

可视化与可视分析 实验技术课程指导书

王松 陈仕杰 胡浩 编著

·

2019年7月编制

**When**

**Data Visualization**

**Meets**

**Data Analytics**

**可视化**

**让数据分析触手可及**

**目录**

[第一章 数据预处理的基本方法 1](#_Toc30740)

[1.1 基本含义 1](#_Toc3309)

[1.2 Python 2](#_Toc12466)

[1.2.1 Python基本介绍 2](#_Toc26853)

[1.2.2 Python安装 3](#_Toc13945)

[1.2.3 Python自定义函数 5](#_Toc2339)

[1.3 Pandas 5](#_Toc17704)

[1.3.1 Pandas基本介绍 5](#_Toc636)

[1.3.2 如何使用Pandas 6](#_Toc3593)

[1.3.3 Pandas数据处理 7](#_Toc6580)

[第二章 可视化基本图形绘制 20](#_Toc16617)

[2.1 D3.js 20](#_Toc11547)

[2.1.1 D3.js是什么 20](#_Toc16836)

[2.1.2 如何学习和使用D3 20](#_Toc15353)

[2.1.3 安装D3 20](#_Toc6396)

[2.1.4 饼状图的制作 21](#_Toc23056)

[2.1.5 柱状图的制作 23](#_Toc5642)

[2.2 本章小作业 26](#_Toc23847)

[第三章 可视化地图技术 28](#_Toc3361)

[3.1 MapBox地图实例 28](#_Toc18932)

[3.1.1 热力地图 28](#_Toc22311)

[3.1.2 Echarts结合Mapbox实例 32](#_Toc9916)

[3.2 本章小作业 34](#_Toc19422)

[第四章 可视化交互技术 36](#_Toc29830)

[4.1 d3可视化交互 36](#_Toc30551)

[4.1.1 添加交互 36](#_Toc27062)

[4.1.2带有交互的柱状图 37](#_Toc3735)

[4.2 Echarts可视化交互 38](#_Toc9895)

[4.2.1 Echarts交互介绍 38](#_Toc6734)

[4.2.2.Echarts交互组件 40](#_Toc23556)

[4.2.3.基于Echarts的多视图交互 42](#_Toc4171)

[4.3 本章小作业 47](#_Toc22765)

[第五章 可视化综合案例 51](#_Toc6548)

[5.1 系统设计 51](#_Toc26405)

[5.2本章小作业 64](#_Toc29265)

[第六章 可视化系统设计与评估 65](#_Toc19937)

[6.1 数据可视化设计流程 66](#_Toc31204)

[6.2 数据可视化系统开发流程 68](#_Toc22517)

[6.2.1数据采集 68](#_Toc31970)

[6.2.2数据处理和转换 68](#_Toc24683)

[6.2.3可视化展示（映射） 69](#_Toc20435)

[6.2.4用户交互 69](#_Toc16583)

[6.2.5可视化评估 69](#_Toc18648)

[6.2.6 课题小组讨论 70](#_Toc5770)

# 数据预处理的基本方法

## 1.1 基本含义

现实世界中数据大体上都是不完整，不一致的脏数据，无法直接进行数据挖掘，或挖掘结果差强人意。为了提高数据挖掘的质量产生了数据预处理技术。 　数据预处理有多种方法：数据清理，数据集成，数据变换，数据归约等。数据预处理对最终结果起着至关重要的作用，如果你的数据没有进行数据清洗等操作，则可能导致最终模型或结论出现完全错误的结果。

数据的预处理是指对所收集数据进行分类或分组前所做的审核、筛选、排序等必要的处理。其主要步骤如下：

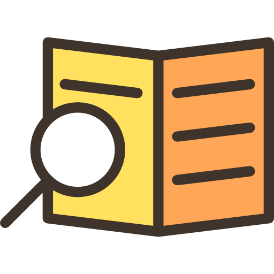
对于目前主流数据清洗软件，主要有EXCEL、SPSS、R语言、Python语言、MySQL等，本文主要采用Python中的第三方库Pandas进行数据预处理，故此只针对其进行详细介绍，其余请另行百度。

## 1.2 Python

想要通过Python来开启数据可视化之旅的朋友，需要什么预备知识呢？

• C语言：了解计算机基本语言的语法格式

• 高等数学：一定的数学功底，对一些基本算法有一定的研究



### 1.2.1 Python基本介绍

Python语言又名蟒蛇语言，是一种高度结合了解释性、编译性、互动性、与面向对象的脚本语言，其代码风格具有较强的可读性，相较于其他编程语言，Python可采用中文作为变量，但一般不建议这么操作，另其语法结构明确，是数据处理、数据挖掘的不二之选。

对于初学者而言，Python是最好的入门语言，其第三方库不断增加，减少了大部分程序员的负担。

下面就是一个最简单的Python演示程序。

Python语言具有以下特点：

1.易于学习：Python有相对较少的关键字，结构简单，和一个明确定义的语法，学习起来更加简单。

2.易于阅读：Python代码定义的更清晰。

3.易于维护：Python的成功在于它的源代码是相当容易维护的。

4.一个广泛的标准库：Python的最大的优势之一是丰富的库，跨平台的，在UNIX，Windows和Macintosh兼容很好。

5.互动模式：互动模式的支持，您可以从终端输入执行代码并获得结果的语言，互动的测试和调试代码片断。

6.可移植：基于其开放源代码的特性，Python已经被移植（也就是使其工作）到许多平台。

7.可扩展：如果你需要一段运行很快的关键代码，或者是想要编写一些不愿开放的算法，你可以使用C或C++完成那部分程序，然后从你的Python程序中调用。

8.数据库：Python提供所有主要的商业数据库的接口。

9.GUI编程：Python支持GUI可以创建和移植到许多系统调用。

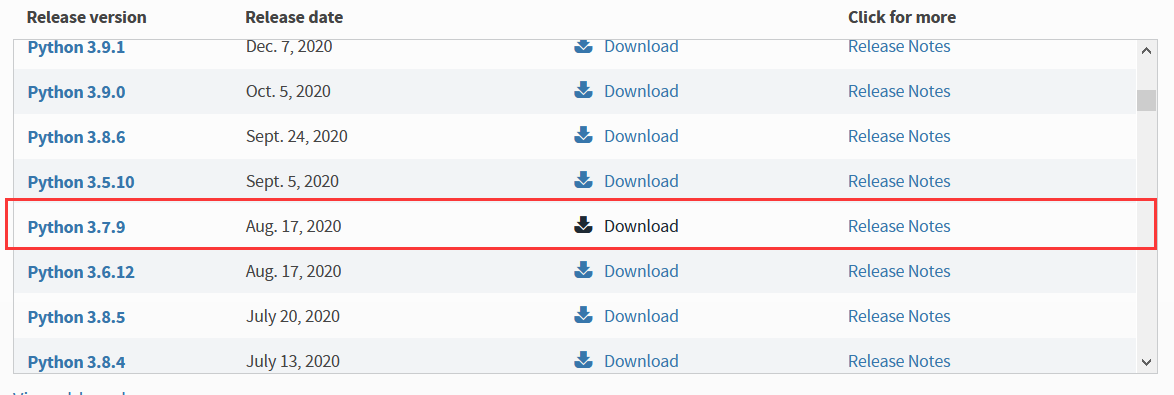
10.可嵌入: 你可以将Python嵌入到C/C++程序，让你的程序的用户获得"脚本化"的能力。



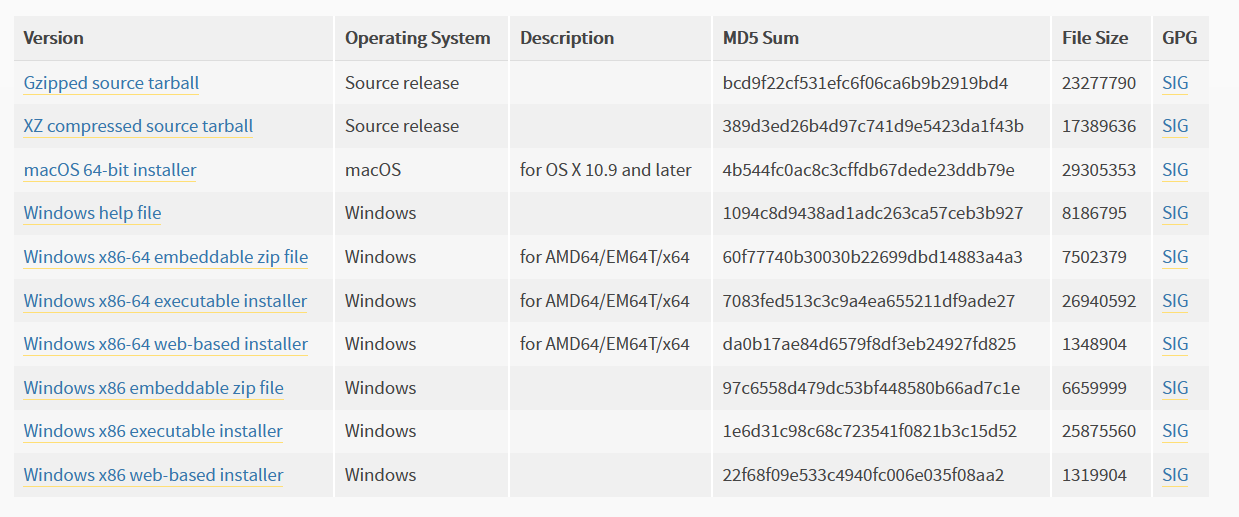
1. print("hello world")

### 1.2.2 Python安装

打开python官网访问：https://www.python.org/downloads/，下载python安装包。截至目前，Python已更新至3.9.7，但本文采用3.7版本，因为稳定。

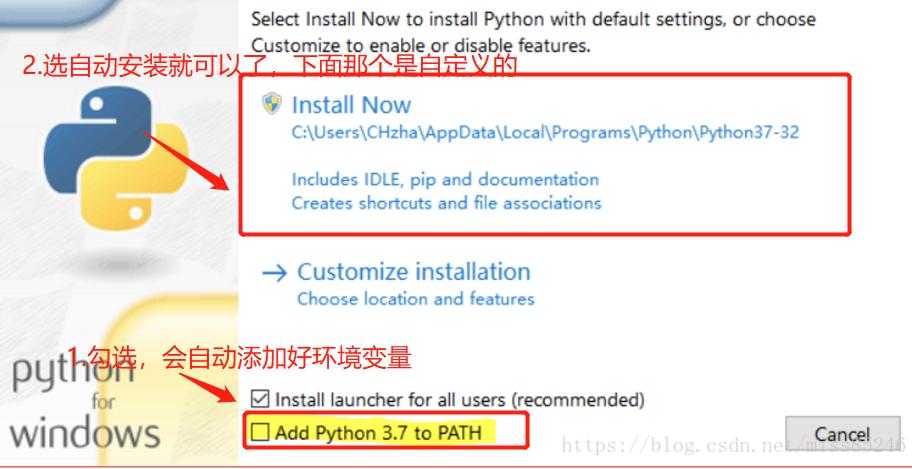


点击上图中的版本号或者“Download”按钮进入对应版本的下载页面，滚动到最后即可看到各个平台的 Python 安装包。

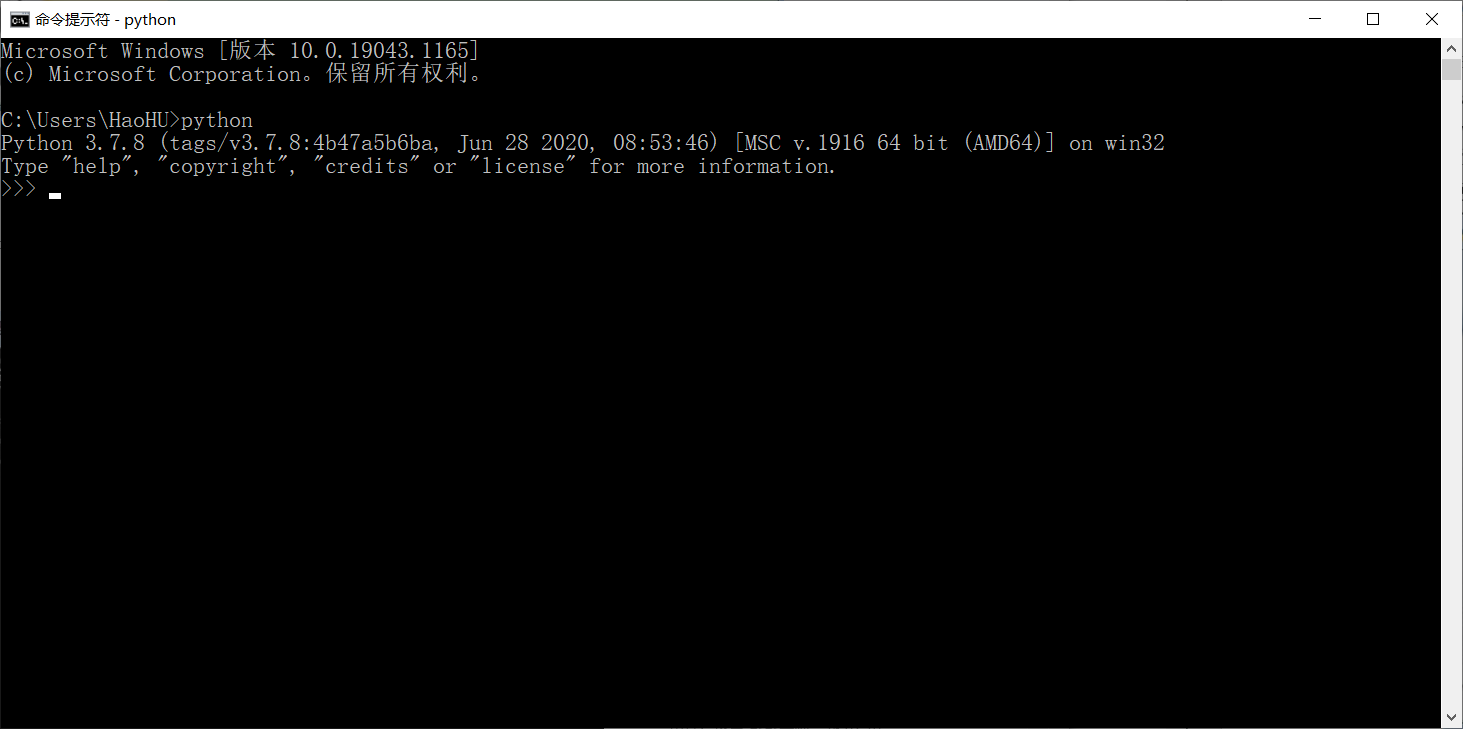


用户可根据个人设备系统要求进行安装，其中windows分为x86与x64版本，Mac用户可下载macOS版本进行安装，或使用brew进行安装，本文仅针对Windows版本用户进行详述。

本文选择的是**Windows x86-64 executable installer，**即64位的完整版安装包，双击下载得到的python可执行文件。



为保证正常运行，请一定勾选add Path选项，安装完成后，请打开DOS窗口，win+R键，输入cmd敲回车，然后输入python敲回车，若提示信息如下所示，则安装成功。



由于Python安装第三方库采用的是国外pypi.org，故此下载速度较慢，本文建议采用国内源进行替换，速度得到成倍提升，以下进行简述，打开windows资源管理器，在地址栏输入C:\Users\<你的用户名>，如下所示



此处新建pip文件夹，另新建文件名为pip，后缀为ini的文件，其中内容为

1. [global]
2. index-url = https://pypi.tuna.tsinghua.edu.cn/simple/

此处也提供其他源路径，如下：

1. https://pypi.mirrors.ustc.edu.cn/simple/ *#中国科技大学*
2. http://mirrors.aliyun.com/pypi/simple/ *#阿里云*
3. http://pypi.douban.com/simple/ *#豆瓣(douban)*
4. https://pypi.tuna.tsinghua.edu.cn/simple/ *#清华大学*
5. http://pypi.mirrors.ustc.edu.cn/simple/  *#中国科学技术大学*

### 1.2.3 Python自定义函数

函数就是执行具体任务的代码块。说得具体一点，函数的特殊之处表现在它可以接收参数，可以返回值。圆括号可以调用（执行）函数。如果函数需要参数（输入值），那么就在调用时把参数放到圆括号中。Python中有多种方式可以定义函数，下面介绍一种比较简单的方式。

1. def fun(参数)
2. # 函数执行体
3. return 变量
4. fun(参数) #调用方法

Python还提供了强大的第三方库函数，下文将对第三方库函数Pandas进行详细介绍。

## 1.3 Pandas

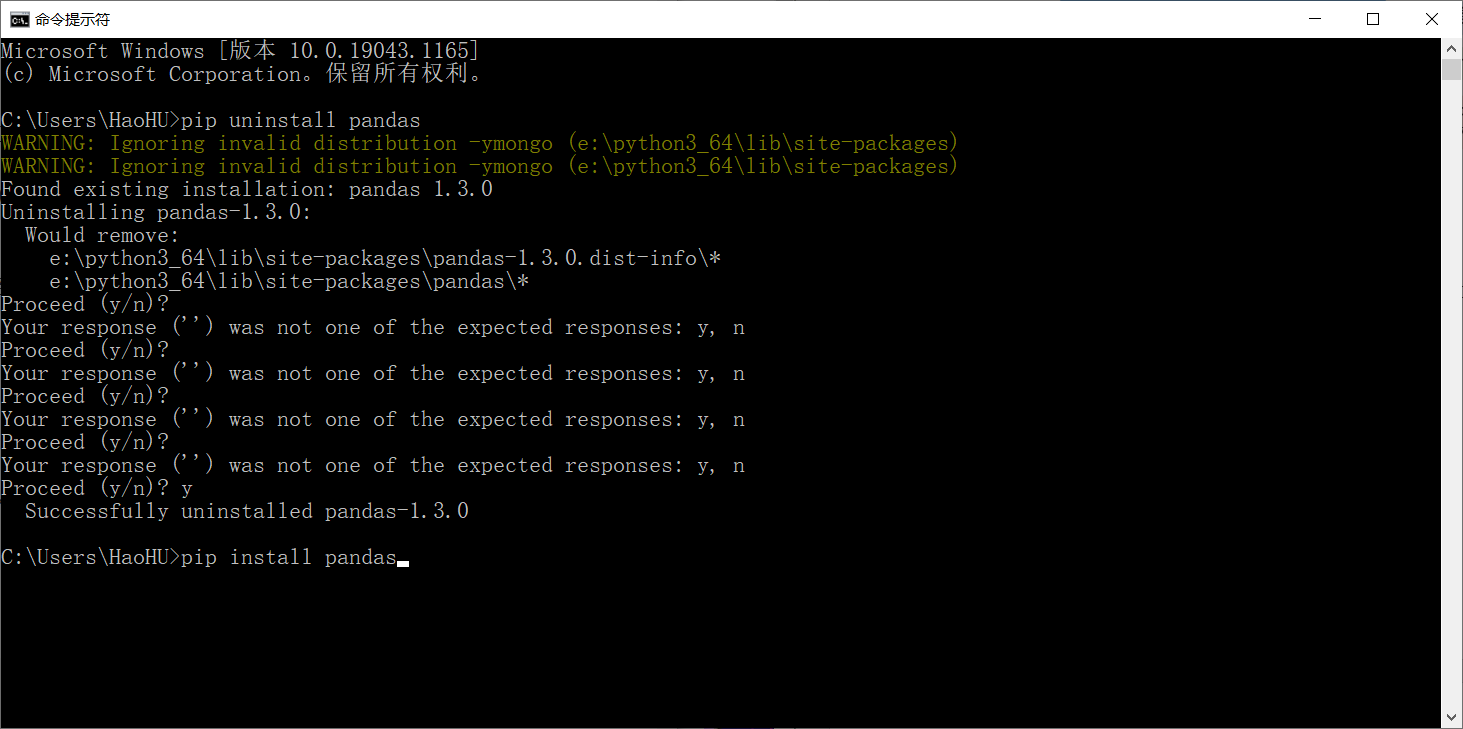
### 1.3.1 Pandas基本介绍

Pandas是python的一个数据分析包，最初由AQR Capital Management于2008年4月开发，并于2009年底开源出来，目前由专注于Python数据包开发的PyData开发team继续开发和维护，属于PyData项目的一部分。Pandas最初被作为金融数据分析工具而开发出来，因此，pandas为时间序列分析提供了很好的支持。 Pandas的名称来自于面板数据（panel data）和python数据分析（data analysis）。panel data是经济学中关于多维数据集的一个术语，在Pandas中也提供了panel的数据类型。

