中国研究生创新实践系列大赛
“华为杯”第十八届中国研究生
数学建模竞赛

学　　校	西安电子科技大学
参赛队号	**21107010054**
队员姓名	**1.吴俊**
	2.夏龙云
	3.麦兴国

中国研究生创新实践系列大赛

“华为杯”第十八届中国研究生
数学建模竞赛

题 目　　空气质量预报二次建模

摘　　要：

建立空气质量预报模型，提前获知可能发生的大气污染过程并采取相应控制措施，是减少大气污染对人体健康和环境等造成的危害，提高环境空气质量的有效方法之一。然而目前常用 WRF-CMAQ 模拟空气质量一次预报模型受制于模拟的气象场以及排放清单的不确定性等因素预报结果并不理想。

本文针对以上问题在一次预报模型的基础上结合更多的数据源进行再建模，建立**二次预报模型**，以提高预报的准确性，并对模型的性能进行评价。

针对问题 1：问题 1 是一个**数据处理问题**，采用**特定的数据整定方法**，对数据样本进行预处理。本文对服从正态分布的数据依据 **3σ 准则**检测异常数据并剔除，对于不服从正态分布的数据依据**箱形图**检测异常数据并剔除，对异常值依据特定方法取附近的相近值取平均值进行填充，对处理后仍然异常的异常值扩大取值范围，直至异常值少到一定标准，完成数据预处理。最后根据附录中提供的计算方法，计算监测点 A 从 2020 年 8 月 25 日到 8 月 28 日每天实测的 AQI 和首要污染物。

针对问题 2：问题 2 是一个**相关性问题**，要求根据气象条件对于污染物浓度的影响程度，对气象条件进行合理分类，并阐述气象条件的特征。本文首先构建**相关性模型**计算处六种污染物和五种气象条件之间的 **Pearson 相关系数**，然后结合对于**一次污染物与二次污染物之间的相关性分析**以及**气象条件变化趋势**对于污染物浓度变化趋势的分析，将气象条件从污染物沉降、扩散以及产生三个方面分为六类，并阐述各类气象条件的特征。

针对问题 3：问题 3 是一个预测问题，在问题 1 和问题 2 对气象条件的特征以及对污染物浓度的影响因素进行充分研究的基础上，建立同时适用于 A、B、C 三个监测点的**二次预报数学模型**（A、B、C 三个监测点互不影响）。本文首先进行异常数据处理；然后对处理后的数据进行**周期性分析**，得出数据样本具有周期性的结论；然后**使用 LSTM 建立预测网络**，使用数据样本进行训练，得到预测模型，并最终使用该模型预测出监测点 A、B、C 在 2021 年 7 月 13 日至 7 月 15 日 6 种常规污染物的单日浓度值，并使用问题 1 的方法计算相应的 AQI 和首要污染物。经过**与附件中提供的实测值进行对比，预测模型的预测效果良好**。

针对问题 4：问题 4 是在问题 3 的二次预测模型基础上，建立一个区域协同二次预测模型。本问题主要考虑问题 2 中总结**风速、风力、风向**对于污染物扩散的影响程度以及**相邻监测点之间的相对位置。**本文首先针对相邻监测点之间的相对位置、风速、风力、风向对风力进行**受力分析，预测本监测点污染物对附近监测点产生影响的时间**，建立一个**污染物扩散模型**。然后依据污染物扩散模型分别预测出各个监测点对于其他监测点的污染物浓度影响数据。最后，在污染物扩散模型的基础上，建立一个**二层 LSTM 预测网络**，通过本监测点的污染物浓度预测和其他监测点对于本监测点污染物浓度影响预测进行训练，得到预测模型。使用该二层 LSTM 预测模型计算出各个监测点的污染物浓度，并使用问题 1 的方法计算相应的 AQI 和首要污染物。最终，经过与附件中提供的实测值进行对比，得出与

问题 3 的模型相比，**协同预报模型能提升监测点的污染物浓度预报准确度的结论**。

关键词：二次预报模型　LSTM Pearson 相关系数 污染物扩散模型 二层 LSTM 预测网络

目录

一、问题重述

1.1 问题背景

大气污染是指大气当中的污染物浓度达到有害程度，危害人体或动物健康、导致生态系统破坏和人类生存环境受到威胁的一种环境污染[1]。大气污染物是指 由于人类活动或自然过程排入大气的并对环境或人产生有害影响的那些物质。大气污染物由人为源或者天然源进入大气（输入），参与大气的循环过程，经过一定的滞留时间之后，又通过大气中的化学反应、生物活动和物理沉降从大气中去除（输出）。如果输出的速率小于输入的速率，就会在大气中相对集聚，造成大气中某种物质的浓度升高。当浓度升高到一定程度时，就会直接或间接地对人、生物或材料等造成急性、慢性危害。大气污染物既包括粉尘、烟、雾等小颗粒状的污染物，也包括二氧化碳、一氧化碳等气态污染物。

大气污染极具危害性[2]，它不仅能够破坏自然环境、影响生 态循环，对人类健康、动植物生长、农业生产和全球环境等都 会造成很大的伤害。为此，以下将分析大气污染的危害性。

WRF 是为大气研究和业务预报应用而设计的下一代中尺度数值天气预报系统。它具有两个动态核心、一个数据同化系统和一个支持并行计算和系统可扩展性的软件架构。WRF 模型具有三维、四维同化系统。可通过机器并行运算，极大扩展了计算能力。

CMAQ 模型是一种数值空气质量模型，它依赖于科学首要原则预测空气中气体和颗粒物的浓度，以及这些污染物回到地球表面的沉积。CMAQ 模型主要由化学传输模块，初始值模块，边界值模块，气象化学接口模块以及光化学模块组成。

在 WRF-CMAQ 模型中，WRF 和 CMAQ 被同时集成，来自 CMAQ 的信息，如气溶胶浓度，被传递到 WRF，以便化学反应可以影响天气。具体来说，WRF-CMAQ 模型为用户提供了将气溶胶光学特性传递给 WRF（气溶胶直接辐射效应）中的辐射模块的选项，并调整 WRF 和 CMAQ 的调用频率，以平衡模拟的准确性和计算成本。但 WRF-CMAQ 模拟计算空气质量的过程中需要详细的大气排放物清单，现有的模拟场不能完全提供所有排放物完整合理的生成机理，这使得模型计算的准确性大打折扣，因此我们需要在 WRF-CMAQ 等一次建模的基础上进行二次建模，提高模型预测的准确率。

1.2 需要解决的问题

基于一次预报数据及实测数据（见附件）进行空气质量预报二次数学建模，完成以下四个问题。请注意，实际工作中会遇到数据为空值或异常值的情况（见附录），故要求建立的模型具有一定的鲁棒性。

问题 1：使用附件 1 中的数据，按照附录中的方法计算监测点 A 从 2020 年 8 月 25 日到 8 月 28 日每天实测的 AQI 和首要污染物，将结果按照附录“AQI 计算结果表”的格式放在正文中。

问题 2：在污染物排放情况不变的条件下，某一地区的气象条件有利于污染物扩散或沉降时，该地区的 AQI 会下降，反之会上升。使用附件 1 中的数据，根据对污染物浓度的影响程度，对气象条件进行合理分类，并阐述各类气象条件的特征。

问题 3：使用附件 1、2 中的数据，建立一个同时适用于 A、B、C 三个监测点（监测点两两间直线距离>100km，忽略相互影响）的二次预报数学模型，用来预测未来三天 6 种常规污染物单日浓度值，要求二次预报模型预测结果中 AQI 预报值的最大相对误差应尽量小，且首要污染物预测准确度尽量高。并使用该模型预测监测点 A、B、C 在 2021 年 7 月 13 日至 7 月 15 日 6 种常规污染物的单日浓度值，计算相应的 AQI 和首要污染物，将结果依照附录“污染物浓度及 AQI 预测结果表”的格式放在论文中。

问题 4：相邻区域的污染物浓度往往具有一定的相关性，区域协同预报可能会提升空气质量预报的准确度。如图 4，监测点 A 的临近区域内存在监测点 A1、A2、A3，使用附件 1、3 中的数据，建立包含 A、A1、A2、A3 四个监测点的协同预报模型，要求二次模型预测结果中 AQI 预报值的最大相对误差应尽量小，且首要污染物预测准确度尽量高。使用该模型预测监测点 A、A1、A2、A3 在 2021 年 7 月 13 日至 7 月 15 日 6 种常规污染物的单日浓度值，计算相应的 AQI 和首要污染物，将结果依照附录“污染物浓度及 AQI 预测结果表”的格式放在论文中。**并讨论：**与问题 3 的模型相比，协同预报模型能否提升针对监测点 A 的污染物浓度预报准确度？说明原因。

二、模型假设与符号说明

2.1 模型假设

假设 1：污染物浓度仅受温度、湿度、气压、风速、风向五个因素影响，且污染物排放量不变；

假设 2：监测点 A、B、C 两两间直线距离>100km，污染物浓度互相不影响；

假设 3：协同预报模型中仅存在监测点 A、A1、A2、A3，不会受到外来污染物扩散；

假设 4：协同预报模型中污染物从一个监测点扩散到另外一个监测点的过程是稳定的，不会发生沉降、扩散、改变方向等状况；

假设 5：协同预报模型中一个监测点的来自不同方向的污染物其浓度是简单的叠加，不考虑其他状况。

2.2 符号说明

符号	符号说明
σ	样本标准差
μ	样本均值
x_i	样本数据值
$IAQI_p$	污染物P的空气质量分指数，结果进位取整数
C_p	污染物P的质量浓度值
$BP_{Hi,}BP_{L0}$	与C_p相近的污染物浓度限值的高位值与低位值
$IAQI_{Hi,}IAQI_{L0}$	与BP_{Hi},BP_{Lo}对应的空气质量分指数
z_i	Z-score 的归一化数据
V	风速
β	风向与正北方向的夹角
d	A1-A2 之间的距离

γ	A1-A2 与正北方向的夹角
α	风向与 A1-A2 之间的夹角

三、问题 1 模型的建立与求解

3.1 问题分析

问题 1 要求使用附件 1 中的数据，按照附录的方法计算监测点 A 从 2020 年 8 月 25 日到 8 月 28 日每天实测的 AQI 和首要污染物。

根据问题要求，将附件 1《监测点 A 空气质量预报基础数据》中提供的“监测点 A 逐日污染物浓度实测数据”作为待处理的原始数据，进行预处理。原始数据样本包括 2019 年 4 月 16 日至 2021 年 7 月 12 日的六种污染物（SO_2、NO_2、PM10、PM2.5、O_3、CO ）浓度实测数据。虽然问题 1 仅要求计算监测点 A 从 2020 年 8 月 25 日到 8 月 28 日每天实测的 AQI 和首要污染物，但是样本容量过小，仅从这几天的样本中无法确定样本是否存在异常数据，因此需要基于大的样本进行异常数据分析，并进行数据预处理，并将处理后的数据分别加入附件 1 中，以供其他问题继续研究使用。将预处理后的数据按照附录的方法进行计算，即可得到监测点 A 从 2020 年 8 月 25 日到 8 月 28 日每天实测的 AQI 和首要污染物。

3.2 数据预处理

为了得到数据预处理的结果，需要对样本 2019 年 4 月 16 日至 2021 年 7 月 12 日（包括 2020 年 8 月 25 日到 8 月 28 日）的六种污染物浓度实测数据进行异常数据数据整定，具体操作如下：

3.2.1 异常数据剔除

首先使用 Pyhton 编程，判断六种污染物实测数据是否服从正态分布，并绘制样本数据箱盒图，并根据箱盒图结果以及实际情况对异常数据类型进行总结。样本异常数据可以分为过大值，过小值，空值，负值以及突出值。

对于服从正态分布的样本，根据拉依达准则（3σ 准则），对大于 3σ 的数据剔除。在正态分布中 σ 代表标准差，μ 代表均值。标准差和均值的计算方法如下：

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_i - \mu\right)^2}$$

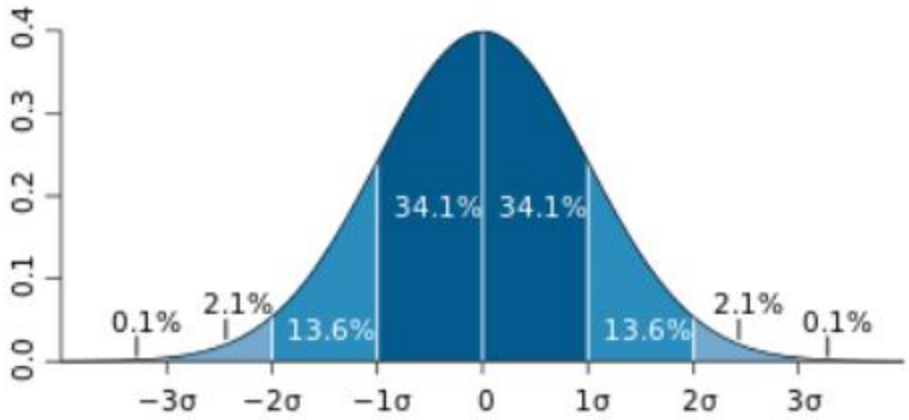


图 3.1 正态分布数值分布情况

如图 3.1 所示，数值分布在（μ-σ,μ+σ)中的概率为 0.6827，数值分布在（μ-2σ,μ+2σ)中的概率为 0.9545，数值分布在（μ-3σ,μ+3σ)中的概率为 0.9973。可以认为，数据的取值几乎全部集中在（μ-3σ,μ+3σ)区间内，超出这个范围的可能性仅占不到 0.3%。最后比较样本数据的每个值，是否大于标准差的 3 倍。如果存在大于 3 倍标准差的数据，该数据被剔除。

对于不服从正态分布的样本，根据箱盒图监测异常数据并剔除。箱型图相较拉依达准则（3σ 准则）而言不需事先假定数据服从特定的分布，没有对数据做限制要求，真实直观低呈现数据本来面貌，且箱型图以四分位数和四分位距作为基础，具有一定耐抗性，箱形图识别异常值的结果比较客观。

如图所示，箱盒图包含六个数据节点分别为上四分位数(Q1)、中位数(Q2)、下四分位数(Q3)、四分位距（IQR = Q3-Q1）、上限（Q3+1.5IQR）、下限（Q1－1.5IQR）。异常值，被定义为小于 Q1-1.5IQR 或大于 Q3+1.5IQR 的值，将属于异常的值进行剔除。

