 160 |
| skew | 0.59 | 2.68 | 0.66 | 2.75 | 0.83 | 3.06 |
| kurt | 0.31 | 2.9 | 2.96 | 0.67 | 3.32 | 0.73 |

表 6　附件 2 各指标汇总统计

| | PM2.5 | PM10 | CO | NO2 | SO2 | O3 | 风速 | 压强 | 降水量 | 湿度 |
|---|---|---|---|---|---|---|---|---|---|---|
| Mean | 64.57 | 64.57 | 102.97 | 0.63 | 55.01 | 15.58 | 62.21 | 1017.38 | 120.14 | 67.27 |
| Std | 32.22 | 54.01 | 0.48 | 28.50 | 1.59 | 29.36 | 0.68 | 8.64 | 83.30 | 22.38 |
| Min | 1 | 2 | 0 | 1 | 10 | 1 | 0 | 997 | 0 | 10 |
| 25% | 40 | 65 | 0 | 28 | 14 | 39 | 0 | 1010 | 51 | 50 |
| 50% | 57 | 91 | 1 | 51 | 16 | 57 | 1 | 1018 | 128 | 71 |
| 75% | 85 | 133 | 1 | 75 | 17 | 82 | 1 | 1024 | 181 | 86 |
| max | 168 | 261 | 1 | 145 | 21 | 154 | 9 | 1040 | 324 | 100 |
| skew | 0.53 | 2.59 | -0.46 | 3.97 | 0.33 | 2.50 | 0.71 | 2.9870 | 0.4951 | 2.74 |
| kurt | 3.03 | 1.96 | 9.08 | -0.11 | 2.49 | 2.21 | 0.48 | 2.33 | -0.58 | 2.44 |

**2.图形 EDA：**　将预处理完毕后的数据导入 Python 绘制箱形图，同时得到了附件 1、2 各指标的四分位数，分别见表 7、表 8。

表 7　附件 1 各指标四分位数

| | PM2.5 | PM10 | CO | NO2 | SO2 | O3 |
|---|---|---|---|---|---|---|
| 25 | 30 | 46.5 | 0.75 | 25 | 8 | 20 |
| 50 | 47 | 72 | 1.02 | 37 | 11 | 44 |
| 75 | 72 | 104 | 1.354 | 54 | 15 | 70.5 |

表 8　附件 2 各指标四分位数

| d2 | PM2.5 | PM10 | CO | NO2 | SO2 | O3 |
|---|---|---|---|---|---|---|
| 25 | 40 | 66 | 0.4 | 28 | 14 | 39 |
| 50 | 57 | 91 | 0.5 | 51 | 16 | 57 |
| 75 | 85 | 134 | 0.7 | 75 | 17 | 82 |

以附件 2 中的 SO2 与 CO 指标为例，其箱形图分别为图 10 中的左图、右图，其他指标箱形图见附录二。

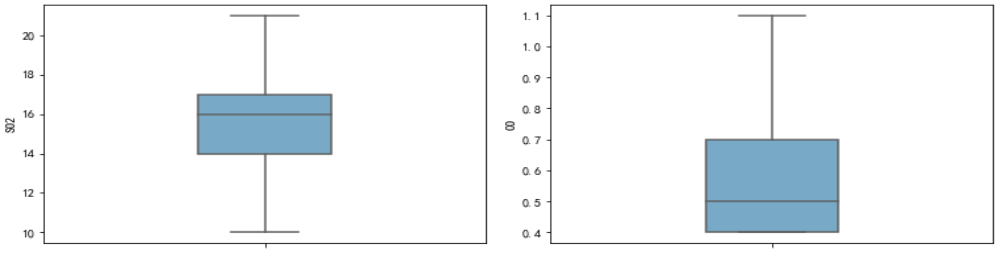



图 10　附件 2 中 SO2 与 CO 指标的箱形图

结合表 8 与图 8 可知，附件 2 中的 SO2 指标数值在区间[16,17]内较为集中，CO 的指标数值在区间[0.4,0.5]内较为集中，其他指标的数值集中区间见表 7 和表 8。

### 6.1.3 国控点、自建点数据对比分析

由于附件 1 中国控点各指标均以小时为时间单位记录、附件 2 中自控点各指标以分为时间单位记录，因此将附件 2 中各指标每 60 分钟进行求和平均，将其记为小时测量值，从而完成对国控点和自建点数据的统一化。

最后，用 Python 将处理后的数据进行相应时间点的匹配，从而实现对国控点和自建点数据差异的统计性描述。统计结果见表 9，代码见附录二。

表 9 “两尘四气”国控点与自建点浓度差值特征汇总表

| 因变量/统计量 | 均值 | 标准差 | 最小值 | 最大值 | 中位数 | 众数 |
|---|---|---|---|---|---|---|
| PM2.5 两点浓度差值 | 17 | 13.88 | -75.00 | 40.00 | -16.00 | -19.00 |
| PM10 两点浓度差值 | 45 | 43.79 | -196.00 | 122.00 | -35.00 | -42.00 |
| CO 两点浓度差值 | 0.51 | 0.40 | -0.62 | 1.81 | 0.46 | 0.24 |
| NO2 两点浓度差值 | 21.98 | 21.96 | -108.00 | 66.00 | -18.00 | -29.00 |
| SO2 两点浓度差值 | 5.34 | 4.91 | -16.00 | 13.00 | -5.00 | -8.00 |
| O3 两点浓度差值 | 33.04 | 40.17 | -121.00 | 85.00 | -3.00 | 11.00 |

备注：均值以绝对差$|nop(i) - pop（i）|$为计算基础

从表中数据可以得出 CO 和 SO2 两点浓度差值各项指标普遍偏最小，PM10、NO2、O3 两点浓度差值区间较大，其中 PM10 的两点浓度误差区间最大为[-196,122]。

## 6.2 问题二的求解

### 6.2.1 浓度差影响因素分析

题目中明确提及，国控点和自建点差异存在原因包括四项：1）零点漂移；2）量程漂移；3）非常规气态污染物（气）浓度变化对传感器存在交叉干扰；4）天气因素对传感器的影响。零点漂移和量程漂移跟传感器的功能参数有较大关系，因此将其定义为系统内部干扰因素。而其余两项，则属于外部环境影响因素。

**1、 内部干扰因素**

对于内部干扰因素，本文根据相关公式进行改进和计算，在实现量化分析的基础上进行自建点校准。

**（1）零点漂移**

本文将自建点某指标在 2018/11/14 10:02 时的值记为$T_0$，将每隔一个小时后的自建点数值记为$T_{0_i}$，按照公式（1）计算零点漂移$\delta_{0_i}$，取最大绝对值为自建点零点漂移值。

$$\delta_{0_i} = \frac{T_{0_i} - T_0}{T} \times 100\% \quad （1）$$

在公式（1）的基础上，本文将分子改为邻近一小时自建点数值的差值，如公式（2）。

$$\delta_{0_i} = \frac{T_{0_{i+1}} - T_{0_i}}{T} \times 100\% \quad （2）$$

对附件 2 中“两尘四气”六个指标分别按上述两个公式进行求值，测量值如表 10 所示，测量仪器中各指标的量程通过查阅化工仪器网[8]可得。

表 10　附件 2 各指标零点漂移测量结果

| | PM2.5 | PM10 | CO | NO2 | SO2 | O3 |
|---|---|---|---|---|---|---|
| 量程 | 500 | 2000 | 20000 | 200000 | 50 | 20000 |
| 减初始 | 0.236 | 0.0815 | 0.0002 | 0.0004 | 0.12 | 0.0054 |
| 减相邻 | 0.216 | 0.0865 | 0.0003 | 0.0005 | 0.14 | 0.0076 |

