Assignment 02

Number：12532721	Name：谢正扬

1. Significant earthquakes since 2150 B.C.

The Significant Earthquake Database contains information on destructive earthquakes from 2150 B.C. to the present. On the top left corner, select all columns and download the entire significant earthquake data file in .tsv format by clicking the Download TSV File button. Click the variable name for more information. Read the file (e.g., earthquakes-2025-10-29_15-11-32_+0800.tsv) as an object and name it Sig_Eqs.

⑴ [5 points] Compute the total number of deaths caused by earthquakes since 2150 B.C. in each country, and then print the top ten countries along with the total number of deaths.

⑵ [10 points] Compute the total number of earthquakes with magnitude larger than 6.0 (use column Mag as the magnitude) worldwide each year, and then plot the time series. Do you observe any trend? Explain why or why not?

⑶ [10 points] Write a function CountEq_LargestEq that returns both (1) the total number of earthquakes since 2150 B.C. in a given country AND (2) the date of the largest earthquake ever happened in this country. Apply CountEq_LargestEq to every country in the file, report your results in a descending order.

1.1.过程分析

⑴首先读取地震数据文件（TSV 格式），确保关键列如年份、国家、震级和死亡人数等均存在。然后对缺失值进行清理，仅保留包含完整信息的记录。接着通过 groupby('Country') 按国家聚合，计算每个国家地震造成的总死亡人数，并按降序排列，输出前十个死亡人数最多的国家。

⑵研究全球震级大于 6.0 的地震在时间上的变化趋势。思路是先筛选出 Mag > 6.0 的地震事件，再按年份统计事件数目。最后使用 matplotlib 绘制每年强震数量的时间序列图，观察地震发生频率随时间的变化趋势。

⑶针对每个国家，输出两个信息：该国地震事件的总次数；该国历史上震级最大的地震日期。实现思路为：编写函数 CountEq_LargestEq(country)，筛选出目标国家的地震数据，统计总数量，并用 idxmax() 找出最大震级事件，格式化输出日期。最后将所有国家的结果整理为 DataFrame，并按地震次数降序排列。

1.2.代码实现

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

# -------------------------
# 1. 读取并预处理数据
# -------------------------
file_path = r"C:\Users\xiezhengyang\Desktop\earthquakes-2025-10-29_21-04-
53_+0800.tsv"

Sig_Eqs = pd.read_csv(file_path, sep='\t', low_memory=False)
print("✅ 数据读取成功！ ")

# 检查关键列
required_cols = ['Year', 'Mo', 'Dy', 'Country', 'Mag', 'Total Deaths', 'Region', 'Location Name']
for col in required_cols:
   if col not in Sig_Eqs.columns:
      Sig_Eqs[col] = None

Sig_Eqs = Sig_Eqs.dropna(subset=['Year', 'Country', 'Mag', 'Total Deaths'])
Sig_Eqs['Year'] = Sig_Eqs['Year'].astype(int)

print(f"数据清洗完成，剩余 {len(Sig_Eqs)} 条有效记录。\n")

# -------------------------
# 1.1 各国地震死亡总数 Top 10
# -------------------------
deaths_by_country = (
   Sig_Eqs.groupby('Country')['Total Deaths']
   .sum()
   .sort_values(ascending=False)
   .head(10)
)
print("===== 1.1 各国地震死亡总人数 Top 10 =====")
print(deaths_by_country.to_string(), "\n")

# -------------------------
# 1.2 每年震级 > 6.0 的地震次数
# -------------------------
eqs_gt6 = Sig_Eqs[Sig_Eqs['Mag'] > 6.0]
eqs_per_year = eqs_gt6.groupby('Year').size()

print("===== 1.2 每年震级 > 6.0 的地震次数 =====\n")
plt.figure(figsize=(12, 5))
plt.plot(eqs_per_year.index, eqs_per_year.values, color='steelblue', linewidth=1.5)
plt.title("Number of Earthquakes (Mag > 6.0) per Year", fontsize=14)
plt.xlabel("Year")
plt.ylabel("Number of Earthquakes")
plt.grid(True, alpha=0.3)
plt.tight_layout()
plt.show()


# -------------------------
# 1.3 各国地震总数与最大地震信息（格式化 YYYY-MM-DD，高亮）
# -------------------------
def CountEq_LargestEq(country):
    sub_df = Sig_Eqs[Sig_Eqs['Country'] == country]
    total_count = sub_df.shape[0]
    max_eq = sub_df.loc[sub_df['Mag'].idxmax()]

    year = int(max_eq['Year'])
    month = int(max_eq['Mo']) if pd.notna(max_eq['Mo']) else 1
    day = int(max_eq['Dy']) if pd.notna(max_eq['Dy']) else 1
    date_str = f"{year:04d}-{month:02d}-{day:02d}"

    return {
        'Total_Count': total_count,
        'Max_Magnitude': max_eq['Mag'],
        'Max_Date': date_str,
        'Total_Deaths': max_eq['Total Deaths'],
        'Region': max_eq['Region'] if pd.notna(max_eq['Region']) else '',
        'Location': max_eq['Location Name'] if pd.notna(max_eq['Location Name']) else ''
    }

countries = Sig_Eqs['Country'].unique()
result_df = pd.DataFrame([CountEq_LargestEq(c) for c in countries], index=countries)
result_df = result_df.sort_values(by='Total_Count', ascending=False)

# -------------------------
# 高亮函数
# -------------------------
def highlight_max_magnitude(val):
    color = 'background-color: yellow' if val == result_df['Max_Magnitude'].max() else ''
    return color

def highlight_top_deaths(val):
    color = 'background-color: lightcoral' if val >= result_df['Total_Deaths'].max() else ''
    return color

# 输出带高亮的表格（Jupyter Notebook）
print("===== 1.3 各国地震总数与最大地震信息 =====\n")
styled_df = result_df.style.applymap(highlight_max_magnitude, subset=['Max_Magnitude'])
\
                    .applymap(highlight_top_deaths, subset=['Total_Deaths']) \
                    .set_caption("各国地震总数与最大地震信息（高亮）")
display(styled_df)
```

1.3.输出结果

===== 1.1 各国地震死亡总人数 Top 10 =====