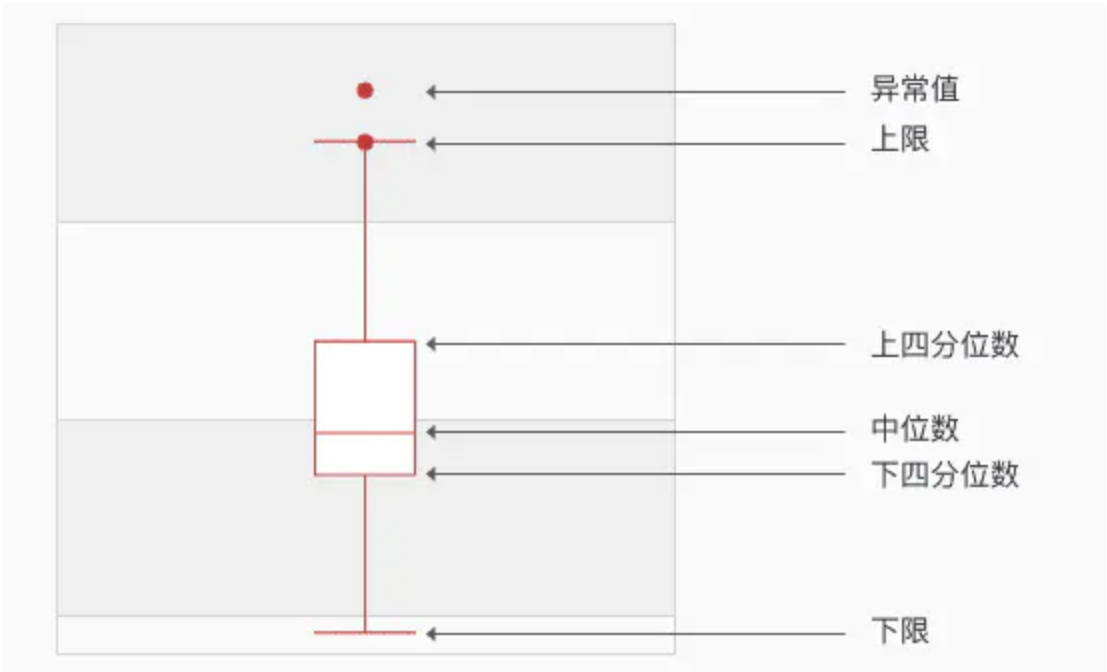


图 3.2 箱型图数据特征示意图

3.2.2 异常数据填充

将异常值剔除后，样本数据中就只剩下数据缺失异常，如果不对数据进行填充，则会造成较大的计算误差。

本次数据预处理的异常数据填充策略为，计算缺失值前后各两天的数据的平均值。但是仅处理一次并不能保证处理后的数据不再异常，因为样本中的数据有可能连续异常多天；因此对全部异常数据处理一遍之后，需要判断填充数据是否仍为异常值，如果异常则继续处理异常数据。在第一次处理异常数据之后，取前后各二天的数据的以及去年同一时期当天以及前后各二天的数据的平均值，对全部异常数据处理一遍之后，判断填充数据是否仍为异常值，如果异常则增加天数取平均值，直到不再异常为止。

3.2.3 异常数据处理效果

异常数据处理前、后箱型图分别如图 3.3、3.4 所示，异常数据处理效果明显。

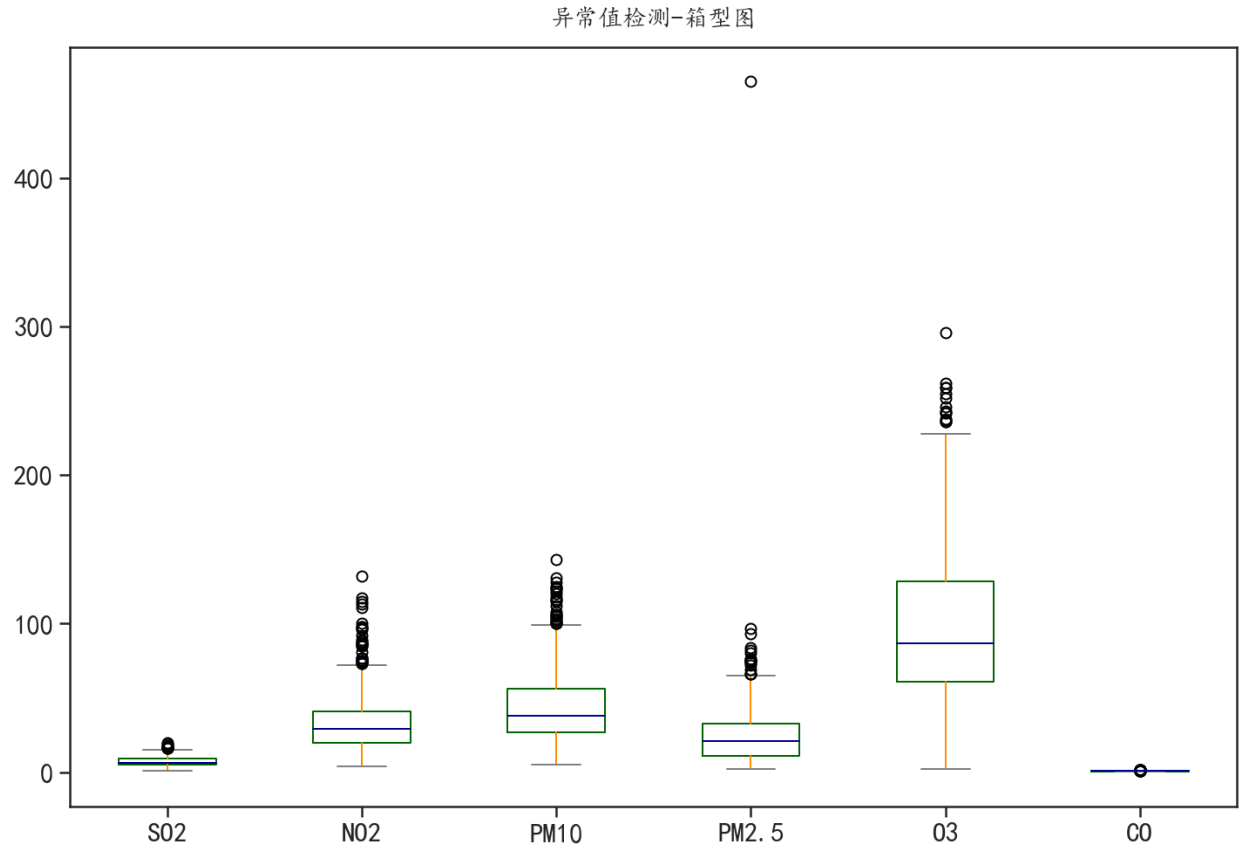


图 3.3 异常数据处理前箱型图

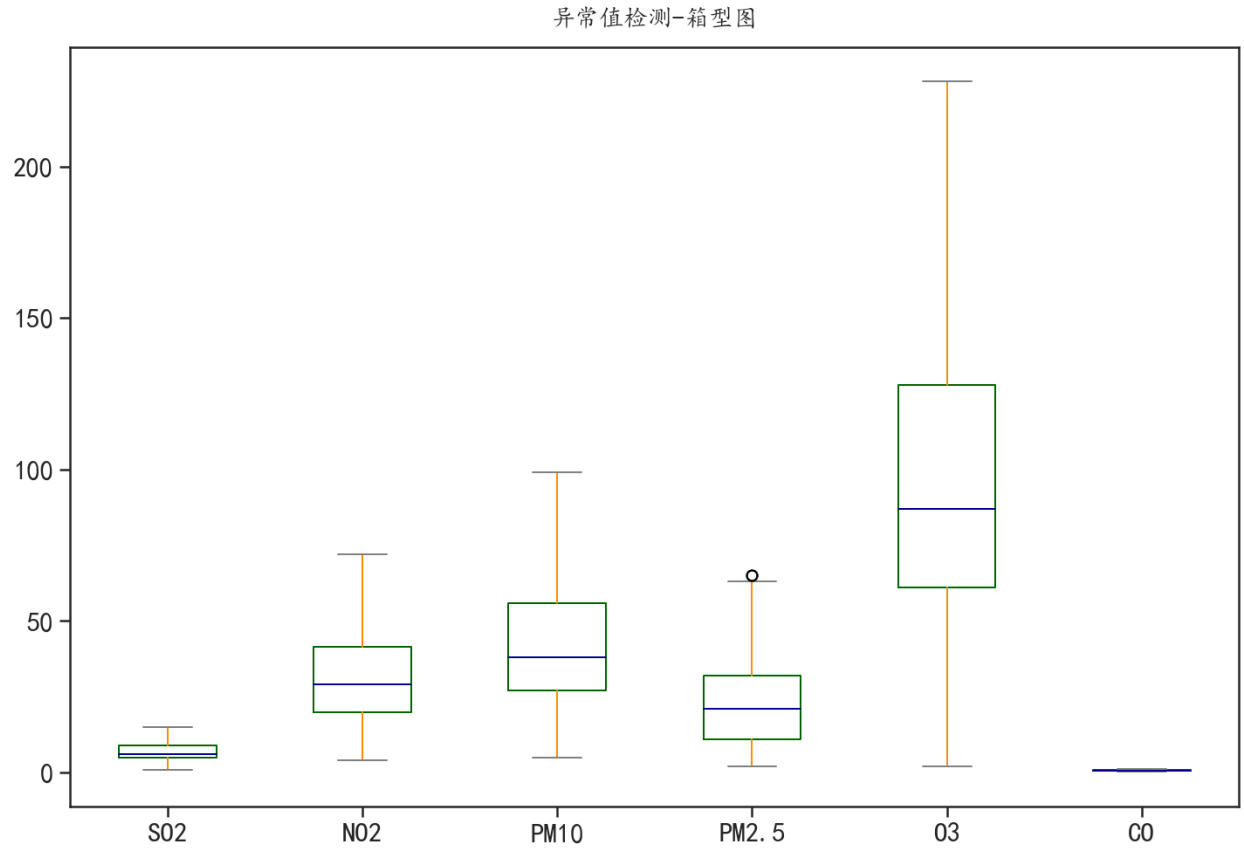


图 3.4 异常数据处理前箱型图

3.3 问题求解

使用 Python 编程，首先分别计算六种污染物的空气质量分指数（IAQI），其计算公式如下：

$$IAQI_P = \frac{IAQI_{Hi} - IAQI_{Lo}}{BP_{Hi} - BP_{Lo}} \cdot (C_p - BP_{Lo}) + IAQI_{Lo}$$

式中各符号含义如下：

$IAQI_P$　　污染物P的空气质量分指数，结果进位取整数；

C_p　　污染物P的质量浓度值；

BP_{Hi}，BP_{Lo}　　与C_p相近的污染物浓度限值的高位值与低位值；

$IAQI_{Hi}$，$IAQI_{Lo}$　　与BP_{Hi},BP_{Lo}对应的空气质量分指数。

各项污染物项目浓度限值及对应的空气质量分指数级别见表 1。

表 1 空气质量分指数（IAQI）及对应的污染物项目浓度限值

序	指数或污染物项目	空气质量分指数	单位

号		及对应污染物浓度限值								
0	空气质量分指数（IAQI）	0	50	100	150	200	300	400	500	-
1	一氧化碳（CO）24 小时平均	0	2	4	14	24	36	48	60	mg/m^3
2	二氧化硫（SO_2）24 小时平均	0	50	150	475	800	1600	2100	2620	$\mu g/m^3$
3	二氧化氮（NO_2）24 小时平均	0	40	80	180	280	565	750	940	
4	臭氧（O_3）最大 8 小时滑动平均	0	100	160	215	265	800	-	-	
5	粒径小于等于 10μm颗粒物（PM_{10}）24 小时平均	0	50	150	250	350	420	500	600	
6	粒径小于等于 2.5μm颗粒物（$PM_{2.5}$）24 小时平均	0	35	75	115	150	250	350	500	

注：(1) 臭氧（O_3）最大 8 小时滑动平均浓度值高于 800 $\mu g/m^3$ 的，不再进行其空气质量分指数计算。

(2) 其余污染物浓度高于 IAQI=500 对应限值时，不再进行其空气质量分指数计算。

获得六种污染物的空气质量分指数后，依据公式

$$AQI = \max\{IAQI_{SO_2}, IAQI_{NO_2}, IAQI_{PM_{10}}, IAQI_{PM_{2.5}}, IAQI_{O_3}, IAQI_{CO}\}$$

计算出实测的 AQI。

空气质量等级范围根据 AQI 数值划分，等级对应的 AQI 范围见表 2。

表 2 空气质量等级及对应空气质量指数（AQI）范围

空气质量等级	优	良	轻度污染	中度污染	重度污染	严重污染
空气质量指数（AQI）范围	[0,50]	[51,100]	[101,150]	[151,200]	[201,300]	[301,+∞)

当 AQI 小于或等于 50（即空气质量评价为“优”）时，称当天无首要污染物；

当 AQI 大于 50 时，IAQI 最大的污染物为首要污染物。若 IAQI 最大的污染物为两项或两项以上时，并列为首要污染物。

3.4 结果与分析

AQI 计算结果表：

监测日期	地点	AQI 计算	
		AQI	首要污染物
2020/8/25	监测点 A	60	O_3
2020/8/26	监测点 A	46	无
2020/8/27	监测点 A	109	O_3
2020/8/28	监测点 A	138	O_3

四、问题 2 模型的建立与求解

4.1 问题分析

问题 2 要求使用附件 1 中的数据，根据对污染物浓度的影响程度，对气象条件进行合理分类，并阐述各类气象条件的特征。

假定在污染物排放情况不变、不考虑除自然因素以外影响条件（例如工业排放），当某一地区的气象条件有利于污染物扩散或沉降时，该地区的 AQI 会下降，反之会上升。根据问题要求，将附件 1 中“监测点 A 逐小时污染物浓度与气象实测数据”作为原始数据样本。

由于需要观察气象条件对于 AQI 的影响，因此需要利用问题 1 建立的模型计算原始数据样本的 AQI。由于不同气象条件对于不同污染物的影响效果不同，因此需要分别计算原始数据 SO_2、NO_2、PM10、PM2.5、O_3、CO 的 IAQI 值，以便于分析。

由于污染物分为一次污染物和二次污染物，一次污染物是指直接从污染源排到大气中的原始污染物质，二次污染物是指由一次污染物与大气中已有组分或几种一次污染物之间经过一系列化学或光化学反应而生成的与一次污染物性质不同的新污染物质。根据查阅的文献资料本数据样本需要考虑的污染物中属于一次污染物的有 SO_2、NO_2、CO，属于二次污染物的有 PM10、PM2.5、O_3。根据文献资料调研，臭氧与氮氧化物之间存在相互转化，因此 NO_2 是形成 O_3 的关键前体，PM10、PM2.5 主要成分包括含碳颗粒、硫酸盐、硝酸盐，SO_2 是形成硫酸盐的关键，NO_2 是形成硝酸盐的关键，CO 属于含碳颗粒，因此 PM10、PM2.5 与 SO_2、NO_2、CO 都有关联。

根据附件 1 中“监测点 A 逐小时污染物浓度与气象实测数据”，假设天气条件仅受温度、湿度、气压、风速、风向影响。即考虑五种天气条件与六种污染物之间的相关性。考虑到这五种天气条件对于六种污染物可能具有不同的影响，因此要分别对五种天气条件和六种污染物的相关性两两进行分析。由于涉及到天气条件的变化趋势，因此需要对五种天气条件与六种污染物的差值进行分析。