对比两种公式下输出的测量值，发现改进后的公式求出的数值略大于减初始的数值、更具可比性，因此本文选用第二种方法的值。

**（2）量程漂移**

取前 3 次测定值的算术平均值为初始测定值$x_0$，以后续测定值$x_i$与初始测定值$x_0$的差值为分母，以工作量程上限 A 为分母，量程漂移用 RD 表示，计算过程见公式（3），结果见表 11。

$$\mathrm{RD} = \frac{x_i - x_0}{A} \tag{3}$$

表 11　附件 2 量程漂移测量值

| | PM2.5 | PM10 | CO | NO2 | SO2 | O3 |
|---|---|---|---|---|---|---|
| 量程 | 500 | 2000 | 20000 | 200000 | 50 | 20000 |
| 量程漂移 | 0.1846 | 0.0820 | 0.0002 | 0.0002 | 0.0782 | 0.0033 |

根据上述结果，零点漂移和量程漂移的影响较小。其自校过程即为基于对应影响浓度差的补偿，从而为下文的模型建立提供了数据蓝本。

**2.外部环境影响因素分析**

外部环境因素的分析，以 6.2.1 系统内部干扰校准后的自建点数据为基础。为了说明国控点和自建点污染物浓度的差异形成机制，本文以国控点和自建点污染物浓度差值为因变量，产生影响的相关外部环境因素为自变量，建立预测模型进行分析。

**（1）模型准备**

由于不同大气污染物的特性不同，“两尘四气”六种污染物浓度的国控点和自建点差异需要分别进行预测和分析。为便于下文模型的建立，在此本文对不同污染物差异预测模型中自变量与因变量的选择与取值进行如下说明。

（1）因变量和自变量选择见表 12

表 12　两点浓度差预测模型的变量设置说明

| 污染物类型 | 污染物 | 因变量 | 自变量 |
|---|---|---|---|
| 固体颗粒物污染物 | PM2.5 | PM2.5 两点浓度差 | PM2.5 自建点浓度、温度、湿度、风速、气压、降水量 |
| | PM10 | PM10 两点浓度差 | PM10 自建点浓度、温度、湿度、风速、气压、降水量 |
| 气态污染物 | CO | CO 两点浓度差 | CO 自建点浓度、温度、湿度、风速、气压、降水量、NO2 自建点浓度、SO2 自建点浓度、O3 自建点浓度 |
| | NO2 | NO2 两点浓度差 | NO2 自建点浓度、温度、湿度、风速、气压、降水量、CO 自建点浓度、SO2 自建点浓度、O3 自建点浓度 |
| | SO2 | SO2 两点浓度差 | SO2 自建点浓度、温度、湿度、风速、气压、降水量、CO 自建点浓度、NO2 自建点浓度、O3 自建点浓度 |
| | O3 | O3 两点浓度差 | O3 自建点浓度、温度、湿度、风速、气压、降水量、CO 自建点浓度、NO2 自建点浓度、SO2 自建点浓度 |

（3）预测模型中，为了消除量纲差异的干扰，对数据进行了标准化处理。

（4）随机选取其中的 75%作为训练数据集，其余 25%则为测试数据集。

**（2）回归预测模型**

大气污染物两点浓度差的回归预测方程，固体颗粒物污染物以 PM2.5 为例，气体污染物以 NO2 为例，标准式见公式（4）。

$$dc(PM_{2.5}) = a_0 + a_1 pop(PM_{2.5}) + a_2 T + a_3 H + a_4 V + a_5 Ap + a_6 P$$
$$dc(\mathrm{NO_2}) = a_0 + a_1 pop(\mathrm{NO_2}) + a_2 T + a_3 H + a_4 V + a_5 Ap + a_6 P + a_7 \mathrm{pop}(CO) + a_8 \mathrm{pop}(SO_2) + a_9 \mathrm{pop}(O_3) \quad （4）$$

利用软件 Minitab，对“两尘四气”的六种污染物的两点浓度差进行多元回归分析，结果汇总如表 13。

表 13　两点浓度差回归预测模型的参数说明

| | 自变量（自建点数据） | | | | | | | | | | | 拟合优度 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 常数项 | 湿度 | 温度 | 降水量 | 气压 | 风速 | PM2.5 | PM10 | CO | NO2 | SO2 | O3 | |
| PM2.5 两点浓度差 | -15.08 | -14.35 | -2.21 | -1.26 | -8.33 | -1.70 | -10.58 | | | | | | 0.45 |
| | 0.00 | 0.00 | 0.01 | 0.00 | 0.00 | 0.02 | 0.00 | | | | | | |
| PM10 两点浓度差 | -33.64 | -51.49 | -8.68 | -6.78 | -34.10 | -10.73 | | -51.38 | | | | | 0.77 |
| | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | | 0.00 | | | | | |
| CO 两点浓度差 | 0.53 | -0.17 | -0.08 | 0.12 | -0.68 | -0.13 | | | -0.24 | 0.16 | 0.57 | -0.05 | 0.36 |
| | 0.00 | 0.00 | 0.04 | 0.00 | 0.00 | 0.00 | | | 0.00 | 0.00 | 0.00 | 0.00 | |
| NO2 两点浓度差 | -33.27 | -16.68 | -7.70 | 0.21 | -14.11 | -27.19 | | | 7.27 | -36.55 | 16.01 | 0.29 | 0.61 |
| | 0.00 | 0.00 | 0.00 | 0.70 | 0.00 | 0.00 | | | 0.00 | 0.00 | 0.00 | 0.76 | |
| SO2 两点浓度差 | -1.12 | -7.10 | 1.20 | 0.92 | -1.21 | -0.60 | | | -0.51 | 2.07 | -1.01 | -2.19 | 0.61 |
| | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.01 | | | 0.01 | 0.00 | 0.00 | 0.00 | |
| O3 两点浓度差 | -40.89 | -11.73 | 34.47 | -0.54 | 12.81 | 18.50 | | | -31.60 | -21.77 | 6.00 | -44.62 | 0.81 |
| | 0.00 | 0.00 | 0.00 | 0.00 | 0.50 | 0.00 | | | 0.00 | 0.00 | 0.00 | 0.00 | |

备注：各污染物模型中第一行为各项自变量系数，第二行为对应显著性水平，空值对应自变量为非干扰因素

根据表 2，可以得到各个污染物两点浓度差的回归模型，以拟合优度最高的臭氧模型为例，回归结果如公式（5）。

$$dc(O_3) = -40.89 - 44.62 pop(O_3) + 35.47T - 11.37H + 18.5V + 12.81Ap - 0.54P - 31.6\mathrm{pop}(CO) + 6\mathrm{pop}(SO_2) - 44.62\mathrm{pop}(O_3) \quad （5）$$

根据上述结果，对各个污染物模型中影响较大的前三项因素做以下罗列。固体颗粒物 PM2.5 和 PM10 模型皆为湿度、对应自建点浓度和气压；CO 模型为气压、SO2 浓度和对应自建点浓度；NO2 模型为对应自建点浓度、风速和湿度；SO2 模型为湿度、O3 浓度和 NO2 浓度；O3 模型为对应自建点浓度、温度和 CO 浓度。

### 6.2.2 预测模型优化

根据参考文献[4],从方法角度进行分类，大气污染的定量预测模型有统计预测、智能预测和数值预测三大类。其中常用的统计预测模型主要包括灰色预测和回归预测；智能预测主要包括神经网络和支持向量机；数值预测则是基于大气动力学相关数据进行分析。

基于本文的数据情况和分析目的，选择调整后的回归模型、支持向量机模型和基于交叉验证的梯度提升回归树模型进行准确的差异预测。

**1.调整因变量后的回归预测模型**

从上述模型的拟合优度发现，PM2.5 和 CO 模型效果差，NO2 和 SO2 模型的效果也不是很好。由于进行两点浓度差的预测，最终是为了实现国控点污染物浓度的精准预测。因此调整因变量为国控点的污染物浓度，自变量不变，建立多元回归预测模型，其结果如表 14。