```
Country
CHINA           2049509.0
TURKEY           713726.0
IRAN             624312.0
HAITI            323782.0
JAPAN            314031.0
ITALY            300894.0
INDONESIA        279704.0
PAKISTAN         143350.0
ECUADOR          129445.0
TURKMENISTAN     110412.0
```

===== 1.2 每年震级 > 6.0 的地震次数 =====

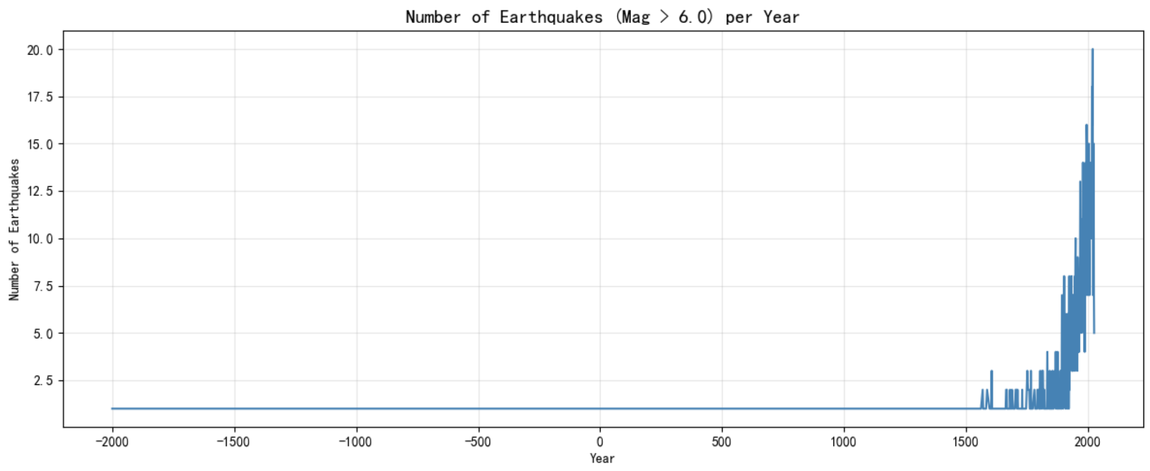


图 1-1 每年震级 > 6.0 的地震次数时间序列图

趋势与原因分析：从图表可以看出，每年的地震总数呈上升趋势。我认为主要原因是现代地震监测技术的进步。其次，我怀疑近年来地壳运动变得更加剧烈。

===== 1.3 各国地震总数与最大地震信息 =====

各国地震总数与最大地震信息（高亮）

	Total_Count	**Max_Magnitude**	**Max_Date**	**Total_Deaths**	**Region**	**Location**
CHINA	267	8.500000	1668-07-25	42578.000000	30.000000	CHINA: SHANDONG PROVINCE

各国地震总数与最大地震信息（高亮）

	Total_Count	**Max_Magnitude**	**Max_Date**	**Total_Deaths**	**Region**	**Location**
INDONESIA	158	9.100000	2004-12-26	227899.000000	60.000000	INDONESIA: SUMATRA: ACEH: OFF WEST COAST
IRAN	148	7.900000	0856-12-22	200000.000000	140.000000	IRAN: DAMGHAN, QUMIS
JAPAN	117	9.100000	2011-03-11	18428.000000	30.000000	JAPAN: HONSHU
TURKEY	107	7.800000	1939-12-26	32700.000000	140.000000	TURKEY: ERZINCAN
PERU	82	8.600000	1619-02-14	350.000000	160.000000	PERU: TRUJILLO, PIURA, SANTA
GREECE	67	8.000000	0365-07-21	5000.000000	130.000000	GREECE: CRETE: KNOSSOS
MEXICO	66	8.600000	1787-03-28	11.000000	150.000000	MEXICO: SAN MARCOS, OAXACA
TAIWAN	66	8.200000	1920-06-05	5.000000	30.000000	TAIWAN
PHILIPPINES	61	8.700000	1897-09-21	13.000000	170.000000	PHILIPPINES: MINDANAO, ZAMBOANGA, SULU, ISABELA
CHILE	56	9.500000	1960-05-22	2226.000000	160.000000	CHILE: PUERTO MONTT, VALDIVIA
USA	53	9.200000	1964-03-28	139.000000	150.000000	ALASKA
ITALY	53	7.500000	1915-01-13	29978.000000	130.000000	ITALY: MARSICA, AVEZZANO, ABRUZZI

各国地震总数与最大地震信息（高亮）						
	Total_Count	**Max_Magnitude**	**Max_Date**	**Total_Deaths**	**Region**	**Location**
COLOMBIA	38	8.100000	1979-12-12	600.000000	160.000000	COLOMBIA: OFF SHORE, PACIFIC OCEAN
AFGHANISTAN	37	7.500000	2015-10-26	399.000000	40.000000	AFGHANISTAN: HINDU KUSH
INDIA	34	8.600000	1950-08-15	1530.000000	60.000000	INDIA-CHINA
PAKISTAN	29	8.000000	1945-11-27	300.000000	60.000000	PAKISTAN: MAKRAN COAST
PAPUA NEW GUINEA	27	8.000000	2000-11-16	2.000000	170.000000	PAPUA NEW GUINEA: NEW IRELAND, DUKE OF YORK
ECUADOR	23	8.800000	1906-01-31	1000.000000	160.000000	ECUADOR: OFF COAST
ALGERIA	23	7.100000	1980-10-10	5000.000000	15.000000	ALGERIA: NORTHERN
GUATEMALA	20	7.500000	1773-07-29	100.000000	100.000000	GUATEMALA: ANTIGUA
VENEZUELA	20	8.000000	1530-09-01	4.000000	90.000000	VENEZUELA: CUMANA
RUSSIA	15	9.000000	1952-11-04	10000.000000	50.000000	RUSSIA: KAMCHATKA PENINSULA
MYANMAR (BURMA)	15	7.700000	2025-03-28	3815.000000	60.000000	MYANMAR (BURMA); THAILAND
TAJIKISTAN	14	7.400000	1907-10-21	12000.000000	40.000000	TAJIKISTAN: KARATAG

各国地震总数与最大地震信息（高亮）						
	Total_Count	**Max_Magnitude**	**Max_Date**	**Total_Deaths**	**Region**	**Location**
NEPAL	13	8.000000	1934-01-15	10600.000000	60.000000	NEPAL; INDIA: BIHAR
EL SALVADOR	12	7.700000	2001-01-13	844.000000	100.000000	EL SALVADOR; GUATEMALA
COSTA RICA	12	7.600000	1941-12-05	6.000000	100.000000	COSTA RICA-PANAMA
ARGENTINA	12	7.500000	1977-11-23	65.000000	160.000000	ARGENTINA: SAN JUAN PROVINCE: MENDOZA
HAITI	9	8.000000	1842-05-07	5300.000000	90.000000	HAITI: CAP-HAITIEN
NEW ZEALAND	9	7.800000	2016-11-13	2.000000	170.000000	NEW ZEALAND: AMBERLEY
SPAIN	9	6.800000	1680-10-09	70.000000	130.000000	SPAIN: MALAGA
SOUTH AFRICA	9	6.300000	1969-09-29	12.000000	10.000000	SOUTH AFRICA: CAPE PROVINCE
SOLOMON ISLANDS	9	8.100000	1977-04-21	18.000000	170.000000	SOLOMON ISLANDS
AZERBAIJAN	8	6.900000	1667-11-01	80000.000000	40.000000	AZERBAIJAN: SHEMAKHA (SAMAXI)
POLAND	7	4.200000	2016-11-29	8.000000	120.000000	POLAND: RUDNA
CONGO	6	7.000000	1992-09-11	8.000000	10.000000	CONGO: KABALO

各国地震总数与最大地震信息（高亮）

	Total_Count	Max_Magnitude	Max_Date	Total_Deaths	Region	Location
BANGLADESH	6	6.900000	1885-07-14	75.000000	60.000000	BANGLADESH:
ALBANIA	6	6.600000	1851-10-12	2000.000000	130.000000	ALBANIA: VLORE (VALONA),BERAT; TURKEY
CROATIA	5	7.200000	1667-04-06	5000.000000	130.000000	BALKANS NW: CROATIA: DUBROVNIK: RAGUSA
RWANDA	5	5.800000	2015-08-07	3.000000	10.000000	RWANDA: BUKAVU
TANZANIA	5	6.000000	1964-05-07	1.000000	10.000000	TANZANIA
NICARAGUA	5	7.600000	1992-09-02	170.000000	100.000000	NICARAGUA: MASACHAPA; COSTA RICA
PORTUGAL	5	8.500000	1755-11-01	50000.000000	130.000000	PORTUGAL: LISBON
AZORES (PORTUGAL)	5	6.900000	1980-01-01	69.000000	70.000000	AZORES: TERCEIRA, ANGRA DO HEROISMO