另外，与问题 1 中的数据样本类似，附件 1 中“监测点 A 逐小时污染物浓度与气象实测数据”也可能存在异常数据，需要先对异常数据进行处理。

4.2 数据预处理

为了得到数据预处理的结果，需要对样本 2019 年 4 月 16 日至 2021 年 7 月 13 日的六种污染物浓度与五种天气条件实测数据进行异常数据数据整定，具体操作如下：

4.2.1 异常数据剔除

首先使用 Pyhton 编程，判断六种污染物与五种天气条件实测数据是否服从正态分布，并绘制样本数据箱盒图。对于服从正态分布的样本，根据拉依达准则（3σ 准则），对于不服从正态分布的样本，根据箱盒图监测异常数据并剔除。

4.2.2 异常数据填充

数据预处理的异常数据填充策略为，计算缺失值前后各两个小时的数据的平均值。对全部异常数据处理一遍之后，判断填充数据是否仍为异常值，如果异常则继续处理异常数据。在第一次处理异常数据之后，取前后各两个小时的数据的以及去年同一时期当天当时刻以及前后各两个小时的数据的平均值，对全部异常数据处理一遍之后，判断填充数据是否仍为异常值，如果异常则增加小时数取平均值，直到不再异常为止。

4.2.3 异常数据处理效果

4.3 模型的建立与求解

为了对气象条件进行合理分类，建立一个相关性模型，来验证六种污染物与五种天气

条件之间的相关性。

对 11 个参数两两计算 Pearson 相关系数[3]，Pearson 相关系数计算公式如下：

$$r_{xy} = \frac{\sum x_i y_i - n\bar{x}\bar{y}}{(n-1)s_x s_y} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}}.$$

Pearson 相关系数的值介于-1.0——1.0 之间，其相关系数的显著性如下图所示：

相关性	负	正
无相关性	-0.09 to 0.0	0.0 to 0.09
弱相关性	-0.3 to -0.1	0.1 to 0.3
中相关性	-0.5 to -0.3	0.3 to 0.5
强相关性	-1.0 to -0.5	0.5 to 1.0

图 4.3 Pearson 相关系数的显著性

11 个参数两两计算 Pearson 相关系数结果如下：

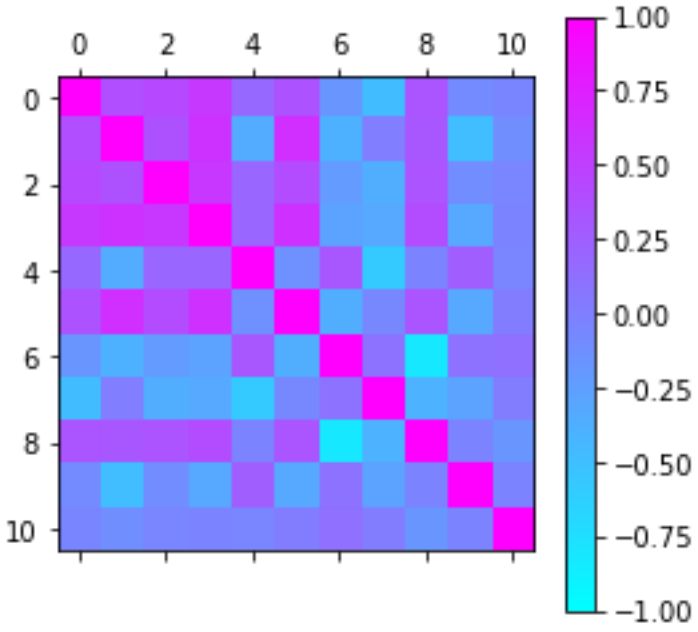


图 4.4 11 个参数 Pearson 相关系数图

根据相关系数图可以获知：

（1）温度与 SO_2、NO_2、PM10、PM2.5、CO 污染物浓度呈负相关，因此温度高的天气利于污染物扩散；

（2）湿度与 SO_2、PM10、PM2.5、CO 污染物浓度呈负相关，因此湿度增大利于污染物沉降；

（3）高气压与 SO_2、NO_2、PM10、PM2.5、CO 污染物浓度呈正相关，因为高气压时大气呈稳定状态，风速较小，污染物不易扩散，导致污染物浓度增加，使大气污染加重。

（4）风速与 SO_2、NO_2、PM10、PM2.5、CO 污染物浓度呈负相关，因此风速增大有利于污染物浓度扩散；风速与 O_3 污染物浓度呈正相关，因为 NO_2 的减少不利于 O_3 的合成；

（5）风向与五种污染物浓度全都不相关；

（6）大气压与湿度呈强负相关；

（7）温度与湿度呈弱相关；

（8）风速与温度呈负相关；

（9）气压与风速呈若负相关；

由此上述特性结合相关文献资料，我们将气象条件从污染物沉降、扩散以及产生三个方面分为六类，具体如下：

（1）不利于污染物沉降，其气象条件特征为大气压高，降水少，湿度低；

（2）利于污染物沉降，其气象条件特征为大气压低，降水多，湿度高；

（3）不利于污染物扩散，其气象条件特征为风速低；

（4）利于污染物扩散，其气象条件特征为风速高；

（5）易增加污染物产生，其气象条件为温度低；

（6）易减少污染物产生，其气象条件为温度高。

五、问题 3 模型的建立与求解

5.1 问题分析

问题 3 要求使用附件 1、2 中的数据，建立一个同时适用于 A、B、C 三个监测点的二次预报模型，用来预测未来三天六种常规污染物单日浓度值，并要求二次预报模型预测结果中 AQI 预报值的最大相对误差应尽量小。假设监测点两两间直线距离>100km，忽略相互影响。

在问题 2 中我们已经对气象条件进行了分类，因此问题 3 若想更加准确的建立二次预测模型应当建立在问题 2 的基础上。首先，我们需要判断数据是否具有周期性，根据数据类型选择不同的预测模型;然后在问题 1 的基础上对数据进行预处理剔除并替换异常数据；最后在问题 2 的基础上，结合预测模型对污染物浓度进行预测。

5.2 数据预处理

5.2.1 异常数据剔除

首先使用 Pyhton 编程，判断六种污染物与五种天气条件实测数据是否服从正态分布，并绘制样本数据箱盒图。对于服从正态分布的样本，根据拉依达准则（3σ 准则），对于不服从正态分布的样本，根据箱盒图监测异常数据并剔除。

5.2.2 异常数据填充

数据预处理的异常数据填充策略为，计算缺失值前后各两个小时的数据的平均值。对全部异常数据处理一遍之后，判断填充数据是否仍为异常值，如果异常则继续处理异常数据。在第一次处理异常数据之后，取前后各两个小时的数据的以及去年同一时期当天当时刻以及前后各两个小时的数据的平均值，对全部异常数据处理一遍之后，判断填充数据是否仍为异常值，如果异常则增加小时数取平均值，直到不再异常为止。