表 14　国控点污染物浓度回归预测模型的参数说明

| | 自变量（自建点数据） | | | | | | | | | | | 拟合 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 常数项 | 湿度 | 温度 | 降水量 | 压强 | 风速 | PM2.5 | PM10 | CO | NO2 | SO2 | O3 | 优度 |
| PM2.5 国 | 70.30 | -14.35 | -2.21 | -1.26 | -8.33 | -1.70 | 70.16 | | | | | | 0.90 |
| 控点浓度 | 0.00 | 0.00 | 0.01 | 0.00 | 0.00 | 0.02 | 0.00 | | | | | | |
| PM10 国 | 100.18 | -51.49 | -8.68 | -6.78 | -34.10 | -10.73 | | 75.62 | | | | | 0.77 |
| 控点浓度 | 0.00 | | 0.00 | 0.00 | 0.00 | 0.00 | | 0.00 | | | | | |
| CO 国控 | 1.28 | -0.17 | -0.08 | 0.12 | -0.68 | -0.13 | | | 0.11 | 0.16 | 0.57 | -0.05 | 0.38 |
| 点浓度 | 0.00 | 0.00 | 0.04 | 0.00 | 0.00 | 0.00 | | | 0.00 | 0.00 | 0.00 | 0.06 | |
| NO2 国控 | 47.53 | -16.68 | -7.70 | 0.21 | -14.11 | -27.19 | | | 7.27 | 21.26 | 16.01 | 0.29 | 0.63 |
| 点浓度 | 0.00 | 0.00 | 0.00 | 0.70 | 0.00 | 0.00 | | | 0.00 | 0.00 | 0.00 | 0.76 | |
| SO2 国控 | 14.17 | -7.10 | 1.20 | 0.92 | -1.21 | -0.60 | | | -0.51 | 2.07 | 4.49 | -2.19 | 0.52 |
| 点浓度 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.01 | | | 0.01 | 0.00 | 0.00 | 0.00 | |
| O3 国控 | 40.00 | -11.73 | 34.47 | -0.54 | 12.81 | 18.50 | | | -31.60 | -21.77 | 6.00 | 19.31 | 0.69 |
| 点浓度 | 0.00 | 0.00 | 0.00 | 0.50 | 0.00 | 0.00 | | | 0.00 | 0.00 | 0.00 | 0.00 | |

备注：各污染物模型中第一行为各项自变量系数，第二行为对应显著性水平，空值对应自变量为非干扰因素

根据表 3，对比于浓度差的回归模型，PM2.5 模型的拟合优度有了明显提升，NO2 模型的拟合优度略提升了一点，PM10 模型的拟合优度没有变化，其他污染物模型的拟合优度下降。其中提升的 PM2.5 国控点浓度回归方程如公式（6）。

$$dc(PM_{2.5}) = 70.3 + 70.16pop(PM_{2.5}) - 2.21T - 14.35H - 1.7V - 8.33Ap - 1.26P \quad (6)$$

**2.支持向量机模型**

支持向量机 (SVM) 是 Cherkassky 等在统计学习理论基础上提出的一种机器学习方法，至今在金融、医学以及图像等领域均有成果。研究表明, SVM 在高维度的情况下对数据的分类和拟合能够产出较优的结果[9]。支持向量机分为线性支持向量机和非线性支持向量机，由于上述线性回归预测模型的结果个别较好、普遍较差，因此本文采用非线性支持向量机对两点浓度差的影响因素进行分析。

由于问题要求对“两尘四气”六种污染物进行浓度差分析，且六种污染物的 SVM 模型处理方式相同。因此，下文以 PM2.5 两点浓度差的 SVM 模型为例，利用 python 软件进行检验和计算。在该模型中，自变量和因变量的选择与回归预测模型相同，即因变量为两点浓度差，因变量为相关的天气因素和自建点的污染物浓度。为了消除量纲差异的影响，将数据进行 z-score 标准化。

由于 SVM 模型的主要参数是核函数和惩罚因子$C$，其参数的选择会对模型的性能产生极大的影响。因此，本文通过交叉验证法和网格搜索法来选取合适的核函数和惩罚因子。

**（1）PM2.5 两点浓度差 SVM 模型的核函数选择**

采用不同核函数将反映相同训练数据集的不同特性，常见的核函数有以下三种[10]，见表 15。

表 15　三种核函数表达式

| 函数 | 表达式 |
|---|---|
| 线性核函数 | $k(x,y)=x^t*y$ |
| 多项式核函数 | $k(x,y)=(ax^ty+c)^d$ |
| 径向基函数 | $k(x,y)=exp\left\{-\frac{\|x-y\|^2}{2\sigma^2}\right\}$ |

基于自建点中 PM2.5 的训练集数据，选择不同的核函数建立 SVM 模型，其预测的拟合优度、均方根误差、平均相对误差如表 16。

表 16　三种核函数的检验表

| 核函数 | 拟合优度 | 均方根误差 | 平均相对误差 |
|---|---|---|---|
| RBF 核函数 | 0.54 | 84.38 | 6.81 |
| 线性核函数 | 0.42 | 105.36 | 7.77 |
| 多项式核函数 | 0.48 | 94.45 | 7.43 |

根据表 16 的检验结果，PM2.5 两点浓度差的模型选择 RBF 核函数。

**（2）PM2.5 两点浓度差 SVM 模型的惩罚因子 C 分析**

惩罚因子的取值表示此数据集对超出误差的样本容忍程度。当$C \to \infty$时，模型便容易成为过拟合模型，提高模型的复杂程度但大大降低模型的泛化能力；当$C \to 0$时模型容易成为欠拟合模型，虽然提高模型模型的泛化能力，但往往因起泛化能力过强导致其模型拟合优度过低。

针对此问题，python 机器学习库中封装的网格搜索法(GridSearchCV)法和交叉验证法，不仅可以自动搜索最优惩罚因子 C 还可以防止模型过拟合或欠拟合，处理过程如图 11。

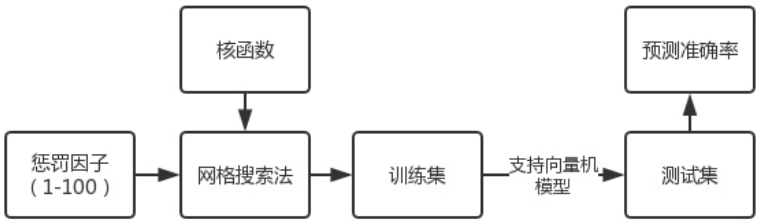



图 11　网格搜索法内在机制

利用 Python 软件，计算得到 PM2.5 浓度差 SVM 模型的惩罚因子 C 为 10，拟合优度为 0.54。

本题中支持向量机回归模型、交叉验证、网格搜索法的 Python 代码均见附录二。

**（3）PM10、CO、NO2、SO2 和 O3 的两点浓度差的 SVM 模型结果**

表 17　剩余污染物模型的结果说明表

| 因变量 | 选择核函数 | 惩罚因子 C | 拟合优度 |
|---|---|---|---|
| PM10 差值 | RBF | 40 | 0.77 |
| CO 差值 | RBF | 40 | 0.68 |
| SO2 差值 | RBF | 20 | 0.63 |
| NO2 差值 | RBF | 10 | 0.80 |
| O3 差值 | RBF | 6 | 0.90 |

从表 17 可以看出，O3、NO2 和 PM10 有较好的拟合效果，CO 和 SO2 的拟合效果一般。

**3.基于交叉验证的梯度提升回归树模型**

梯度提升回归树[11]是基于梯度提升的树模型。首先，提升指的是在优度低的结果上进行一种函数的优化方法，再$m$次循环后，选择减少残差最少的方向使损失函数在梯度方向最优；其次，梯度是指，每次更新数据时选择梯度下降的方向，从而确保得到最好的回归结果，最后每次模型在弱模型累加过程中，按照加权的方式来集成所有"弱回归树"，从而形成一个好的模型。其运作机理如图 12，理论推导过程见附录二。