ROMANIA	5	7.500000	1977-03-04	1641.000000	110.000000	ROMANIA: BUCHAREST
TURKMENISTAN	4	7.500000	1946-11-04	400.000000	40.000000	TURKMENISTAN
EGYPT	4	7.200000	1995-11-22	11.000000	15.000000	EGYPT: NUWAYBI; SAUDI ARABIA; ISRAEL; JORDAN
MOROCCO	4	6.800000	2023-09-08	2946.000000	15.000000	MOROCCO: MARRAKECH, SAFI

各国地震总数与最大地震信息（高亮）

	Total_Count	Max_Magnitude	Max_Date	Total_Deaths	Region	Location
GEORGIA	4	7.000000	1991-04-29	270.000000	40.000000	GEORGIA: DZHAVA, CHIATURA, AMBROLAURI
BOLIVIA	4	8.200000	1994-06-09	5.000000	160.000000	BOLIVIA-PERU: AREQUIPA, FELT IN N AND S AMERICA
DOMINICAN REPUBLIC	4	7.900000	1946-08-04	1790.000000	90.000000	DOMINICAN REPUBLIC: NORTHEASTERN COAST
CANADA	4	7.700000	2012-10-28	1.000000	150.000000	CANADA: QUEEN CHARLOTTE ISLANDS
KYRGYZSTAN	4	8.000000	1911-01-03	450.000000	40.000000	KAZAKHSTAN: ALMA-ATA, TURKESTAN; AFGHANISTAN
BULGARIA	4	7.200000	1901-03-31	4.000000	110.000000	BULGARIA: BALCHIK
BRAZIL	4	5.100000	1922-01-27	1.000000	160.000000	BRAZIL: SAO PAULO
MALAWI	3	6.100000	1989-03-10	9.000000	10.000000	MALAWI: SALIMA, DEDZA, MOHINJI
MACEDONIA	3	6.000000	1963-07-26	1070.000000	130.000000	BALKANS NW: MACEDONIA: SKOPJE
BOSNIA-HERZEGOVINA	3	5.700000	2022-04-22	1.000000	130.000000	BALKANS NW: BOSNIA-HERZEGOVINA: MOSTAR, STOLAC
SLOVENIA	3	6.500000	1511-03-26	15.000000	130.000000	BALKANS NW: SLOVENIA: IDRIJA,SKOFJA LOKA

各国地震总数与最大地震信息（高亮）						
	Total_Count	**Max_Magnitude**	**Max_Date**	**Total_Deaths**	**Region**	**Location**
USA TERRITORY	3	7.300000	1867-11-18	24.000000	90.000000	VIRGIN ISLANDS
SOUTH KOREA	3	6.300000	0780-01-01	100.000000	30.000000	SOUTH KOREA: KYONGJU
ISRAEL	3	6.600000	1759-10-30	2000.000000	140.000000	ISRAEL: ZEFAT (SAFED)
VANUATU	3	7.500000	1999-11-26	10.000000	170.000000	VANUATU ISLANDS: PENTECOST
ETHIOPIA	3	6.200000	1961-06-01	160.000000	10.000000	ETHIOPIA: KARAKORE
SERBIA	3	5.700000	2002-04-24	1.000000	130.000000	BALKANS NW: KOSOVO; MACEDONIA: N
YEMEN	3	6.000000	1982-12-13	2800.000000	60.000000	YEMEN: DHAMAR
UGANDA	3	6.600000	1966-03-20	157.000000	10.000000	UGANDA: KICHWAMBA, BONDIBOGYO; TANZANIA; DR CONGO
HONDURAS	3	7.300000	2009-05-28	7.000000	90.000000	HONDURAS: NORTHERN; BELIZE
GUADELOUPE	2	8.300000	1843-02-08	5000.000000	90.000000	GUADELOUPE: POINTE-A-PITRE
TRINIDAD AND TOBAGO	2	7.500000	1888-01-10	1.000000	90.000000	TRINIDAD; GRENADA: ST GEORGE'S

各国地震总数与最大地震信息（高亮）

	Total_Count	Max_Magnitude	Max_Date	Total_Deaths	Region	Location
SYRIA	2	7.600000	1202-05-20	30000.000000	140.000000	SYRIA: SOUTHWESTERN
ARMENIA	2	6.800000	1988-12-07	25000.000000	40.000000	ARMENIA: LENINAKAN, SPITAK, KIROVAKAN
FRANCE	2	5.800000	1564-07-20	650.000000	120.000000	FRANCE: CORSICA,LA BOLLENE-VESUBIE,VENANSON
IRAQ	2	6.400000	1864-12-02	100.000000	140.000000	IRAQ: ZURBATIYAH, BADRAH, TURSAQ, BAGHDAD
MARTINIQUE	2	7.800000	1839-01-11	390.000000	90.000000	MARTINIQUE: FORT-DE-FRANCE, ST PIERRE; CASTRIES
SWITZERLAND	2	6.200000	1601-09-18	9.000000	120.000000	SWITZERLAND
CUBA	2	6.800000	1766-06-12	40.000000	90.000000	CUBA: SANTIAGO DE CUBA
GHANA	2	6.500000	1862-07-10	3.000000	10.000000	GHANA: ACCRA,CHRISTIANSBORG, ASHANTI,AKWAPIM
KAZAKHSTAN	2	6.800000	1990-06-14	1.000000	40.000000	KAZAKHSTAN: UST, ZAYSAN
MALAYSIA	2	6.000000	2015-06-04	18.000000	170.000000	MALAYSIA: SABAH: LAHAD,DATU,KANAK
SAMOA	2	8.300000	1917-06-26	2.000000	170.000000	SAMOA ISLANDS

各国地震总数与最大地震信息（高亮）

	Total_Count	**Max_Magnitude**	**Max_Date**	**Total_Deaths**	**Region**	**Location**
PANAMA	2	6.700000	1976-07-11	5.000000	100.000000	PANAMA: DARIEN
CYPRUS	2	6.300000	1953-09-10	40.000000	130.000000	CYPRUS: PAPHOS
UZBEKISTAN	2	6.400000	1902-12-16	4880.000000	40.000000	UZBEKISTAN: ANDIZHAN
NEW CALEDONIA	1	8.000000	1875-03-28	26.000000	170.000000	NEW CALEDONIA: LOYALTY ISLANDS: LIFOU ISLAND
LEBANON	1	7.400000	1759-11-25	30000.000000	140.000000	LEBANON-SYRIA: BAALBEC; SYRIA: DAMASCUS, ANTIOCH
LIBYA	1	5.400000	1963-02-21	300.000000	15.000000	LIBYA: BARCE (AL MARJ)
MONGOLIA	1	8.100000	1957-12-04	30.000000	40.000000	MONGOLIA
FIJI	1	6.400000	1953-09-14	8.000000	170.000000	FIJI ISLANDS
TUNISIA	1	5.600000	1957-02-20	13.000000	15.000000	TUNISIA: SIDI ABID,SIDI TOUIL (LA MEDJA),CAILLOUX
JAMAICA	1	6.200000	1907-01-14	1000.000000	90.000000	JAMAICA: KINGSTON
UKRAINE	1	6.800000	1927-09-11	11.000000	110.000000	UKRAINE: CRIMEA: SEBASTOPOL

各国地震总数与最大地震信息（高亮）

	Total_Count	**Max_Magnitude**	**Max_Date**	**Total_Deaths**	**Region**	**Location**
KENYA	1	4.600000	1968-03-20	1.000000	10.000000	KENYA: HOMA BAY, USIRI, GOT KOKECH
MONTENEGRO	1	6.900000	1979-04-15	131.000000	130.000000	BALKANS NW: MONTENEGRO