5.2.3 异常数据处理效果

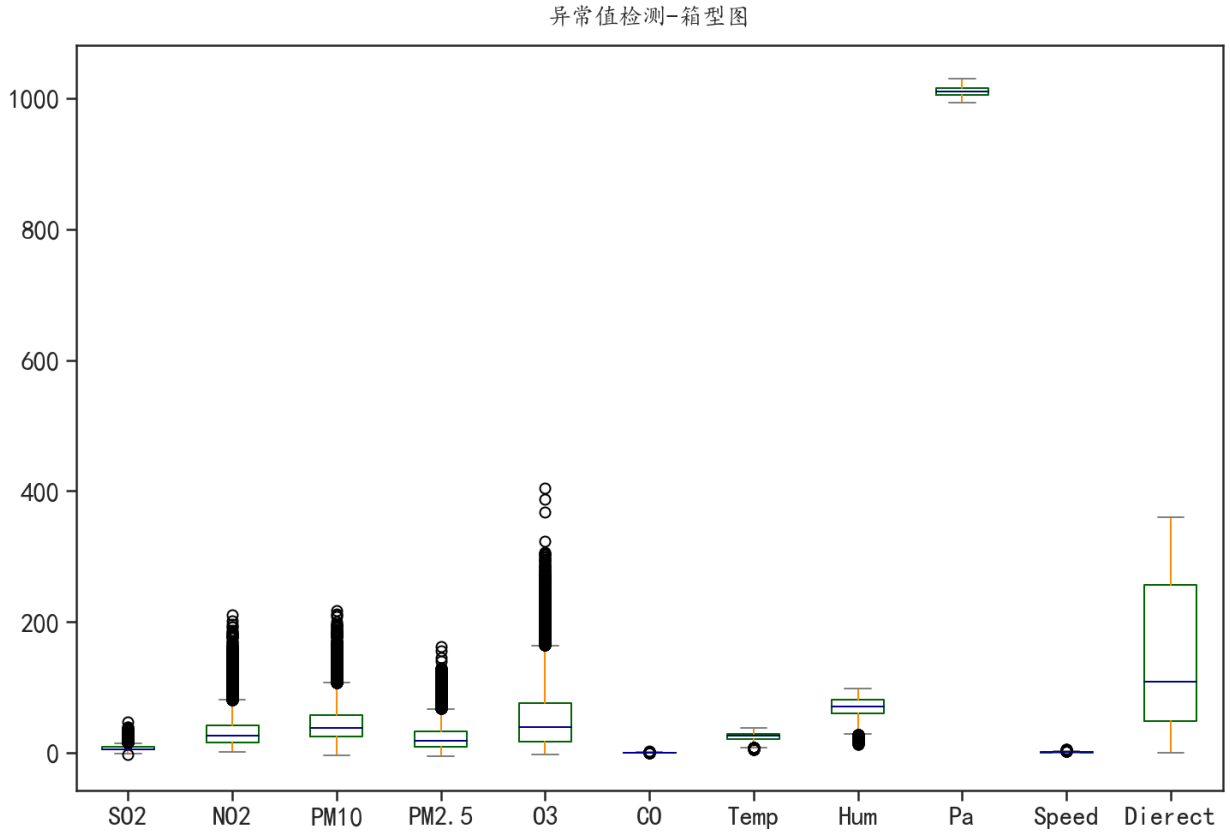


图 5.1 监测点 A 异常数据处理前箱型图

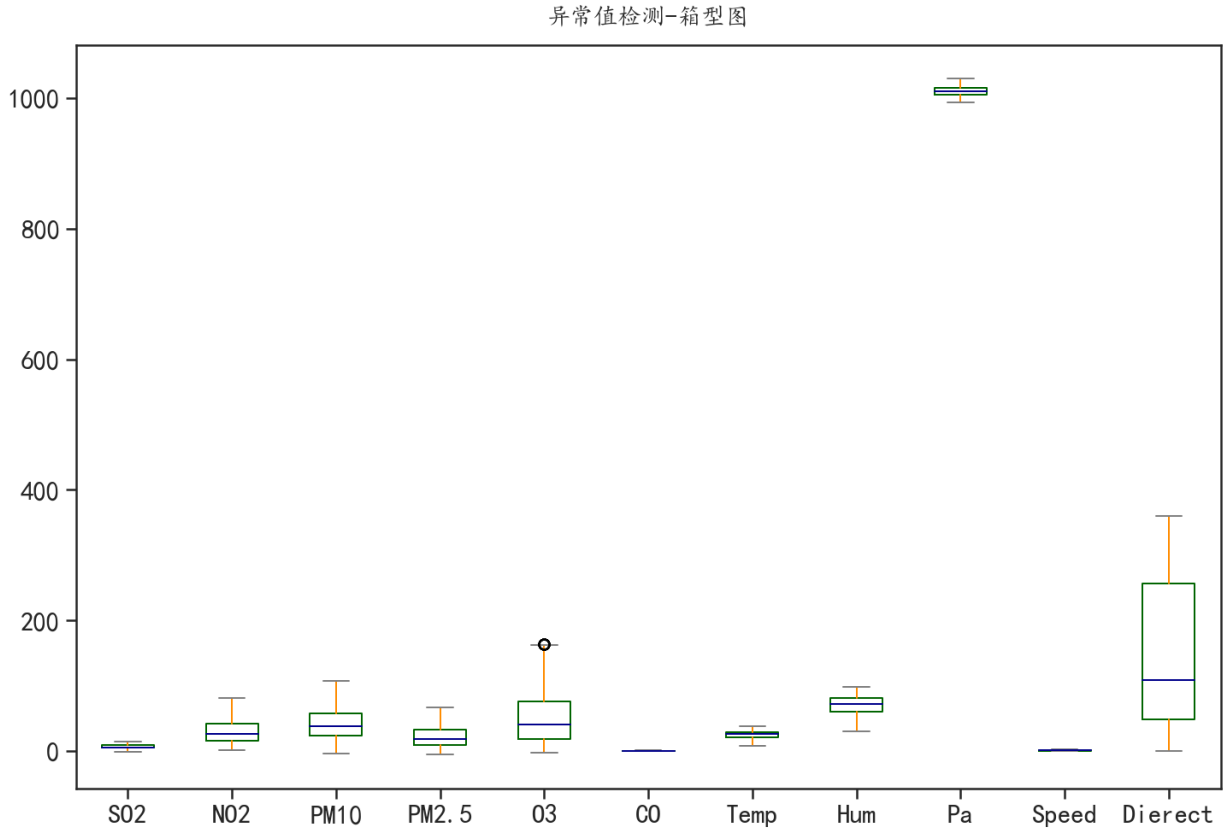


图 5.2 监测点 A 异常数据处理后箱型图

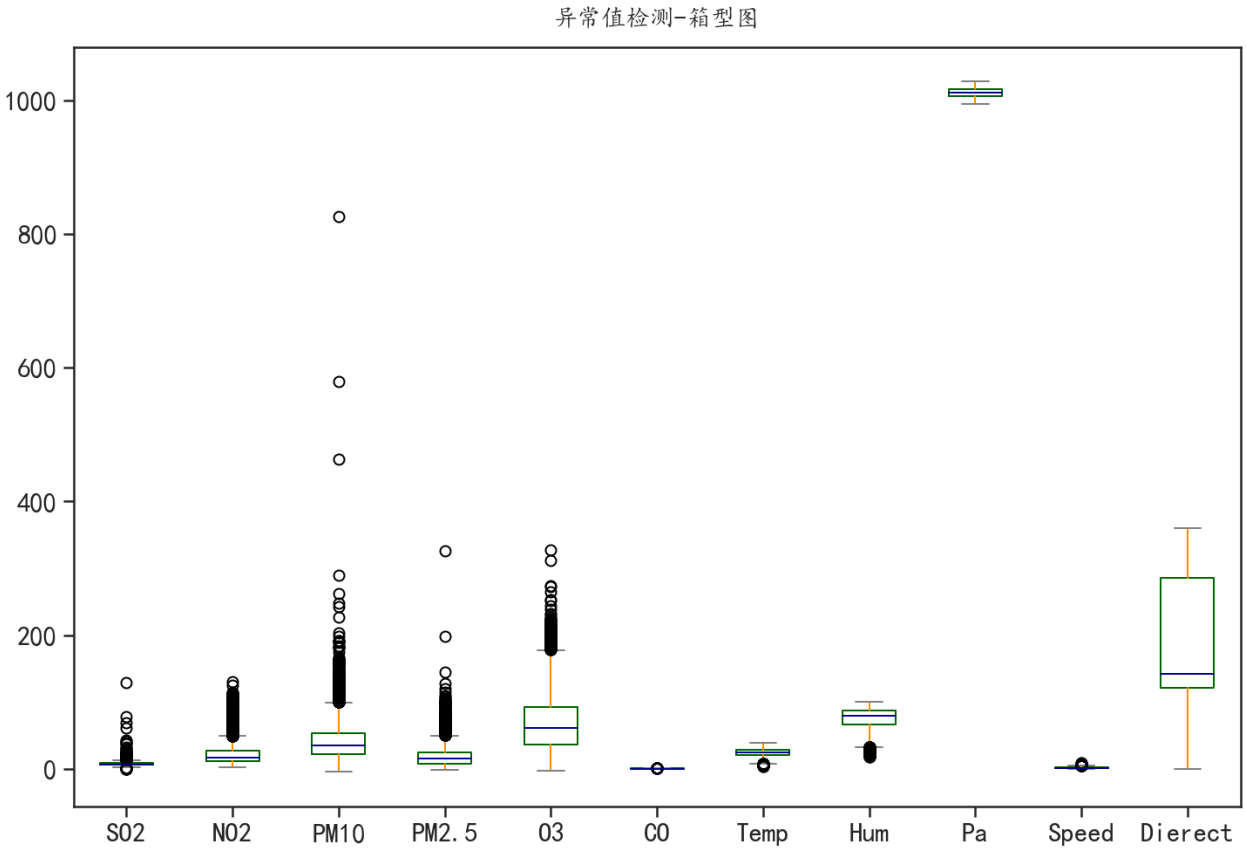


图 5.3 监测点 B 异常数据处理前箱型图

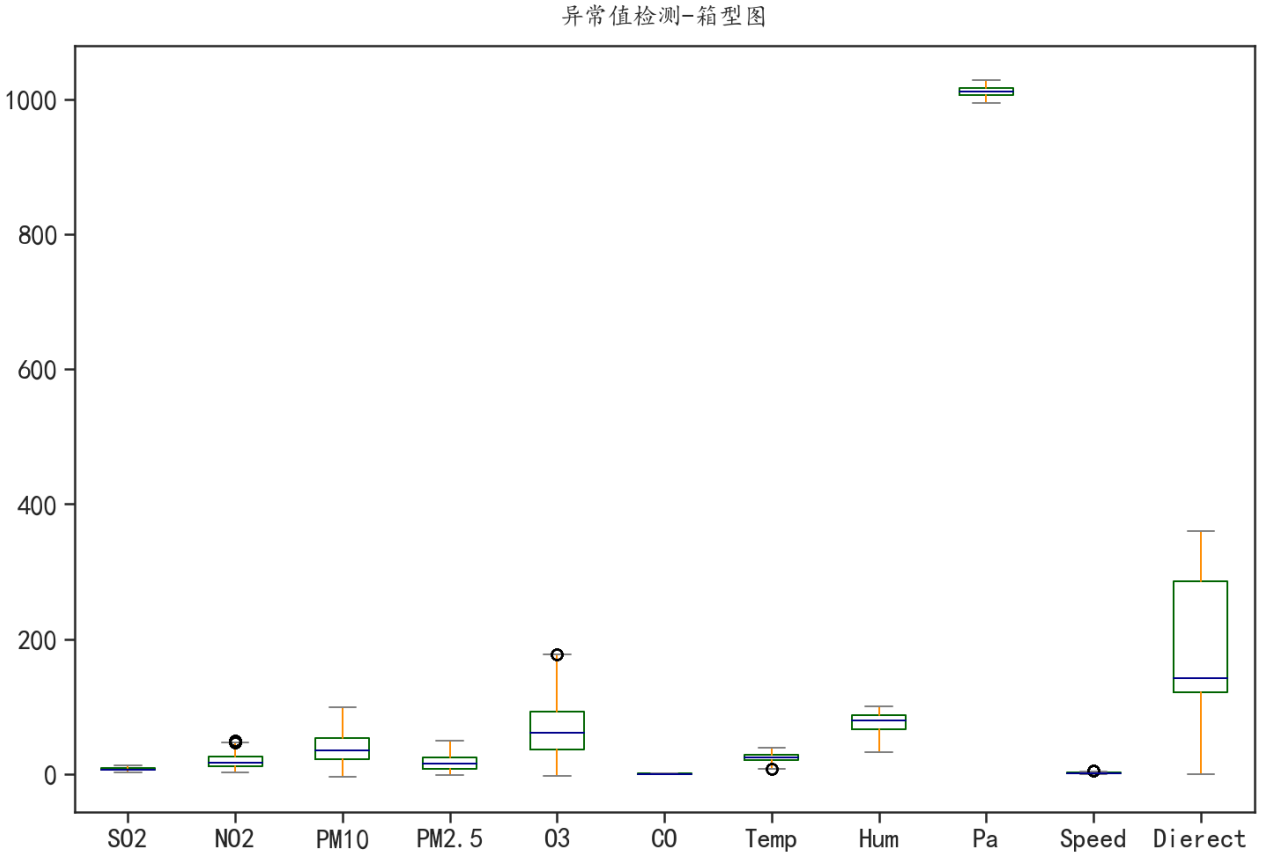


图 5.4 监测点 B 异常数据处理后箱型图

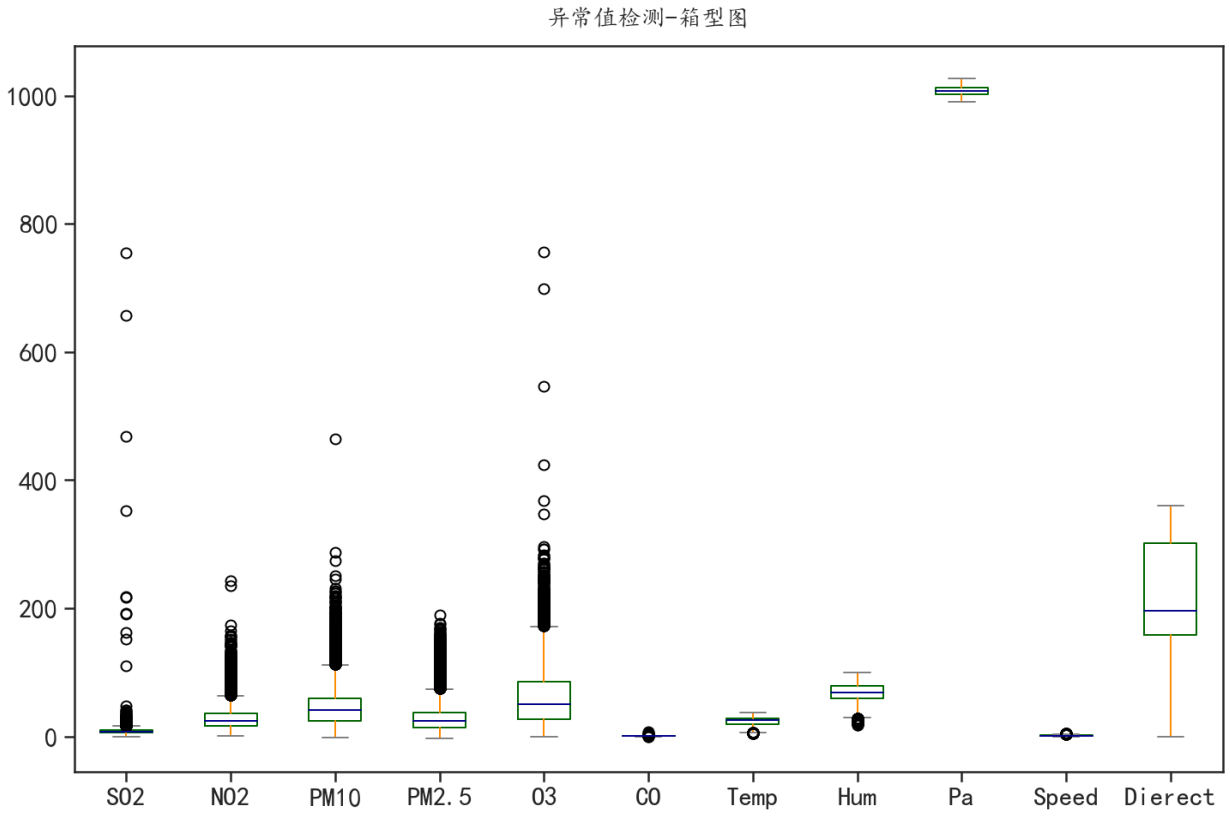


图 5.5 监测点 C 异常数据处理前箱型图

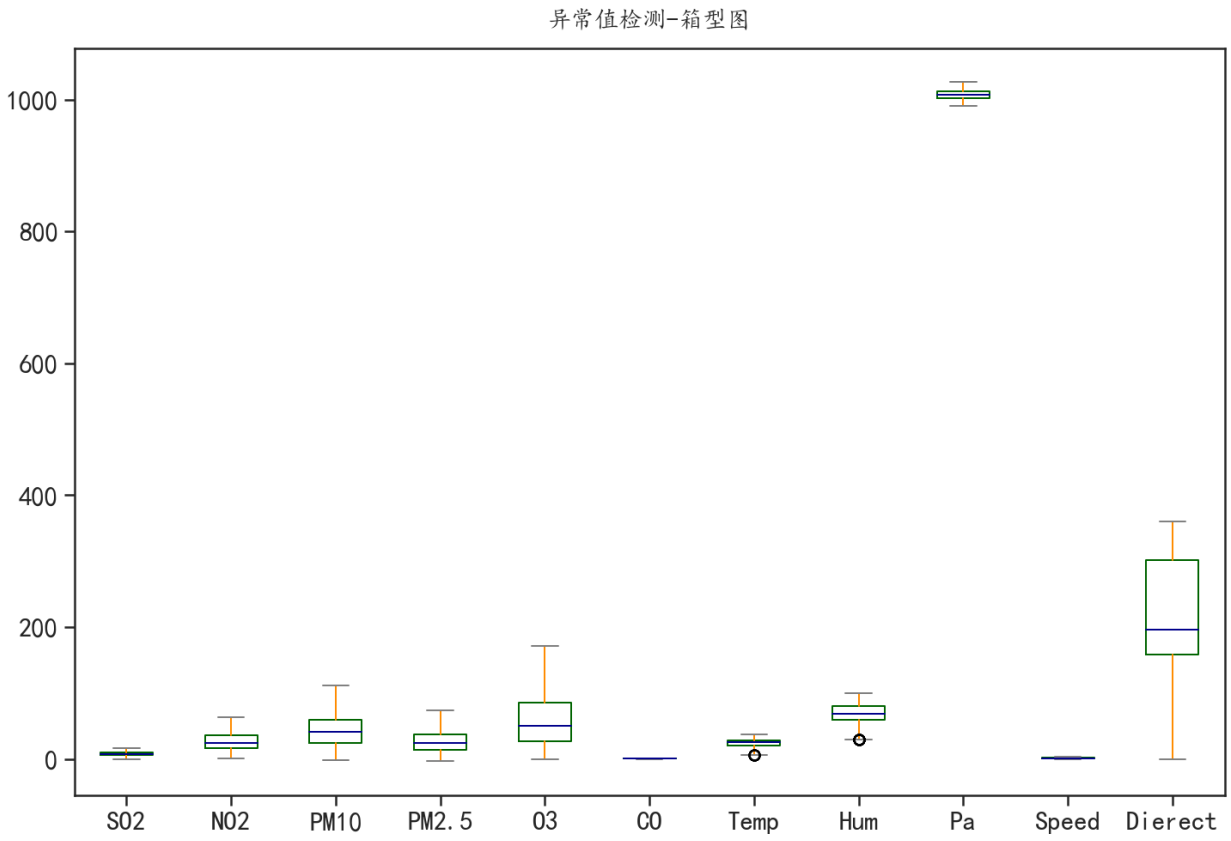


图 5.6 监测点 C 异常数据处理后箱型图

5.2.4 数据周期性判断

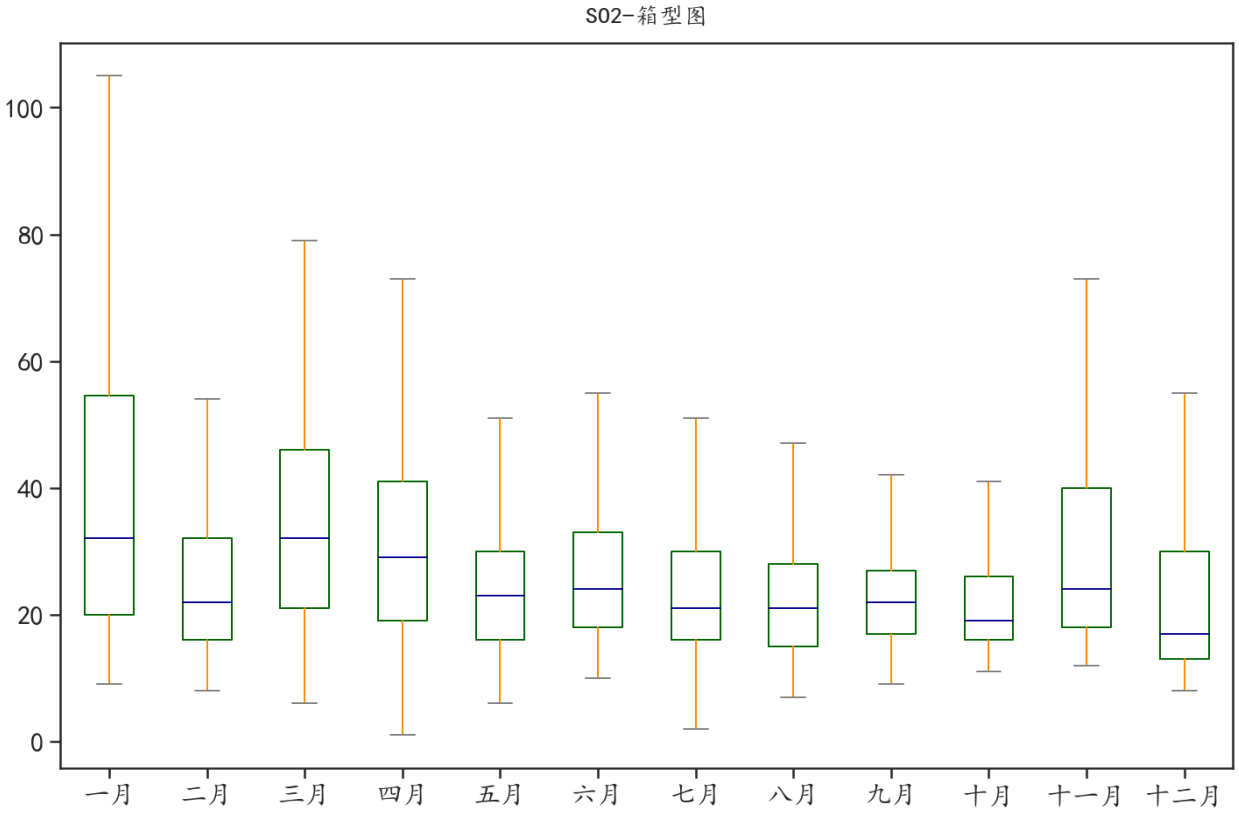


图 5.7　SO_2 周期性变化箱型图

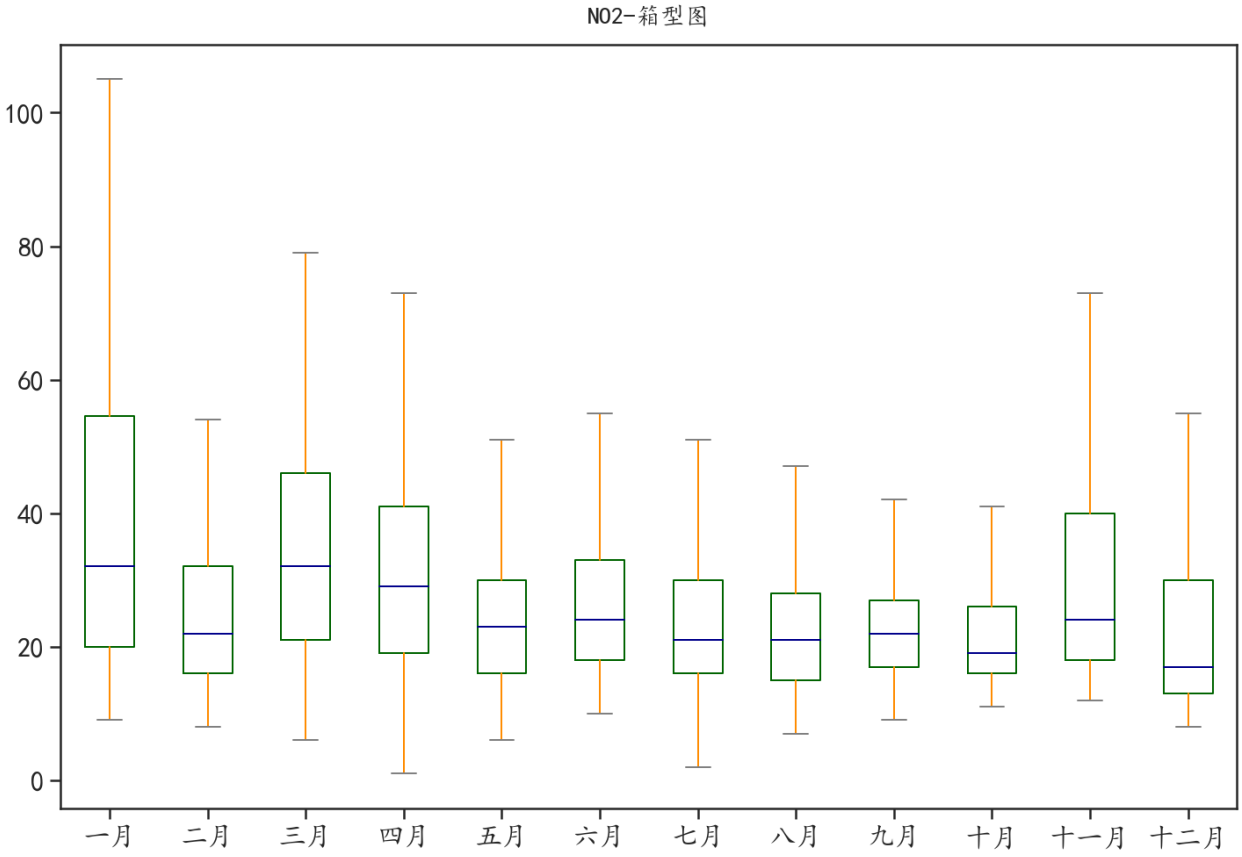


图 5.8　NO_2 周期性变化箱型图

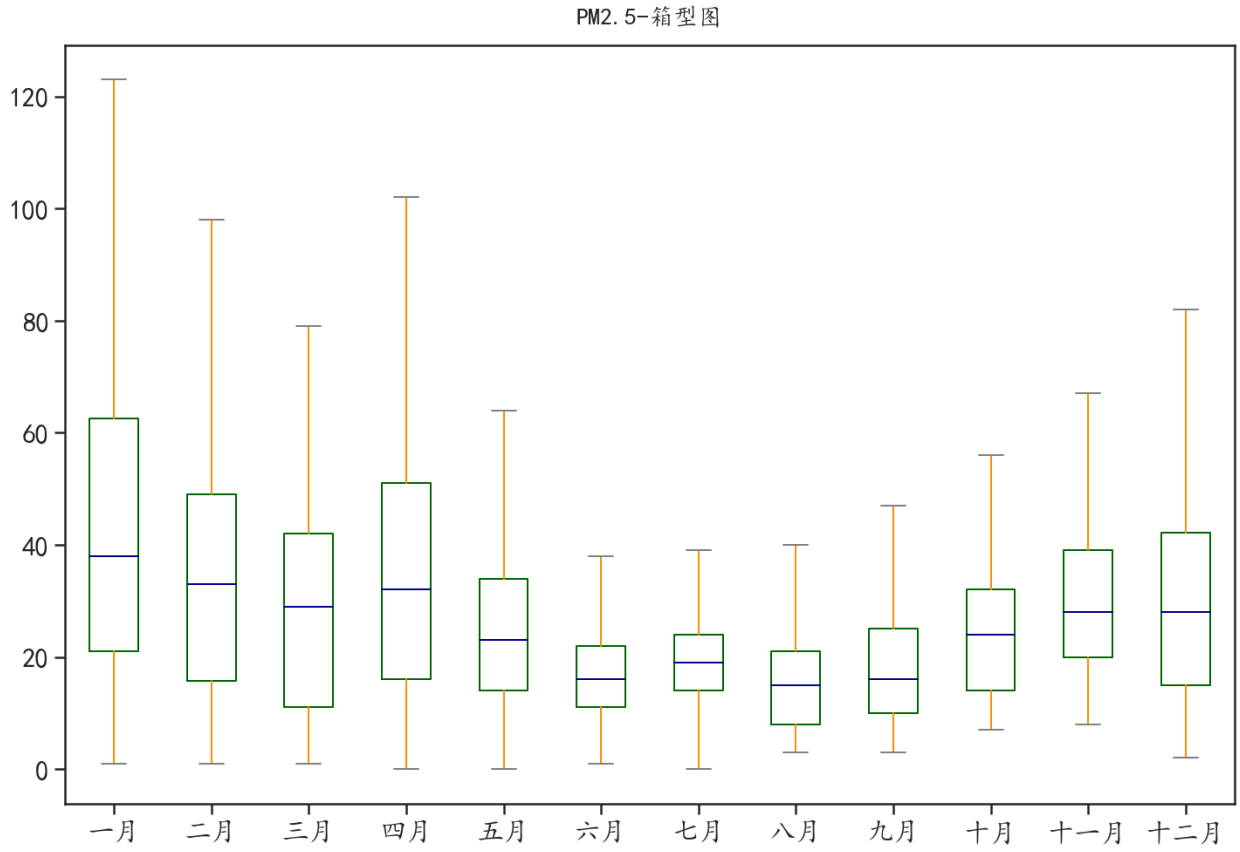


图 5.9　PM2.5 周期性变化箱型图

5.3 模型的建立

气象条件、污染物浓度等是按时间顺序排列、随时间变化且互相关的数据序列。这些序列具有趋势性、季节性和非平稳性。使用 LSTM（长短期记忆，Long short-term memory），可以处理序列变化的数据，使用 LSTM 建立预测的网络。

如下图所示，图中的 X 表示输入的时间序列，给定的数据一共共包含 11 个不同的特征，如温度、湿度等。数据中每小时记录一次值，一天有 24 个观测值。我们选择了五天的观察时间，创建一个包含 120 个观测值的窗口以训练模型。即使用某一时间下的前 120 个数据用于预测该时间点的气候条件。模型包含一个 LSTM 层和一个线性层，LSTM 层有 50 个 LSTM。使用时间跨度在 2020-7-23 ~ 2021-7-13 的数据对 LSTM 模型进行训练，得到训练好的预测模型。将 2021 年 7 月 7 日至 7 月 12 日的数据输入预测模型，得到 2021 年 7 月 13 日至 7 月 15 日 6 种常规污染物的单日浓度的预测值。