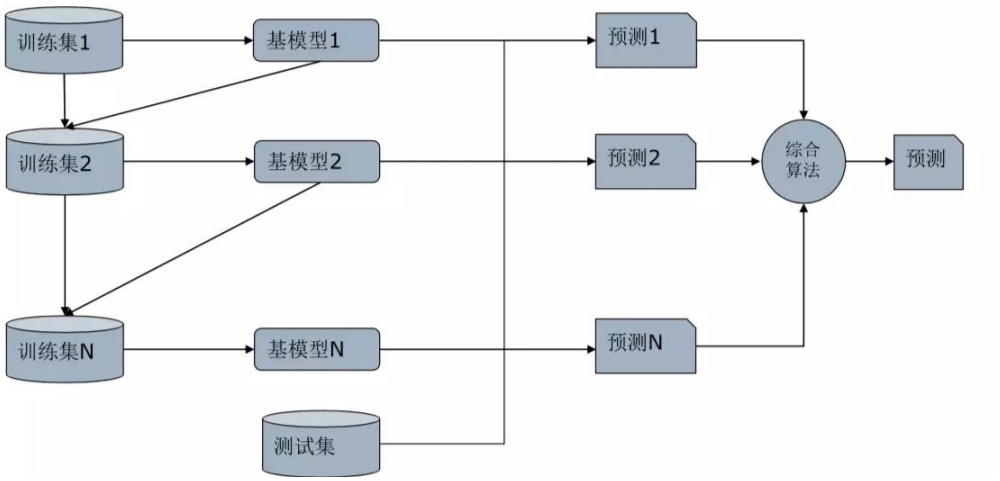



图 12　基于交叉验证的梯度提升回归树模型的机理图

利用 python 软件，进行编程计算得到结果，见表 18，代码见附录二。

表 18　梯度提升树的模型结果说明

| 梯度提升树 | 拟合优度 | 均方误差 | 平均绝对误差 | 方差得分 |
|---|---|---|---|---|
| PM2.5 差值 | 0.59 | 0.47 | 0.38 | 0.59 |
| PM10 差值 | 0.78 | 0.35 | 0.22 | 0.78 |
| CO 差值 | 0.73 | 0.39 | 0.26 | 0.73 |
| NO2 差值 | 0.78 | 0.35 | 0.22 | 0.78 |
| SO2 差值 | 0.74 | 0.38 | 0.28 | 0.74 |
| O3 差值 | 0.86 | 0.27 | 0.13 | 0.86 |

从表 18 可以看出，O3、PM10、NO2、SO2 和 CO 的拟合效果都较好，而 PM2.5 的拟合效果较差。

**6.2.3"两尘四气"误差模型的综合分析**

**1.影响因素及其影响程度**

零点漂移和量程漂移作为内部干扰因素，自主校准即可，其数据影响较小。以PM2.5 为例，零点漂移数值上造成的浓度差在 0.236-0.216，量程漂移在数值上造成的浓度差约为 0.173。

对气态污染物的交叉干扰和天气因素的影响进行分析，各个污染物模型中影响较大的前三项因素做以下罗列。固体颗粒物PM2.5 和PM10 模型皆为湿度、对应自建点浓度和气压；CO模型为气压、SO2 浓度和对应自建点浓度；NO2 模型为对应自建点浓度、风速和湿度；SO2 模型为湿度、O3 浓度和NO2 浓度；O3 模型为对应自建点浓度、温度和CO浓度。

**2.模型优化和对比**

由于上文中针对自建点“两尘四气”污染物选择了四种不同的预测模型，进行浓度差的分析，而拟合优度为衡量模型拟合效果的重要指标。因此汇总整理六种污染物、四大模型的拟合优度，并进行对比分析，如表 19。

表 19　模型对比分析表

| | 拟合优度 | | | | |
|---|---|---|---|---|---|
| 污染物种类 | 回归预测模型 | 调整后回归预测模型 | 支持向量机模型 | 梯度提升树模型 | 最终选择模型 |
| PM2.5 | 0.45 | **0.90** | 0.54 | 0.59 | 调整后的回归模型 |
| PM10 | 0.77 | 0.77 | 0.77 | **0.78** | 梯度提升树 |
| CO | 0.36 | 0.38 | 0.68 | **0.73** | 梯度提升树 |
| NO2 | 0.61 | 0.63 | **0.80** | 0.78 | 支持向量机 |
| SO2 | 0.61 | 0.52 | 0.63 | **0.74** | 梯度提升树 |
| O3 | 0.81 | 0.69 | **0.90** | 0.86 | 支持向量机 |

## 6.3 问题三模型建立和求解

当下，在环境空气质量数据监测上，主要有两种实现途径：一为高成本、长滞后、高准确定的国控点数据，二为低成本、实时监测、误差较大的自建点数据。为了将来实现环境空气质量数据的实时监测和播报，校准并推广自建点空气质量数据就显得尤为重要。

基于这样的目的，本小题在上文的基础上，进行了两方面的分析，一方面是基于问题二模型的，自建点空气质量数据的校准；另一方面是对全文数据校准的流程说明和展示，有助于进行校准方法的推广。

### 6.3.1 自建点空气质量校准模型

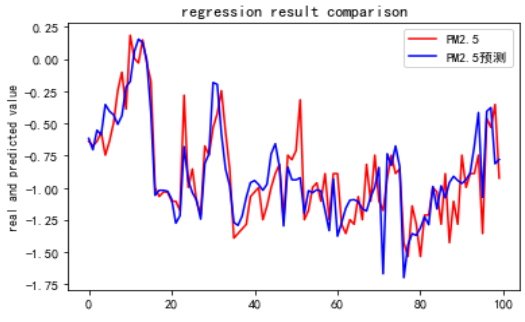
基于问题二中对国控点和自建点浓度差进行预测得到的模型，将其进行转换即可得到以国控点数据为因变量的自建点空气质量校准模型。即将问题二中的模型

$$\mathrm{f(x)} = dc(i) \quad i = CO, NO_2 SO_2, O_3 \tag{7}$$

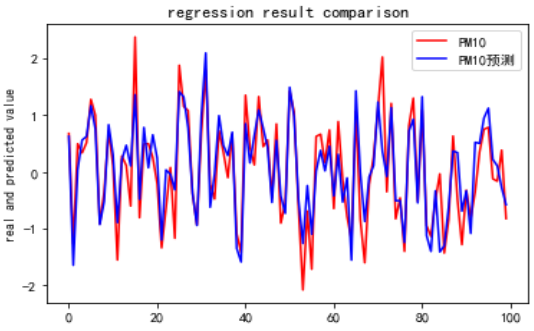
转换为

$$spop(i) = pop(i) + dc(i) \quad i = CO, NO_2 SO_2, O_3 \tag{8}$$

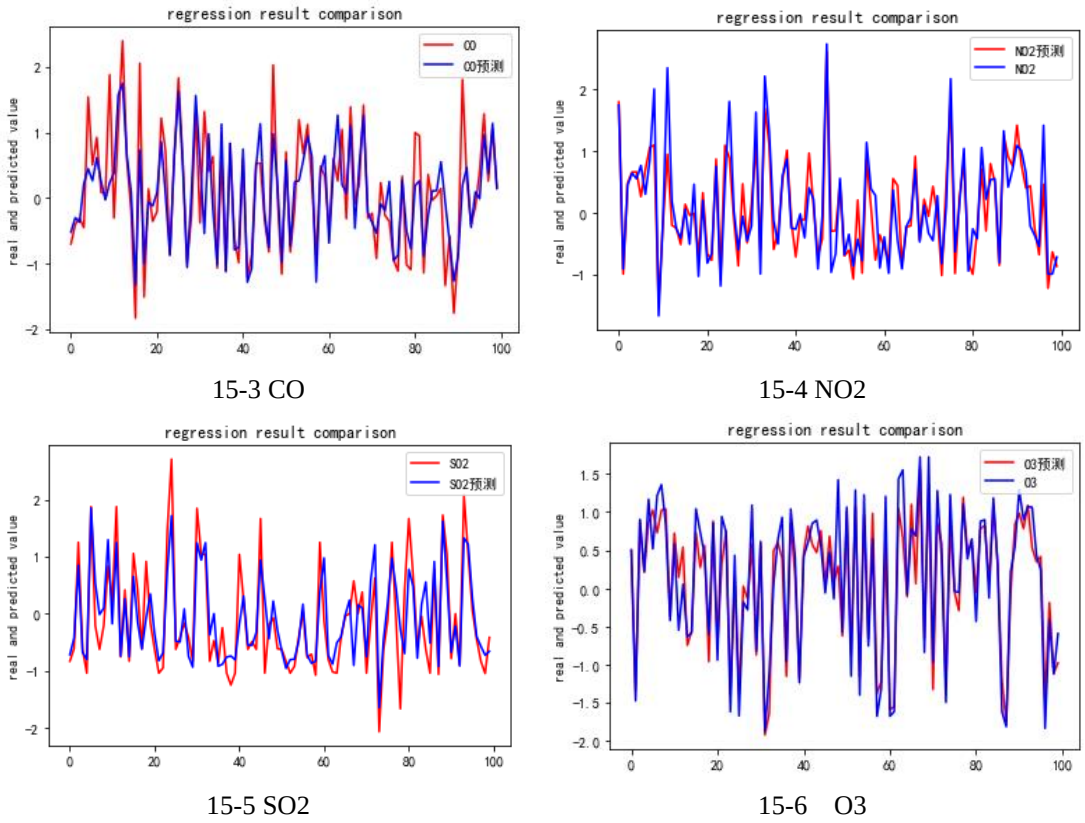
基于 25%的测试数据集，运行 python 程序，利用自建点空气质量校准模型得到校准后的污染物浓度数据，并将其与国控点的污染物浓度进行对比分析，结果如图 15。