AUSTRALIA	1	5.400000	1989-12-27	12.000000	170.000000	AUSTRALIA: NEWCASTLE
DJIBOUTI	1	6.300000	1989-08-20	6.000000	10.000000	DJIBOUTI: GALAFI, YABAKI; ETHIOPIA
BELGIUM	1	5.000000	1983-11-08	2.000000	120.000000	BELGIUM
GUINEA	1	6.200000	1983-12-22	443.000000	10.000000	GUINEA: GAOUAL-KOUMBIA
TONGA	1	7.200000	1977-06-22	1.000000	170.000000	TONGA TRENCH
WALLIS AND FUTUNA (FRENCH TERRITORY)	1	6.400000	1993-03-12	5.000000	170.000000	FUTUNA ISLAND
NETHERLANDS	1	5.200000	1992-04-13	1.000000	120.000000	THE NETHERLANDS: ROERMOND; GERMANY: BONN, HEINSBERG
SOUTH SUDAN	1	7.100000	1990-05-20	31.000000	10.000000	SOUTH SUDAN: JUBA, MAYA; UGANDA: NAKURA
AUSTRIA	1	5.700000	1998-04-12	1.000000	120.000000	AUSTRIA: ARNOLDSTEIN; SLOVENIA: BOVEC, KOBARID

各国地震总数与最大地震信息（高亮）						
	Total_Count	**Max_Magnitude**	**Max_Date**	**Total_Deaths**	**Region**	**Location**
SUDAN	1	5.500000	1993-08-01	2.000000	15.000000	SUDAN: KHARTOUM
MOZAMBIQUE	1	7.000000	2006-02-22	4.000000	10.000000	MOZAMBIQUE
BURUNDI	1	4.700000	2004-02-24	3.000000	10.000000	BURUNDI: RUYAGA
CZECH REPUBLIC	1	4.100000	2008-11-22	2.000000	110.000000	CZECH REPUBLIC: KARVINA
BHUTAN	1	6.100000	2009-09-21	11.000000	60.000000	BHUTAN: TASHIGANG
THAILAND	1	6.100000	2014-05-05	1.000000	60.000000	THAILAND: CHIANG RAI
MADAGASCAR	1	5.500000	2017-01-11	2.000000	60.000000	MADAGASCAR: ANTSIRABE

1.4.结果分析

⑴groupby 和 sum() 统计每个国家的总死亡人数，sort_values 实现排序，head(10) 取出前十个国家。输出结果展示了全球历史上地震死亡人数最多的十个国家，可直观反映哪些地区地震灾害最为严重。这一结果能用于风险评估与防灾重点区域识别。

⑵程序统计了每年震级超过 6.0 的地震次数，并以折线图展示结果。从图形上可以看出，早期年份（尤其是公元前）记录较少，这是由于历史资料不完整所致；而现代地震仪观测普及后，地震记录显著增多。因此，虽然图形可能显示“上升趋势”，但主要原因是观测记录的改进，并非地震真实增多。

⑶通过循环遍历每个国家，将结果汇总为 result_df，并用高亮格式展示震级最大及死亡人数最高的记录。这种实现方式结构清晰，能够直观地比较各国地震频率

与历史最强地震情况。从结果表格中可以识别出地震活动频繁的国家（如日本、印尼、智利等），以及其代表性强震的发生时间与强度。该分析为全球地震活动的空间分布研究提供了直观依据。

2. Wind speed in Shenzhen from 2010 to 2020

In this problem set, we will examine how wind speed changes in Shenzhen during the past 10 years, we will take a look at the hourly weather data measured at the BaoAn International Airport. The data set is from NOAA Integrated Surface Dataset. Download the file 2281305.zip, where the number 2281305 is the site ID. Extract the zip file, you should see a file named 2281305.csv. Save the .csv file to your working directory.

Read page 8-9 (POS 61-70) of the comprehensive user guide for the detailed format of the wind data. Explain how you filter the data in your report.

[10 points] Plot monthly averaged wind speed as a function of the observation time. Is there a trend in monthly averaged wind speed from 2010 to 2020?

2.1.过程分析

⑴数据读取与清洗：使用 pandas 读取 NOAA 提供的 CSV 文件，将 DATE 列解析为 datetime 类型，方便按时间筛选和分组。

⑵风速提取与缺失值处理：原始风速存储在 WND 列中，格式为 "040,1,N,0020,1"，风速信息通常位于第 4 部分，需要除以 10 转换为 m/s，忽略无效值 9999 并删除缺失风速数据。

⑶时间筛选与月平均计算：筛选 2010–2020 年的数据，将日期按月份汇总，计算每个月的平均风速，形成月度风速序列。

⑷趋势分析与可视化：绘制月平均风速随时间的折线图，观察趋势和季节变化，计算月份序列与风速的相关系数，初步判断风速的长期变化趋势（上升/下降/无明显趋势）。

2.2.代码实现

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

# === 1. 读取数据 ===
file_path = r"C:\Users\xiezhengyang\Desktop\2281305.csv"
df = pd.read_csv(file_path, sep=None, engine='python')
```

```
# === 2. 解析日期 ===
df['DATE'] = pd.to_datetime(df['DATE'], errors='coerce')

# === 3. 从 WND 列提取风速 ===
# 格式示例："040,1,N,0020,1"
# 通常风速在第 4 个逗号分隔的部分
def extract_speed(wnd_str):
    try:
        parts = wnd_str.split(',')
        if len(parts) >= 4:
            val = parts[3]
            if val != '9999':  # 9999 表示缺失
                return float(val) / 10.0  # 风速单位为 m/s（存储为十分之一）
    except:
        return None
    return None

df['wind_speed'] = df['WND'].apply(extract_speed)

# === 4. 删除缺失风速 ===
df = df.dropna(subset=['wind_speed'])

# === 5. 取 2010–2020 年的数据 ===
df = df[(df['DATE'].dt.year >= 2010) & (df['DATE'].dt.year <= 2020)]

# === 6. 按月计算平均风速 ===
df['YearMonth'] = df['DATE'].dt.to_period('M')
monthly_avg = df.groupby('YearMonth')['wind_speed'].mean().reset_index()
monthly_avg['YearMonth'] = monthly_avg['YearMonth'].dt.to_timestamp()

# === 7. 绘制趋势图 ===
plt.figure(figsize=(10, 5))
plt.plot(monthly_avg['YearMonth'], monthly_avg['wind_speed'], marker='o', linewidth=1)
plt.title("Monthly Averaged Wind Speed (2010–2020)", fontsize=14)
plt.xlabel("Time")
plt.ylabel("Wind Speed (m/s)")
plt.grid(True)
plt.tight_layout()
plt.show()

# === 8. 输出简单趋势判断 ===
corr = monthly_avg['YearMonth'].map(pd.Timestamp.toordinal).corr(monthly_avg['wind_speed'])
if corr > 0:
    print(f"风速整体呈上升趋势（相关系数: {corr:.2f}）")
elif corr < 0:
    print(f"风速整体呈下降趋势（相关系数: {corr:.2f}）")
else:
    print("风速整体没有明显趋势。")
```