其数据结构类型不同于传统数据类型，主要包含有：

### 1.3.2 如何使用Pandas

首先我们需要对Pandas进行安装，打开cmd窗口，输入命令pip install pandas。敲下回车键。



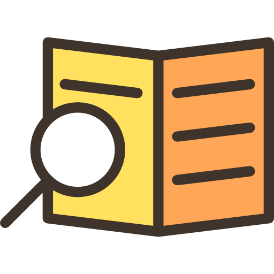
当出现以下字符时，



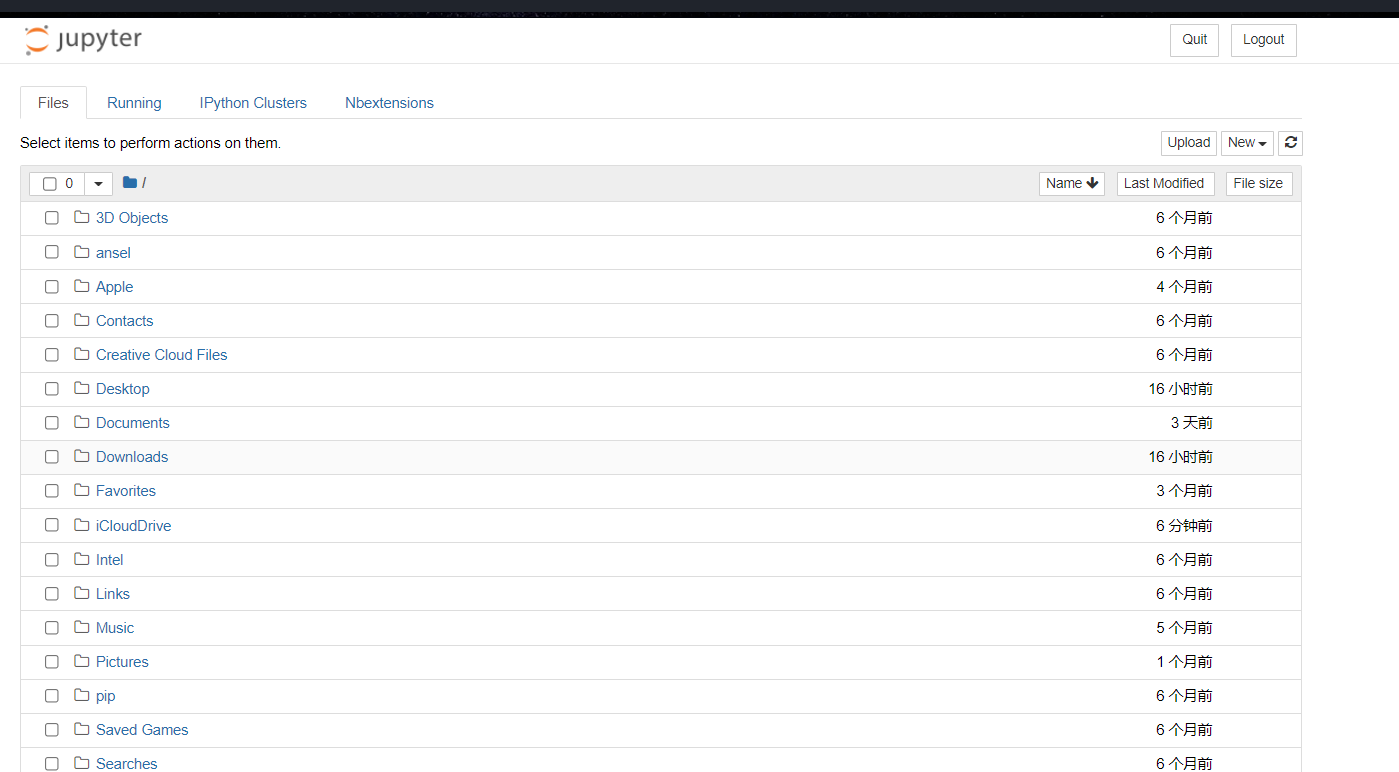
说明已经成功安装上了，在正式教学前，我们还将引入一个非常好用的编译器，Jupyter notebook。

Jupyter Notebook是基于网页的用于交互计算的应用程序。其可被应用于全过程计算：开发、文档编写、运行代码和展示结果。

简而言之，Jupyter Notebook是以网页的形式打开，可以在网页页面中直接编写代码和运行代码，代码的运行结果也会直接在代码块下显示。如在编程过程中需要编写说明文档，可在同一个页面中直接编写，便于作及时的说明和解释。



安装方式与pandas安装大同小异，此处不在详述，安装命令为pip install jupyter notebook，安装完成后，在cmd窗口中输入命令 jupyter notebook即可打开如图所示界面，



或在某一文件夹地址栏输入jupyter notebook命令也可打开对应文件夹下内容。

### 1.3.3 Pandas数据处理

#### 1.3.3.1 数据预处理

Pandas需要先读取表格类型的数据，然后进行分析，下图将展示pandas能够读取的各种类型

例如读取csv文件可用函数：

1. data = pd.read\_csv(filepath\_or\_buffer, sep=',', delimiter=None, header='infer', names=None, index\_col=None, usecols=None,squeeze=False,prefix=None, mangle\_dupe\_cols=True, dtype=None, engine=None, converters=None, true\_values=None, false\_values=None, skipinitialspace=False,skiprows=None, nrows=None, na\_values=None, keep\_default\_na=True, na\_filter=True, verbose=False, skip\_blank\_lines=True, parse\_dates=False,infer\_datetime\_format=False, keep\_date\_col=False, date\_parser=None, dayfirst=False, iterator=False, chunksize=None, compression='infer', thousands=None, decimal=b'.', lineterminator=None, quotechar='"', quoting=0, escapechar=None, comment=None, encoding=None, dialect=None, tupleize\_cols=False, error\_bad\_lines=True, warn\_bad\_lines=True, skipfooter=0, skip\_footer=0, doublequote=True, delim\_whitespace=False, as\_recarray=False, compact\_ints=False, use\_unsigned=False, low\_memory=True, buffer\_lines=None, memory\_map=False, float\_precision=None)

函数中参数大致如下：

1.filepath\_or\_buffer:（这是唯一一个必须有的参数，其它都是按需求选用的）

文件所在处的路径

2.sep：

指定分隔符，默认为逗号','

3.delimiter : str, default None

定界符，备选分隔符（如果指定该参数，则sep参数失效）

4.header：int or list of ints, default ‘infer’

指定哪一行作为表头。默认设置为0（即第一行作为表头），如果没有表头的话，要修改参数，设置header=None

5.names：

指定列的名称，用列表表示。一般我们没有表头，即header=None时，这个用来添加列名就很有用啦！

6.index\_col:

指定哪一列数据作为行索引，可以是一列，也可以多列。多列的话，会看到一个分层索引

7.prefix:

给列名添加前缀。如prefix="x",会出来"x1"、"x2"、"x3"酱纸

8.nrows : int, default None

需要读取的行数（从文件头开始算起）

9.encoding:

乱码的时候用这个就是了，官网文档看看用哪个：

https:*//docs.python.org/3/library/codecs.html#standard-encodings*

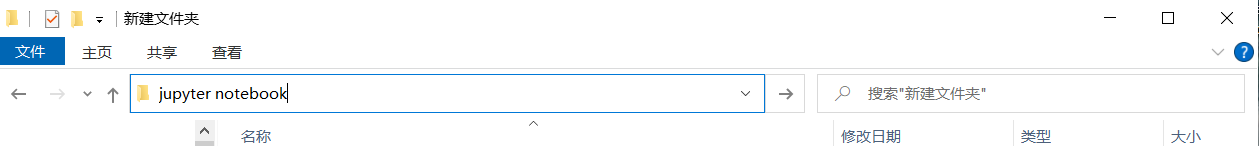
10.skiprows : list-like or integer, default None

需要忽略的行数（从文件开始处算起），或需要跳过的行号列表（从0开始）

当我们读取完数据之后，我们可以则可以进行数据处理工作，接下来的数据处理工作，我们将在实验作业中进行讲述。

#### 1.3.3.3 实验作业：

本文将以某一竞赛中的气象数据进行案例展示，首先我们打开Jupyter，这种打开方式能够有效的基于某一文件夹下对数据进行复制粘贴等操作。



首先将本文数据导入文件夹中，此处我们对数据字段进行简要描述（本数据集来自ChinaVis2021数据挑战赛）。

**每日分析数据**: 日均值数据

|  |  |  |
| --- | --- | --- |
| 时间范围 | 2018年1月 | 共31个csv文件，每个文件以日期命名，包含当月的日均值数据。  采用csv文件格式，分隔符为逗号。包含当日污染物浓度和气象要素的日均值。 |
| 空间范围 | 中国 |
| 空间分辨率 | 15公里 |
| 位置变量 | 网格中心纬度（）、网格中心经度（） |
| 污染物变量 | 细颗粒物（）、可吸入颗粒物（）、臭氧（）、一氧化碳（）、二氧化硫（）、二氧化氮（） |
| 气象变量 | 纬向风速（）、经向风速（）、温度（）、相对湿度（）、地面气压（） |

字段说明：

每个csv文件总共包含 13 列，分别给出了我国6项常规污染物、5 个常用气象要素以及所在网格点的经纬度值。数据集中包含的变量及其具体说明如下表所示：

|  |  |
| --- | --- |
| 变量名 | 释义 |
|  | 空气动力学直径小于 2.5 微米的颗粒物，单位：微克每立方米。形成灰霾污染的关键污染物。 |
|  | 臭氧，单位：微克每立方米。 光化学污染的关键污染物。 |
|  | 空气动力学直径小于10微米的颗粒物，单位：微克每立方米。在环境空气中长期飘浮的悬浮微粒，对大气能见度影响很大。 |
|  | 二氧化硫，单位：微克每立方米。的重要前体物之一，可在大气中被氧化形成硫酸盐气溶胶。 |
|  | 二氧化氮，单位：微克每立方米。和的重要前体物，一方面可在大气中被氧化形成硝酸盐气溶胶，另一方面参与大气光化学反应形成臭氧。 |
|  | 一氧化碳，单位：微克每立方米。含碳物质燃烧不完全时的产物，高浓度时能使人出现不同程度中毒症状。 |
|  | 纬向风速，单位：米每秒。影响大气污染物浓度变化的关键气象要素之一。 |
|  | 经向风速，单位：米每秒。影响大气污染物浓度变化的关键气象要素之一。 |
|  | 气温，单位：开尔文（）。影响大气污染物浓度变化的关键气象要素之一。 |
|  | 相对湿度，指空气中水汽压与相同温度下饱和水汽压的百分比。影响大气污染物浓度变化的关键气象要素之一。 |
|  | 地面气压，单位帕斯卡（）。影响大气污染物浓度变化的关键气象要素之一。 |
|  | 网格中心纬度（北纬），单位：° |
|  | 网格中心经度（东经），单位：° |

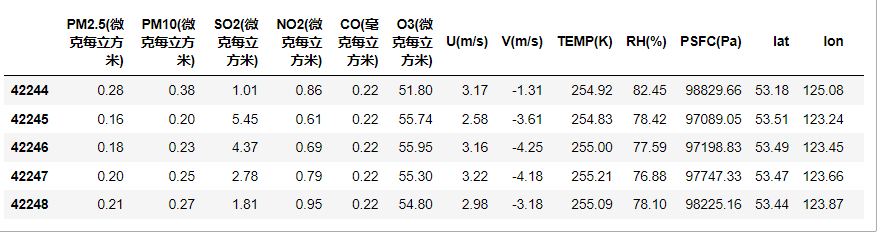
导入后，正式开始书写数据处理代码，首先，我们读入一个csv文件进行预读取。

data = pd.read\_csv("201801/CN-Reanalysis-daily-2018010100.csv")

数据读取完成后，我们可以使用如下函数进行数据查看：

1. data.head() *# 查看前5行数据*
2. data.tail() *# 查看最后5行数据*

即会出现前5行数据或后5行数据:



到这里我们将带入需求来讲解我们的代码设计。

假设小明、小刚、以及李华同学合作，正在做一个气象数据可视化系统，小明负责后端数据处理部分，李华负责前端框架，小刚负责端茶倒水喊大佬yyds，小刚需要小明处理天气数据并转为json文件，提出了两个需求：

需求1：需要将2018年1月1日的气象数据前10000中的 𝑃𝑀2.5>0.4 且 𝐶𝑂>0.3 数据提出，且将其 𝑃𝑀2.5 、 𝑃𝑀10 、 𝐶𝑂 、 𝐿𝑎𝑡 、 𝐿𝑜𝑛 字段提取出来并组成一个json文件，json格式如下：

1. [{
2. "PM25": 22.81,
3. "PM10": 39.28,
4. "CO": 0.53,
5. "lat": 18.34,
6. "lon": 109.25
7. },
8. ..............
9. {
10. "PM25": 55.35,
11. "PM10": 66.6,
12. "CO": 0.69,
13. "lat": 29.37,
14. "lon": 121.83
15. }]

李华对小刚提出了一个无理的要求，希望小刚能够从找出这些数据的一些要求如下： 求出这个月各个经纬度的 𝑃𝑀2.5 的平均值， 𝑃𝑀10 的最大值以及 𝐶𝑂 的最大值减最小值的差，并找到这些数据中前100条所记录的城市（PS：可根据经纬度求出城市名称）

1. [{
2. "PM25": 0.33,
3. "PM10": 0.36,
4. "CO": 0.1,
5. "lat": 32.64,
6. "lon": 85.09,
7. "city": "西藏自治区"
8. },
9. ..............
10. {
11. "PM25": 7.65,
12. "PM10": 16.0,
13. "CO": 0.17,
14. "lat": 29.96,
15. "lon": 94.71,
16. "city": "西藏自治区"
17. }]

扩展功能：这些城市的聚类，方法使用KMeans聚类。

详细讲解：

首先我们需要进行导入包函数，在导入包函数前，我们需要进行安装，安装包使用命令：pip install pandas，pandas安装完成后，则可以导入包。

1. import pandas as pd
2. *# import 第三方库*
3. *# as 则是指定别名，后期可以使用pd来代表pandas*

当我们把pandas导入之后，我们则可以进行数据的读入操作，键入以下代码：

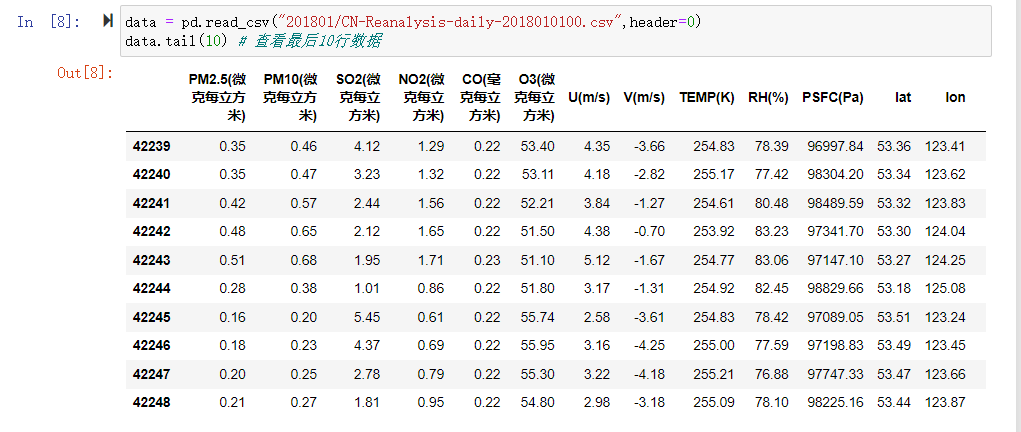
1. data = pd.read\_csv("201801/CN-Reanalysis-daily-2018010100.csv",header=0)

其中header代表表头，当header为0是，则是将第一行作为表头，指定为1则是将第二行作为表头。

读完数据后，输入以下代码

1. data.head() *# 查看前5行数据*
2. data.tail() *# 查看最后5行数据*
3. data.head(10) *# 查看前10行数据*
4. data.tail(10) *# 查看最后10行数据*

其中，head(tail)函数是默认输出前（后）5行数据，当我们在括号中输入如数字10时，则输出前（后）10行数据，如下所示

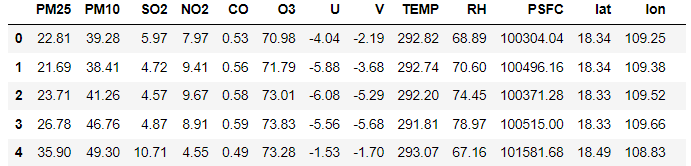


这时我们的数据就顺利读取完成了，这是我们看表头，表头非常复杂，故此需要进行重命名，故此我们使用了一个新函数。

1. data = data.rename({"PM2.5(微克每立方米)":"PM25"," PM10(微克每立方米)":"PM10"," SO2(微克每立方米)":"SO2",
2. " NO2(微克每立方米)":"NO2"," CO(毫克每立方米)":"CO"," O3(微克每立方米)":"O3",
3. " U(m/s)":"U"," V(m/s)":"V"," TEMP(K)":"TEMP"," RH(%)":"RH"," PSFC(Pa)":"PSFC",
4. " lat":"lat"," lon":"lon"},axis=1)

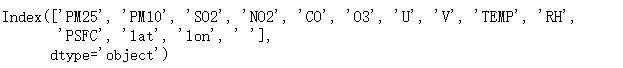
这时候我们如何去评判是否修改是否，成功这时可以输入函数，则可以查看是否修改成功。

1. data.head()



当然也可以使用函数

1. data.columns

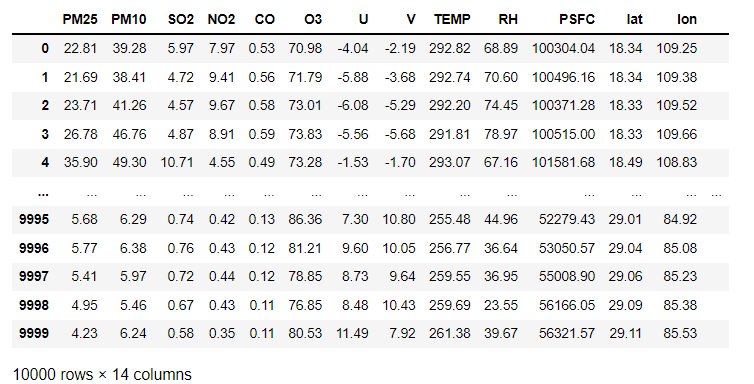


在rename函数中存在一个参数inplace，默认为False,由于pandas的机制，当inplace为false时，pandas会复制一份数据，进行修改并返回修改后的数据，反之则直接在原数据上进行修改，代码如下所示。

1. data.rename({"PM2.5(微克每立方米)":"PM25"," PM10(微克每立方米)":"PM10"," SO2(微克每立方米)":"SO2",
2. " NO2(微克每立方米)":"NO2"," CO(毫克每立方米)":"CO"," O3(微克每立方米)":"O3",
3. " U(m/s)":"U"," V(m/s)":"V"," TEMP(K)":"TEMP"," RH(%)":"RH"," PSFC(Pa)":"PSFC",
4. " lat":"lat"," lon":"lon"},axis=1,inplace=True)

我们将表头修改后，则继续进行数据的提取，在需求1中，首先提取出前10000条数据，代码如下

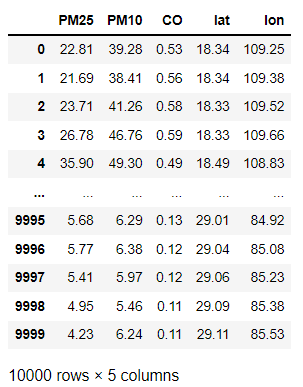
1. data1 = data[:10000]



随后我们提取相关字段，代码如下

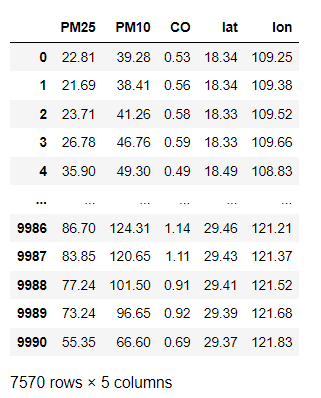
1. data1 = data[:10000][["PM25","PM10","CO","lat","lon"]]