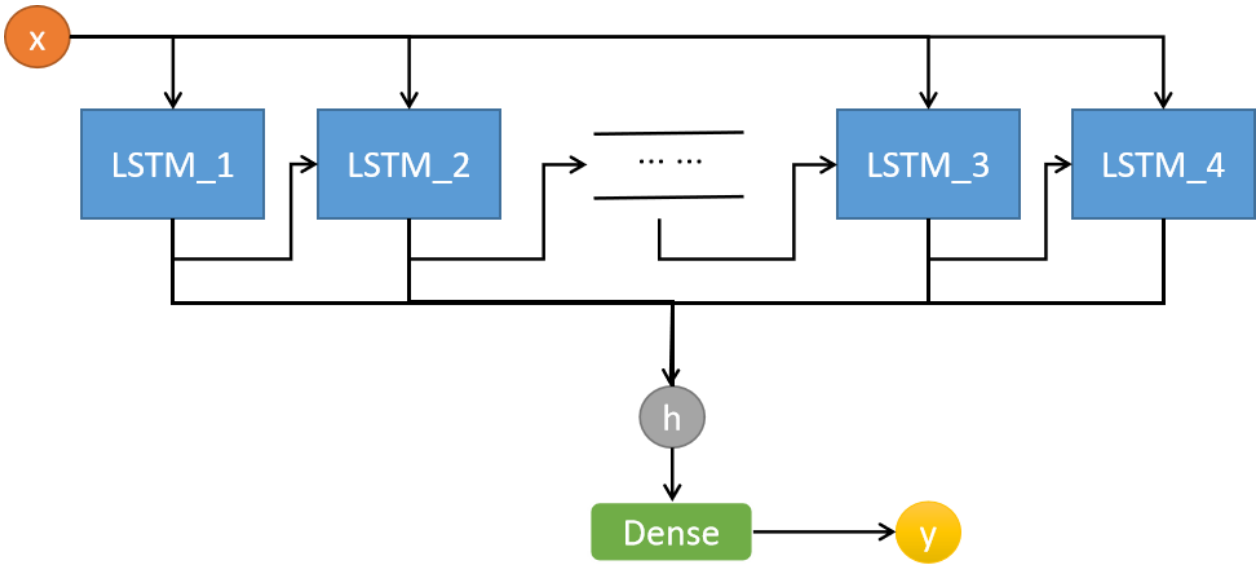


图 5.10 预测模型

5.4 模型的求解

5.4.1 监测点 A 污染物浓度及 AQI 预测结果表

预报日期	地点	二次模型日值预测							
		SO_2 (μg/m³)	NO_2 (μg/m³)	PM_{10} (μg/m³)	$PM_{2.5}$ (μg/m³)	O_3 最大八小时滑动平均 (μg/m³)	CO (mg/m³)	AQI	首要污染物
2021/7/13	监测点 A	5	17	28	10	73	0.4	38	无
2021/7/14	监测点 A	5	18	27	10	72	0.4	51	O_3
2021/7/15	监测点 A	5	18	26	9	71	0.4	39	无

5.4.2 监测点 B 污染物浓度及 AQI 预测结果表

预报日期	地点	二次模型日值预测							
		SO_2 (μg/m³)	NO_2 (μg/m³)	PM_{10} (μg/m³)	$PM_{2.5}$ (μg/m³)	O_3 最大八小时滑动平均 (μg/m³)	CO (mg/m³)	AQI	首要污染物
2021/7/13	监测点 B	6	12	24	8	73	0.4	36	无
2021/7/14	监测点 B	6	12	24	8	72	0.4	36	无
2021/7/15	监测点 B	6	12	23	8	71	0.4	35	无

5.4.3 监测点 C 污染物浓度及 AQI 预测结果表

预报日期	地点	二次模型日值预测							
		SO_2 (μg/m³)	NO_2 (μg/m³)	PM_{10} (μg/m³)	$PM_{2.5}$ (μg/m³)	O_3 最大八小时滑动平均 (μg/m³)	CO (mg/m³)	AQI	首要污染物
2021/7/13	监测点 C	6	19	30	17	111	0.6	72	O_3
2021/7/14	监测点 C	6	19	31	18	113	0.6	74	O_3
2021/7/15	监测点 C	6	19	31	18	114	0.5	74	O_3