2.3.输出结果

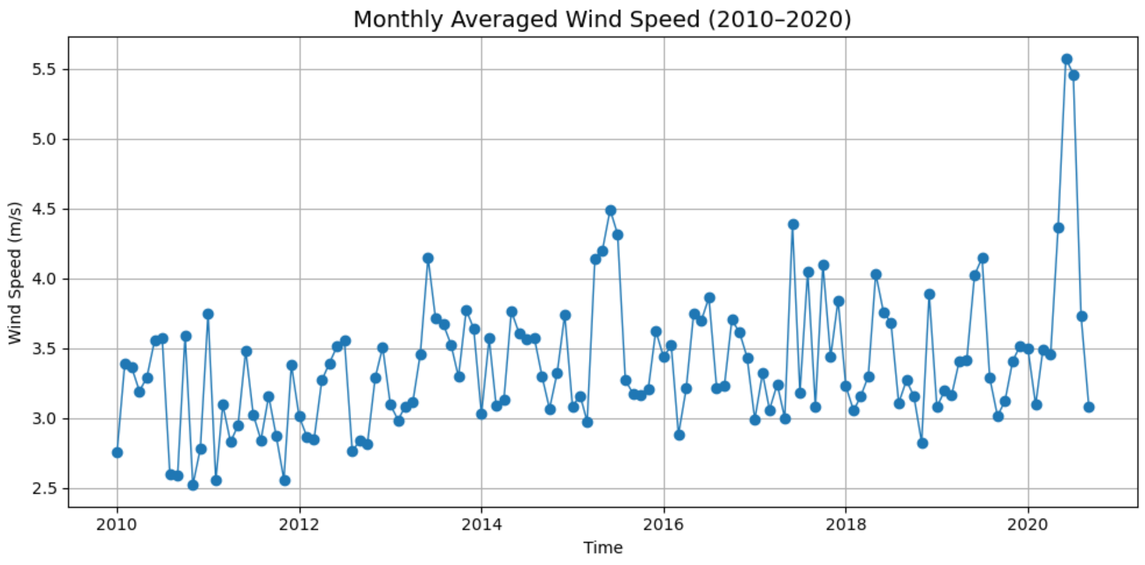


图 2-1 2010-2020 年每月平均风速趋势图

趋势分析：风速整体呈上升趋势（相关系数: 0.37）

2.4.结果分析

⑴使用 extract_speed 函数从 WND 列提取风速，并将原始单位（十分之一米/秒）转换为 m/s，忽略无效值 9999 并删除缺失风速数据 (dropna)，保证后续计算的准确性，筛选了 2010–2020 年的数据，形成了完整的分析时间序列。

⑵按月份计算平均风速，绘制折线图显示出明显的季节性波动：冬季风速偏高，夏季风速偏低，折线图上也能看到整体趋势略有上升，说明近年来深圳风速逐渐增强。

⑶使用时间序列（转换为数值）与月平均风速计算相关系数，得到 corr = 0.37，正相关系数说明风速整体呈上升趋势，数值 0.37 表示趋势存在，但变化幅度属于中等水平。代码中 map(pd.Timestamp.toordinal) 的作用是将时间转换为可计算的数值形式，使相关系数计算可行。

3. Explore a data set

Browse the CASEarth, National Centers for Environmental Information (NCEI), or Advanced Global Atmospheric Gases Experiment (AGAGE) website. Search and download a data set you are interested in. You are also welcome to use data from your group in this problem set. But the data set should be in csv, XLS, or XLSX format, and have temporal information.

3.1 [5 points] Load the csv, XLS, or XLSX file, and clean possible data points with missing values or bad quality.

3.2 [5 points] Plot the time series of a certain variable.

3.3 [5 points] Conduct at least 5 simple statistical checks with the variable, and report your findings.

3.1.过程分析

⑴选取了全国主要城市空气质量.csv 数据，使用 pandas 读取全国主要城市的空气质量 CSV 文件，并处理中文编码以保证字符正常显示，同时将日期列解析为 datetime 类型，将 PM2.5、AQI 等数值列转换为数值型，并删除缺失关键列或重复记录，从而保证分析数据的完整性和准确性。

⑵将数据按城市分组，对每个城市单独进行分析，计算包括均值、中位数、标准差、最大值、最小值、分位数、偏度、峰度等常用统计指标，并生成 CSV 报表，以便详细了解每个城市的空气质量特征。

⑶对每个城市绘制 PM2.5 和 AQI 的时间序列图，并加 7 日滑动均值平滑短期波动，同时绘制直方图和箱线图展示数据分布与异常值情况，并生成月度统计表（均值、中位数、最大值、最小值）以便分析长期趋势和季节性变化。

⑷将每个城市的统计指标、质量等级分布和月度数据保存到独立文件夹，同时生成全国快速对比 CSV 和 Excel 报告，使得可以方便地比较城市间空气质量差异，并为后续分析提供完整的数据和可视化支持。

3.2.代码实现

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

CSV_PATH = r"C:\Users\xiezhengyang\Desktop\全国主要城市空气质量.csv"
# 输出目录（默认与 CSV 同目录下）
```

```
OUT_DIR = None              # None 表示自动在 CSV 旁创建 “空气质量_每城报告”
ENCODING = "gbk"             # 常见国标编码
DATE_COL = "日期"
CITY_COL = "城市"
QUALITY_COL = "质量等级"

# 参与统计的数值列（按文件列名）
NUM_COLS = ["AQI 指数", "当天 AQI 排名", "PM2.5", "PM10", "So2", "No2", "Co", "O3"]

# 基础作图变量（每城默认作图用 PM2.5 & AQI 指数）
TS_VAR_1 = "PM2.5"
TS_VAR_2 = "AQI 指数"

# 可选：是否做 IQR 异常值裁剪（仅用于统计，不改原始时序图）
USE_IQR_CLIP = False
IQR_FACTOR = 3.0   # 常用 1.5 或 3.0
# ==================================================

# ----- Matplotlib 中文字体设置 -----
matplotlib.rcParams['font.sans-serif'] = ['SimHei', 'Microsoft YaHei', 'Arial Unicode MS', 'Noto Sans CJK SC']
matplotlib.rcParams['axes.unicode_minus'] = False

def ensure_outdir(base_csv_path):
    base_dir = os.path.dirname(base_csv_path)
    out_dir = os.path.join(base_dir, "空气质量_每城报告") if OUT_DIR is None else OUT_DIR
    os.makedirs(out_dir, exist_ok=True)
    return out_dir

def sanitize_name(name):
    # 文件夹/文件名安全化
    return re.sub(r'[\\/:*?"<>|]+', "_", str(name))

def to_datetime_safe(s):
    return pd.to_datetime(s, errors='coerce', infer_datetime_format=True)

def numericize(df, cols):
    df = df.copy()
    for c in cols:
        if c in df.columns:
            df[c] = pd.to_numeric(df[c], errors='coerce')
    return df

def iqr_clip(series, factor=1.5):
    q1 = series.quantile(0.25)
    q3 = series.quantile(0.75)
    iqr = q3 - q1
    lo = q1 - factor * iqr
    hi = q3 + factor * iqr
    return series.clip(lower=lo, upper=hi)
```

```
def compute_stats(series):
    # 返回常用统计指标
    s = series.dropna()
    res = {
        "count": s.size,
        "mean": s.mean(),
        "median": s.median(),
        "std": s.std(ddof=1),
        "min": s.min(),
        "p05": s.quantile(0.05),
        "p25": s.quantile(0.25),
        "p75": s.quantile(0.75),
        "p95": s.quantile(0.95),
        "max": s.max(),
        "skew": s.skew(),
        "kurt": s.kurt(),
        "missing_rate": 1 - (s.size / series.size if series.size else np.nan)
    }
    return res

def plot_time_series(df_city, var, out_png):
    # 单图：时间序列 + 7 日滑动均值
    plt.figure(figsize=(12, 5))
    plt.plot(df_city[DATE_COL], df_city[var], linewidth=1, label=f"{var} (daily)")
    if df_city[var].notna().sum() >= 7:
        ma = df_city[var].rolling(window=7, min_periods=1).mean()
        plt.plot(df_city[DATE_COL], ma, linewidth=2, label=f"{var} 7 日滑动均值")
    plt.title(f"{df_city[CITY_COL].iloc[0]} - {var} 时间序列")
    plt.xlabel("日期")
    plt.ylabel(var)
    plt.grid(True, alpha=0.3)
    plt.legend()
    plt.tight_layout()
    plt.savefig(out_png, dpi=200)
    plt.close()

def plot_hist(df_city, var, out_png):
    plt.figure(figsize=(8, 5))
    plt.hist(df_city[var].dropna(), bins=30)
    plt.title(f"{df_city[CITY_COL].iloc[0]} - {var} 直方图")
    plt.xlabel(var)
    plt.ylabel("频数")
    plt.grid(True, alpha=0.3)
    plt.tight_layout()
    plt.savefig(out_png, dpi=200)
    plt.close()