15-1 PM2.5




15-2 PM10




15-3 CO　　15-4 NO2

15-5 SO2　　15-6　O3

图 15　测试集数据的误差分析图

从图 15 中的六张子图可以看出，测试集数据得到的自建点空气质量校准数据和对应的国控点数吻合度较高，模型误差小，模型效果较好。

### 6.3.2 空气质量数据的校准说明书

由于 6.3.1 中自建点空气质量校准模型最终结果较好，因此在该模型的基础上结合全文的分析，建立空气质量校准仪的机制说明书，见图 16。

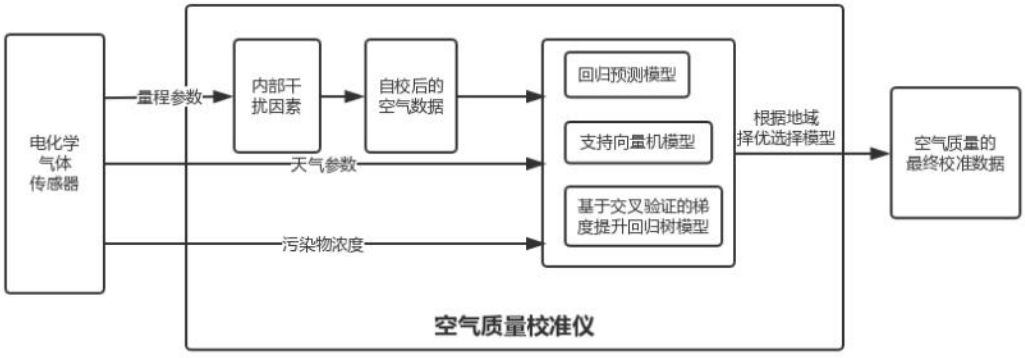



图 16　空气质量校准仪机制图

该空气质量校准仪的工作流程如下：

1.以电化学气体传感器自身的量程参数以及输出的天气参数、污染物浓度，作为天气质量校准仪的输入值；

2.空气质量校准仪对电化学气体传感器自身的量程参数进行自校，与天气参数、污染物浓度一起根据地域择优选择模型进行校准，模型包括：回归预测模型、支持向量机模型、基于交叉验证的梯度提升回归树模型；

3.输出空气质量的最终校准数据。

# 七、模型的灵敏性和稳健性分析

## 7.1 灵敏度分析

由于问题中所给的空气质量数据的地点，是某一地区的国家监测控制站点及其附近的自建点。文中天气因素是影响自建点空气质量校准模型的重要参数，而不同地理位置存在气象差异。本文有五个天气因素指标、六种污染物、四大预测模型。对应地，此处灵敏性分析选择温度指标、PM2.5 的浓度、调整后的回归预测模型进行灵敏度分析。

19 年 9 月 15 日中午，根据天气预报显示，中国南北温差约为 9 摄氏度。因此，本文将自建点温度连续两次上下浮动 5 摄氏度，其他数据保持不变来进行灵敏性分析，其分析结果如表 20。

表 20　温度变化的模型灵敏度说明

| 自建点的温度变化 | 拟合优度 | 拟合优度绝对变化 | 拟合优度相对变化 |
|---|---|---|---|
| 下调 10 度 | 0.9026 | -0.0012 | -0.1328% |
| 下调 5 度 | 0.9026 | -0.0012 | -0.1328% |
| 原始数据 | 0.9038 | — | — |
| 上调 5 度 | 0.9027 | -0.0011 | -0.1217% |
| 上调 10 度 | 0.9026 | -0.0012 | -0.1328% |

备注：由于数据差异小，此处保留小数点后四位

表 23 说明，温度变化对模型的影响不大，即模型对温度的灵敏性较低。因此本文所建立的空气质量校准模型在其他地域皆可使用，模型可推广性强。

## 7.2 稳健性分析

在交叉验证法的学习中，模型结果的好坏在一定程度上会受训练集和测试集的划分比例的影响。上文中数据集的划分比例是python软件默认的 0.75：0.25。为了明确划分比例对模型稳健性的影响，更改 0.5：0.5 和 0.9：0.1 两种较为极端的划分比例，以O3 的支持向量机模型为例，运行python程序，进行模型拟合优度的对比分析，如表 21。

表 21　数据划分比例的模型稳健性说明

| 数据集划分比例 | 拟合优度 | 拟合优度绝对变化 | 拟合优度相对变化 |
|---|---|---|---|
| 0.5：0.5 | 0.9032 | -0.0011 | -0.1216% |
| 0.75:0.25(正文模型使用） | 0.9043 | — | — |
| 0.9：0.1 | 0.9159 | 0.0116 | 1.2828% |

备注：由于数据差异小，此处保留小数点后四位

表 24 说明，数据集的划分比例的变化，对模型影响很小，模型稳健。因此，本文选择 0.75：0.25 的模型准确且有效。

# 八、模型拓展

1.构建生理预警系统。大气污染日益严重，污染物浓度超标会损害人们身体健康。因此以本文的空气质量较准模型为内核，对照专业文献中人体健康生理可承受的污染物浓度范围，嵌套智能健康手环系统，可以在大气污染物方面实现对人们健康的警示和护卫。

2.构建空气质量校准仪的自我预警系统，定期进行仪器的维护。计算历史某段时间内的国控点和自建点浓度差绝对值的最大值，并以此为阈值上限。如果自建点的空气质量数据误差超过阈值上限则预警，进行空气质量校准仪的维护。

3.由于传感器内部原因导致的零点漂移和量程漂移精准校准较困难，因此在仪器的日常运行中增加零点漂移的量程漂移的记录，有助于精准化校准。表22为零点漂移和量程漂移的记录模板。

表 22　零点漂移和量程漂移记录模板

**自动监测设备零点漂移、量程漂移校准原始记录**

测试人员：　　设备生产厂：

测试地点：　　设备型号、编号：

测试位置：　　设备原理：

标准物质浓度或校准器件响应值：　　污染物名称：

| 序号 | 日期 | 时间 | 计量单位（ug/m3、mg/m3、ppb……） | | | | | | | | 备注 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 零点读数 | | 零点漂移绝对误差 | % 满量程 | 上标校准读数 | | 量程漂移绝对误差 | % 满量程 | |
| | | | 起始 $Z_0$ | 最终 $Z_1$ | $Z = Z_1 - Z_0$ | | 起始 $S_0$ | 最终 $S_0$ | $S = S_1 - S_0$ | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |

## 九、模型评价及推广

### 9.1 模型的评价

**1.模型优点：**

（1）分阶段校准，逻辑清晰严谨。系统干扰因素根据仪器参数进行自较，外部环境干扰因素以国控点为因变量进行预测校准；

（2）多模型对比分析，校准外部环境干扰所导致的误差，更为精准，模型体系完整，增强模型可推广性，适用于大多数地区和污染物指标的校准。

（3）问题二的智能预测模型（支持向量机模型和基于交叉验证的梯度提升回归树模型）采取了交叉验证和网格搜索法，极大地提高了模型的优度。通过python封装的机器学习库进行操作，在一定程度上降低了编码的复杂性，使其易于操作。