运行结果如下所示



数据字段提取完成后，我们这时候进行数据提取输入以下代码

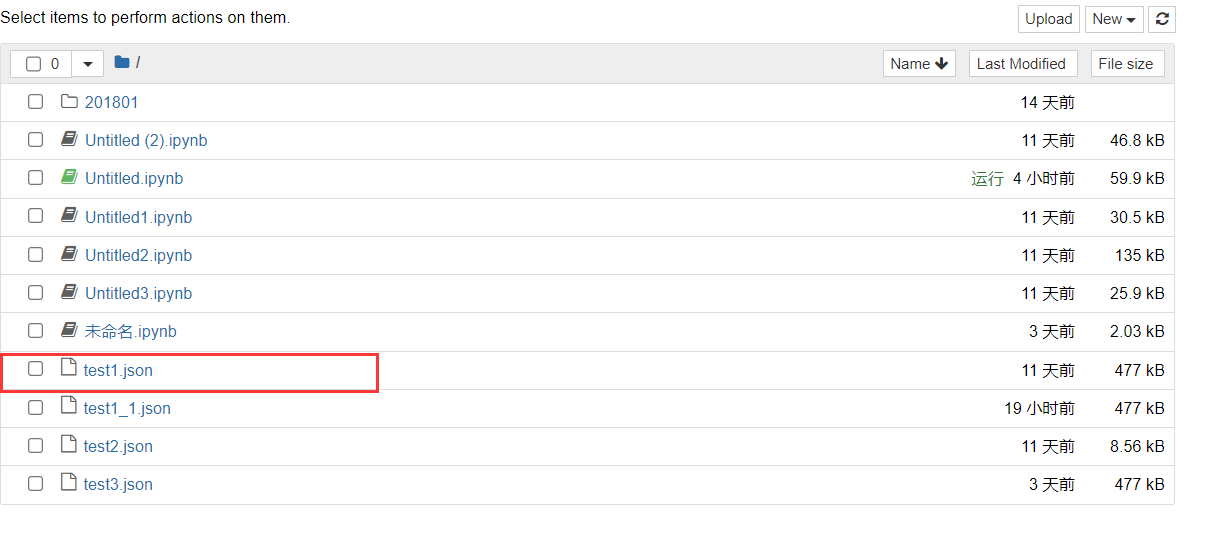
1. data1 = data1[(data1["PM25"]>20) & (data1["CO"]>0.3)]



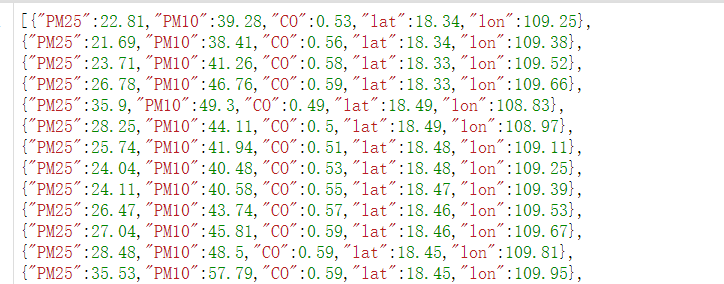
此时需求1任务则已经完成，故此需要将数据导出文件，此时使用函数

1. data1.to\_json("test1.json",orient="records")

这时候文件就导出到我们的文件系统中了。



这时，点开文件则可以看到如下图所示的文件。



针对需求2：首先我们需要导入两个库函数

1. import os
2. import datetime

导入完成后，定义一个命名为readfile的子函数

1. def readfile(file):
2. d = datetime.datetime.strptime(file.split('/')[-1].split('-')[3].split('.')[0][:-2],'%Y%m%d').date()
3. data = pd.read\_csv(file)
4. data = data.rename({"PM2.5(微克每立方米)":"PM25"," PM10(微克每立方米)":"PM10"," SO2(微克每立方米)":"SO2"," NO2(微克每立方米)":"NO2"," CO(毫克每立方米)":"CO"," O3(微克每立方米)":"O3"," U(m/s)":"U", " V(m/s)":"V"," TEMP(K)":"TEMP"," RH(%)":"RH"," PSFC(Pa)":"PSFC"," lat":"lat"," lon":"lon"},axis=1)
5. data['date'] = pd.to\_datetime(d)
6. return data

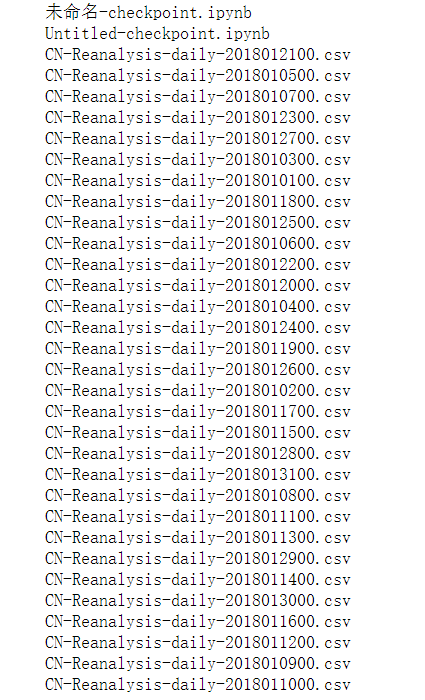
其中函数主要左右为取得文件名并设计date字段，date字段数据为当天的时间。子函数设计完成后，则可以进行后面的代码设计。

其余代码如下所示

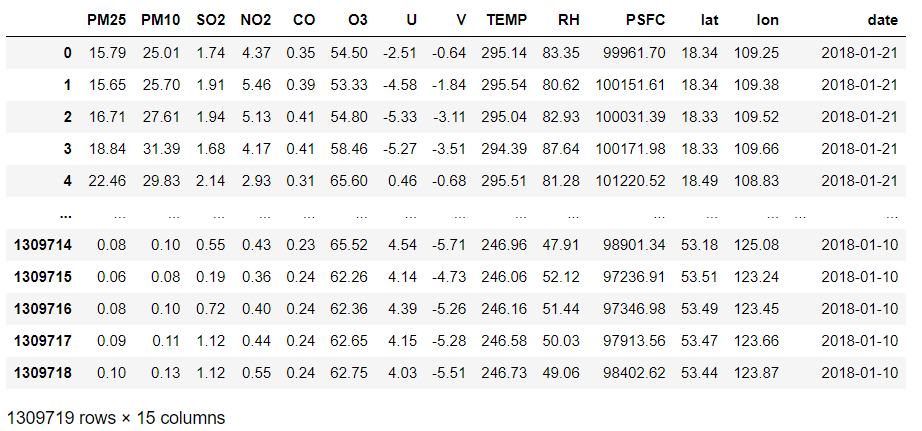
1. paths= os.listdir(os.getcwd()) *#得到文件夹下的所有文件名称*
2. data2 = pd.DataFrame()
3. for path in paths: *#遍历文件夹*
4. if os.path.isdir(path): *#判断是否是文件夹，不是文件夹才打开*
5. files=os.listdir(path)
6. for file in files:
7. if(file.split('.')[-1]=='csv'):
8. data2 = data2.append(readfile(path+'/'+file),ignore\_index=True)

函数os.listdir(os.getcwd())主要作用为读出文件夹下所有文件和文件夹的名称并返回数组类型，接下来进行文件的遍历，找到文件后，进行后缀名判断，当文件后缀名为csv时，即其为csv文件时则进行读取和加入。

运行结果如下所示：



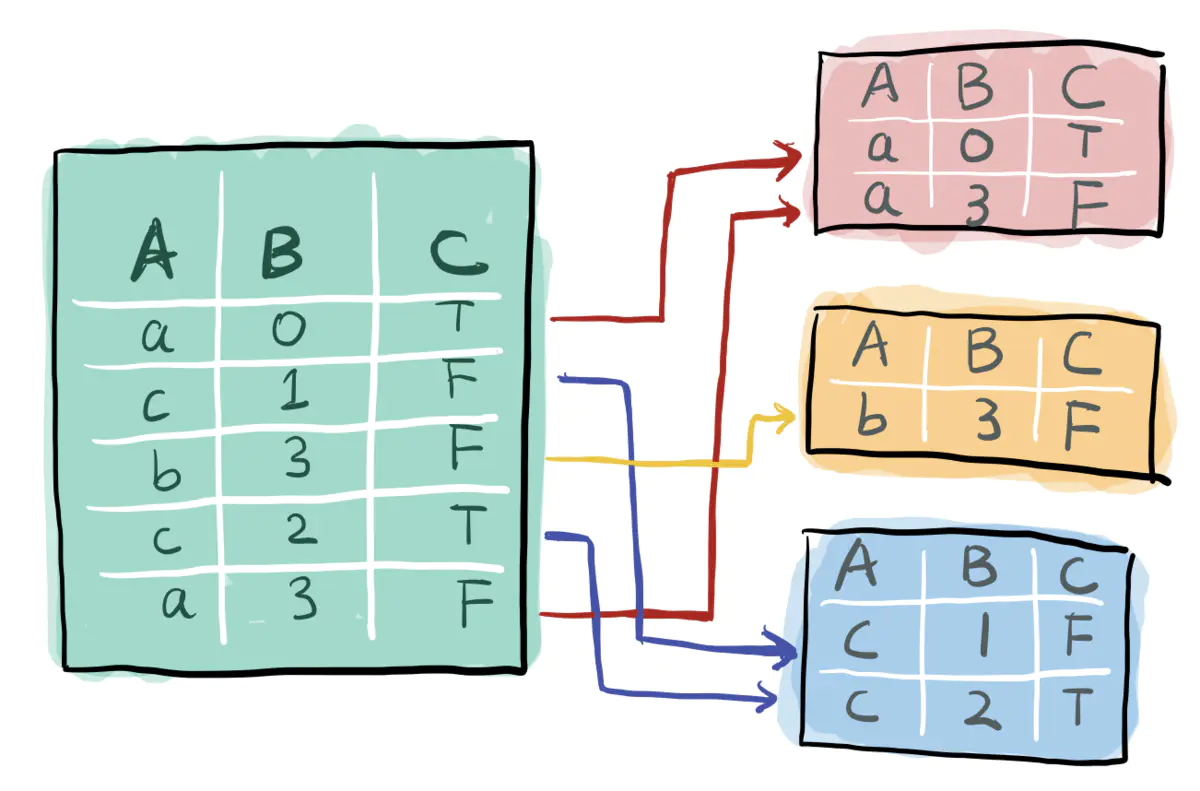
此时我们输出数据后，我们则可以看到数据全部加入如下的集合中



数据读入后我们则可以进行数据的处理，此处处理思想为取出在这个月的平均值，****在这个月的最大值，以及在这月的极差情况。

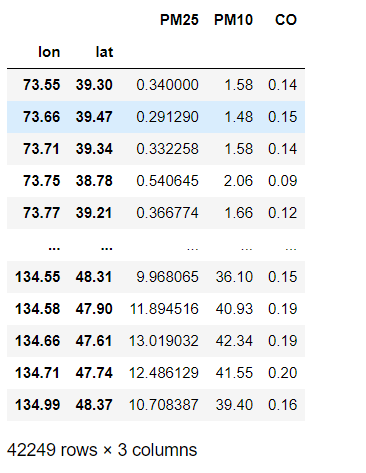
1. def peak\_range(df):
2. return df.max() - df.min()
3. data2 = data2[["PM25","PM10","CO","lat","lon",'date']]
4. data2.groupby(['lon','lat']).agg({"PM25":"mean","PM10":"max","CO":peak\_range})

groupby就是按xx分组,它也确实是用来实现这样功能的. 比如, 将一个数据集按A进行分组, 效果是这样



agg是一个聚合函数，聚合函数操作始终是在轴（默认是列轴，也可设置行轴）上执行，不同于 numpy聚合函数（np.sum() //求和；np.prod() //所有元素相乘；np.mean() //平均值；np.std() //标准差；np.var() //方差；np.median() //中数；np.power() //幂运算；np.sqrt() //开方；np.min() //最小值；np.max() //最大值；np.argmin() //最小值的下标；np.argmax() //最大值的下标；np.inf //无穷大；np.exp(10) //以e为底的指数；np.log(10) //对数）。

以上代码运行结果如下所示：



数据提取完成后，接着完成下面的需求：

1. data2 = data2.sample(n=100)
2. import requests
3. import json
4. def findcity(row):
5. *#     print(row)*
6. url = 'https://restapi.amap.com/v3/geocode/regeo?key=3e7a24eb63ccd0362f20ef75c17ab511&location={},{}'.format(row['lon'],row['lat'])
7. res = requests.get(url)
8. json\_data = json.loads(res.text)
9. *#     print(json\_data)*
10. city = json\_data['regeocode']['addressComponent']['province']
11. *#     print(city)*
12. return city
13. data2['city'] = data2.apply(findcity,axis=1)

此时我们用到了一个新的第三方库 requests，requests是一个很实用的Python HTTP客户端库，编写爬虫和测试服务器响应数据时经常会用到，Requests是Python语言的第三方的库，专门用于发送HTTP请求。安装库函数

1. pip install requests

安装完成后继续运行代码，则此时可输出以下结果



此时导出文件即可。

# 可视化基本图形绘制

## 2.1 D3.js

### 2.1.1 D3.js是什么

D3 的全称是（Data-Driven Documents），顾名思义可以知道是一个被数据驱动的文档。听名字有点抽象，说简单一点，其实就是一个 JavaScript的函数库，使用它主要是用来做数据可视化的。

### 2.1.2 如何学习和使用D3

以下是几个学习D3的站点：

* 官方网站  
  [http://d3js.org/](http://d3js.org/" \t "_blank)  
  包含有很多示例和API，要使用D3，熟悉API是避不开的。
* Mike Bostock 的博客和作品展示板  
  [http://bost.ocks.org/mike/](http://bost.ocks.org/mike/" \t "_blank)
* Web 数据可视化工具  
  [https://github.com/jobbole/awesome-javascript-cn#data-visualization](https://github.com/jobbole/awesome-javascript-cn" \l "data-visualization" \t "_blank)
* 可视化技术栈及学习计划  
  [https://segmentfault.com/a/1190000010175892](https://segmentfault.com/a/1190000010175892" \t "_blank)
* API  
  [https://github.com/d3/d3/wiki/API–%E4%B8%AD%E6%96%87%E6%89%8B%E5%86%8C](https://github.com/d3/d3/wiki/API--%E4%B8%AD%E6%96%87%E6%89%8B%E5%86%8C" \t "_blank)
* D3.js中文版api-接口文档  
  https://blog.mn886.net/chenjianhua/show/773c07b3abce/index.html

### 2.1.3 安装D3

D3是一个JavaScript函数库，并不需要通常所说的安装。它只有一个文件，在HTML中引用即可。有两种方法：  
（1）下载D3.js的文件



解压后，在HTML文件中包含相关的js文件即可。  
（2）直接包含网络的链接

1. **<script src="https://cdn.bootcdn.net/ajax/libs/d3/3.5.9/d3.js"></script>**

### 2.1.4 饼状图的制作

饼图适合展示离散数据点百分占比或者大小比较。

1. **绘制画布**
2. **var** width=500;
3. **var** height = 500; //画布的高度
4. **var** svg = d3.select("body") //选择文档中的body元素
5. .append("svg") //添加一个svg元素
6. .attr("width", width) //设定宽度
7. .attr("height", height) //设定高度
8. .style('border','black solid 1px');//为画布添加边界线

在D3.js中，append函数将进行元素的添加，attr函数将为实例添加相应的属性，如“width”与“height”，style函数将为实例修改css样式。

1. **数据**



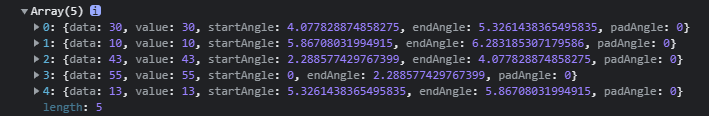
这样的值是不能直接绘制图形的，例如绘制饼图的一部分，需要知道一段弧度的起始位置和终止角度，这些值都不存在于数组的dataset中，因此需要用到布局

1. **布局(数据转换):**

计算出适合于作图的数据

1. **var** pie = d3.layout.pie();
2. **var** piedata = pie(dataset);

5个整数倍转换成了5个对象，每个对象都有：起始角度（startAngle）和终止角度（endAngle），还有原数据（属性名称为 data）。这些都是绘图需要的数据。



1. **绘制图形**

SVG 有一个元素，叫做路径 path，是 SVG 中功能最强的元素，它可以表示其它任意的图形。绘制饼状图需要使用弧生成器。它可以生成弧的路径。

弧生成器返回的结果赋值给arc。此时，arc可以当做一个函数使用，把 piedata 作为参数传入，即可得到路径值。

1. var outerRadius = 150; // 外半径
2. var innerRadius = 0; // 内半径
3. var arc = d3.svg.arc() // 弧生成器
4. .innerRadius(innerRadius) //设置内半径
5. .outerRadius(outerRadius); //设置外半径

在 svg 里添加足够数量的分组元素（g），每一个分组用于存放一段弧的相关元素。

1. var arcs = svg.selectAll("g")
2. .data(piedata)
3. .enter()
4. .append("g")
5. .attr("transform","translate("+ (width/2) +","+ (width/2) +")");

接下来对每个g元素，添加path。

1. arcs.append("path")
2. .attr("fill",(d,i)=>{
3. return color(i); // 颜色比例尺
4. })
5. .attr("d", d=> {
6. return arc(d); // 使用弧生成器获取路径
7. });

然后在每一个弧线中心添加文本。 arc.centroid(d) 能算出弧线的中心坐标，返回一个[x,y]数组。 text-anchor文本相对基点（x，y）的位置。

1. arcs.append("text")
2. .attr("transform",function(d){
3. return "translate(" + arc.centroid(d) + ")"; // 算出弧线的中心
4. })
5. .attr("text-anchor","middle")
6. .text(function(d){
7. return d.data; // d.data，而不是d，因为被绑定的数据是对象
8. })

运行代码，在浏览器可以看见绘制的饼状图如图2.1所示：

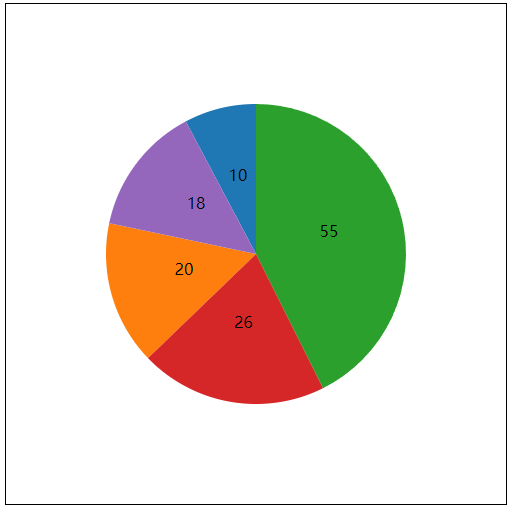


图 2.1饼状图

### 2.1.5 柱状图的制作

1. var padding = {left:30, right:30, top:20, bottom:20};//画布周边的空白
2. **数据**
3. var dataset = [10, 20, 30, 40, 33, 24, 12, 5];
4. **定义坐标系**
5. //x轴的比例尺
6. var xScale = d3.scale.ordinal()
7. .domain(d3.range(dataset.length))
8. .rangeRoundBands([0, width - padding.left - padding.right]);
9. //y轴的比例尺
10. var yScale = d3.scale.linear()
11. .domain([0,d3.max(dataset)])
12. .range([height - padding.top - padding.bottom, 0]);
13. //定义x轴
14. var xAxis = d3.svg.axis()
15. .scale(xScale)
16. .orient("bottom");
18. //定义y轴
19. var yAxis = d3.svg.axis()
20. .scale(yScale)
21. .orient("left");
22. //添加x轴
23. svg.append("g")
24. .attr("class","axis")
25. .attr("transform","translate("+padding.left+","+(height - padding.bottom) + ")")
26. .call(xAxis);
28. //添加y轴
29. svg.append("g")
30. .attr("class","axis")
31. .attr("transform","translate(" + padding.left + "," + padding.top + ")")
32. .call(yAxis);