5.5 结果分析

5.5.1 监测点 A AQI 预测结果与实际结果比对图

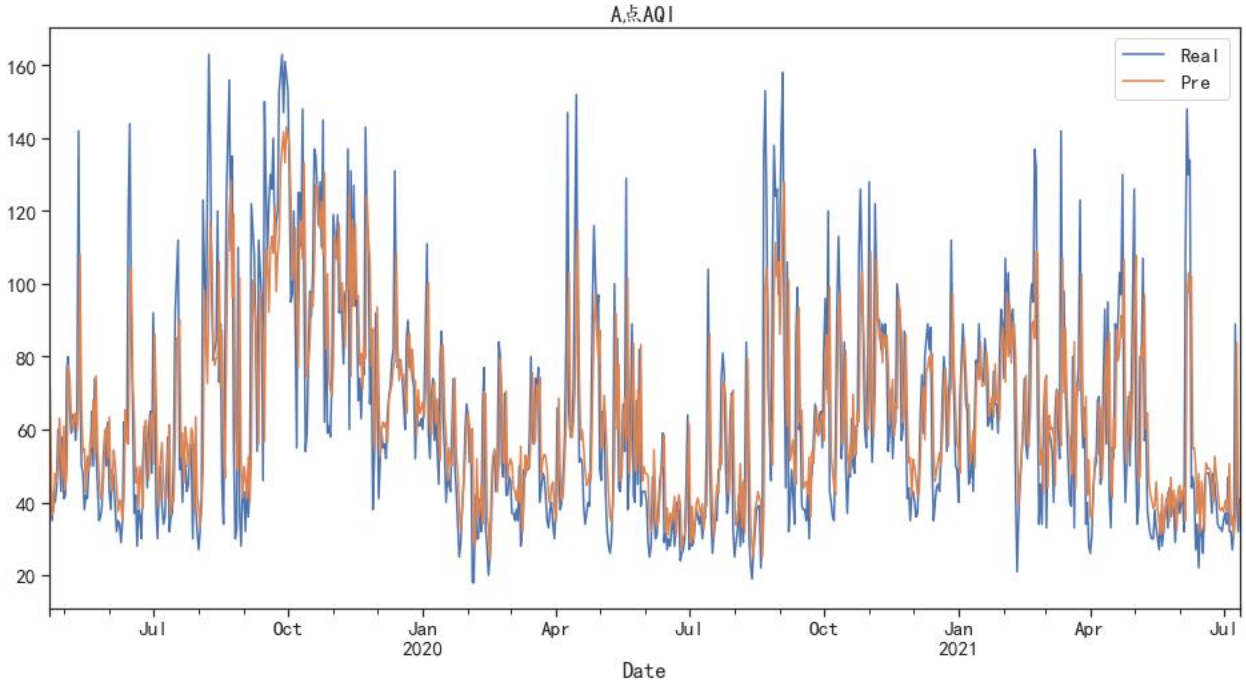


图 5.11 监测点 A AQI 预测结果与实际结果比对图

5.5.2 监测点 B AQI 预测结果与实际结果比对图

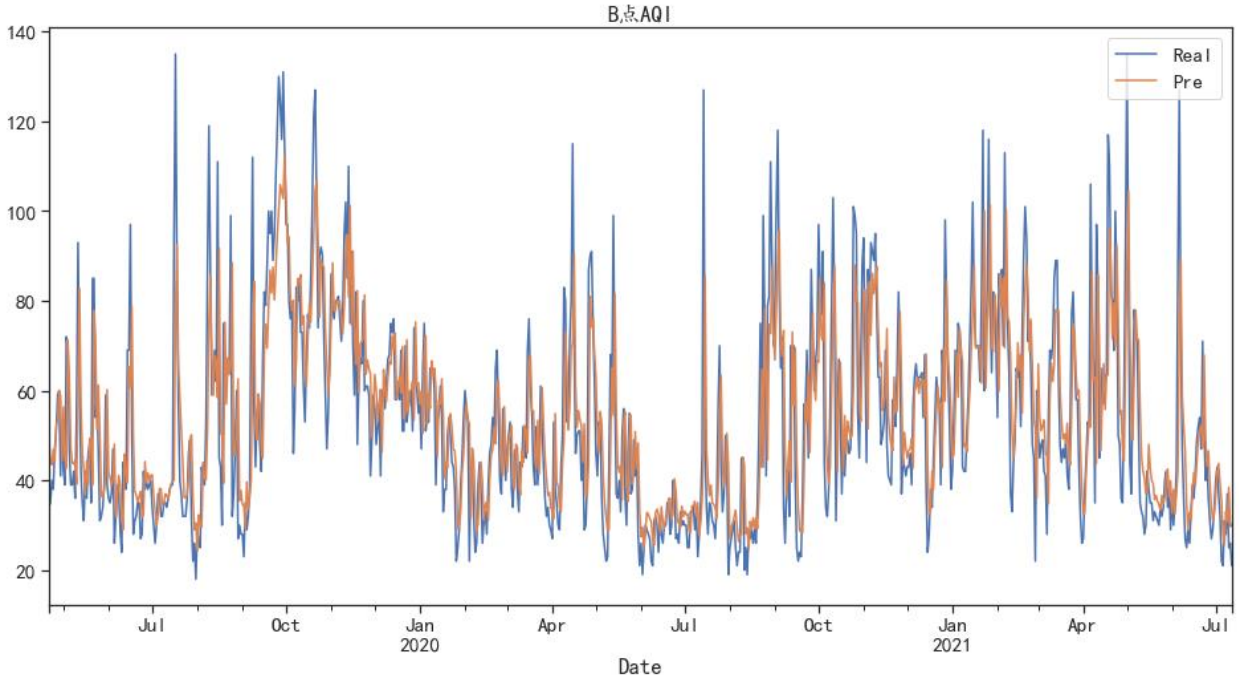


图 5.12 监测点 B AQI 预测结果与实际结果比对图

5.5.3 监测点 C AQI 预测结果与实际结果比对图

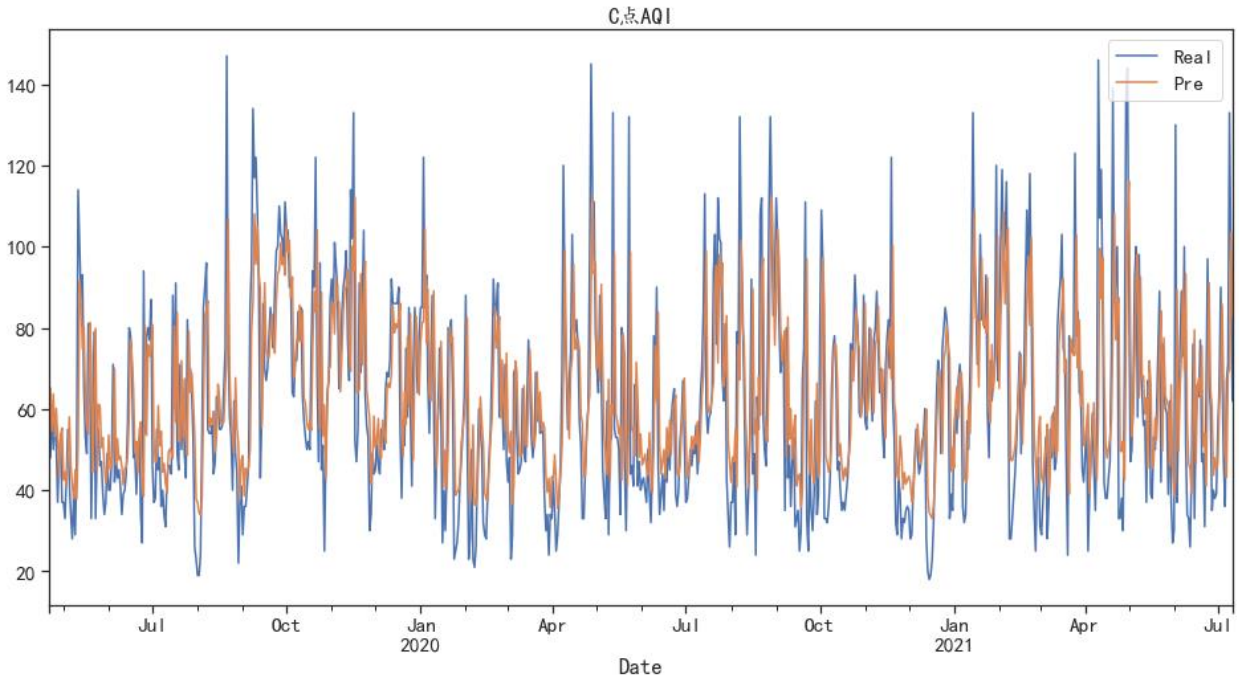


图 5.13 监测点 C　AQI 预测结果与实际结果比对图

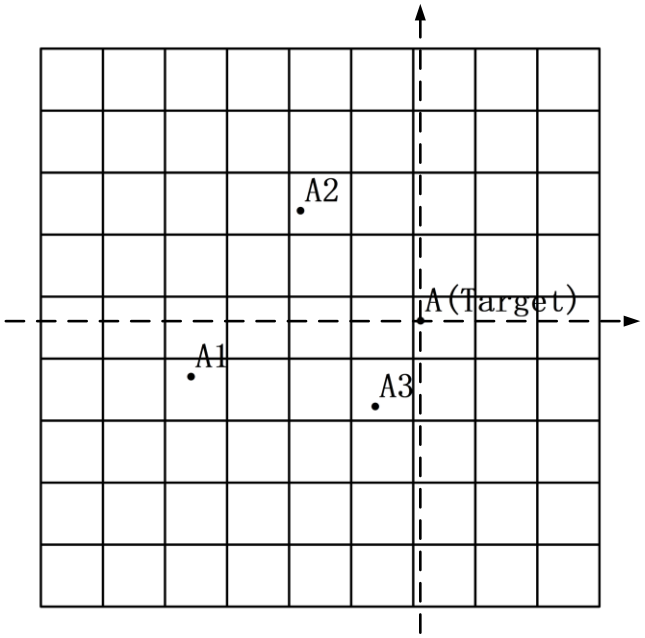
5.5.3 分析

通过监测点 A、B、C AQI 预测结果与实际结果比对图可以发现预测结果与实际结果相关度很高，因此可以得出预测模型的预测效果良好的结论。

六、问题 4 模型的建立与求解

6.1 问题分析

问题 4 要求建立包含 A、A1、A2、A3 四个监测点的协同预报模型，使二次模型预测结果中 AQI 预报值的最大相对误差应尽量小，且首要污染物预测准确度尽量高。由于相邻区域的污染物浓度往往具有一定的相关性，因此需要在问题 1、2、3 的基础上考虑监测点 A、A1、A2、A3 之间的影响。假设 A、A1、A2、A3 之间的相对位置如图 6.1 所示。根据实际情况分析，某一监测点的变化趋势除了受到本监测点天气变化的影响，还会受到其他监测点的污染物扩散的影响，而影响污染物扩散的因素主要为风速、距离、风向，因此需要在问题 3 模型的基础上，对风速、风向、距离对于其他监测点的影响进行建模。



A (0, 0)　　A1 (-14.4846, -1.9699)　　A2 (-6.6716, 7.5953)　　A3 (-3.3543, -5.0138)

图 6.1 各监测站点相对位置示意图，正东方向为 x 轴，正北方向为 y 轴，单位：km

6.2 数据预处理

6.2.1 异常数据剔除

首先使用 Pyhton 编程，判断污染物与天气条件实测数据是否服从正态分布，并绘制样本数据箱盒图。对于服从正态分布的样本，根据拉依达准则（3σ 准则），对于不服从正态分布的样本，根据箱盒图监测异常数据并剔除。

与问题 1、2、3 不同，问题 4 中由于相邻区域的额污染物具有相关性，如果存在 A、A1、A2、A3 中任意一监测点的数值与其他地区数值相差较大的也应当被当作异常值来处理。对于该类异常值，选择适应 Z-score 处理异常值，将远离标准差 3 倍距离以上的数据点视为异常值剔除掉。其中，Z-score 是一维或低维特征空间中的参数异常检测方法。该技术假定数据是高斯分布，异常值是分布尾部的数据点，因此远离数据的平均值。距离的远近取决于使用公式计算的归一化数据点 z i 的设定阈值 Zthr：

$$z_i = \frac{x_i - \mu}{\sigma}$$

其中 x_i 是一个数据点，μ 是所有点 x_i 的平均值，δ 是所有点 x_i 的标准偏差。然后经过标准化处理后，异常值也进行标准化处理，其绝对值大于 Zthr 值设置为 3.0。

6.2.2 异常数据填充

问题 4 数据预处理的异常数据填充策略也与问题 1、2、3 不同，需要计算缺失值前后各一个小时的数据以及相邻区域前后各一个小时的数据的平均值。对全部异常数据处理一遍之后，判断填充数据是否仍为异常值，如果异常则继续处理异常数据。在第一次处理异常数据之后，取本区域与相邻区域前后各两个小时的数据的以及去年同一时期当天当时刻以及前后各两个小时的数据的平均值，对全部异常数据处理一遍之后，判断填充数据是否仍为异常值，如果异常则增加小时数取平均值，直到不再异常为止。

6.3 模型的建立

6.3.1 污染物扩散模型

各个监测点的污染物浓度预测由本监测点的气象条件以及附近监测点的气象条件共同组成。本监测点的污染物浓度预测与问题 3 的污染物浓度预测一致。附近监测点的污染物浓度影响主要由风速、风向、距离和污染物浓度所决定。如图 6.4 所示，以 A1 监测点对于 A2 监测点的污染物浓度影响为例，构建各个监测点之间的污染物浓度影响模型。

（1）A1 监测点的风向为 A1-B,风速为 v；

（2）风向与正北方向的夹角为 β；

（3）A1-A2 之间的距离为 d；

（4）A1-A2 与正北方向的夹角为 γ；

（5）风向与 A1-A2 之间的夹角为 $\alpha = \beta - \gamma$；

（6）对风 A1-B 进行受力分析,得到风 A1-B 在 A1-A2 方向上的分力 $F' = F * \cos\alpha = F * \cos(\beta - \gamma)$;

（7）此时刻 A1 污染物对于 A2 污染物的影响时间为 $t = d / (v * \cos\alpha) = d / (v * \cos(\beta - \gamma))$；

（8）A1 处污染物浓度为 x_i，A2 处污染物浓度为 $y_i = x_i * \cos\alpha = x_i * (\beta - \gamma)$。

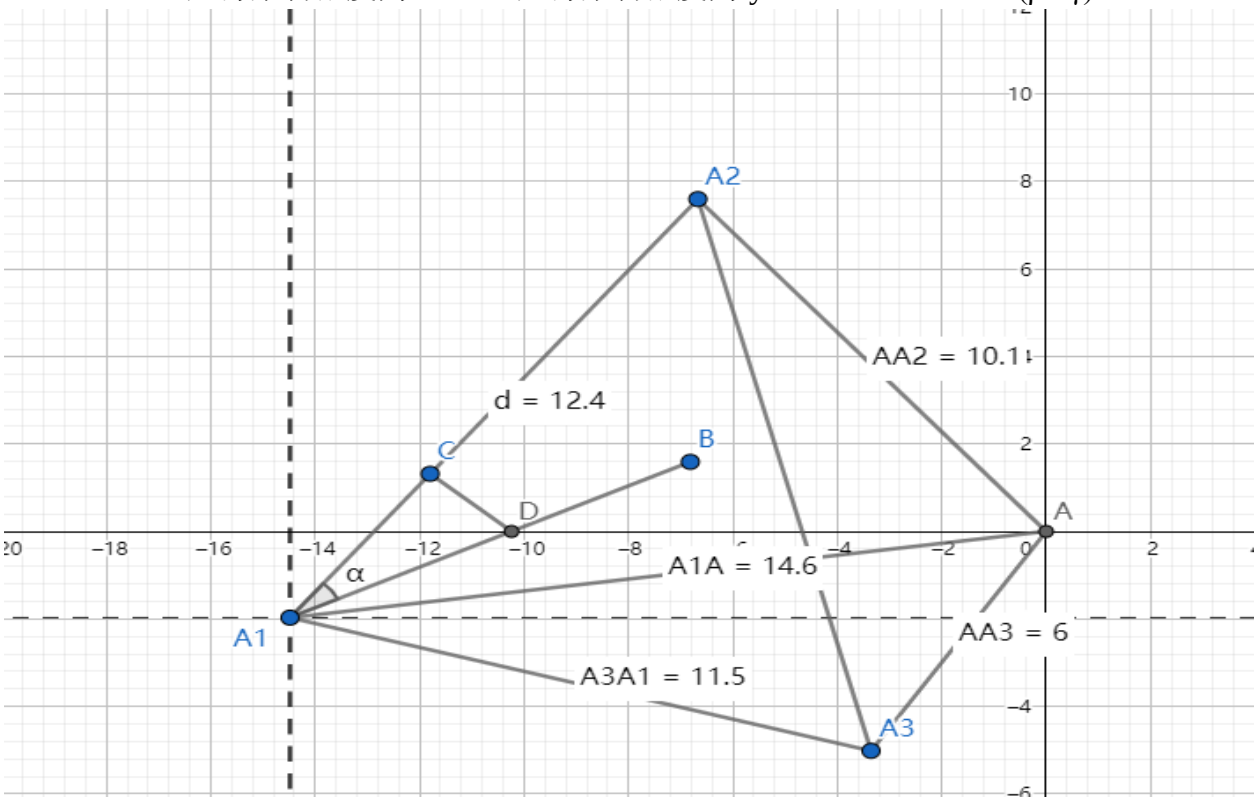


图 6.4 污染物扩散模型

6.3.2 协同预报模型

协同预报分为三层，第一层为异常数据处理，第二层为本监测地污染物浓度预报以及对其他检测地污染物扩散预报，第三层为协同预报模型，如图 6.5 所示。

首先进行异常数据预处理，得到有好的数据样本。然后分别使用监测点 A、A1、A2、A3 的时间跨度在 2020-7-23 ~ 2021-7-13 的数据在污染物扩散模型基础上对 LSTM 模型进行训练，分别得到训练好的对于四个监测点的预测模型（例如对 A 进行预测得到 A 对 A 本身以及对 A1、A2、A3 的预测模型）。最后使用本监测点的数据以及其他监测点对于本监测点的污染物扩散数据对 LSTM 模型进行训练，得到训练好的训练模型。分别将监测点