**2.模型缺点：**

（零点漂移和量程漂移的校准没有使用模型进行分析，文章中只是利用仪器参数进行简单校准，可能仍然存在较小的误差。

### 9.2 模型的推广

1. 模型的地域可推广性强。在灵敏性分析中，以温度为例的天气因素的灵敏性较低，说明在不同地域不同温度下，模型依然可使用。

2. 预测和检验思想的可推广性强。大数据时代的数据信息充裕，在进行预测分析时，提前将数据划分成训练集和测试集，模型检验更加完整有效。

3. 多模型对比择优思想的可推广性强。相关文献中同一目的的实现模型较

多，建立系统的模型组，对比分析并择优，能实现模型效果的优化。

# 十、参考文献

[1]万方分析
http://miner.wanfangdata.com.cn/themeBootPage/explainAndstatistics.do?themeWord=零点漂移
[2]李汲峰,马建东.浅谈在后续检定中气体检测仪漂移检定的必要性[J].石油工业技术监督,2012,28(05):39-41.
[3]隋峰,孙倩芸,杨中元,高捷.烟气分析仪的交叉干扰及检定规程中存在的问题
[4]李勇,白云,李川.大气污染物 SO_2 预测模型研究综述[J].四川环境,2016,35(01):144-148.
[5]中华人民共和国国家标准
http://img.jingbian.gov.cn/upload/CMSjingbian/201806/201806210853050.pdf
[6]王翔宇,张引琼.基于 MATLAB 的时间序列异常检测方法探讨[J].电脑知识与技术,2012,8(04):866-872.
[7]张荣明、邹湘军、顾邦军、罗陆锋、周艳琼，基于探索性分析的时序数据研究[J].系统仿真学报，2006，18（增刊 2）：791-793.
[8]李飞,蒋敏兰.基于支持向量机回归的蛋鸡产蛋率预测模型[J].江苏农业科学,2019,47(13):249-252.
[9]化工仪器网：http://www.chem17.com/
[10]陈俏. 支持向量机应用于大气污染物浓度预测[D].西安科技大学,2010.]：
[11]江佳伟,符芳诚,邵蓥侠,崔斌.面向高维特征和多分类的分布式梯度提升树[J].软件学报,2019,30(03):784-798.

# 十一、附录

## 附录一：数据预处理附录

**1、利用 Python 数据预处理**

**筛选缺失值**

```
import pandas as pd
d1=pd.read_csv("D1.csv")
d1.isna()
```

**筛选异常值**

```
import numpy as np
#new_nums = list(set(deg)) #剔除重复元素
deg=d1["O3"]
#["PM2.5"],["PM10"],["CO"],["NO2"],["SO2"],["O3"]
mean = np.mean(deg)
var = np.var(deg)
percentile = np.percentile(deg, (25, 50, 75), interpolation='midpoint')
Q1 = percentile[0]#上四分位数
```

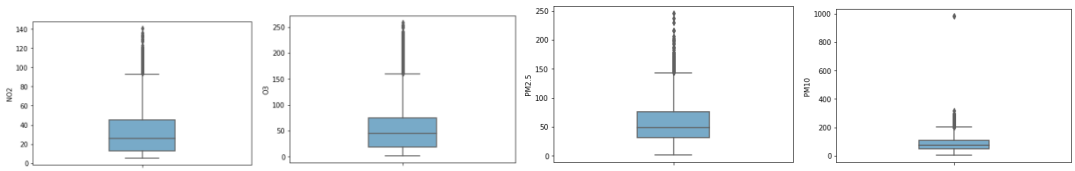
Q3 = percentile[2]#下四分位数
IQR = Q3 - Q1#四分位距
ulim = Q3 + 1.5*IQR#上限 非异常范围内的最大值
llim = Q1 - 1.5*IQR#下限 非异常范围内的最小值

**筛选重复值**

```
import pandas as pd
d2=pd.read_csv("D22.csv",encoding="gbk")
d2["shifou"]=d2.duplicated()
d2=d2[d2['shifou'].isin([False])]
d2.to_csv("D23.csv")
```

**箱型图**

```
import seaborn as sns
import pandas as pd
d1=pd.read_csv("D1.csv",encoding="gbk")
sns.boxplot(y="PM2.5",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="PM10",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="CO",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="NO2",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="SO2",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="O3",data=d1,width=0.3,palette="Blues")
```



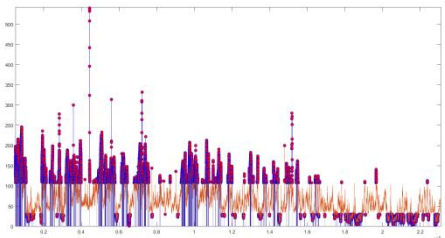

**2.利用 MATLAB 进行数据处理的合理性检验**

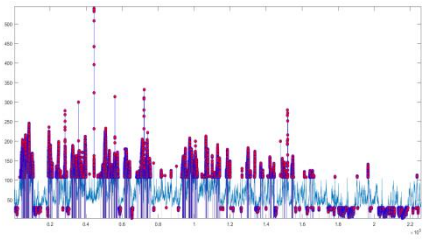
**以 PM2.5 为例**

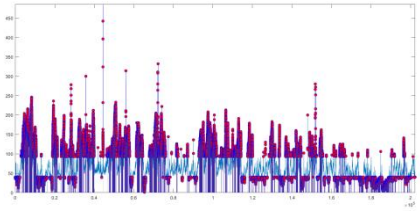
```
%求四分位差距和中位数
A=xlsread('附件 2.csv','A1:L234718');
x=A(:,1);
a=a1-a2;
b0=median(x);
b1=b0+a;
b2=b0-a;
%%找到异常点
bOriginal=NaN(length(x),1);
U=x-b1;
bOriginal(find(U>0))=x(find(U>0));
L=b2-x;
bOriginal(find(L>0))=x(find(L>0));
plot(x)
hold on
plot(bOriginal,'b-o','Markerface','r')
bar(bOriginal)
```

```
epose=find(~isnan(bOriginal));%%找出异常结果
Wrong=length(epose)
Coverage=length(epose)/length(x)%%求覆盖率
RIGHT=1-Coverage%%正确率

%%%方差法检测
A=xlsread('附件 2.csv','A1:L234718');
x=A(:,1);
c0=mean(x);
cs=std(x);
c1=c0+cs;
c2=c0-cs;
%%%%找异常点
bOriginal=NaN(length(x),1);
U=x-c1;
bOriginal(find(U>0))=x(find(U>0));
L=c2-x;
bOriginal(find(L>0))=x(find(L>0));
plot(x)
hold on
plot(bOriginal,'b-o','Markerface','r')
bar(bOriginal)
epose=find(~isnan(bOriginal));%%找出异常结果
Wrong=length(epose)
Coverage=length(epose)/length(x)%%求覆盖率
RIGHT=1-Coverage%%正确率

%%%5000-8000 的详细图
%%%%%%%%%%滑动四分位法
A=xlsread('附件 2.csv','A1:L234718');
x=A(:,1);Win=10;
Meanv=NaN(1,length(x));
DATUpper=NaN(1,length(x));
DATLower=NaN(1,length(x));
for i=(Win/2):(length(x)-(Win/2))
DATUpper=prctile(x((i-(Win/2)+1):i+(Win/2)),75);
DATLower=prctile(x((i-(Win/2)+1):i+(Win/2)),25);
End

%%%%找异常点
bOriginal=NaN(length(x),1);
U=x-DATUpper;
bOriginal(find(U>0))=x(find(U>0));
L=DATLower-x;
```

```
bOriginal(find(L>0))=x(find(L>0));
plot(x)
hold on
plot(bOriginal,'b-o','Markerface','r')
bar(bOriginal)
epose=find(~isnan(bOriginal));%%找出异常结果
Wrong=length(epose)
Coverage=length(epose)/length(x)%%求覆盖率
RIGHT=1-Coverage%%正确率

%%四分位法
A=xlsread('附件 2.csv','A1:L234718');
x=A(:,1);
a1=prctile(x,75);
a2=prctile(x,25);
%找到异常点
bOriginal=NaN(length(x),1);
U=x-a1;
bOriginal(find(U>0))=x(find(U>0));
L=a2-x;
bOriginal(find(L>0))=x(find(L>0));
plot(x)
hold on
plot(bOriginal,'b-o','Markerface','r')
bar(bOriginal)
epose=find(~isnan(bOriginal));%%找出异常结果
Wrong=length(epose)
Coverage=length(epose)/length(x)%%求覆盖率
RIGHT=1-Coverage%%正确率
```