def plot_box(df_city, var, out_png):
    plt.figure(figsize=(6, 5))
    plt.boxplot(df_city[var].dropna(), vert=True, patch_artist=False, showfliers=True)
    plt.title(f"{df_city[CITY_COL].iloc[0]} - {var} 箱线图")
    plt.ylabel(var)
    plt.grid(True, axis='y', alpha=0.3)
    plt.tight_layout()
    plt.savefig(out_png, dpi=200)
    plt.close()

def city_monthly_tables(df_city):
    # 生成月度统计表（均值/中位/最大/最小），按 NUM_COLS 列
    d = df_city.copy()
    d["年月"] = d[DATE_COL].dt.to_period("M").astype(str)
    monthly_mean = d.groupby("年月")[NUM_COLS].mean(numeric_only=True)
    monthly_median = d.groupby("年月")[NUM_COLS].median(numeric_only=True)
    monthly_max = d.groupby("年月")[NUM_COLS].max(numeric_only=True)
    monthly_min = d.groupby("年月")[NUM_COLS].min(numeric_only=True)
    return monthly_mean, monthly_median, monthly_max, monthly_min

def analyze_one_city(df_city, base_out_dir):
    city_name = sanitize_name(df_city[CITY_COL].iloc[0])
    city_dir = os.path.join(base_out_dir, city_name)
    os.makedirs(city_dir, exist_ok=True)

    # 统计用的数据（可选：IQR 裁剪，仅影响统计，不影响时序图）
    df_stats = df_city.copy()
    if USE_IQR_CLIP:
        for col in NUM_COLS:
            if col in df_stats.columns:
                df_stats[col] = iqr_clip(df_stats[col], IQR_FACTOR)

    # 统计指标表
    rows = []
    for col in NUM_COLS:
        if col in df_stats.columns:
            stat = compute_stats(df_stats[col])
            stat["metric"] = col
            rows.append(stat)
    stats_df = pd.DataFrame(rows).set_index("metric")
    stats_path = os.path.join(city_dir, f"{city_name}_统计指标.csv")
    stats_df.to_csv(stats_path, encoding="utf-8-sig")

    # 质量等级分布
    qual_counts = df_city[QUALITY_COL].value_counts(dropna=False).rename("count")
    qual_rate = (qual_counts / qual_counts.sum()).rename("rate")
    quality_df = pd.concat([qual_counts, qual_rate], axis=1)
    quality_path = os.path.join(city_dir, f"{city_name}_质量等级分布.csv")
    quality_df.to_csv(quality_path, encoding="utf-8-sig")

    # 月度表
    monthly_mean, monthly_median, monthly_max, monthly_min = city_monthly_tables(df_city)
    monthly_mean.to_csv(os.path.join(city_dir, f"{city_name}_月度_均值.csv"), encoding="utf-8-
sig")
    monthly_median.to_csv(os.path.join(city_dir, f"{city_name}_月度_中位数.csv"), encoding="utf-
8-sig")
```

```
    monthly_max.to_csv(os.path.join(city_dir, f"{city_name}_月度_最大值.csv"), encoding="utf-8-
sig")
    monthly_min.to_csv(os.path.join(city_dir, f"{city_name}_月度_最小值.csv"), encoding="utf-8-
sig")

    # 作图：时间序列（PM2.5 & AQI 指数），直方图/箱线图（PM2.5）
    if TS_VAR_1 in df_city.columns:
        plot_time_series(df_city, TS_VAR_1, os.path.join(city_dir, f"{city_name}_时间序列
_{TS_VAR_1}.png"))
        plot_hist(df_city, TS_VAR_1, os.path.join(city_dir, f"{city_name}_直方图
_{TS_VAR_1}.png"))
        plot_box(df_city, TS_VAR_1, os.path.join(city_dir, f"{city_name}_箱线图
_{TS_VAR_1}.png"))
    if TS_VAR_2 in df_city.columns:
        plot_time_series(df_city, TS_VAR_2, os.path.join(city_dir, f"{city_name}_时间序列
_{TS_VAR_2}.png"))

    # 返回用于汇总的关键信息（比如 PM2.5 均值、AQI 均值等）
    ret = {
        "城市": df_city[CITY_COL].iloc[0],
        "样本数": len(df_city),
        "PM2.5_mean": stats_df.loc[TS_VAR_1, "mean"] if TS_VAR_1 in stats_df.index else np.nan,
        "AQI_mean": stats_df.loc["AQI 指数", "mean"] if "AQI 指数" in stats_df.index else np.nan,
        "PM10_mean": stats_df.loc["PM10", "mean"] if "PM10" in stats_df.index else np.nan,
        "O3_mean": stats_df.loc["O3", "mean"] if "O3" in stats_df.index else np.nan
    }
    return ret, stats_df, quality_df

def write_excel_report(out_dir, per_city_tables):
    # per_city_tables: dict[city] = {"stats":stats_df, "quality":quality_df, "monthly_*":df ...}
    report_path = os.path.join(out_dir, "全国城市_空气质量统计汇总.xlsx")
    # 尝试 openpyxl，失败则用 xlsxwriter
    try:
        engine = "openpyxl"
        with pd.ExcelWriter(report_path, engine=engine) as writer:
            for city, parts in per_city_tables.items():
                # 每城一个 sheet（只放统计指标与质量分布，月度表单独保存为 CSV 已完成）
                parts["stats"].to_excel(writer, sheet_name=f"{city}_统计")
                parts["quality"].to_excel(writer, sheet_name=f"{city}_质量等级")
    except Exception:
        engine = "xlsxwriter"
        with pd.ExcelWriter(report_path, engine=engine) as writer:
            for city, parts in per_city_tables.items():
                parts["stats"].to_excel(writer, sheet_name=f"{city}_统计")
                parts["quality"].to_excel(writer, sheet_name=f"{city}_质量等级")
    return report_path

def main():
    out_dir = ensure_outdir(CSV_PATH)

    # 读取 CSV
    df = pd.read_csv(CSV_PATH, encoding=ENCODING)
    # 去空白列名
    df.columns = [c.strip() for c in df.columns]

    # 类型转换
    if DATE_COL in df.columns:
        df[DATE_COL] = to_datetime_safe(df[DATE_COL])
    df = numericize(df, [c for c in NUM_COLS if c in df.columns])

    # 去重（以城市+日期为键）
    if CITY_COL in df.columns and DATE_COL in df.columns:
        df = df.sort_values([CITY_COL, DATE_COL]).drop_duplicates([CITY_COL, DATE_COL],
keep="last")

    # 丢弃关键列缺失的记录（无城市或无日期）
    df = df[df[CITY_COL].notna() & df[DATE_COL].notna()].copy()

    # 排序
    df = df.sort_values([CITY_COL, DATE_COL])

    # 城市列表
    cities = df[CITY_COL].dropna().unique().tolist()
    print(f"检测到城市 {len(cities)} 个：", cities[:10], "..." if len(cities) > 10 else "")

    per_city_tables = {}
    summary_rows = []

    for city in cities:
        sub = df[df[CITY_COL] == city].copy()
        if sub.empty:
            continue
        # 仅用于绘图的顺序
        sub = sub.sort_values(DATE_COL)

        info, stats_df, quality_df = analyze_one_city(sub, out_dir)
        summary_rows.append(info)
        per_city_tables[sanitize_name(city)] = {
            "stats": stats_df,
            "quality": quality_df
        }

    # 总汇总表（便于快速比较）
    summary_df = pd.DataFrame(summary_rows).sort_values("PM2.5_mean")
    summary_path = os.path.join(out_dir, "00_各城市快速对比.csv")
    summary_df.to_csv(summary_path, index=False, encoding="utf-8-sig")