A、A1、A2、A3 时间跨度在 2021 年 7 月 7 日至 7 月 12 日的数据输入预测模型，分别得到 2021 年 7 月 13 日至 7 月 15 日 6 种常规污染物的单日浓度的预测值。

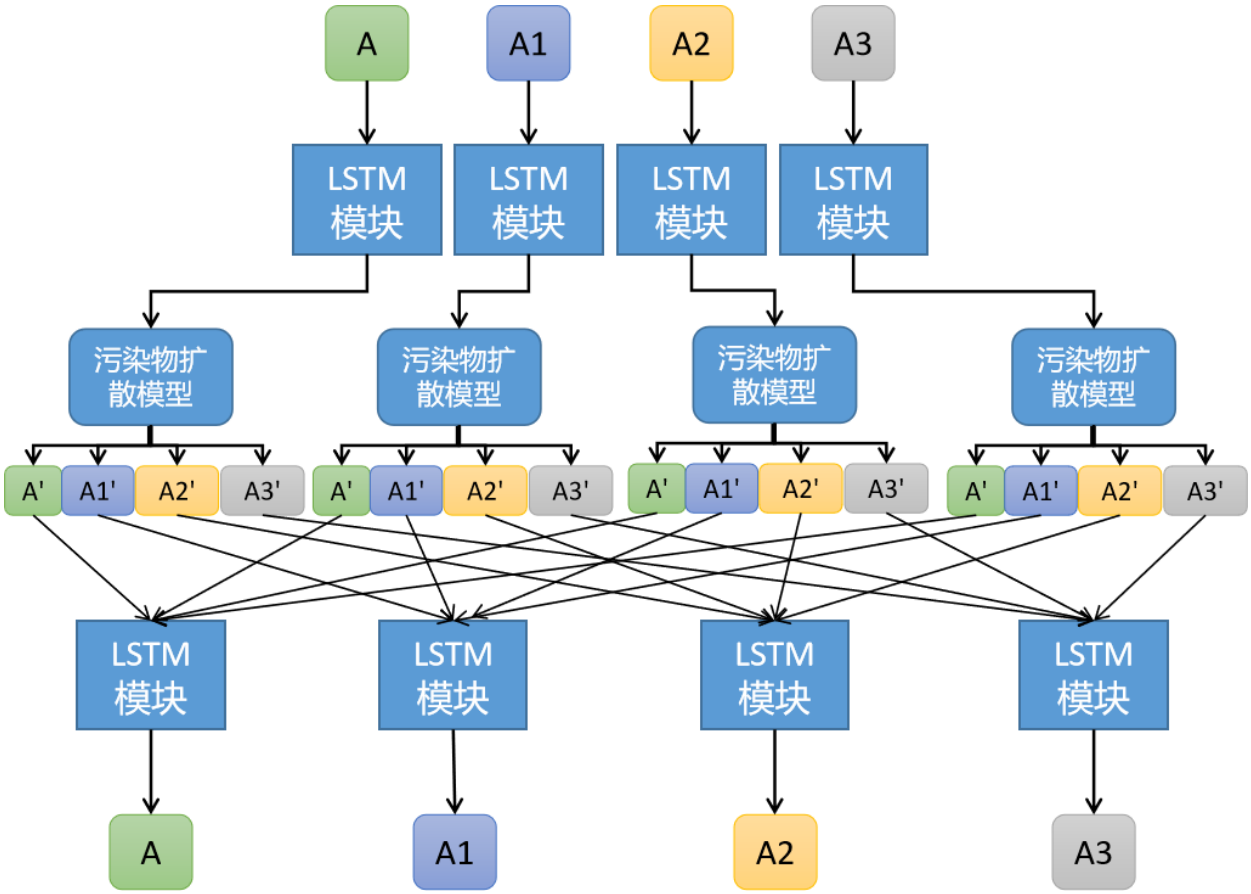


图 6.5 协同预报模型

6.4 模型的求解

6.4.1 监测点 A 污染物浓度及 AQI 预测结果表

预报日期	地点	二次模型日值预测							
		SO_2 ($\mu g/m^3$)	NO_2 ($\mu g/m^3$)	PM_{10} ($\mu g/m^3$)	$PM_{2.5}$ ($\mu g/m^3$)	O_3 最大八小时滑动平均 ($\mu g/m^3$)	CO (mg/m^3)	AQI	首要污染物
2021/7/13	监测点 A	5	17	28	10	73	0.4	38	无
2021/7/14	监测点 A	5	18	27	10	72	0.4	51	O_3
2021/7/15	监测点 A	5	18	26	9	71	0.4	39	无

6.4.2 监测点 A1 污染物浓度及 AQI 预测结果表

预报日期	地点	二次模型日值预测							
		SO_2 ($\mu g/m^3$)	NO_2 ($\mu g/m^3$)	PM_{10} ($\mu g/m^3$)	$PM_{2.5}$ ($\mu g/m^3$)	O_3 最大八小时滑动平均 ($\mu g/m^3$)	CO (mg/m^3)	AQI	首要污染物
2021/7/13	监测点 A1	6	18	31	12	84	0.5	50	无
2021/7/14	监测点 A1	6	17	31	12	85	0.5	52	O_3
2021/7/15	监测点 A1	6	17	31	12	85	0.5	52	O_3

6.4.3 监测点 A2 污染物浓度及 AQI 预测结果表

预报日期	地点	二次模型日值预测							
		SO_2 ($\mu g/m^3$)	NO_2 ($\mu g/m^3$)	PM_{10} ($\mu g/m^3$)	$PM_{2.5}$ ($\mu g/m^3$)	O_3 最大八小时滑动平均 ($\mu g/m^3$)	CO (mg/m^3)	AQI	首要污染物
2021/7/13	监测点 A2	5	19	32	11	91	0.5	49	无
2021/7/14	监测点 A2	5	19	33	12	90	0.5	49	无
2021/7/15	监测点 A2	5	19	33	12	92	0.5	48	无

6.4.4 监测点 A3 污染物浓度及 AQI 预测结果表

预报日期	地点	二次模型日值预测							
		SO_2 ($\mu g/m^3$)	NO_2 ($\mu g/m^3$)	PM_{10} ($\mu g/m^3$)	$PM_{2.5}$ ($\mu g/m^3$)	O_3 最大八小时滑动平均 ($\mu g/m^3$)	CO (mg/m^3)	AQI	首要污染物
2021/7/13	监测点 A3	4	13	20	10	80	0.4	40	无
2021/7/14	监测点 A3	4	14	21	10	81	0.4	39	无
2021/7/15	监测点 A3	4	14	21	11	81	0.4	40	无