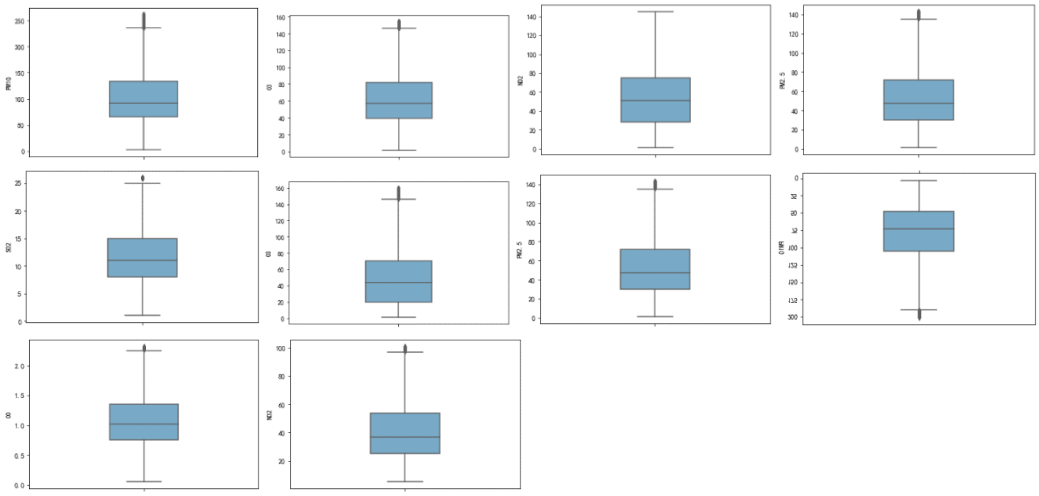




## 附录二：问题一附录

**1.利用 python 统计描述**

```
import seaborn as sns
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
from statsmodels.graphics.api import qqplot
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
d1=pd.read_csv("D12.csv",encoding="gbk")
d2=pd.read_csv("D2.csv",encoding="gbk")
cc1=d1.describe()
cc1=d2.describe()
sns.boxplot(y="PM2.5",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="PM10",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="CO",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="NO2",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="SO2",data=d1,width=0.3,palette="Blues")
sns.boxplot(y="O3",data=d1,width=0.3,palette="Blues")
```



**2.利用 Matlab 统计描述**

```
A=xlsread('匹配  全指标.xls','sheet1','A1:Q3516');
for i=1:1:17
X=A(:,i);
skewness(X)
kurtosis(X)
End
```

**3. 处理后国控点、自建点数据的匹配**

```
import pandas as pd
d1=pd.read_excel("tongyihoupiaoyi.xls",encoding="gbk")
data =d1.groupby("xiaoshi").mean()
```

```
d2=pd.read_csv("D12.csv",encoding="gbk")
data1=pd.merge(data,d2,on='xiaoshi')
data1=data1.drop("index",1)
data1.to_csv("pipeihoupiaoyi.csv")
```

## 附录三：问题二和问题三附录

**1.回归模型利用 minirab 软件直接计算**

**2.Python 支持向量机回归模型**

```
# -*- coding: utf-8 -*-
import pandas as pd
from sklearn.cross_validation import train_test_split
from sklearn.preprocessing import StandardScaler
import numpy as np
from sklearn.model_selection import    GridSearchCV
import matplotlib.pyplot as plt
d1=pd.read_excel("913yiwanshang.xlsx",encoding="gbk")
y=np.array(d1["chazhino2"])
X=np.array(d1[["fensu","yaqiang","jiangshuiliang","wendu","shidu","CO_x","NO2_x
","SO2_x","O3_x"]])
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 33, test_size =
0.25)
ss_X = StandardScaler()
ss_y = StandardScaler()
y_train=np.array(y_train).reshape(-1,1)
y_test=np.array(y_test).reshape(-1,1)
#X_train = ss_X.fit_transform(X_train)
#X_test = ss_X.transform(X_test)
#y_train = ss_y.fit_transform(y_train)
#y_test = ss_y.transform(y_test)
from sklearn.svm import SVR
linear_svr = SVR(kernel = 'linear')
linear_svr.fit(X_train, y_train)
linear_svr_y_predict = linear_svr.predict(X_test)
poly_svr = SVR(kernel = 'poly')
poly_svr.fit(X_train, y_train)
poly_svr_y_predict = poly_svr.predict(X_test)
rbf_svr = SVR(kernel = 'rbf')
rbf_svr.fit(X_train, y_train)
rbf_svr_y_predict = rbf_svr.predict(X_test)
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
print ('R-squared value of linear SVR is: ', linear_svr.score(X_test, y_test))
print ('The mean squared error of linear SVR is: ',
mean_squared_error(ss_y.inverse_transform(y_test),
ss_y.inverse_transform(linear_svr_y_predict)))
```

```
print ('The mean absolute error of lin SVR is: ',
mean_absolute_error(ss_y.inverse_transform(y_test),
ss_y.inverse_transform(linear_svr_y_predict)))
print ('R-squared of ploy SVR is: ', poly_svr.score(X_test, y_test))
print ('the value of mean squared error of poly SVR is: ',
mean_squared_error(ss_y.inverse_transform(y_test),
ss_y.inverse_transform(poly_svr_y_predict)))
print ('the value of mean ssbsolute error of poly SVR is: ',
mean_absolute_error(ss_y.inverse_transform(y_test),
ss_y.inverse_transform(poly_svr_y_predict)))
print ('R-squared of rbf SVR is: ', rbf_svr.score(X_test, y_test))
print ('the value of mean squared error of rbf SVR is: ',
mean_squared_error(ss_y.inverse_transform(y_test),
ss_y.inverse_transform(rbf_svr_y_predict)))
print ('the value of mean ssbsolute error of rbf SVR is: ',
mean_absolute_error(ss_y.inverse_transform(y_test),
ss_y.inverse_transform(rbf_svr_y_predict)))
#%画图 PM2.5 /15  00：14
plt.plot(rbf_svr_y_predict[:100,],c="r")
plt.plot(y_test[:100,],c="b")
plt.xlabel("测试集")
plt.ylabel("标准化处理后的数据")
plt.title("PM2.5 数据预测")
```

**Python 交叉验证**

```
from sklearn.model_selection import cross_val_score
cross_val_score(rbf_svr, X_train, Y_train,cv=5,scoring='neg_mean_absolute_error')
```

**Python 网格搜索法**

```
import pandas as pd
from sklearn.cross_validation import train_test_split
from sklearn.preprocessing import StandardScaler
import numpy as np
from sklearn.model_selection import  GridSearchCV
d1=pd.read_excel("913yiwanshang.xlsx",encoding="gbk")
y=np.array(d1["chazhio3"])
X=np.array(d1[["fensu","yaqiang","jiangshuiliang","wendu","shidu","CO_x","NO2_x
","SO2_x","O3_x"]])
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 33, test_size =
0.25)
ss_X = StandardScaler()
ss_y = StandardScaler()
y_train=np.array(y_train).reshape(-1,1)
y_test=np.array(y_test).reshape(-1,1)
X_train = ss_X.fit_transform(X_train)
```

```
X_test = ss_X.transform(X_test)
y_train = ss_y.fit_transform(y_train)
y_test = ss_y.transform(y_test)
parameters = {'kernel':('linear', 'rbf',"poly"), 'C':[1, 2, 4,6,9,10,20,40,80,100]}
from sklearn import svm
from sklearn import grid_search
from sklearn.datasets import load_iris
iris = load_iris()
svr = svm.SVR()
clf = grid_search.GridSearchCV(svr, parameters)
clf.fit(X_train, y_train)
print (clf.best_params_)       # 最好的参数
```