默认的坐标轴样式不太美观，下面提供一个常见的样式：

1. <style>
2. .axis path,
3. .axis line {
4. fill: none;
5. stroke: black;
6. shape-rendering: crispEdges;
7. }
8. .axis text {
9. font-family: sans-serif;
10. font-size: 11px;
11. }
12. .MyText {
13. fill: white;
14. text-anchor: middle;
15. }
16. </style>
17. **绘图柱状图**
18. //矩形之间的空白
19. var rectPadding = 4;
21. //添加矩形元素
22. var rects = svg.selectAll(".MyRect")
23. .data(dataset)
24. .enter()
25. .append("rect")
26. .attr("class","MyRect")
27. .attr("transform","translate(" + padding.left + "," + padding.top + ")")
28. .attr("x", function(d,i){
29. return xScale(i) + rectPadding/2;
30. } )
31. .attr("y",function(d){
32. return yScale(d);
33. })
34. .attr("width", xScale.rangeBand() - rectPadding )
35. .attr("height", function(d){
36. return height - padding.top - padding.bottom - yScale(d);
37. });
38. **添加文字标签**
39. var texts = svg.selectAll(".MyText")
40. .data(dataset)
41. .enter()
42. .append("text")
43. .attr("class","MyText")
44. .attr("transform","translate(" + padding.left + "," + padding.top + ")")
45. .attr("x", function(d,i){
46. return xScale(i) + rectPadding/2;
47. } )
48. .attr("y",function(d){
49. return yScale(d);
50. })
51. .attr("dx",function(){
52. return (xScale.rangeBand() - rectPadding)/2;
53. })
54. .attr("dy",function(d){
55. return 20;
56. })
57. .text(function(d){
58. return d;
59. });

运行代码，结果如图2.2所示：

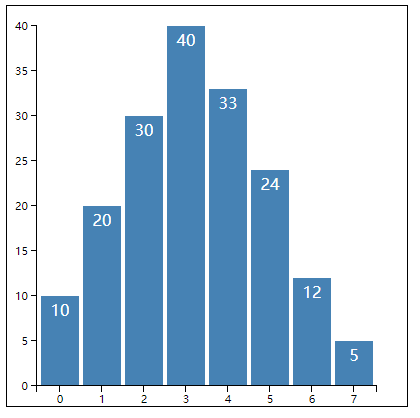


图 2.2 饼状图

## 2.2 本章小作业

1. 将第一章小作业中需求二保存的json文件使用d3读取，然后使用前五组数据中的pm2.5值进行饼状图绘制。

1. d3.json('./json/test2.json', data => {
2. let values = []//存放pm2.5值
3. for (let i = 0; i < 5; i++) {
4. values.push(data[i]['PM25'])
5. }
6. ......
7. })

使用d3读取json文件，使用数组存放数据，最后将2.1.4节绘制饼状图的代码添加在省略号之后，保存代码并运行，在浏览器中查看结果。

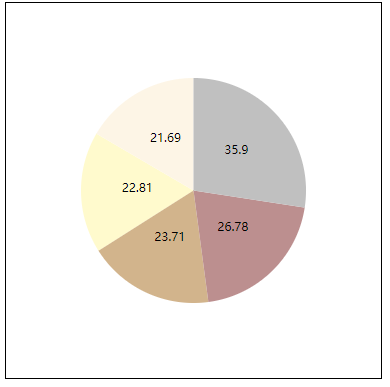


图 2.3 作业1结果

2 . 基于第一题读取的数据，第二题将统计PM10，将其柱状图中绘制出来，并且坐标轴横坐标用城市名标注。

1. d3.json('./json/test2.json', data => {
2. let values = []//存放pm2.5值
3. let cities = []//存放城市
4. for (let i = 0; i < 5; i++) {
5. //判断data[i]['city']是否在cities数组中
6. if (!cities.includes(data[i]['city'].slice(0,2))) {
7. values.push(data[i]['PM10'])
8. cities.push(data[i]['city'].slice(0,2))
9. }
10. }
11. ......
12. })

预计浏览器结果

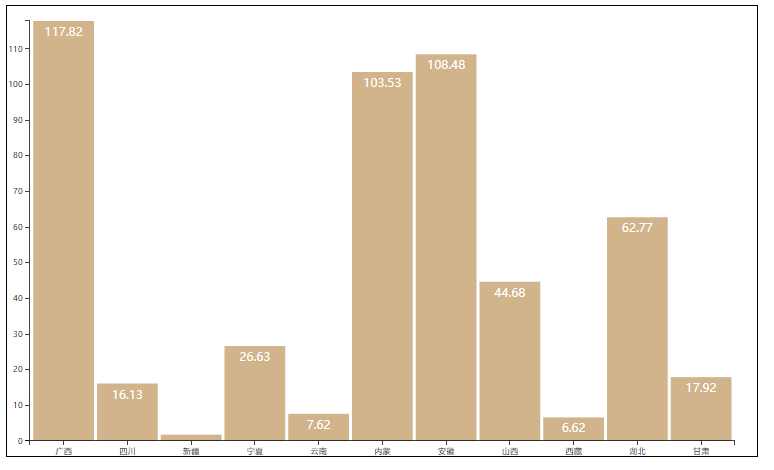


图 2.4 作业2结果

3、扩展：经过本章几个简单视图的绘制，大家已经对D3有了初步的了解，大家可以使用D3绘制其他的基本图形。

# 可视化地图技术

可视化直观理解就是转化为视觉所能感知。可视化的基本含义是将科学计算中产生的大量非直观的、抽象的或者不可见的数据，借助计算机图形学和图像处理等技术，以图形图像信息的形式，直观、形象地表达出来，并进行交互处理。地理信息可视化是运用图形学、计算机图形学和图像处理技术，将地学信息输入、处理、查询、分析以及预测的结果和数据以图形符号、图标、文字、表格、视频等可视化形式显示并进行交互的理论、方法和技术。

在地理信息系统中，可视化则以地理信息科学、计算机科学、地图学、认知科学、信息传输学与地理信息系统为基础，并通过计算机技术、数字技术、多媒体技术动态，直观、形象地表现、解释、传输地理空间信息并揭示其规律，是关于信息表达和传输的理论、方法与技术的一门学科。

在可视化中用于前端绘制的地图工具主要有百度地图api、Mapbox、高德地图api等。本章将主要介绍Mapbox的使用方法。

## 3.1 MapBox地图实例

### 3.1.1 热力地图

**1.引入js文件**

1. <script src='https://api.tiles.mapbox.com/mapbox-gl-js/v1.1.1/mapbox-gl.js'></script>
2. <link href='https://api.tiles.mapbox.com/mapbox-gl-js/v1.1.1/mapbox-gl.css' rel='stylesheet' />
3. <script src="https://cdn.bootcdn.net/ajax/libs/jquery/3.5.1/jquery.js"></script>

**2.构建地图容器**

1. <div id='map'></div>

**3.初始化Mapbox GL地图对象**

1. mapboxgl.accessToken = 'your token';
2. var map = new mapboxgl.Map({
3. container: 'map',
4. style: 'mapbox://styles/mapbox/dark-v9',
5. center: [-74.50, 40],
6. zoom: 9
7. });

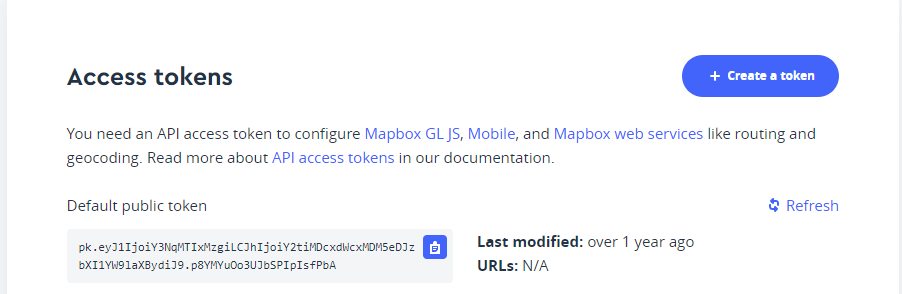
在使用MapBox时，首先需要添加accessToken ，上面代码中‘your token’需要在mapbox官网注册账户，会生成token。链接如下：（https://account.mapbox.com/）。引入后，使用new mapboxgl.Map( )创建MapBox实例对象。

“container”后需要设置容器名称，对应到<div id=”map”></div>中的id（“map”）。

“style”为地图设置风格样式，MapBox拥有属于自己的一套样式库，初学者可以上MapBox官网查询（https://www.mapbox.com/）。

“center”设置地图中心。

“zoom”为地图的初始缩放比例。



1. **地图展示**

只需仅仅几行代码，我们就可以使用MapBox在前端页面中绘制出地图。

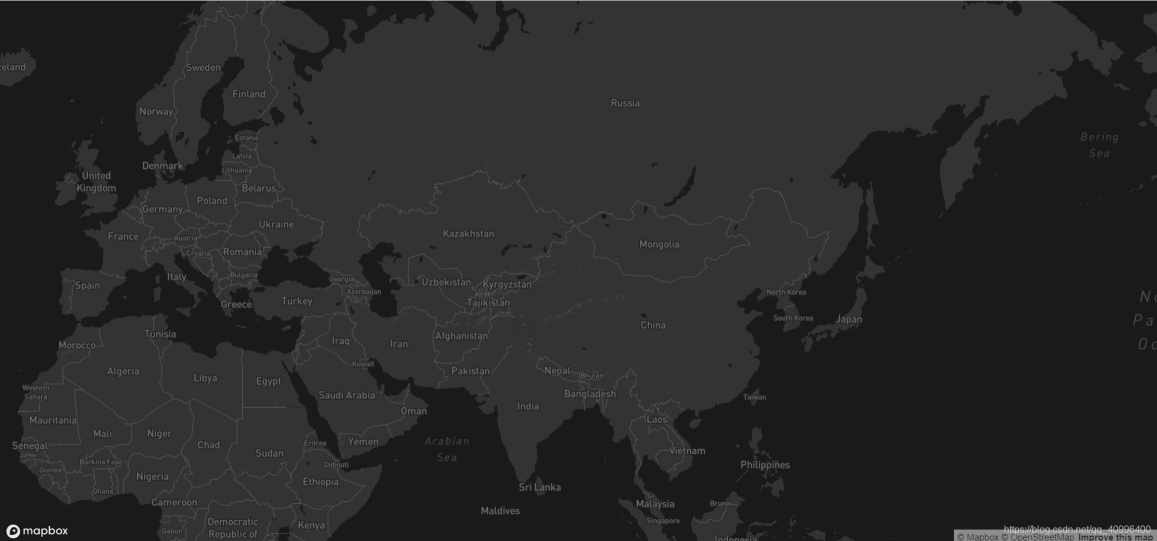


图 3.1 地图

**5.添加数据源**

Mapbox加载数据为geojson格式数据，geojson是用json的语法表达和存储地理数据，可以说是json的子集。data中可以为本地文件数据，或者引用网络数据。

1. map.addSource('earthquakes', {
2. "type": "geojson",
3. "data": "json/earthquakes.geojson"
4. //“https://docs.mapbox.com/mapbox-gl-js/assets/earthquakes.geojson”
5. });

**6.添加热力图图层**

MapBox中使用addLayer( )为地图添加相应瓦片图形，本例中我们在地图中添加热力图。

1. map.addLayer(
2. {
3. id: "earthquakes-heat",
4. type: "heatmap",
5. source: "earthquakes",//注意source对应数据源名
6. maxzoom: 9,
7. paint: {
8. }
9. }
10. );

对象中，“id”为对应的图层的名称。“type”为图例的类型，如“circle”，“heatmap”等等。另外，初学者需特别注意，“source”中为引入的数据源名称，这里对应了“map.addSource('earthquakes',······”中的“earthquakes”，通俗说就是“source”中设置的数据源名称必须与addSource( )中定义的数据源名称一致。

**7.效果展示**

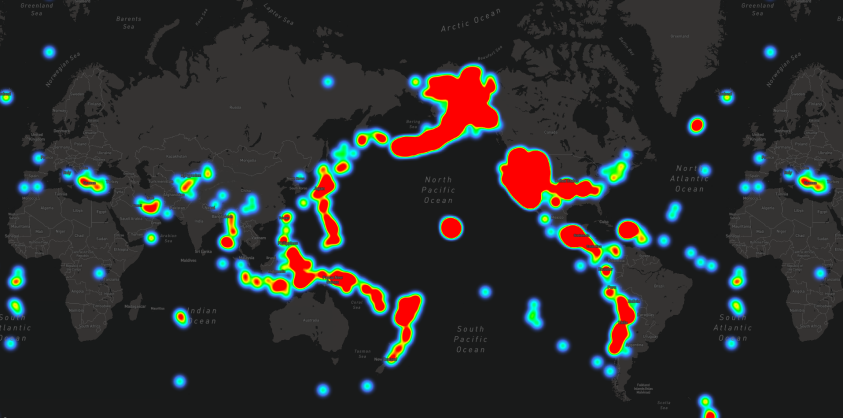


图 3.2 初始热力图

很明显，这幅图的效果十分不好，首先，其存在着严重的“热力失真”现象，在地图缩放级别很小的情况下，中国的整个东边都好像被感染了，这明显是不符合现实情况的；再者，如果在不同的地图缩放等级下，均采用相同的热力半径，会导致更严重的可视化失衡，因此我们需要进一步改善。

**8.改进**

在MapBox中，我们在“paint”属性中添加相应的属性来修改我们的图层样式。下面代码将展示如何使用热力图中对应的样式属性对我们的热力图进行修改，使其美观，便于我们进行数据分析工作。

1. "heatmap-weight": [
2. "interpolate",//插值
3. ["linear"],//线性
4. ["get", "mag"],
5. 0, 0,
6. 6, 1
7. ],

根据properties中的mag值来设置每一个点对热力图强度的贡献，mag在0-6,贡献从0-1线性变换，本文数据源无mag值，所以配置每个mag为1。

1. "heatmap-intensity": [
2. "interpolate",
3. ["linear"],
4. ["zoom"],
5. 0, 1,
6. 9, 3
7. ],

根据zoom设置热力图强度变化，本文设置地图最小缩放为0，热力图最大层级为9，所以此处配置zoom在0-9之间，热力图强度在1-3之间线性变换。

1. "heatmap-color": [
2. "interpolate",
3. ["linear"],
4. ["heatmap-density"],
5. 0, "rgba(33,102,172,0)",
6. 0.2, "rgb(103,169,207)",
7. 0.4, "rgb(209,229,240)",
8. 0.6, "rgb(253,219,199)",
9. 0.8, "rgb(239,138,98)",
10. 1, "rgb(178,24,43)"
11. ],

颜色线性变化。

1. "heatmap-radius": [
2. "interpolate",
3. ["linear"],
4. ["zoom"],
5. 0, 2,
6. 9, 20
7. ],

根据zoom设置热力图计算半径。

1. "heatmap-opacity": [
2. "interpolate",
3. ["linear"],
4. ["zoom"],
5. 7, 1,
6. 9, 0
7. ],

根据zoom设置热力图透明度。

运行代码后，可在浏览器中看见改进后的热力图明显优于之前版本。

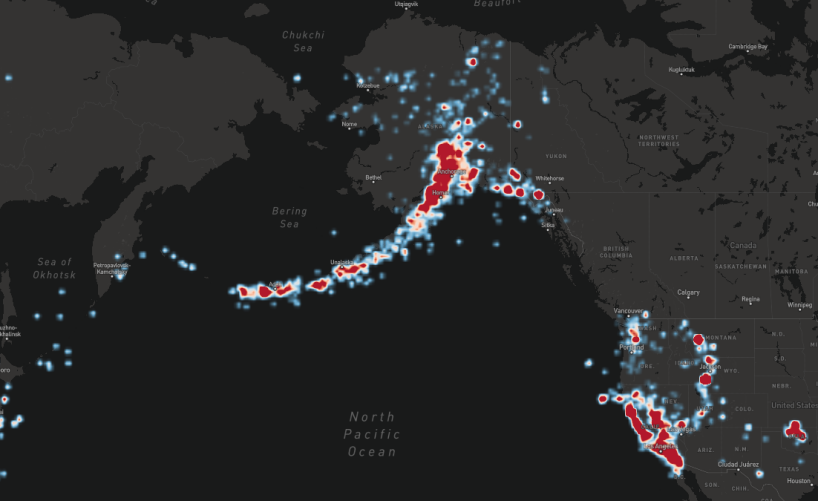


图 3.3 完善后热力图

### 3.1.2 Echarts结合Mapbox实例

**1.在页面引入相关依赖**

基于 mapbox-gl-js 的地理组件。支持在 mapbox 的地图上绘制三维的散点图，飞线图，柱状图和地图。你可以利用 Mapbox 强大的地图服务和 ECharts GL 丰富的可视化和渲染效果实现你想要的可视化作品。

在使用 mapbox 组件之前你需要先引入 mapbox 的官方 sdk。

<script src='https://api.mapbox.com/mapbox-gl-js/v0.38.0/mapbox-gl.js'></script>

<link href='https://api.mapbox.com/mapbox-gl-js/v0.38.0/mapbox-gl.css' rel='stylesheet' />

在Echarts使用网络引入Mapbox会造成前端等待时间过长，所以建议本地引入Mapbox相关文件。

1. <script src="js/echarts.min.js"></script>
2. <script src="js/echarts-gl.min.js"></script>
3. <script src="js/mapbox-gl.js"></script>
4. <link href="js/mapbox-gl.css" rel="stylesheet" />
5. <script src="http://d3js.org/d3.v3.min.js"></script>

**2.创建容器**

1. <div id="main"></div>

在css中设置“main”长和宽。

1. #main {
2. position: absolute;
3. width: 100%;
4. height: 100%;
5. }

**3.读取数据**

本次案例将读取本地JSON文件，该数据是关于全国空气质量指数，这里使用D3读取JSON文件。

1. d3.json('json/data.json', data => {
2. console.log(data)
3. })

将读取的JSON文件在控制台中输出可以看见以下结果。控制台输出为一个对象，对象中包含一个键值为“map”的二维数组，每个数组中包含一个坐标的经度，纬度与AQI指数。

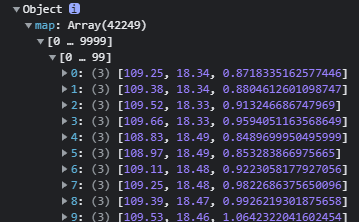


图 3.4 数据输出

1. **视图绘制**

同样，在Echarts中使用Mapxbox也需要accessToken。

1. mapboxgl.accessToken = '你的 token';
2. var chart = echarts.init(document.getElementById('main'));
3. var option = {
4. mapbox: {
5. center: [108.55, 36.27],
6. zoom: 4,
7. pitch: 60,
8. },
9. visualMap: {
10. min: 0,
11. max: 1,
12. color: ['rgb(238,222,223)', 'rgb(200,226,233)', 'rgb(173,206,215)'],
13. },
14. series: [{
15. type: 'scatter3D',
16. coordinateSystem: 'mapbox',
17. data: data.map
18. }]
19. }
20. chart.setOption(option);

在使用Echarts绘制可视化视图时。我们需要使用echarts.init( )来定义我们的画布。定义完画布后，我们需要进行我们的配置项设置，在“option”中我们设置了“mapbox”为我们的坐标系统，并且将我们的“scatter3D”通过“coordinateSystem”属性绑定到MapBox中。