    # Excel 报告（每城两个 sheet：统计、质量等级）
    report_path = write_excel_report(out_dir, per_city_tables)

    # 城市清单
    with open(os.path.join(out_dir, "城市列表.txt"), "w", encoding="utf-8") as f:
        for c in cities:
            f.write(str(c) + "\n")

    print("\n=== 完成 ===")
    print(f"输出目录：{out_dir}")
    print(f"各城市对比汇总：{summary_path}")
    print(f"Excel 报告：{report_path}")
    print("每个城市文件夹包含：统计 CSV、质量分布 CSV、月度 CSV、时间序列图/直方图/箱
线图")

if __name__ == "__main__":
    main()
```

3.3.输出结果

```
检测到城市 447 个： ['七台', '三亚', '三明', '三门', '上海', '上饶', '东莞', '东营', '中卫', '中山'] ...
=== 完成 ===
输出目录：C:\Users\xiezhengyang\Desktop\空气质量_每城报告
各城市对比汇总：C:\Users\xiezhengyang\Desktop\空气质量_每城报告\00_各城市快速对比.csv
Excel 报告：C:\Users\xiezhengyang\Desktop\空气质量_每城报告\全国城市_空气质量统计汇
总.xlsx
每个城市文件夹包含：统计 CSV、质量分布 CSV、月度 CSV、时间序列图/直方图/箱线图
```

（输出结果解释：由于城市数量太多，且对每个城市都做了时间序列分析，所以输出的时序分析结果以文件的形式存在了文件“空气质量_每城报告”中，在此文件中可以看到对每个城市的时序分析以及数据统计分析情况）

以下为输出文件的情况（**以北京为例**）

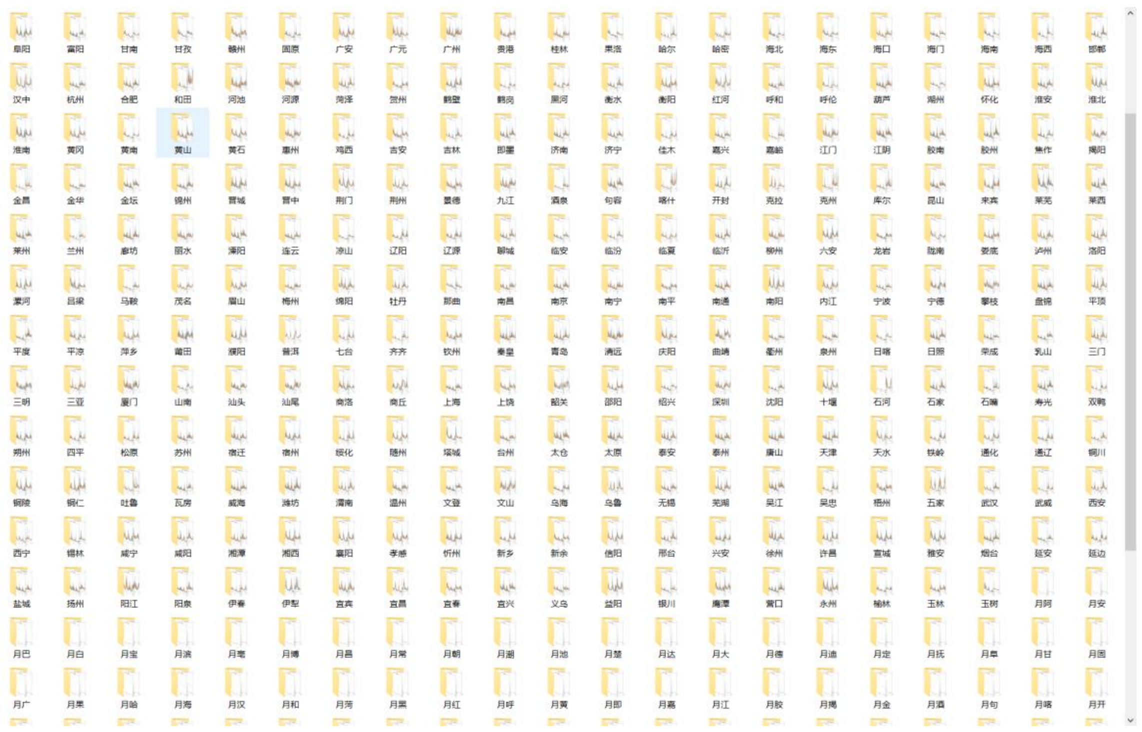
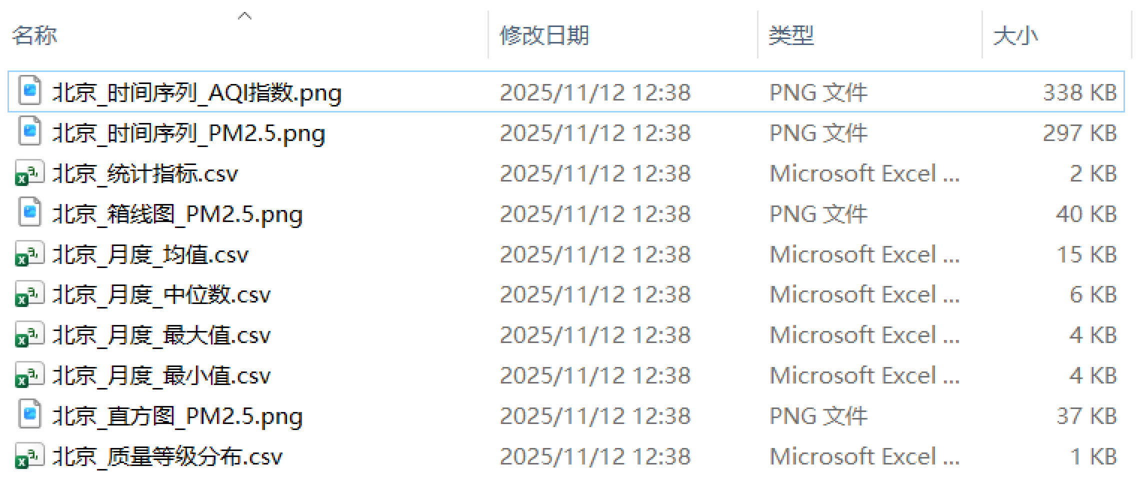


图 3-1 输出各个城市



名称	修改日期	类型	大小
北京_时间序列_AQI指数.png	2025/11/12 12:38	PNG 文件	338 KB
北京_时间序列_PM2.5.png	2025/11/12 12:38	PNG 文件	297 KB