**3.Python 基于交叉验证的梯度提升树回归模型**

```
import numpy as np    # numpy 库
from sklearn.linear_model import BayesianRidge, LinearRegression, ElasticNet    # 批量导入要实现的回归算法
from sklearn.svm import SVR    # SVM 中的回归算法
from sklearn.ensemble.gradient_boosting import GradientBoostingRegressor    # 集成算法
from sklearn.model_selection import cross_val_score    # 交叉检验
from sklearn.metrics import explained_variance_score, mean_absolute_error, mean_squared_error, r2_score    # 批量导入指标算法
import pandas as pd    # 导入 pandas
import matplotlib.pyplot as plt    # 导入图形展示库
from sklearn.model_selection import train_test_split #导入切分数据
from sklearn.preprocessing import StandardScaler #导入标准化

# 数据准备
d1=pd.read_excel("913yiwanshang.xlsx",encoding="gbk")    #导入数据集
y=np.array(d1["chazhiPM2.5"])    #选择 X 值
X=np.array(d1[["fensu","yaqiang","jiangshuiliang","wendu","shidu","CO_x","NO2_x","SO2_x","O3_x"]]) #选择 Y 值
X_train, X_test, Y_train, Y_test = train_test_split(X, y, test_size=0.25)    #切分数据级
ss_X = StandardScaler() #标准化
ss_y = StandardScaler()#标准化
Y_train=np.array(Y_train).reshape(-1,1) #转置    不然会报错  一个 warning
Y_test=np.array(Y_test).reshape(-1,1)#转置    不然会报错  一个 warning
X_train = ss_X.fit_transform(X_train)    #模型
X_test = ss_X.transform(X_test)    #模型
Y_train = ss_y.fit_transform(Y_train) #模型
Y_test = ss_y.transform(Y_test) #模型
```

```
# 训练回归模型
n_folds = 8  # 设置交叉检验的次数
model_br = BayesianRidge()  # 建立贝叶斯岭回归模型对象
model_lr = LinearRegression()  # 建立普通线性回归模型对象
model_etc = ElasticNet()  # 建立弹性网络回归模型对象
model_svr = SVR()  # 建立支持向量机回归模型对象
model_gbr = GradientBoostingRegressor()  # 建立梯度增强回归模型对象
model_names = ['BayesianRidge', 'LinearRegression', 'ElasticNet', 'SVR', 'GBR']  # 不同模型的名称列表
model_dic = [model_br, model_lr, model_etc, model_svr, model_gbr]  # 不同回归模型对象的集合
cv_score_list = []  # 交叉检验结果列表
pre_y_list = []  # 各个回归模型预测的 y 值列表
for model in model_dic:  # 读出每个回归模型对象
    scores = cross_val_score(model, X_train, Y_train, cv=n_folds)  # 将每个回归模型导入交叉检验模型中做训练检验
    cv_score_list.append(scores)  # 将交叉检验结果存入结果列表
    pre_y_list.append(model.fit(X_train, Y_train).predict(X_test))  # 将回归训练中得到的预测 y 存入列表


# 模型效果指标评估
n_samples, n_features = X.shape  # 总样本量,总特征数
model_metrics_name = [explained_variance_score, mean_absolute_error,
mean_squared_error, r2_score]  # 回归评估指标对象集
model_metrics_list = []  # 回归评估指标列表
for i in range(5):  # 循环每个模型索引
    tmp_list = []  # 每个内循环的临时结果列表
    for m in model_metrics_name:  # 循环每个指标对象
        tmp_score = m(Y_test, pre_y_list[i])  # 计算每个回归指标结果
        tmp_list.append(tmp_score)  # 将结果存入每个内循环的临时结果列表
    model_metrics_list.append(tmp_list)  # 将结果存入回归评估指标列表
df1 = pd.DataFrame(cv_score_list, index=model_names)  # 建立交叉检验的数据框
df2 = pd.DataFrame(model_metrics_list, index=model_names, columns=['ev', 'mae',
'mse', 'r2'])  # 建立回归指标的数据框
print ('samples: %d \t features: %d' % (n_samples, n_features))  # 打印输出样本量和特征数量
print (70 * '-')  # 打印分隔线
print ('cross validation result:')  # 打印输出标题
print (df1)  # 打印输出交叉检验的数据框
print (70 * '-')  # 打印分隔线
print ('regression metrics:')  # 打印输出标题
print (df2)  # 打印输出回归指标的数据框
```

```
print (70 * '-')  # 打印分隔线
print ('short name \t full name')  # 打印输出缩写和全名标题
print ('ev \t explained_variance')
print ('mae \t mean_absolute_error')
print ('mse \t mean_squared_error')
print ('r2 \t r2')
print (70 * '-')  # 打印分隔线

# 模型效果可视化
plt.figure()  # 创建画布
plt.plot(np.arange(X_test.shape[0]),Y_test, color='k', label='true y')  # 画出原始值的曲线
color_list = ['r', 'b', 'g', 'y', 'tan']  # 颜色列表
linestyle_list = ['-', '.', 'o', 'v', '*']  # 样式列表
for i, pre_y in enumerate(pre_y_list):  # 读出通过回归模型预测得到的索引及结果
    plt.plot(np.arange(X_test.shape[0]), pre_y_list[i], color_list[i],
label=model_names[i])  # 画出每条预测结果线
plt.title('regression result comparison')  # 标题
plt.legend(loc='upper right')  # 图例位置
plt.ylabel('real and predicted value')  # y 轴标题
plt.show()  # 展示图像

#取前 100 的数据
hundred=[]  #取 qian yibai
for i in range(5):
    hundred.append(pre_y_list[i][:100,])
plt.figure()  # 创建画布
plt.plot(np.arange(X_test[:100,].shape[0]),Y_test[:100,], color='k', label='true y')  # 画出原始值的曲线
color_list = ['r', 'b', 'g', 'y', 'tan']  # 颜色列表
linestyle_list = ['-', '.', 'o', 'v', '*']  # 样式列表
for i, pre_y in enumerate(hundred):  # 读出通过回归模型预测得到的索引及结果
    plt.plot(np.arange(X_test[:100,].shape[0]), hundred[i], color_list[i],
label=model_names[i])  # 画出每条预测结果线
plt.title('regression result comparison')  # 标题
plt.legend(loc='upper right')  # 图例位置
plt.ylabel('real and predicted value')  # y 轴标题
plt.show()  # 展示图像
```

**4.问题二基于交叉验证的梯度提升回归树模型的理论推导说明**

通过每一个样本进行训练，迭代过程中都会产生一个较弱的回归树，然后将这些弱回归树进行累加，则，此模型便可描述为：

$$f_0(x) = \text{argmin} \sum_{m=1}^{M} T(x;\theta_m)$$

其中$T(x;\theta_m)$是回归树，$\theta_m$是每一棵回归树的参数，M 是树的棵树。

然后按照向前迭代的方式，计算每一棵回归树的误差，再沿着最小误差的回归树方向进行再次迭代，计算 t 次迭代的梯度：

$$r_{tm} = \frac{\partial T(x, f_{M-1}(x_i))}{\partial f_{M-1}(x_i)}, i = 1,2,\cdots,m$$

利用$(x_i, r_{tm})$,拟合第 t 棵$f_M(x)$回归树，而其对应的叶子结点区域为$R_{tj}, j =$ 1，2，…，$j$，其中 J 为回归树的叶子结点的个数

对叶子结点区域$j$=1,2,…，$j$，计算最佳拟合值：

$$c_{tj} = argmin \sum_{x_i \in R_{tj}} T(x, f_{M-1}(x_i) + \theta_m)$$

随着迭代次数，更新优回归树：

$$h_t(x) = h_{t-1}(x) + \sum_{j=1}^{J} c_{tj}\, I(x \in R_{tj})$$

其中 I 是最小误差值

最后得到最优回归树：

$$F(x) = h_0(x) + \sum_{t=1}^{T}\sum_{j=1}^{J} c_{tj} I(x \in R_{tj})$$