运行代码后，效果如下图所示。



图 3.5 地图展示

## 3.2 本章小作业

1. 基于第一章中需求二中保存的“test2.json”文件，采用直接调用Mapbox的方式将数据中的地理位置在地图中显示。

**提示：Mapbox调用的数据必须为geojson格式，读出test1.json后需要将数据处理为特定格式。**

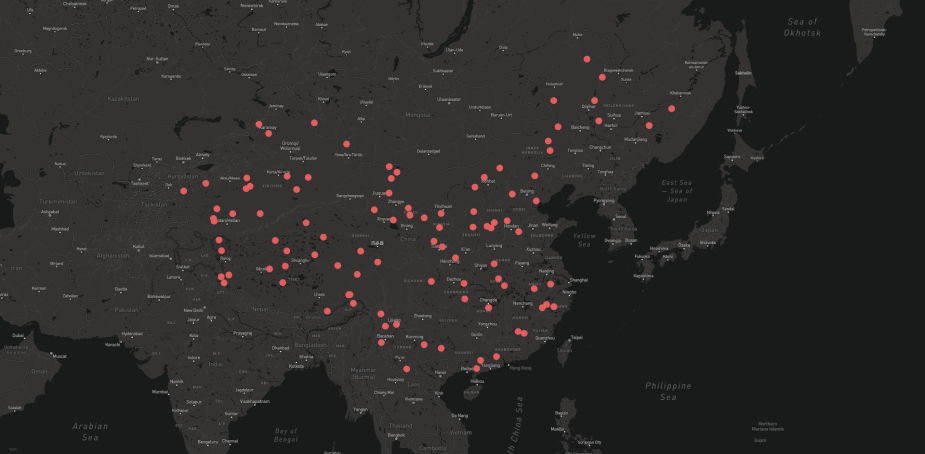


图 3.6 作业1结果

1. 同理基于作业1的需求，这次使用Echarts结合Mapbox将数据中的地理位置展示出来，并且使用颜色深度编码PM2.5大小。

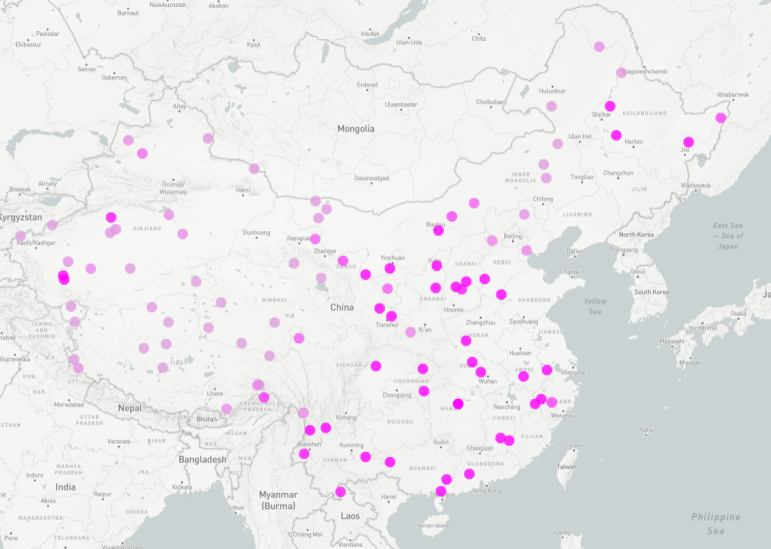


图 3.7 作业二结果

1. 扩展：基于test2.json绘制3D柱状图，使用高度编码PM10的大小。

# 可视化交互技术

交互，指的是用户输入了某种指令，程序接受到指令之后必须做出某种响应。对可视化图表来说，交互能使图表更加生动，能表现更多内容。例如，拖动图表中某些图形、鼠标滑到图形上出现提示框、用触屏放大或缩小图形等等。用户用于交互的工具一般有三种：鼠标、键盘、触屏。

## 4.1 d3可视化交互

### 4.1.1 添加交互

在D3中对某一元素添加交互操作十分简单，代码如下：

1. var circle = svg.append("circle");
2. circle.on("click", function(){
3. //在这里添加交互内容
4. });

这段代码在 SVG 中添加了一个圆，然后添加了一个监听器，是通过 on() 添加的。在 D3 中，每一个选择集都有 on() 函数，用于添加事件监听器。

on( ) 的第一个参数是监听的事件，第二个参数是监听到事件后响应的内容，第二个参数是一个函数。

鼠标事件：

* click：鼠标单击某元素时，相当于 mousedown 和 mouseup 组合在一起。
* mouseover：光标放在某元素上。
* mouseout：光标从某元素上移出来时。
* mousemove：鼠标被移动的时候。
* mousedown：鼠标按钮被按下。
* mouseup：鼠标按钮被松开。
* dblclick：鼠标双击。

键盘事件：

* keydown：当用户按下任意键时触发，按住不放会重复触发此事件。该事件不会区分字母的大小写，例如“A”和“a”被视为一致。
* keypress：当用户按下字符键（大小写字母、数字、加号、等号、回车等）时触发，按住不放会重复触发此事件。该事件区分字母的大小写。
* keyup：当用户释放键时触发，不区分字母的大小写。

### 4.1.2带有交互的柱状图

本例中我们将使用第二章绘制的柱形图进行交互的添加，部分代码修改成如下代码。

1. var rects = svg.selectAll(".MyRect")
2. .data(dataset)
3. .enter()
4. .append("rect")
5. .attr("class", "MyRect")
6. .attr("transform", "translate(" + padding.left + "," + padding.top + ")")
7. .attr("x", function(d, i) {
8. return xScale(i) + rectPadding / 2;
9. })
10. .attr("y", function(d) {
11. return yScale(d);
12. })
13. .attr("width", xScale.rangeBand() - rectPadding)
14. .attr("height", function(d) {
15. return height - padding.top - padding.bottom - yScale(d);
16. })
17. .attr("fill", "steelblue")
18. .on("mouseover", function(d, i) {
19. d3.select(this)
20. .attr("fill", "yellow");
21. })
22. .on("mouseout", function(d, i) {
23. d3.select(this)
24. .transition()
25. .duration(500)
26. .attr("fill", "steelblue");
27. });

这段代码添加了鼠标移入（mouseover），鼠标移出（mouseout）两个事件的监听器。监听器函数中都使用了 d3.select(this)，表示选择当前的元素，this 是当前的元素，要改变响应事件的元素时这么写就好。

mouseover 监听器函数的内容为：将当前元素变为黄色。

mouseout 监听器函数的内容为：缓慢地将元素变为原来的颜色（蓝色）。

运行修改后的程序，在浏览器中将鼠标放在柱状图上，可以看见柱状图变为黄色。

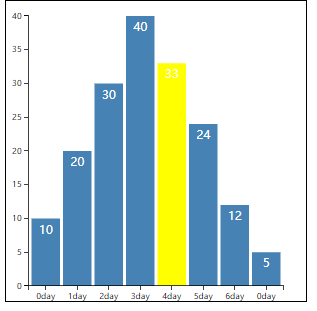


图 4.1 D3交互

## 4.2 Echarts可视化交互

### 4.2.1 Echarts交互介绍

**1. 事件与行为**

在 Apache ECharts 的图表中用户的操作将会触发相应的事件。开发者可以监听这些事件，然后通过回调函数做相应的处理，比如跳转到一个地址，或者弹出对话框，或者做数据下钻等等。

ECharts 中的事件名称对应 DOM 事件名称，均为小写的字符串，如下是一个绑定点击操作的示例。

1. myChart.on('click', function(params) {
2. // 控制台打印数据的名称
3. console.log(params.name);
4. });

在ECharts 中事件分为两种类型：

* 用户鼠标操作点击，或者 hover 图表的图形时触发的事件；
* 用户在使用可以交互的组件后触发的行为事件。

**2. 鼠标事件的处理**

ECharts 支持常规的鼠标事件类型，包括 'click'、 'dblclick'、 'mousedown'、 'mousemove'、 'mouseup'、 'mouseover'、 'mouseout'、 'globalout'、 'contextmenu' 事件。

1. var option = {
2. xAxis: {
3. data: ['衬衫', '羊毛衫', '雪纺衫', '裤子', '高跟鞋', '袜子']
4. },
5. yAxis: {},
6. series: [
7. {
8. name: '销量',
9. type: 'bar',
10. data: [5, 20, 36, 10, 10, 20]
11. }
12. ]
13. };
14. // 使用刚指定的配置项和数据显示图表。
15. myChart.setOption(option);
16. // 处理点击事件并且跳转到相应的百度搜索页面
17. myChart.on('click', function(params) {
18. window.open('https://www.baidu.com/s?wd='+encodeURIComponent(params.name));
19. });

运行代码，在浏览器中点击名称为“雪纺衫”的柱状图，页面会跳转到百度搜索中，且搜索内容为“雪纺衫”。



图 4.2 Echarts交互

**提示：**

所有的鼠标事件包含参数 params，这是一个包含点击图形的数据信息的对象，如下格式：

1. type EventParams = {
2. // 当前点击的图形元素所属的组件名称，
3. // 其值如 'series'、'markLine'、'markPoint'、'timeLine' 等。
4. componentType: string;
5. // 系列类型。值可能为：'line'、'bar'、'pie' 等。当 componentType 为 'series' 时有意义。
6. seriesType: string;
7. // 系列在传入的 option.series 中的 index。当 componentType 为 'series' 时有意义。
8. seriesIndex: number;
9. // 系列名称。当 componentType 为 'series' 时有意义。
10. seriesName: string;
11. // 数据名，类目名
12. name: string;
13. // 数据在传入的 data 数组中的 index
14. dataIndex: number;
15. // 传入的原始数据项
16. data: Object;
17. // sankey、graph 等图表同时含有 nodeData 和 edgeData 两种 data，
18. // dataType 的值会是 'node' 或者 'edge'，表示当前点击在 node 还是 edge 上。
19. // 其他大部分图表中只有一种 data，dataType 无意义。
20. dataType: string;
21. // 传入的数据值
22. value: number | Array;
23. // 数据图形的颜色。当 componentType 为 'series' 时有意义。
24. color: string;
25. };

**4.组件交互的行为事件**

在 ECharts 中基本上所有的组件交互行为都会触发相应的事件，常用的事件和事件对应参数在 events 文档中有列出。

下面是监听一个图例开关的示例：

1. // 图例开关的行为只会触发 legendselectchanged 事件
2. myChart.on('legendselectchanged', function(params) {
3. // 获取点击图例的选中状态
4. var isSelected = params.selected[params.name];
5. // 在控制台中打印
6. console.log((isSelected ? '选中了' : '取消选中了') + '图例' + params.name);
7. // 打印所有图例的状态
8. console.log(params.selected);
9. });

### 4.2.2.Echarts交互组件

在Echarts中，每个视图都有属于自己的交互组件，如“tooltip”，“legend”，“visualMap”等等。

**Legend(图例**）：如图4.3左所示，框选部分为图例，编码环形饼状图中不同的数据维度。点击图例，视图中对应的形状将会消失。

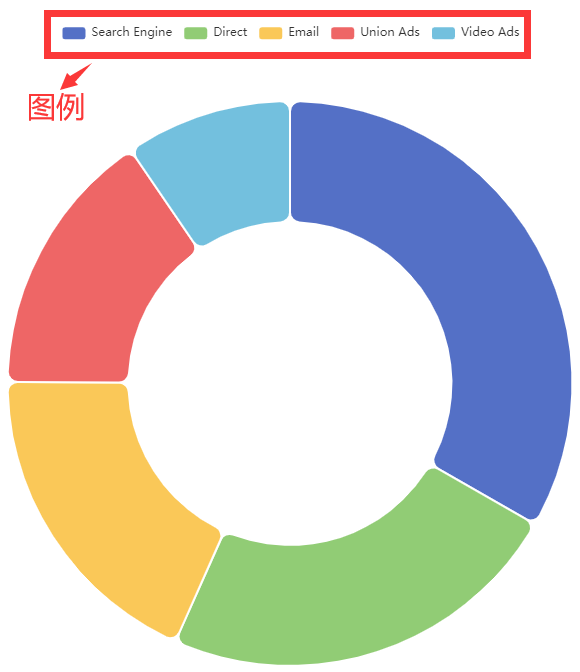
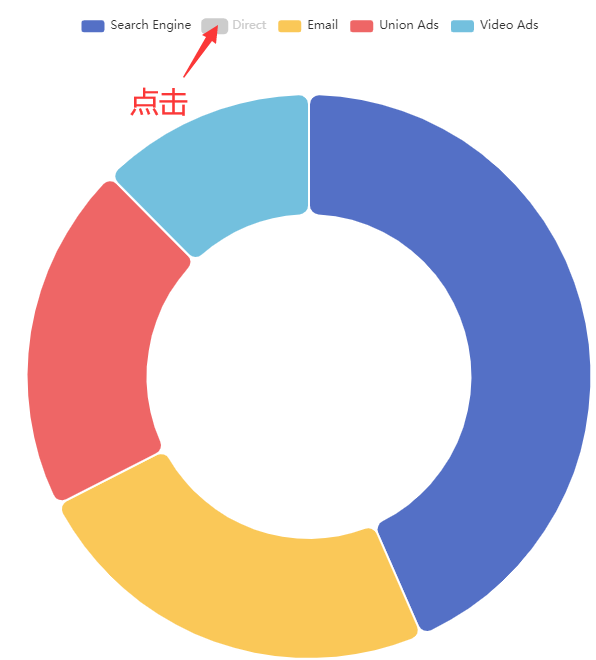
 

图 4.3 legend及其交互

**tooltip:(提示框）**：当鼠标放入饼状图时，将会展示对应图形的数据信息。如图4.4所示。

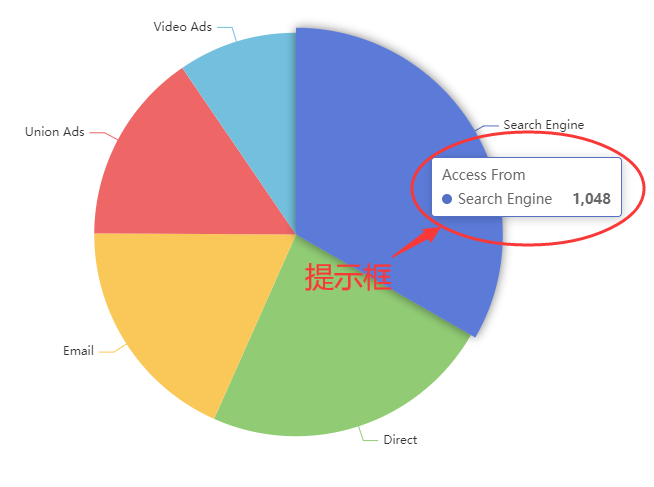


图 4.4 tooltip及其交互

**visualMap（视觉映射）**：用于进行视觉编码，也就是将数据映射到视觉元素（视觉通道），如图4.5右所示，使用视觉映射组件进行数据过滤。

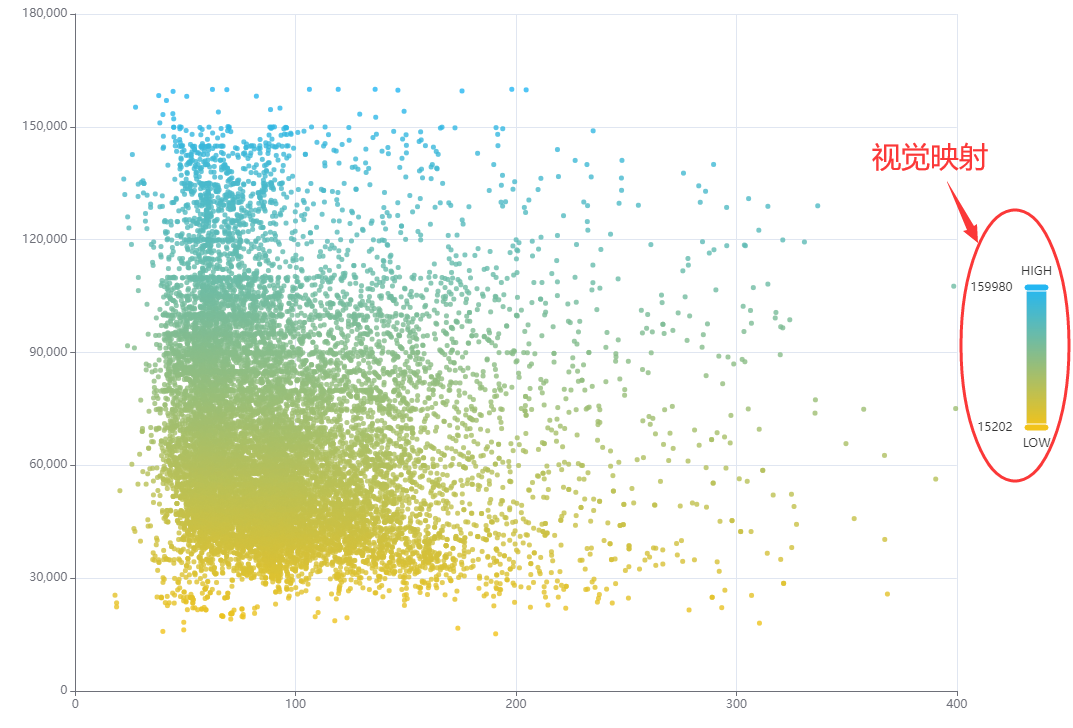
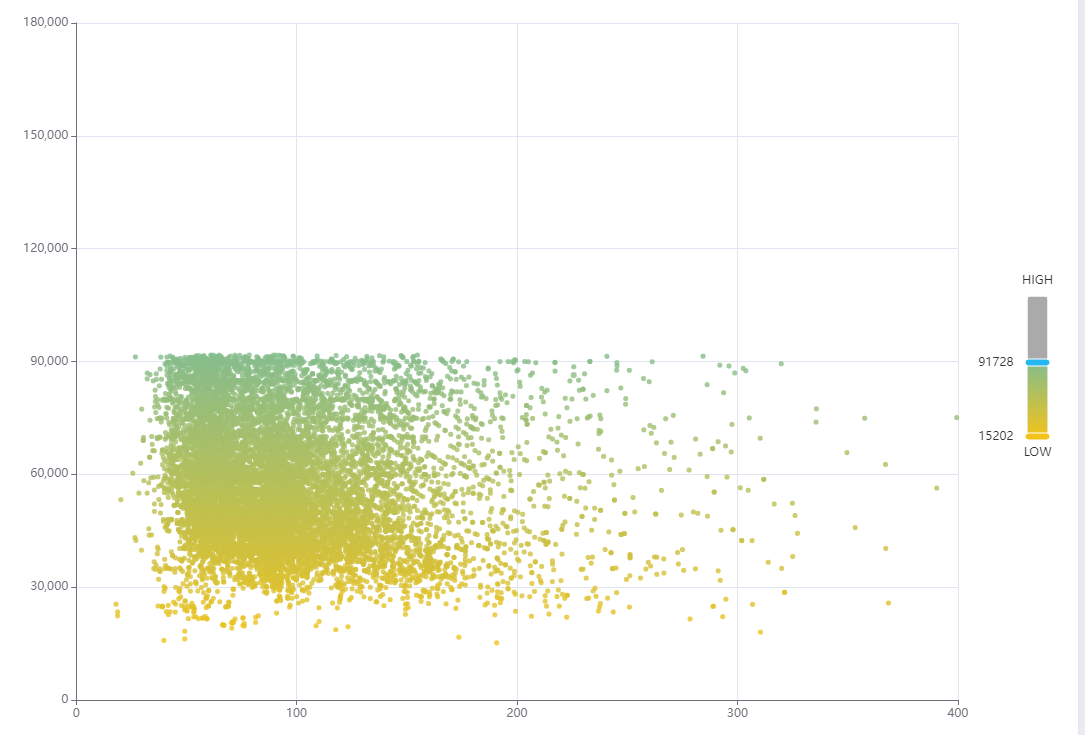
 

图 4.5 visualMap及其交互

**dataZoom**： 组件用于区域缩放，从而能自由关注细节的数据信息，或者概览数据整体，或者去除离群点的影响。该组件拥有两种主要类型：“inside”与“slider”。

“inside”为内置型数据区域缩放组件，所谓『内置』，即内置在坐标系中。设置该属性可以在视图中进行视图的平移与缩放。

“slider”为滑动条型数据区域缩放组件。滑动条型数据区域缩放组件提供了数据缩略图显示，缩放，刷选，拖拽，点击快速定位等数据筛选的功能。如图4.6所示显示了该组件可交互部分。