6.5 结果分析

6.5.1 监测点 A AQI 预测结果与实际结果比对图

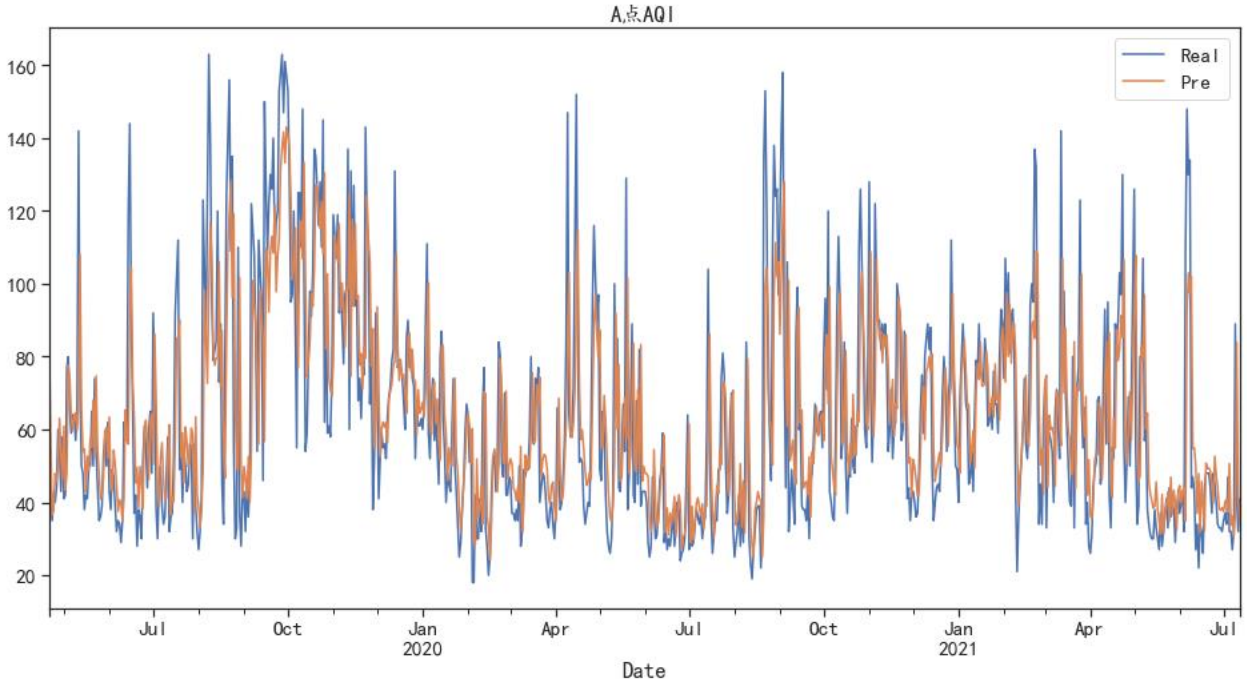


图 6.6 监测点 A　AQI 预测结果与实际结果比对图

6.5.2 监测点 A1 AQI 预测结果与实际结果比对图

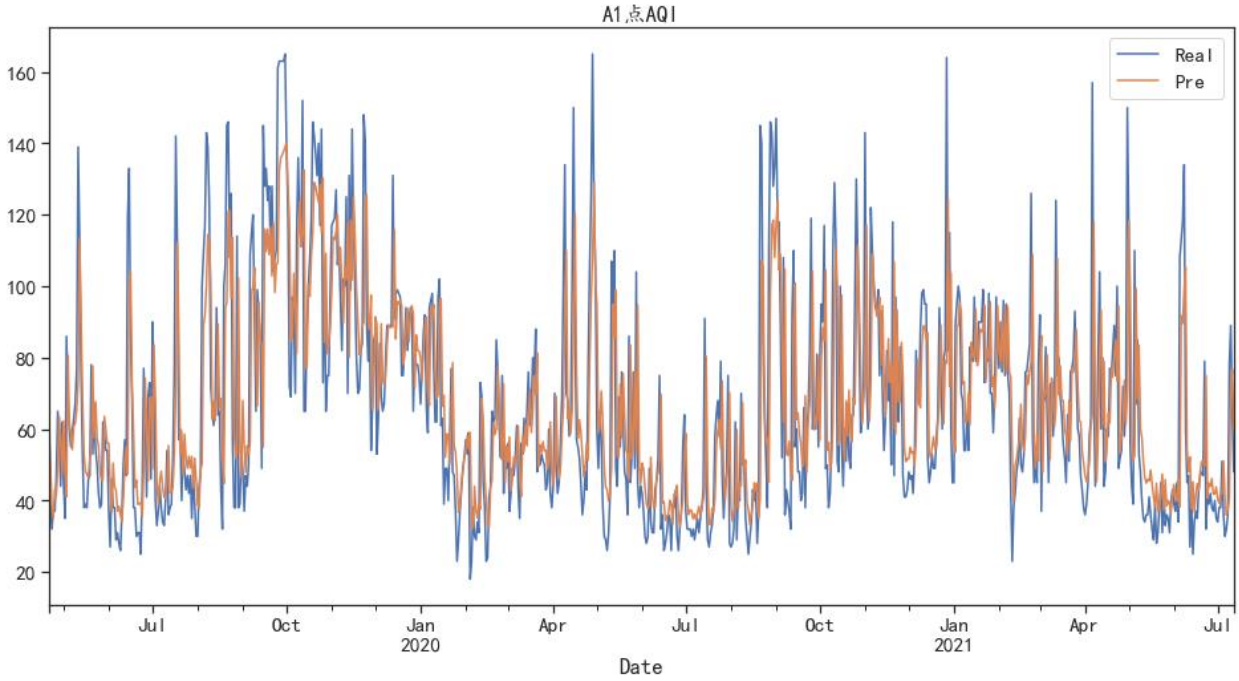


图 6.7 监测点 A1　AQI 预测结果与实际结果比对图

6.5.3 监测点 A2 AQI 预测结果与实际结果比对图

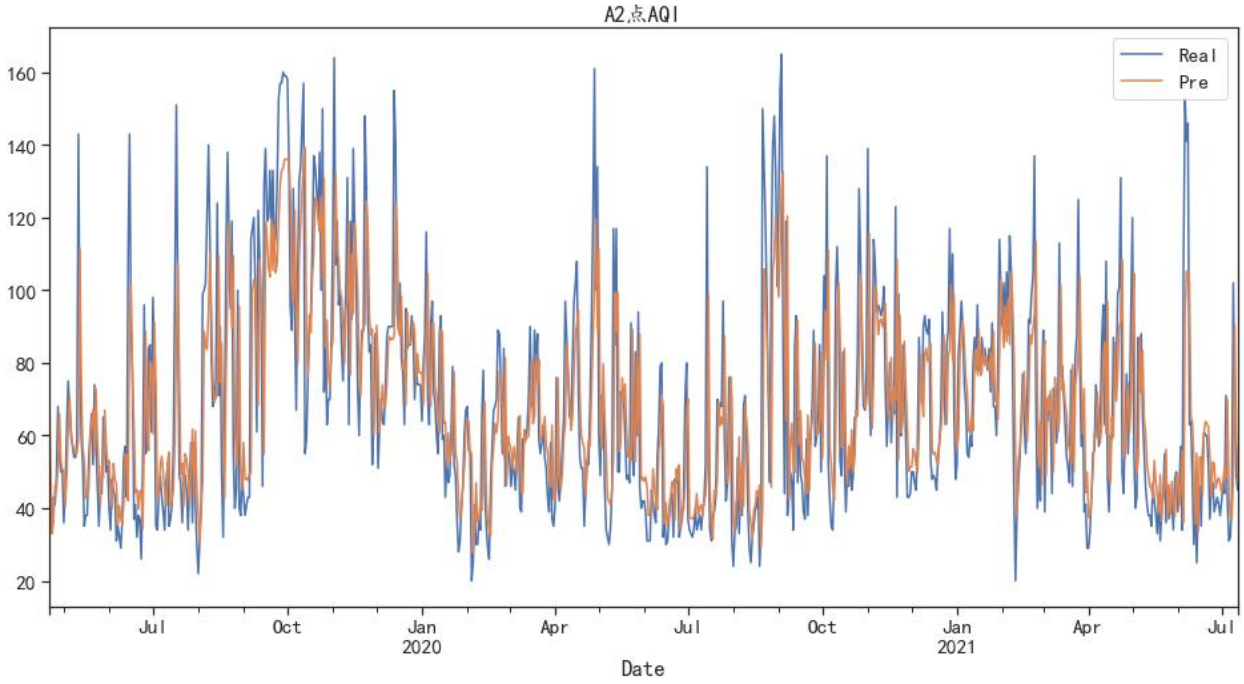


图 6.8 监测点 A2　AQI 预测结果与实际结果比对图

6.5.4 监测点 A3 AQI 预测结果与实际结果比对图

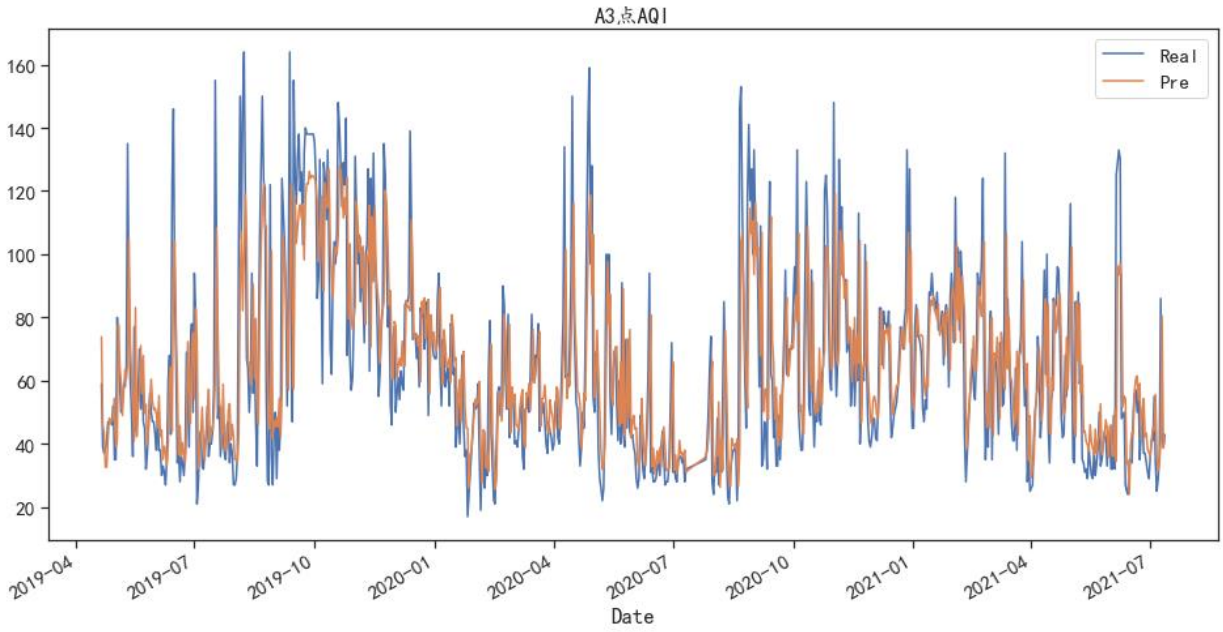


图 6.9 监测点 A3　AQI 预测结果与实际结果比对图

6.5.5 分析

通过监测点 A、A1、A2、A3 AQI 预测结果与实际结果比对图可以发现预测结果与实际结果相关度很高，因此可以得出预测模型的预测效果良好的结论。

七、模型的总结与评价

参考文献

[1] 《环境科学大辞典》编委会. 环境科学大辞典(修订版)[M]. 中国环境科学出版社, 2008.
[2] 刘茜.大气污染的危害及治理对策研究[J].皮革制作与环保科技,2021,2(14):72-74.
[3] 张宇镭,党琰,贺平安.利用 Pearson 相关系数定量分析生物亲缘关系[J].计算机工程与应用, 2005(33):79-82.

附录

附录 1：问题 1 代码

```
# 读取数据
import pandas as pd
dataQ1 = pd.read_excel("Q1_Data.xlsx", sheet_name=1)
# 将最后三行空数据删除
```

```
dataQ1 = dataQ1.iloc[0: -3, :]
# 可视化方法查看缺失的数据
%config InlineBackend.figure_format = 'retina'
%matplotlib inline
import missingno as msno
import matplotlib.pyplot as plt
msno.matrix(dataQ1)
plt.show()
import seaborn as sns
sns.set(font= "Kaiti",style="ticks",font_scale=1.4)
color = dict(boxes='DarkGreen', whiskers='DarkOrange', medians='DarkBl
ue', caps='Gray')  # 颜色设置
box=dataQ1.plot.box(figsize=(12,8),return_type='dict',color=color)
plt.title('异常值检测-箱型图',fontsize=14,pad=12)
from scipy.stats import kstest
import numpy as np

# 正态分布检测
def ksNormDetect(data):
    mu = data.mean()[0]
    sigma = data.std()[0]
    res = kstest(data, "norm", (mu, sigma))[0]
    if res > 0.05:
        return mu, sigma, True
    else:
        return mu, sigma, False

# 异常数据监测
def outLier(data):
    mu, sigma, isKsNorm = ksNormDetect(data)
    if isKsNorm:
        data = data[np.abs(data - mu) <= 3 * sigma]
        return data
    else:
        q1 = data.quantile(q=0.25)
        q3 = data.quantile(q=0.75)
        lowWhisker = q1 - 1.5 * (q3 - q1)
        upWhisker = q3 + 1.5 * (q3 - q1)
        data = data[(data <= upWhisker) & (data >= lowWhisker)]
        return data

# 查找异常值并置为NaN
for _ in dataQ1.columns.to_list()[2: ]:
    data = pd.DataFrame(dataQ1, columns=[_])
```

```
    data = outLier(data)
    dataQ1[_] = data

print(dataQ1.isna().sum())
# 定义修复缺失值的方法
def fixNaN(dataCol):
    fillBackward = dataCol.fillna(method='bfill')
    fillForward = dataCol.fillna(method='ffill')
    return (fillBackward + fillForward) / 2

# 对缺失值进行修复
dataQ1.iloc[:, 2: ] = dataQ1.iloc[:, 2: ].apply(func=fixNaN)
# 可视化方法查看修复后的数据
msno.matrix(dataQ1)
plt.show()

# 查看数据修复后的箱型图
import seaborn as sns
sns.set(font= "Kaiti",style="ticks",font_scale=1.4)
color = dict(boxes='DarkGreen', whiskers='DarkOrange', medians='DarkBl
ue', caps='Gray')  # 颜色设置
box=dataQ1.plot.box(figsize=(12,8),return_type='dict',color=color)
plt.title('箱型图',fontsize=14,pad=12)
# 查找位置
def findIndex(x):
    x = list(iter(x))
    index = [_ for _, x in enumerate(x) if bool(x) == True]
    return index

def IAQIp(Cp, BP, IAQI, isO3=False):
    IAQIpVal = 0
    if isO3 and (Cp > 800):
        IAQIpVal = "NaN"
    else:
        index = findIndex(np.array(BP) <= Cp)[-1]
        IAQIpVal = (IAQI[index + 1] - IAQI[index]) / (BP[index + 1] -
BP[index]) * (Cp - BP[index]) + IAQI[index]
        IAQIpVal = np.ceil(IAQIpVal)
    return IAQIpVal
IAQIpAll = dataQ1.iloc[:, 2: ]
IAQI = [0, 50, 100, 150, 200, 300, 400, 500]
# 计算所有的 IAQIp
# SO2
```

```
IAQIpAll["SO2"] = IAQIpAll["SO2"].apply(func=IAQIp, BP=[0, 50, 150, 47
5, 800, 1600, 2100, 2620], IAQI=IAQI, isO3=False)
# NO2
IAQIpAll["NO2"] = IAQIpAll["NO2"].apply(func=IAQIp, BP=[0, 40, 80, 180
, 280, 565, 750, 940], IAQI=IAQI, isO3=False)
# PM10
IAQIpAll["PM10"] = IAQIpAll["PM10"].apply(func=IAQIp, BP=[0, 50, 150,
250, 350, 420, 500, 600], IAQI=IAQI, isO3=False)
# PM2.5
IAQIpAll["PM2.5"] = IAQIpAll["PM2.5"].apply(func=IAQIp, BP=[0, 35, 75,
 115, 150, 250, 350, 500], IAQI=IAQI, isO3=False)
# O3
IAQIpAll["O3"] = IAQIpAll["O3"].apply(func=IAQIp, BP=[0, 100, 160, 215
, 265, 800], IAQI=IAQI, isO3=True)
# CO
IAQIpAll["CO"] = IAQIpAll["CO"].apply(func=IAQIp, BP=[0, 2, 4, 14, 24,
 36, 48, 60], IAQI=IAQI, isO3=False)

IAQIpAll["Date"] = dataQ1["Date"]
IAQIpAll.isna().sum()
AQIAll = pd.DataFrame({"Date": IAQIpAll["Date"], "AQI": IAQIpAll.iloc[
:, 0: 6].apply(max, axis=1)})
major = []
name = np.array(['SO2', 'NO2', 'PM10', 'PM2.5', 'O3', 'CO'])
for _ in AQIAll.index:
    val = findIndex((IAQIpAll.iloc[_, 0: 6] == AQIAll["AQI"][_]) & (AQ
IAll["AQI"][_] > 50))
    major.append(name[val])
AQIAll["Major"] = major
```

附录 2：问题 2 代码

```
import pandas as pd

dataQ2 = pd.read_excel('Q2_Data.xlsx', sheet_name=0)
from scipy.stats import kstest
import numpy as np

def ksNormDetect(data):
    mu = data.mean()[0]
    sigma = data.std()[0]
    res = kstest(data, "norm", (mu, sigma))[0]

    if res > 0.05:
        return mu, sigma, True
    else:
```

```
        return mu, sigma, False

def outLier(data):
    mu, sigma, isKsNorm = ksNormDetect(data)
    if isKsNorm:
        data = data[np.abs(data - mu) <= 3 * sigma]
        return data
    else:
        q1 = data.quantile(q=0.25)
        q3 = data.quantile(q=0.75)
        lowWhisker = q1 - 1.5 * (q3 - q1)
        upWhisker = q3 + 1.5 * (q3 - q1)
        data = data[(data <= upWhisker) & (data >= lowWhisker)]
        return data

# 查找异常值并置为 NaN
for _ in dataQ2.columns.to_list()[2: ]:
    data = pd.DataFrame(dataQ2, columns=[_])
    data = outLier(data)
    dataQ2[_] = data

print(dataQ2.isna().sum())
# 定义修复缺失值的方法
def fixNaN(dataCol):
    fillBackward = dataCol.fillna(method='bfill')
    fillForward = dataCol.fillna(method='ffill')
    return (fillBackward + fillForward) / 2

# 对缺失值进行修复
dataQ2.iloc[:, 2: ] = dataQ2.iloc[:, 2: ].apply(func=fixNaN)
from sklearn.preprocessing import StandardScaler
import numpy as np

q1 = dataQ2.iloc[:, 2:].values
q1 = np.array(q1).astype('float64')
q1 = StandardScaler().fit_transform(q1)
q1 = q1.transpose(1, 0)
my_rho = np.corrcoef(q1)
print(my_rho)
import matplotlib.pyplot as plt

plt.matshow(my_rho, cmap=plt.cm.cool, vmin=-1, vmax=1)
plt.colorbar()
plt.show()
```

附录 3：问题 3、4 代码

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

dataQ3 = pd.read_excel('Q3_Data.xlsx', sheet_name=0)

dataQ3 = dataQ3.dropna()
print(len(dataQ3))
dataset = dataQ3.iloc[:, 2: 9].values
dataset = dataset.astype('float32')

from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(0, 1))
dataset = scaler.fit_transform(dataset)

def create_dataset(dataset, look_back=5):
    dataX, dataY = [], []
    for i in range(len(dataset) - look_back):
        a = dataset[i: (i + look_back)]
        dataX.append(a)
        dataY.append(dataset[i + look_back])
    return np.array(dataX), np.array(dataY)

look_back = 5
data_X, data_Y = create_dataset(dataset, look_back)
print(len(data_X), len(data_Y))

train_size = int(len(data_X) * 7)
test_size = len(data_X) - train_size
train_X = data_X#[: train_size]
train_Y = data_Y#[: train_size]
test_X = data_X#[train_size:]
test_Y = data_Y#[train_size:]
print(train_X.shape)
print(train_Y.shape)
print(test_X.shape)
print(train_size)
print(len(data_X))

import torch

train_X = train_X.transpose(0, 2, 1)
print(train_X.shape)
```

```
train_Y = train_Y.reshape(-1, 7, 1)
print(train_Y.shape)
test_X = test_X.transpose(0, 2, 1)

train_x = torch.from_numpy(train_X)
train_y = torch.from_numpy(train_Y)
test_x = torch.from_numpy(test_X)

from torch import nn
from torch.autograd import Variable

class lstm(nn.Module):
    def __init__(self, input_size=2, hidden_size=4, output_size=1, num
_layer=2):
        super(lstm, self).__init__()
        self.layer1 = nn.LSTM(input_size, hidden_size, num_layer)
        self.layer2 = nn.Linear(hidden_size, output_size)

    def forward(self, x):
        # x = x.view(len(x), 1, -1)
        x, _ = self.layer1(x)
        s, b, h = x.size()
        x = x.view(s * b, h)
        x = self.layer2(x)
        x = x.view(s, b, -1)

        return x

model = lstm(5, 4, 1, 2)

criterion = nn.MSELoss()
optimizer = torch.optim.Adam(model.parameters(), lr=1e-2)

for _ in range(1000):
    var_x = Variable(train_x)
    var_y = Variable(train_y)

    out = model(var_x)
    loss = criterion(out, var_y)

    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
```

```
if (_ + 1) % 100 == 0:
    print('Epoch: {}, Loss: {:.5f}'.format(_ + 1, loss.item()))
```