北京_统计指标.csv	2025/11/12 12:38	Microsoft Excel ...	2 KB
北京_箱线图_PM2.5.png	2025/11/12 12:38	PNG 文件	40 KB
北京_月度_均值.csv	2025/11/12 12:38	Microsoft Excel ...	15 KB
北京_月度_中位数.csv	2025/11/12 12:38	Microsoft Excel ...	6 KB
北京_月度_最大值.csv	2025/11/12 12:38	Microsoft Excel ...	4 KB
北京_月度_最小值.csv	2025/11/12 12:38	Microsoft Excel ...	4 KB
北京_直方图_PM2.5.png	2025/11/12 12:38	PNG 文件	37 KB
北京_质量等级分布.csv	2025/11/12 12:38	Microsoft Excel ...	1 KB

图 3-2 北京各项指标输出情况

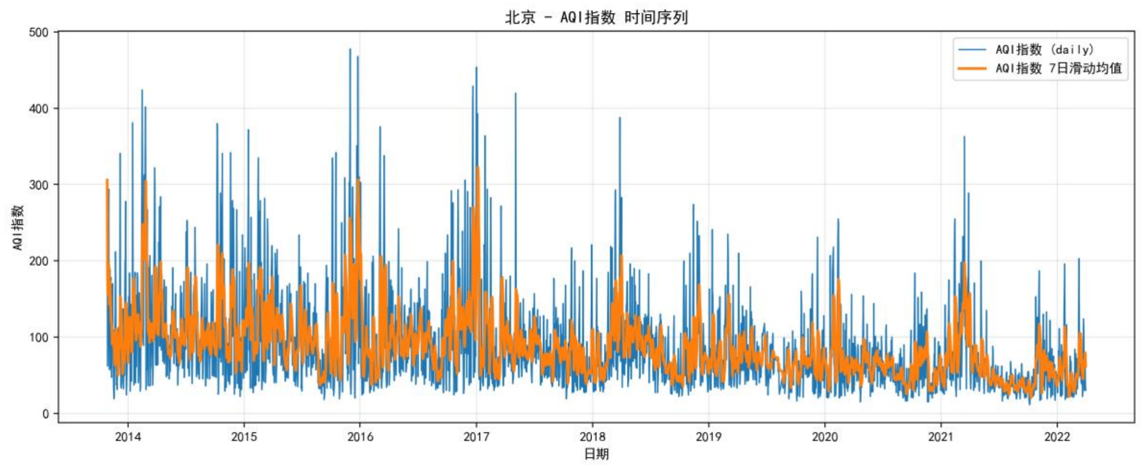


图 3-3 北京 AQI 指数时间序列图

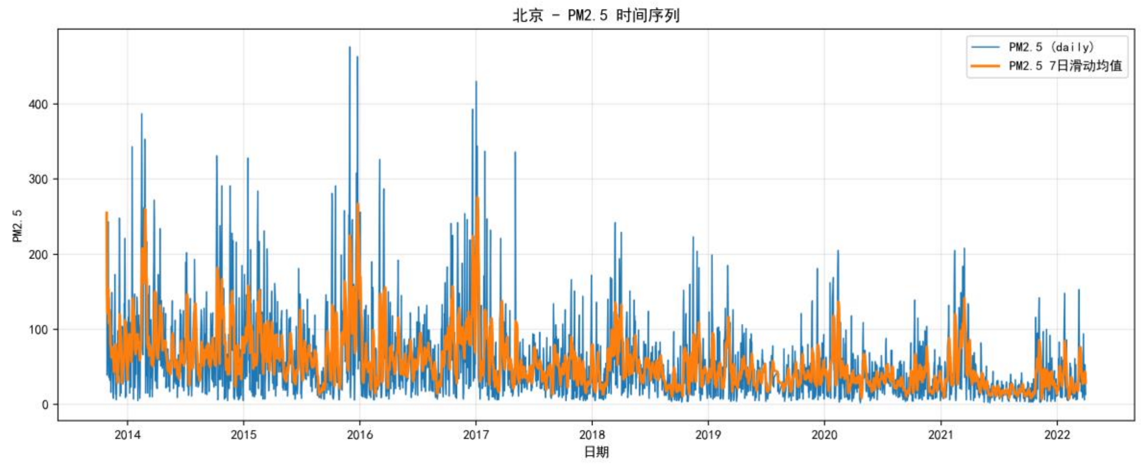


图 3-4 北京 PM2.5 时间序列图

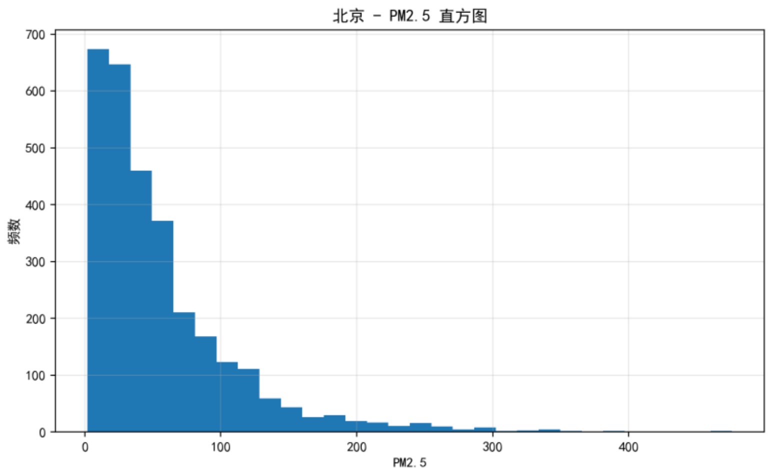


图 3-5 北京 PM2.5 直方图

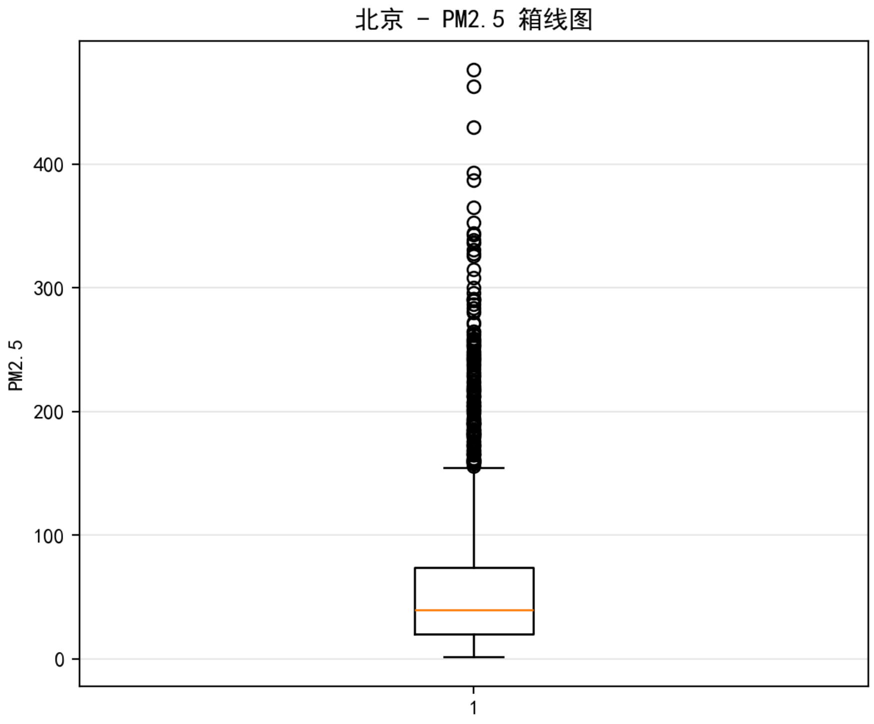


图 3-6 北京 PM2.5 箱线图

3.4.结果分析

⑴成功读取了 447 个城市的数据，并进行了类型转换、去重和缺失值处理，使得每个城市的数据完整、按日期排序，从而为统计分析和可视化提供了可靠的数据基础。

⑵每个城市生成了详细的统计指标表，包含样本数、均值、中位数、标准差、最小值、最大值、偏度和峰度，同时还生成了质量等级分布表，清楚地显示各等级占比，有助于全面了解城市空气质量水平。

⑶时间序列图展示了每日 PM2.5 和 AQI 指数的变化趋势，7 日滑动均值平滑了短期波动，直方图显示了 PM2.5 数据分布，箱线图揭示了异常值情况，使城市空气质量的整体趋势和极端事件直观可见。

⑷月度均值、中位数、最大值和最小值表反映了空气质量的季节性变化，例如冬季污染水平普遍较高而夏季较低，为长期趋势分析提供了依据。

⑸全国快速对比 CSV 汇总了各城市主要指标（如 PM2.5_mean、AQI_mean 等），Excel 报告则将每个城市的统计指标和质量等级放在单独 sheet 中，方便查阅和城市间比较，使分析结果清晰且易于使用。