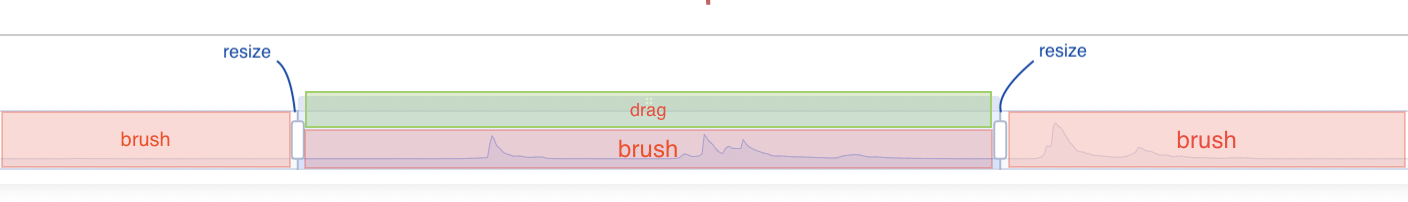
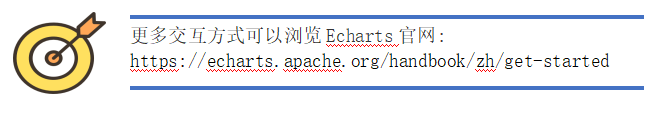


图 4.6 dataZoom中“slider”交互

这里介绍几种通用的Echarts视图交互组件。感兴趣的读者可以进入Echarts官网（https://echarts.apache.org/zh/index.html）进行更高层次的学习。



### 4.2.3.基于Echarts的多视图交互

本案例将会为大家介绍基于Echarts绘制的多个基本视图如何进行数据交互。首先介绍一下本案例需要的JSON的数据内容。

1. {
2. "whole":[
3. {
4. "name":"A",
5. "value":100
6. },
7. {
8. "name":"B",
9. "value":235
10. },
11. {
12. "name":"C",
13. "value":185
14. },
15. {
16. "name":"D",
17. "value":169
18. }
19. ],
20. "A":[
21. {
22. "name":"a1",
23. "value":10
24. },
25. {
26. "name":"a2",
27. "value":25
28. },
29. {
30. "name":"a3",
31. "value":35
32. },
33. {
34. "name":"a4",
35. "value":30
36. }
37. ],
38. ······

数据中包含五个数组，其对应的键值为“whole”，“A”，“B”，“C”，“D”。在“A”，“B”，“C”，“D”所在数组中包函四组子对象，每个对象包含两个键值，分别为“name”，“value”。“name”数据中为一个字符串，“value”为整数。“whole”中也包含四个对象，其每个对象中“value”值分别等于“A”，“B”，“C”，“D”四个对象中的“value”值的总和。

接下来将展示该案例的主要代码。

首先在HTML文件中，我们需要定义两个div容器。

1. <div id="box1"></div>
2. <div id="box2"></div>

接下来我们新建css文件，并在文件中为两个div设置样式。

1. #box1{
2. width: 800px;
3. height: 500px;
4. border: 1px black solid;
5. position: absolute;
6. }
7. #box2{
8. width: 800px;
9. height: 500px;
10. border: 1px black solid;
11. margin-left: 850px;
12. position: absolute;
13. }

接下来将介绍js文件中的相关代码。

1. function drawPie(data,p){
2. var name=[]
3. for(var i=0;i<data.length;i++){
4. name.push(data[i].name)
5. }
7. option={
8. title:{
9. text:"整体饼状图",
10. fontSize:20,
11. left:"center"
12. },
13. legend:{
14. orient:"vertical",
15. left:10,
16. data:name
17. },
18. series:[{
19. type:"pie",
20. data:data,
21. emphasis:{
22. label:{
23. show:true,
24. fontSize:30,
25. fontWeight:"bold"
26. }
27. }
28. }]
29. }
30. p.setOption(option)
31. }

该部分是用于饼状图的绘制，参数“data”是饼状图所需的数据。“p”为绘制饼状图所在的canvas。函数中定义了一个“name”的空数组。用于legend组件“data”所需的数据。

1. function drawLine(data,str){
2. var labels=[]
3. var values=[]
4. for(var i=0;i<data.length;i++){
5. labels.push(data[i].name)
6. values.push(data[i].value)
7. }
8. var line=echarts.init(document.getElementById('box2'))
9. var title=str
10. option={
11. title:{
12. text:str+"折线图",
13. fontSize:20,
14. left:"center"
15. },
16. xAxis:{
17. type:"category",
18. data:labels
19. },
20. yAxis:{
21. type:"value"
22. },
23. series:[{
24. type:"line",
25. data:values,
26. smooth:"true",
27. label:{
28. show:true
29. },
30. lineStyle:{
31. width:2,
32. color:"yellow"
33. },
34. symbol:"rect",
35. symbolSize:20
36. }]
37. }
38. line.setOption(option)
39. }

该函数用于折线图的绘制，参数“data”同理为绘制视图所需的数据，“str”为字符串，主要是为交互视图交互时更改折线图标题内容。labels数组用于储存X轴所需数据，values数组为绘图数据。

1. d3.json("json/LastClass.json",function(data){
2. console.log(data)
3. var pie =echarts.init(document.getElementById('box1'))
4. drawPie(data.whole,pie)
5. drawLine(data.whole,"whole")
6. pie.on("click",function(params){
7. console.log(params)
8. if(params.name=="A"){
9. drawLine(data.A,"A")
10. }else if(params.name=="B"){
11. drawLine(data.B,"B")
12. }else if(params.name=="C"){
13. drawLine(data.C,"C")
14. }else if(params.name=="D"){
15. drawLine(data.D,"D")
16. }
17. })
18. })

首先使用d3.json读取数据并在控制台中输出数据检查数据是否正确读出。接下来获取id为“box1”的DOM元素，并使用Echarts对其进行画布声明。然后调用两个绘图函数，并正确传参。

最后便是本案例的精华部分，我们使用on( )函数对饼状图进行事件监听，当鼠标对饼状图进行点击时，我们在控制台输出params.name中内容，因为后续中我们需要使用params参数中“name”属性储存的数据。我们对其使用if语句进行判断当params.name=="A"时，我们调用drawLine函数，并向“data”参数传入JSON数据中键值“A”中的数据，将“str”参数出入“A”，后续依次对“B”，“C”，“D”进行判断。

在浏览器中运行本案例，结果如图4.7所示。左侧为整体饼状图，标签显示对应数据名称。右侧为折线图，标题显示为“whole折线图”，代表JSON数据中“whole”对应的数据。

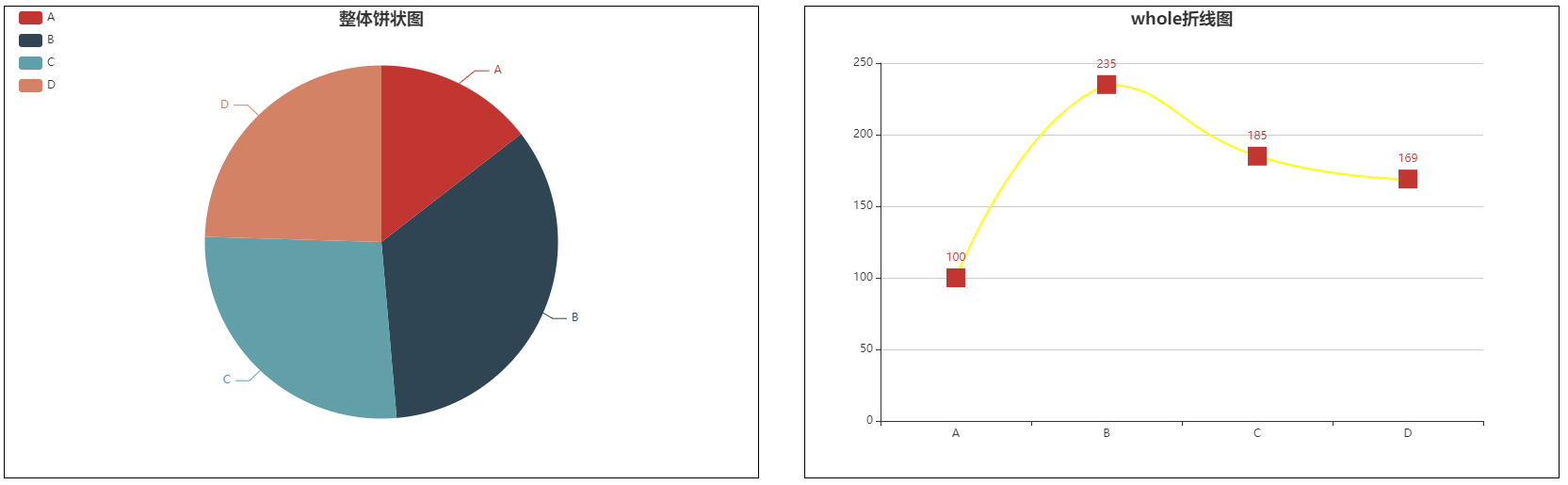


图 4.7 案例结果

接下来点击饼状图标签“A”所在的扇形，观察结果如图4.8所示。折线图更改为JSON数据“A”键值所对应的数据。

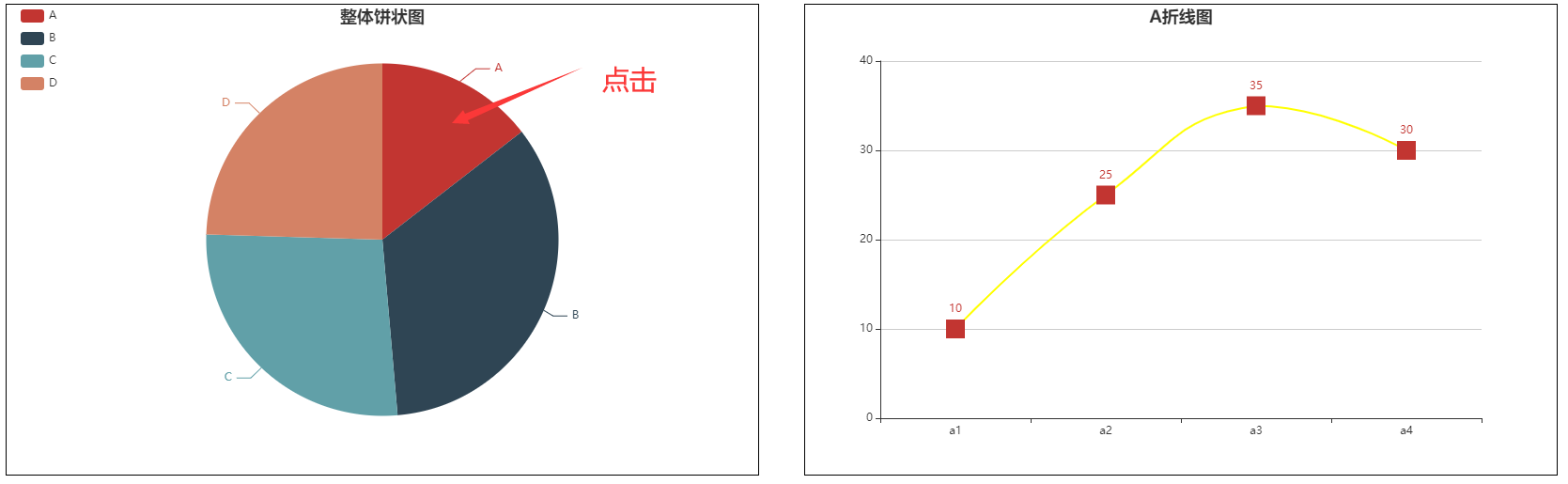


图 4.8 多视图交互

## 4.3 本章小作业

1. 基于2.1.4中绘制的饼状图，我们将为其添加交互式提示框。

首先找到该部分代码，在js代码中使用d3选择“body”元素，在其下添加一个类别为“tooltip”的div，将其透明初始化为0。

1. let tooltip = d3.select("body")
2. .append("div")
3. .attr("class", "tooltip")
4. .style("opacity", 0.0);

然后，定义一个 tooltip 样式，并将其定位方式设置为绝对定位，这一步是重点。其他的属性时关于边框外观和文字显示方式的。

1. .tooltip {
2. position: absolute;
3. width: 120;
4. height: auto;
5. font-family: simsun;
6. font-size: 14px;
7. text-align: center;
8. border-style: solid;
9. border-width: 1px;
10. background-color: white;
11. border-radius: 5px;
12. }

最后，为饼状图的各图形元素定制鼠标事件的监听器，其中包括：鼠标放到到图形上时（mouseover）、鼠标在图形上移动时（mousemove），鼠标移出时（mouseout）。

1. arcs.on("mouseover", function(d) {
2. /\*
3. 鼠标移入时，
4. （1）通过 selection.html() 来更改提示框的文字
5. （2）通过更改样式 left 和 top 来设定提示框的位置
6. （3）设定提示框的透明度为1.0（完全不透明）
7. \*/
8. tooltip.html("PM2.5为"+d.data)
9. .style("left", (d3.event.pageX) + "px")
10. .style("top", (d3.event.pageY - 20) + "px")
11. .style("opacity", 1.0);
12. })
13. .on("mousemove", function(d) {
14. /\* 鼠标移动时，更改样式 left 和 top 来改变提示框的位置 \*/
15. tooltip.style("left", (d3.event.pageX) + "px")
16. .style("top", (d3.event.pageY + 20) + "px");
17. })
18. .on("mouseout", function(d) {
19. /\* 鼠标移出时，将透明度设定为0.0（完全透明）\*/
20. tooltip.style("opacity", 0.0);
21. })

保存代码并运行，在浏览器中将鼠标移入一个扇形上，可以观察到如下图效果。

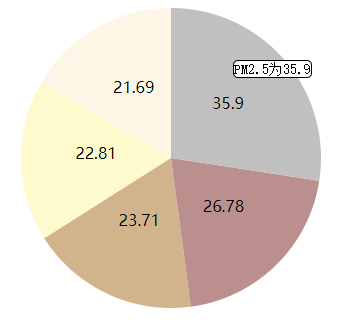


图 4.3 作业一结果

1. 使用d3.behavior 为视图添加拖拽、缩放操作。
2. let width = 1000;
3. let height = 800;
4. let svg = d3.select("body")
5. .append("svg")
6. .attr("width", width)
7. .attr("height", height)
8. .style(“border”, “black solid 1px”);
9. let g = svg.append("g");

第一步依然是创建svg画布，在svg添加<g>标记。在绘图之前，我们需要进行数据准备。

1. function phyllotaxis(radius) {
2. let theta = Math.PI \* (3 - Math.sqrt(5));
3. return function(i) {
4. let r = radius \* Math.sqrt(i),
5. a = theta \* i;
6. return {
7. x: width / 2 + r \* Math.cos(a),
8. y: height / 2 + r \* Math.sin(a)
9. };
10. };
11. }
12. let points = d3.range(500).map(phyllotaxis(10));

接下来将准备好的数据在<g>标记下进行绘制。

1. g.selectAll("circle")
2. .data(points)
3. .enter()
4. .append("circle")
5. .attr("cx", d => {
6. return d.x;
7. })
8. .attr("cy", d => {
9. return d.y;
10. })
11. .attr("r", 2.5);

保存代码并运行，在浏览器中可以看见视图绘制完成。

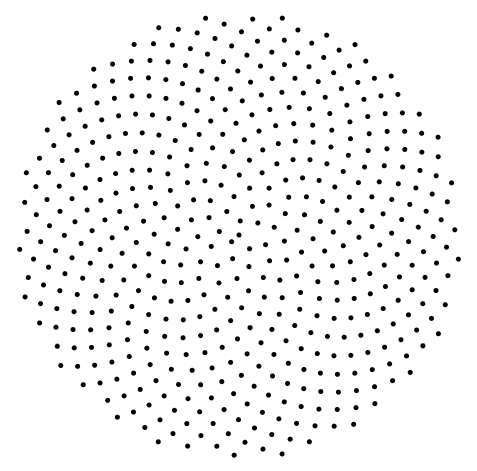


图 4.4 作业二结果

接下来我们需要定义drag函数与zoom函数

1. function dragged(d) {
2. d3.select(this).attr("cx", d.x = d3.event.x).attr("cy", d.y = d3.event.y);
3. }
4. function zoomed() {
5. g.attr("transform", "translate(" + d3.event.translate + ")scale(" + d3.event.scale + ")");
6. }

Dragged函数与zoomed函数定义完成后，首先我们需要使用call函数对整个<g>标记赋予drag操作。

1. g.call(d3.behavior.drag()
2. .on("drag", dragged));

上述代码中我们调用d3里的behavior中的drag行为，在on中监听“drag”事件，当发生“drag”事件时，将会调用dragged函数对视图进行位置更新。同理，我们对svg赋予zoomed函数。

1. svg.call(d3.behavior.zoom()
2. .scaleExtent([0.5, 10])
3. .on("zoom", zoomed));

这里scaleExtent([0.5, 10])是对视图缩放尺度进行规定，最小倍数为0.5倍，最大倍数为10倍。保存代码后，在浏览器中就可以对试图进行拖拽与缩放交互了。

1. 扩展：大家可以通过官网或者相关博客使用d3对视图增加除课程中的其他交互方式。

# 可视化综合案例

本章将会介绍如何从前端部署搭建可视化系统平台，使用的编译器可从以下几个选择其一：

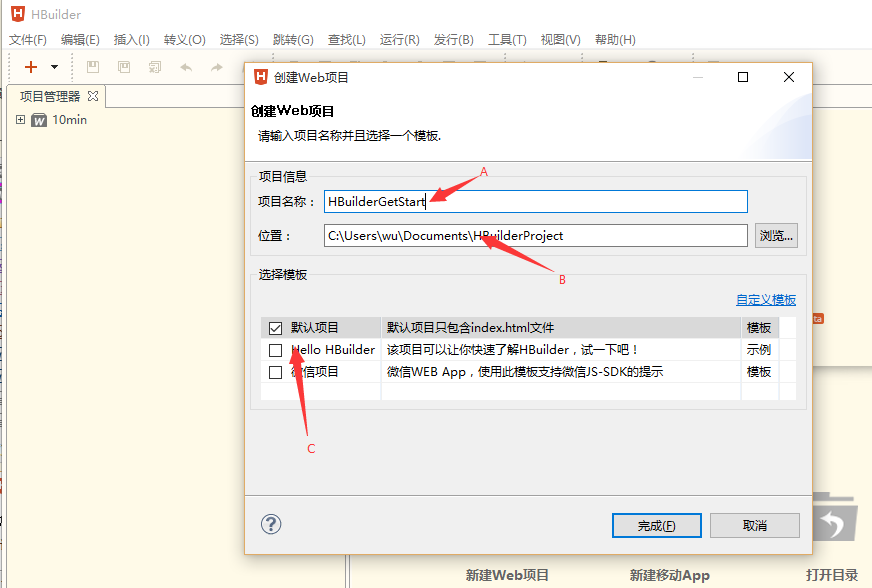
* Hbuilder
* Webstrom
* VScode

...

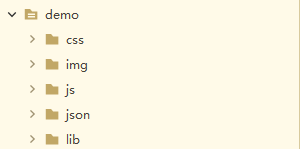
## 5.1 系统设计

1. **项目搭建**

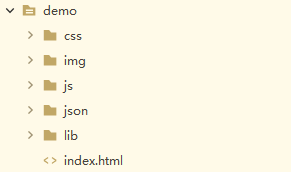
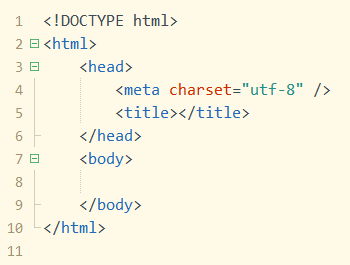
在本次案例中，作者使用Hbuilder进行可视化系统搭建。步骤为：依次点击文件→新建→选择Web项目



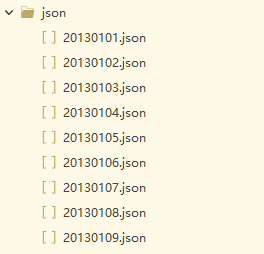
项目创建完成后，接下来在项目中新建以下文件夹用于储存各类文件。



最后在项目中新建index.html文件，项目也就搭建成功了。

1. **数据准备**



在json文件夹中我们存放了关于全国省会城市在2013年1月1日到2013年1月9日pm2.5的指数情况。

1. **初始化容器**

首先需要创建4个div，id分别为：“title”、“map”、“bar”、“pie”。

1. <div id="title">全国空气质量可视系统</div>
2. <div id="map"></div>
3. <div id='bar'></div>
4. <div id="pie"></div>

接下来为4个div创建css样式。在css文件夹下创建一个名称为“style”的css文件，在“style.css”文件中输入以下代码。代码中我们将html与body的内外边距都设置为0。这样做的目的是为了清除浏览器中默认的样式，因为每种浏览器都有一套默认的样式表，在网页制作过程，没有指定相应的样式时，就会按照浏览器内置的样式表来渲染。

1. html,
2. body {
3. margin: 0;
4. padding: 0;
5. }
6. #title {
7. font-size: 36px;
8. color: aliceblue;
9. position: absolute;
10. width: 100%;
11. height: 6%;
12. text-align: center;
13. background-color: gray;
14. }
15. #map {
16. position: absolute;
17. top: 6%;
18. width: 69%;
19. height: 59%;
20. background-color: white;
21. }
22. #pie {
23. position: absolute;
24. top: 6%;
25. left: 69.1%;
26. width: 30.9%;
27. height: 59%;
28. background-color: white;
29. }
30. #bar {
31. position: absolute;
32. top: 65.2%;
33. width: 100%;
34. height: 34.8%;
35. background-color: white;
36. }

最后在“index.htm”l文件中引入“style.css”

1. <link rel="stylesheet" type="text/css" href="css/style.css"/>

运行代码，在浏览器中可以观察到如下图效果：



1. **绘制地图**

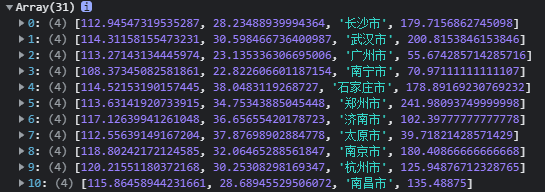
绘制地图的方式这里采用Echarts结合Mapbox的方式进行绘制，首先需要引入项目所需的js文件，这些文件需要我们提前导入到lib文件夹下。

1. <script src="lib/d3.min.js"></script>
2. <script src="lib/echarts-gl.min.js"></script>
3. <script src="lib/echarts.min.js"></script>
4. <script src="lib/mapbox-gl.js"></script>
5. <link rel="stylesheet" type="text/css" href="lib/mapbox-gl.css"/>

接下来在js文件夹中创建一个名称为“map.js”的文件。

1. mapboxgl.accessToken= · 'pk.eyJ1IjoiY3NqMTIxMzgiLCJhIjoiY2tiMDdhb2FwMDU5YjJxbzNidTUzdXM1ZCJ9.LNxHiLbA7Gii1mPdWtZNsw';
2. d3.json('json/data.json', data => {
3. var d = []
4. for(var i=0;i<data['20130109'].length;i++){
5. dd=[data['20130109'][i]['lon'],data['20130109'][i]['lat'],data['20130109'][i]['name'],data['20130109'][i]['pm2.5']]
6. d.push(dd)
7. }
8. console.log(d)
9. drawMap(d)
10. })
11. function drawMap(d) {
12. var chart = echarts.init(document.getElementById('map'));
13. var option = {
14. title:{
15. text:'全国城市PM2.5情况',
16. left:'center'
17. },
18. mapbox: {
19. center: [108.55, 36.27],
20. zoom:3.5,
21. },
22. visualMap: {
23. min: 0,
24. max: 200,
25. color: ['red','yellow' ],
26. },
27. series: [{
28. type: 'scatter3D',
29. symbolSize:25,
30. coordinateSystem: 'mapbox',
31. itemStyle:{
32. borderWidth:0.05
33. },
34. data: d
35. } ]
36. }
37. chart.setOption(option);
38. }

使用d3读取后，我们数值存放了各个城市的信息，在浏览器输出观察，数组中的值依次为经度、纬度、城市名称、pm2.5指数大小。

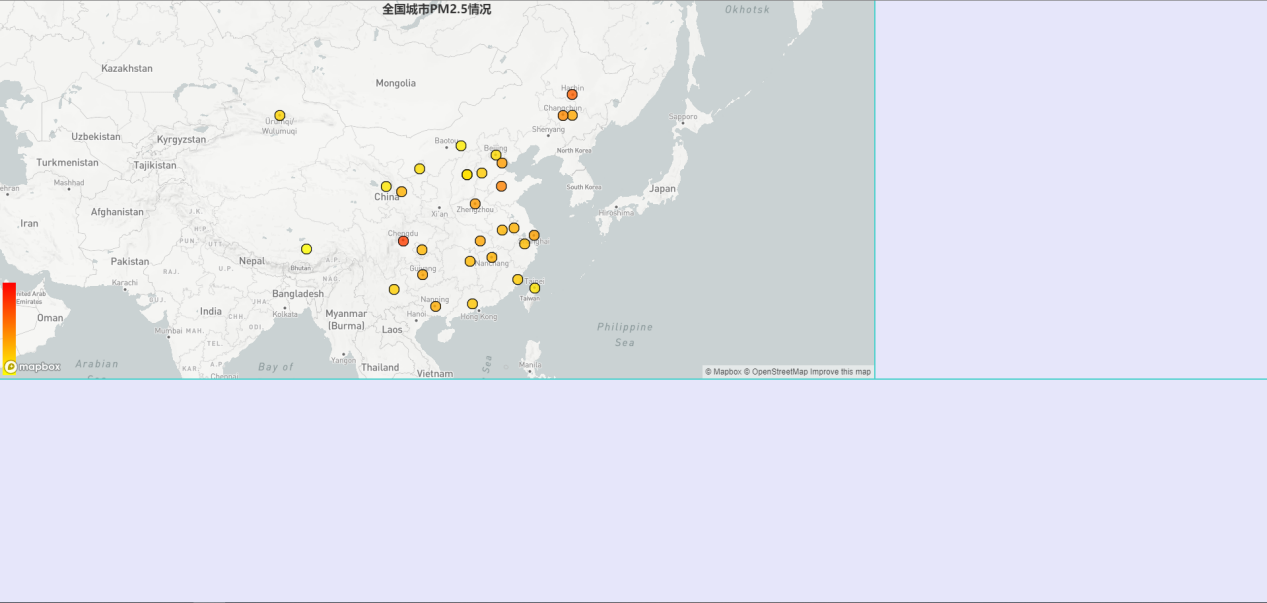


我们将会绘制地图的代码封装在drawMap函数中，将数组作为参数调用函数绘制视图，

在index.html文件中引用map.js文件。

1. <script src="js/map.js"></script>

保存后运行代码，浏览器中可以观察到地图已绘制完成，并且城市已用散点编码，颜色深浅编码pm2.5大小。



1. **绘制饼状图**

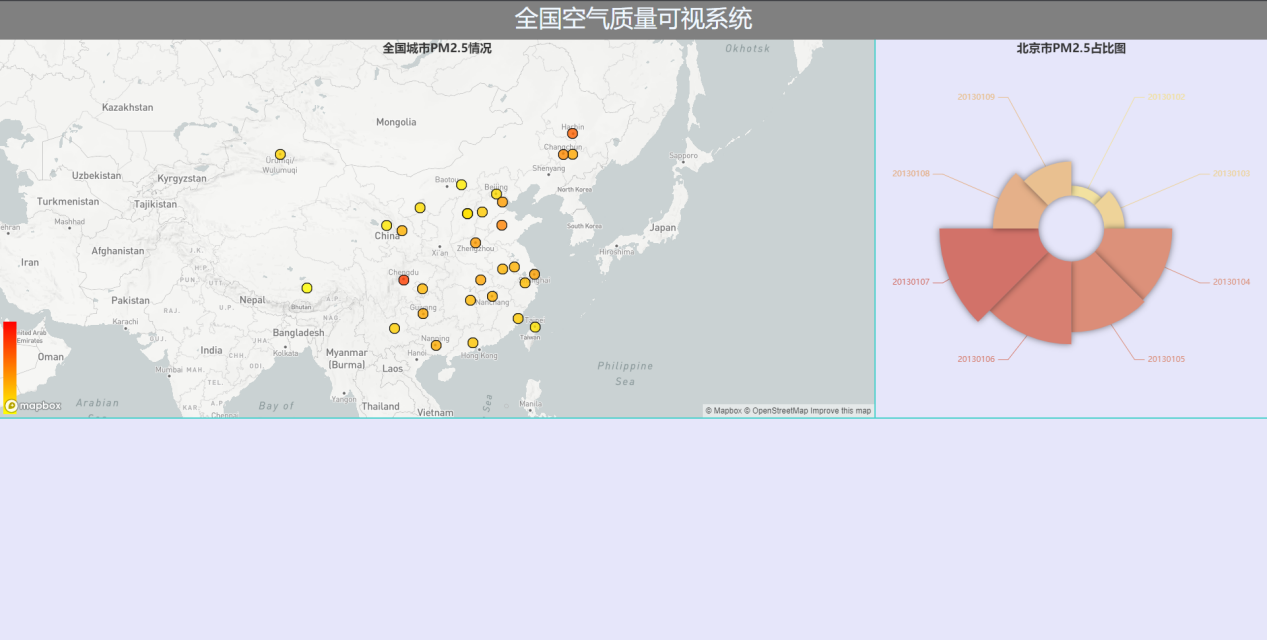
第二个视图将进行饼状图的绘制，饼状图将展示各个城市pm2.5的占比。在js文件夹中新建一个名称为“pie.js”的文件，在文件中写入以下代码：

1. function drawPie(d) {
2. let Pie = echarts.init(document.getElementById('pie'));
3. option = {
4. title: {
5. text: 'PM2.5占比图',
6. left:"center"
7. },
8. visualMap: {
9. show: false,
10. min: 0,
11. max: 200,
12. },
13. series: [{
14. type: 'pie',
15. radius: [50, 200],
16. center: ['50%', '50%'],
17. roseType: 'area',
18. itemStyle: {
19. borderRadius: 100,
20. color: '#FFC0CB',
21. shadowBlur: 10,
22. shadowColor: 'rgba(0, 0, 0, 0.5)'
23. },
24. data: d
25. }]
26. };
27. Pie.setOption(option)
28. }

该部分代码将绘制饼状图封装在drawPie函数中，在读出数据并处理后可直接调用函数。

1. let values= []
2. for (let i = 1; i < 10; i++) {
3. d3.json("json/2013010" + i + ".json", data => {
4. data['data'].forEach(d => {
5. let obj = {}
6. if (d['name'] == ‘北京市’) {
7. obj['value'] = d['pm2.5']
8. obj['name'] = '2013010' + i
9. }
10. values.push(obj)
11. })
12. if (i == 9) {
13. drawPie(values)
14. }
15. })
16. }

上面代码中使用for循环读取全部json文件，将城市名为“北京市”的pm2.5的值存入value数组中，同时将对应的日期存入date数组中，当i自增为9时调用drawBar函数。代码中for循环中定义i的关键字必须使用let声明，感兴趣的同学可以搜索一下es6，去查看var与let声明的区别。最后在index.html文件中引用pie.js文件。保存文件，运行代码，在浏览器中可以观察到饼状图绘制完成。



1. **绘制柱状图**

接下来将进行柱状图的绘制，柱状图将展示单个城市在2013年1月1日至2013年1月9日的pm2.5的情况。首先在js文件夹中创建一个名称为“bar.js”的文件，在文件中写入以下代码。

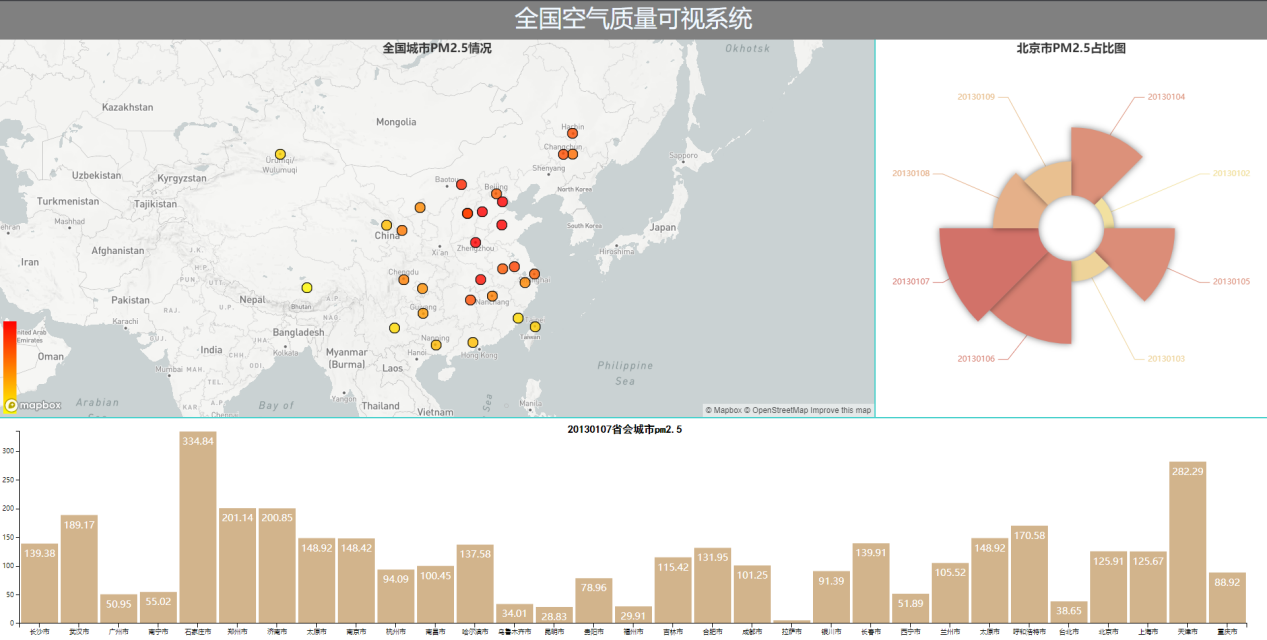
1. function drawBar(v,da) {
2. let width = 1920;
3. let height = 331;
5. let svg = d3.select("#bar")
6. .append("svg")
7. .attr("width", width)
8. .attr("height", height)
10. let padding = {
11. left: 30,
12. right: 30,
13. top: 20,
14. bottom: 20
15. };
16. svg.append("g")
17. .append('text')
18. .attr("x", width / 2 - 100)
19. .attr("y", height / 15)
20. .style('font-weight', 1000)
21. .style('font-family', '黑体')
22. .style('fill', 'black')
23. .text("省会城市pm2.5")
24. let xScale = d3.scale.ordinal()
25. .domain(d3.range(v.length))
26. .rangeRoundBands([0, width - padding.left - padding.right]);
27. let yScale = d3.scale.linear()
28. .domain([0, d3.max(v)])
29. .range([height - padding.top - padding.bottom, 0]);
30. let xAxis = d3.svg.axis()
31. .scale(xScale)
32. .orient("bottom")
33. .tickFormat((d, i) => {
34. return da[i];
35. });
36. let yAxis = d3.svg.axis()
37. .scale(yScale)
38. .orient("left");
39. let rectPadding = 4;
40. let rects = svg.selectAll(".MyRect")
41. .data(v)
42. .enter()
43. .append("rect")
44. .attr("class", (d, i)=> {
45. return cities[i]
46. })
47. .attr("transform", "translate(" + padding.left + "," + padding.top + ")")
48. .attr("x", (d, i)=> {
49. return xScale(i) + rectPadding / 2;
50. })
51. .attr("y", d=> {
52. return yScale(d);
53. })
54. .attr("width", xScale.rangeBand() - rectPadding)
55. .attr("height", d=> {
56. return height - padding.top - padding.bottom - yScale(d);
57. })
58. .attr("fill", "#D2B48C")
59. .on("mouseover", function(d, i) {
60. d3.select(this)
61. .attr("fill", "#696969");
62. })
63. .on("mouseout", function(d, i) {
64. d3.select(this)
65. .transition()
66. .duration(500)
67. .attr("fill", "#D2B48C");
68. })
69. .on("click", params=> {
70. pie(d3.event.path[0].className.animVal)
71. });
72. //添加文字元素
73. let texts = svg.selectAll(".MyText")
74. .data(v)
75. .enter()
76. .append("text")
77. .attr("class", "MyText")
78. .attr("transform", "translate(" + padding.left + "," + padding.top + ")")
79. .attr("x", (d, i)=> {
80. return xScale(i) + rectPadding / 2;
81. })
82. .attr("y",d=>{
83. let min = yScale.domain()[0];
84. return yScale(min);
85. })
86. .transition()
87. .delay((d,i)=>{
88. return i \* 200;
89. })
90. .duration(1000)
91. .ease("bounce")
92. .attr("y",d=>{
93. return yScale(d);
94. })
95. .attr("dx", function() {
96. return (xScale.rangeBand() - rectPadding) / 2;
97. })
98. .attr("dy", d=> {
99. return 20;
100. })
101. .text(d=> {
102. return d.toFixed(2);
103. });
104. //添加x轴
105. svg.append("g")
106. .attr("class", "axis")
107. .attr("transform", "translate(" + padding.left + "," + (height - padding.bottom) + ")")
108. .call(xAxis);
109. //添加y轴
110. svg.append("g")
111. .attr("class", "axis")
112. .attr("transform", "translate(" + padding.left + "," + padding.top + ")")
113. .call(yAxis);
114. }

将绘制柱状图封装在drawBar函数中，接下就是读数据并处理数据，继续在bar.js中写入以下代码。

1. d3.json("json/20130101.json", data => {
2. PM25 = []
3. cities = []
4. for (let i = 0; i < data['data'].length; i++) {
5. PM25.push(data['data'][i]['pm2.5'])
6. cities.push(data['data'][i]['name'])
7. }
8. drawBar(PM25, cities)
9. })

最后在index.html文件中引入bar.js文件。最后将style.css文件中map、pie、bar的背景颜色修改为白色。

保存文件，运行代码，可在浏览器中观察到柱状图绘制完成。



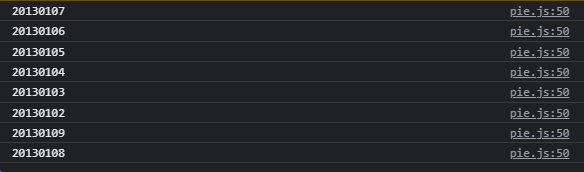
1. **交互设计**

该部分将会为大家介绍点击视图如何在另一视图中进行视图重绘制，重绘制通俗说就是新数据在视图中展示出来。

首先，我们需要在pie.js添加一下代码：

1. Pie.on("click",params=>{
2. console.log(params.name)
3. })

代码中，我们对Pie实例使用on函数进行“click”事件的监听。添加代码后，我们保存并且运行，在浏览器中点击饼状图图并且在控制台中可看见输出了对应柱状图的class名。



最后打开map.js文件，将文件中的代码修改为以下格式。

1. mapboxgl.accessToken = 'pk.eyJ1IjoiY3NqMTIxMzgiLCJhIjoiY2tiMDdhb2FwMDU5YjJxbzNidTUzdXM1ZCJ9.LNxHiLbA7Gii1mPdWtZNsw';
2. draw('20130101')//系统初始化
3. function draw(s) {
4. d3.json('json/'+s+'.json', data => {
5. var d = []
6. for (var i = 0; i < data['data'].length; i++) {
7. dd=[data['data'][i]['lon'],data['data'][i]['lat'], data['data'][i]['name'], data['data'][i]['pm2.5']]
8. d.push(dd)
9. }
10. var chart = echarts.init(document.getElementById('map'));
11. var option = {
12. mapbox: {
13. center: [108.55, 36.27],
14. zoom: 3.5,
15. },
16. visualMap: {
17. min: 0,
18. max: 200,
19. color: ['red', 'yellow'],
20. },
21. series: [{
22. type: 'scatter3D',
23. symbolSize: 25,
24. coordinateSystem: 'mapbox',
25. itemStyle: {
26. borderWidth: 0.05
27. },
28. data: d
29. }]
30. }
31. chart.setOption(option);
32. })
33. }

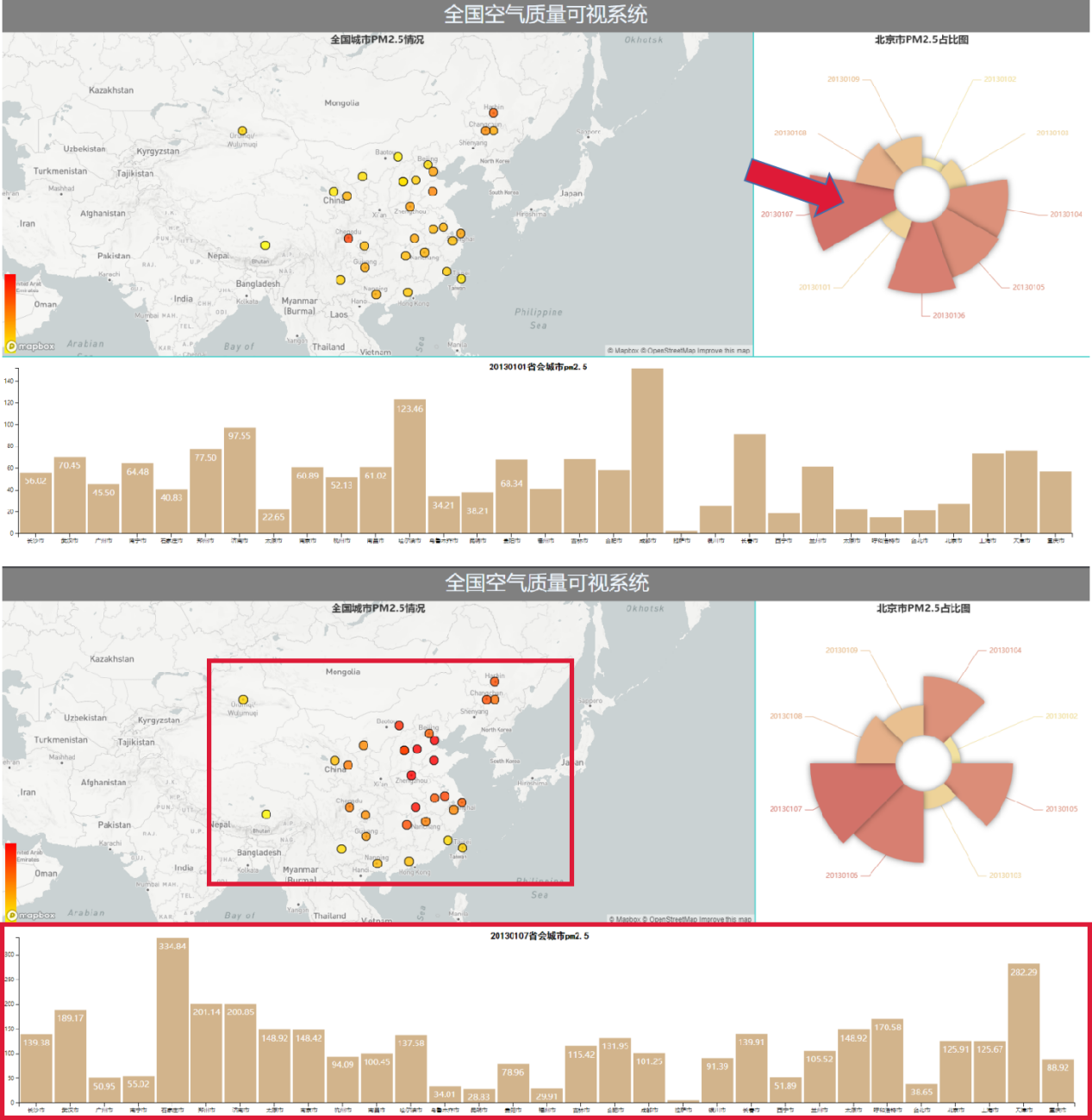
接下来降为柱状图添加交互，首先我们将柱状图数据处理部分封装在函数里。

1. function bar(date){
2. d3.json("json/"+date+".json", data => {
3. PM25 = []
4. cities = []
5. for (let i = 0; i < data['data'].length; i++) {
6. PM25.push(data['data'][i]['pm2.5'])
7. cities.push(data['data'][i]['name'])
8. }
9. drawBar(PM25, cities,date)
10. })
11. }

同时我们需要更改一下代码，在drawBar函数中新增一个title变量，用于更改视图标题的日期。

1. function drawBar(v, da,title) {
2. ......

在修改的代码中的我们可以看见读取json文件的索引修改为点击饼状图时传输的参数。保存文件，运行代码，在浏览器中点击饼状图观察地图与柱状图的变化。

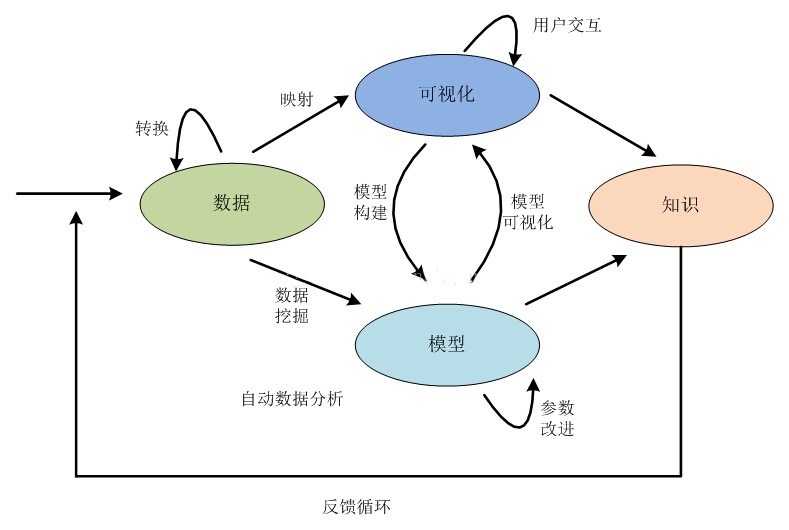


## 5.2本章小作业

在上面的案例中我们只针对各视图日期部分进行了数据交互，未对点击柱状图时饼状图修改为对应城市的数据进行交互添加，本章小作业要求对上述交互进行实现。

# 可视化系统设计与评估

数据可视化不仅是一门包含各种算法的技术，还是一个具有方法论的学科。因此，除了复杂的编程工作外，还需要采用系统化的思维设计数据可视化方法与工具。参考欧洲学者Daniel Keim等人在2008年提出的可视分析标准流程，起点是输入的数据，终点是提炼的知识。从数据到知识有两个途径：交互的可视化方法和自动的数据挖掘方法。两个途径的中间结果分别是对数据的交互可视化结果和从数据中提炼的数据模型。用户既可以对可视化结果进行交互的修正，也可以调节参数以修正模型。



**图6-1 可视分析学标准流程**

数据可视化的设计流程可以简化为四个级联的层次：

第一层，问题刻画层，概括现实生活中用户遇到的问题。设计人员首先要了解目标用户的任务需求和数据属于哪个特定的目标领域，每个领域通常都有其特有的术语来描述数据和问题，在通常情况下，对特定领域工作流程特征的描述是一个详细的问题集或者用户收集异构数据的工作过程。描述务必要细致，因为这可能是对领域问题的直接复述或整个设计过程中数据的描述。设计人员需要收集与问题相关的信息，建立系统原型，并通过观察用户与原型系统的交互过程来判断所提出方案的实际效果。

第二层，抽象层，将特定领域的任务和数据映射到抽象且通用的任务及数据类型。第二层将第一层确定的任务和数据从采用特定领域的专有名词的描述转化为更抽象、更通用的信息可视化术语的描述。高层次的通用任务分类包括不确定性计算、关联分析、求证和参数确定等。与数据相关的底层通用任务包括取值、过滤、统计、极值计算、排序、确定范围、提取分布特征、离群值计算、异常检测、趋势预测、聚簇和关联。在数据抽象过程中，可视化设计人员需要考虑是否要将用户提供的数据集转化为另一种形式，以及使用何种转化方法，以便于选择合适的可视编码，完成分析任务。

第三层，编码层，设计与数据类型相关的视觉编码及交互方法。视觉编码和交互这两个层面通常相互依赖。为应对一些特殊需求，第二层确定的抽象任务应被用于指导视觉编码方法的选取。

第四层，算法层，实现算法和交互。设计与前三个层次匹配的具体算法，相当于一个细节描述的过程。第三层确定应当呈现的内容以及如何呈现，而第四层解决的是如何完成的问题。当然，两层之间相互影响和制约。将可视化设计的层次嵌套模型应用于实际的数据可视化系统设计，需要考虑各个层次面临的潜在风险和对风险的评估方法。

## 6.1 数据可视化设计流程

一个完整的数据可视化过程可以分为四个步骤：

**（1）确定可视化的主题**

首先得确定数据围绕什么样的主题来组织。具体地说，可以是业务的目标、一个需求、一个问题。比如，信贷行业分析不同信用等级的客户按时还款率；制造业分析不同生产线产品的合格率；餐饮业分析客户对某种菜品的评价；物流公司分析不同地区的运输时效等，都可以作为数据可视化的一个主题。

**（2）提炼可视化需要的数据**

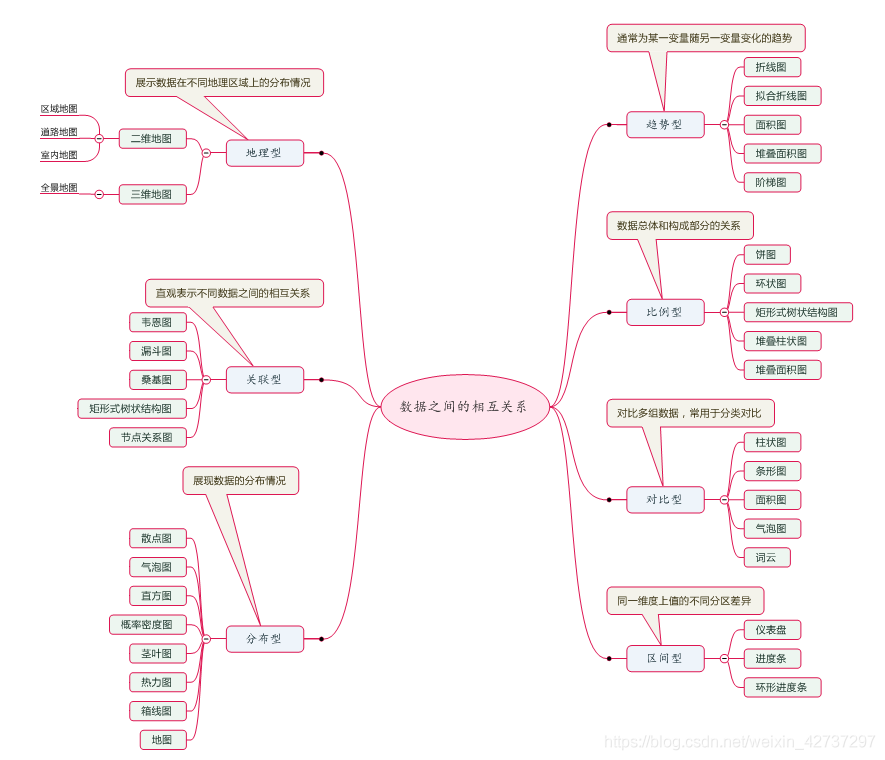
确定好数据可视化的主题之后，我们需要获取并组织该主题需要的数据，这里分三步进行：

* 确定数据衡量指标。不同的问题和数据，思考方式和角度不一样，衡量的指标也就不同，我们应根据业务需求或者用户需求，选择合适的指标关注。
* 明确数据间的关系。
* 确定重点指标。根据用户需求，对数据的指标做一个重要性排序，选择较为重要的几个指标，在后续设计过程中考虑到着重显示。

数据之间的相互关系

**（3） 确定图表**

统计图表作为数据可视化形式之一，常用的图表类型有：柱状图、饼图、折线图、堆叠图、直方图、等值线图、散点图、维恩图、热力图等。我们根据数据之间的相互关系以及想要展示的结果来选择合适的图表，常见的数据关系和图表类型的对应关系如下图：



数据关系和图表类型的对应关系

**（4）可视化设计**

数据可视化的目的是展示数据所包含的信息。主要包括两个方面：一是进行可视化布局的设计，二是数据图形化的呈现。

1、页面布局设计

* 简洁

过于复杂的可视化可能会对用户的理解造成麻烦，甚至会引起对数据的误读，因此在可视化布局中要求设计者避免使用复杂的设计效果，做到简洁明了。

* 聚焦

设计者应该通过适当的排版布局，将用户的注意力集中到可视化结果中最重要的区域，从而使重要的数据信息脱颖而出，抓住用户的注意力，提升用户信息解读的效率。

* 平衡

平衡原则要求充分利用可视化的空间，尽量使重要元素位于可视化空间的中心区域，同时确保各元素在可视化空间中平衡分布。

2、图表制作

影响图表呈现效果的，主要有两个影响因素，一个是数据层面的，一个是非数据层面的。

* 数据层面

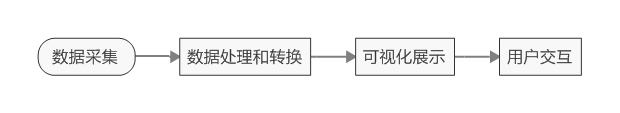
若数据中存在极端值、空值或过多分类项等，会极大影响可视化的效果呈现，如柱形图中柱形条的高度、气泡图中气泡的大小、饼图中的分类项太多等。

* 非数据层面

非数据层面大部分是图表的设计问题，比如图表的背景颜色、网格线的深浅有无、外边框等等，这类元素是辅助用户理解图表的次要元素，但如果不加处理，视觉上就不够聚焦，也不够美观，可能会干扰到你真正想展示的数据信息。

## 6.2 数据可视化系统开发流程

数据可视化的基本流程是一个以数据流向为主线的完整流程，主要包括数据采集、数据处理和转换、可视化展示、用户交互。一个完整的可视化过程，可以看成数据流经过一系列处理模块并得到转化的过程。



数据可视化基本流程

### 6.2.1数据采集

数据采集的方法很多，根据数据来源可分为一手数据和二手数据。一手数据是指通过研究者实施的调查或实验活动获得的数据，因此获取一手数据的方法有调查和实验。二手数据又可以分为内部数据和外部数据，内部数据一般是指企业内部在生产活动和日常运营中产生的业务、渠道、成本、收益等数据，可通过SQL从数据仓库中调用数据或者“埋点”采集数据；外部数据一般是来源公开的数据，例如行业数据、官网公布的数据、统计部门发布的数据等，可通过编写爬虫的方式获取。

### 6.2.2数据处理和转换

数据要想进行有效的可视化展示，必须经过转换处理，数据处理和变换包括数据预处理和数据特征挖掘两个过程。首先，采集到的数据可能含有噪声、误差或者缺失值，数据质量不高；另外，数据特征不明确，通常隐藏在大量的数据之中，因此需要挖掘提取特征。

1、数据预处理问题

* 缺失值填充。由数据本身的缺失值或者记录数据时人为失误造成缺失值，根据数据整体情况进行删除含有缺失值的数据或者用均值填充缺失值。
* 数据标准化。收集的数据可能存在数据的规格不一样的情况，也可能需要将定性数据转为定量数据表示。
* 数据值错误。数据中包含非正常的数据值，比如身高一栏中显示性别.
* 数据不一致。由于数据冗余或者并发控制不当的原因造成同样的信息莫哪些数据显示前后矛盾。
* 离群数据。某些明显不同于大部分数据对象的值。

2、数据特征挖掘

比如用户想了解某个地区最受欢迎的快递公司，我们可以通过从收集到的该地区的快递数据中，人为选择特征，比如对下单量、客户评价这两列赋予大权重，也可以对数据降维处理，选择相关性高的特征。

另外，根据业务需求以及可视化展示需求，可继续对数据做一些处理。

### 6.2.3可视化展示（映射）

数据经过清洗、去噪声等预处理和特征提取之后，就可以进行可视化展示的设计了，可视化展示就是将数据所包含的信息映射成可视化元素的过程，是整个可视化基本流程的核心。可视化元素由三部分组成：可视化空间、标记、视觉通道

1、可视化空间

数据可视化的显示空间通常是二维，但是三维的也可以在二维空间绘图显示。

2、标记

标记，是指数据属性到可视化几何图形元素的映射，用来代表数据属性的归类。也叫图形元素，是出现在空间中的视觉元素，包括点、线、面、体四种，分别对应于常见的散点图、折线图、矩形树图、三维柱状图。

3、视觉通道

视觉通道就是标记的属性，包括标记的大小（点的大小、线条的长度、面的面积、体的体积）、颜色（色调、饱和度）、形状（圆形、方形、六边形）、位置、方向等。

标记通常和视觉通道结合来在可视化展示中描述信息，例如用点的大小表示数量多少，用不同的颜色来区分不同类别。

### 6.2.4用户交互

对数据进行分析和可视化的目的是将数据所包含的信息及其特征直观易懂的传达给用户，交互界面如果混乱不堪，不仅仅影响美观，还对用户获取数据信息造成了困难，降低工作效率。

这里介绍几种常见的交互方式：

1、滚动和缩放：当可视化界面数据较多、图片较大导致无法完整展示时，滚动和缩放是一种非常有效的交互方式，比如地图、折线图的信息细节等；

2、颜色映射的控制：合理的颜色搭配以及对不同数据较明显的色彩对比，有助于美观和用户理解；

3、数据映射方式选择：指用户可以根据自己的需要和喜好选择数据可视化映射元素，探索自己想要的信息；

4、数据细节层次控制：这个指的是隐藏一些详细描述，当点击对应主题时再显示。

### 6.2.5可视化评估

帮助用户认识到可视化的作用，在专业领域或其它使用场景中接受可视化

可视化评估方法：定量评估和定性评估

定量评估：1、列出假设、2、设计实验、3、完成实验、4、分析结果

**1、列出假设**

假设尽可能具体，必须使用可以被证伪的假设，可以使虚假设

例如：可视化方法A比B好——不恰当，无法检验

用户使用可视化方法A完成T任务花费的时间比B少

可视化评估时涉及到的任务要简单

常见的任务有：

识别：用户通过可视化在数据中识别目标

定位：找到指定特征或目标的位置

区分：将数据中不同元素区分开

分类：将数据中不同元素划分为不同类型

聚类：按一定的相似法则将相似的数据聚合成一类

排序：将可视化的对象按一定规则排序

比较：对两个或多个可视化对象进行比较并发现相同或不同之处

关联：判断可视化对象之间的关联

**2、设计实验**

独立变量：可能影响假设验证的因素，如任务、可视化方法、可视化参数

因变量：随独立变量变化而变化的因素，如完成任务的实践、完成任务的准确性等

**3、完成实验**

招募被试：在满足条件的人中尽可能广泛的招募被试人员，如性别比例、年龄范围分布、学历程序分布均衡。

**4、分析结果**

分析对象：一组被试反复实验结果

分析方法：统计学假设检验方法。如显著性检验、卡方检验、似然比检验、t检验、F检验等

定性评估主要方法：访谈、观察

观察：被试人员在操作过程中的问题、情绪、发现等

注意要降低对被试人员的干扰

访谈：引导用户描述细节，善于倾听

### 6.2.6 课题小组讨论

针对课程中发布的多个研究课题，选定其中的一个课题安装如下流程开展分析讨论：

1、分析数据，查看给定数据集的数据特征、数据规模、数据类别等，完成基本数据清洗等预处理操作；

2、确定分析任务，根据对数据集的分析，小组讨论确定具体的分析任务，具体讨论每个任务对应的数据特征挖掘方法、功能验证方法等；

3、系统界面设计，根据每个分析任务，选择合适的可视化编码方式完成可视化映射，多个视图在系统界面上如何布局与呈现。针对单个视图与多个视图之间的关联操作设计交互方式，尤其针对时变数据、层次结构数据、多维属性数据或特征结构数据设计时间刷、分层、过滤等特殊的交互方式；

4、任务分工与时间节点，小组讨论过程中拟定组长，由组长统一完成任务分工，每位小组完成其中某个或多个视图的开发任务。这种多人协同开发的模式，需要提前规范好函数命名规范和变量定义方式等，同时注意关联交互中需要的数据传递。对所有任务设定时间节点，在开发过程中多交流多讨论，保证系统按时按质完成开发；

5、同期撰写需求文档、开发文档和测试文档，录制系统功能视频与答辩ppt。

总之，数据可视化的基本要素包括复杂数据的表示与变换、可扩展的数据智能可视化和支持用户分析决策的交互方法与集成环境等。它引导的分析推理模式，是探索复杂数据中蕴涵的新规律和新现象的催化剂。21世纪初以来，国际上逐步形成了可视化技术的研究热潮。希望大家借助这门课，掌握基本的数据可视化开发技能，为大家做学术研究或找工作就业都能提供