

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
file_path = "D:/桌面/ESE5023_Assignments_12431305/PS2/earthquakes.tsv"
Sig_Eqs = pd.read_csv(file_path, sep="\t")
Sig_Eqs.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6616 entries, 0 to 6615
Data columns (total 39 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Search Parameters                     1 non-null     object
1   Year                                 6615 non-null  float64
2   Mo                                  6202 non-null  float64
3   Dy                                  6045 non-null  float64
4   Hr                                  4502 non-null  float64
5   Mn                                  4289 non-null  float64
6   Sec                                  3831 non-null  float64
7   Tsu                                  2083 non-null  float64
8   Vol                                  81 non-null    float64
9   Location Name                        6615 non-null  object
10  Latitude                             6554 non-null  float64
11  Longitude                             6554 non-null  float64
12  Focal Depth (km)                     3574 non-null  float64
13  Mag                                  4823 non-null  float64
14  MMI Int                              3343 non-null  float64
15  Deaths                              2244 non-null  float64
16  Death Description                    2748 non-null  float64
17  Missing                              24 non-null    float64
18  Missing Description                  26 non-null    float64
19  Injuries                             1451 non-null  float64
20  Injuries Description                1699 non-null  float64
21  Damage ($Mil)                       631 non-null   float64
22  Damage Description                  4833 non-null  float64
23  Houses Destroyed                    923 non-null   float64
24  Houses Destroyed Description        1990 non-null  float64
25  Houses Damaged                      610 non-null   float64
26  Houses Damaged Description          1187 non-null  float64
27  Total Deaths                        2112 non-null  float64
28  Total Death Description              2526 non-null  float64
29  Total Missing                       27 non-null    float64
30  Total Missing Description            32 non-null    float64
31  Total Injuries                      1476 non-null  float64
32  Total Injuries Description           1730 non-null  float64
33  Total Damage ($Mil)                 620 non-null   float64
34  Total Damage Description             4050 non-null  float64
35  Total Houses Destroyed               957 non-null   float64
36  Total Houses Destroyed Description   2075 non-null  float64
37  Total Houses Damaged                 558 non-null   float64
38  Total Houses Damaged Description     1094 non-null  float64
dtypes: float64(37), object(2)
memory usage: 2.0+ MB
```

```
In [41]: # 1.1

# 从 Location Name 中抽取“国家/地区”——
# 规则：有冒号取冒号左侧；无冒号取整串；缺失为 NA；统一去空格
Sig_Eqs["Country"] = (
    Sig_Eqs["Location Name"].astype("string")
    .str.extract(r"^\s*([:: ]+)", expand=False) # 提取开头到第一个冒号前的内容——国家
    .str.strip().replace("", pd.NA).str.upper())

# “死亡人数”的选取（“Total Deaths”还是“Deaths”）
# 优先用 'Total Deaths'，没有就'Deaths'
Deaths_real = "Total Deaths" if "Total Deaths" in Sig_Eqs.columns else "Deaths"

# 把死亡人数列表转为数值（无法解析的记为 NaN）
Sig_Eqs[Deaths_real] = pd.to_numeric(Sig_Eqs[Deaths_real], errors="coerce")

# 按照国家名称分组求和
deaths_by_country = (
    Sig_Eqs.groupby("Country", dropna=True)[Deaths_real].sum(min_count=1).fillna(0).sort_values(ascending=False))

# 只取前 10 名（用 nlargest 少一次全量排序也可）
top10 = deaths_by_country.head(10).astype(int)

print("\n[1.1] 各国地震总死亡人数（前 10）：")
print(top10)
```

```
[1.1] 各国地震总死亡人数（前 10）：
Country
CHINA      2106535
TURKEY     1008342
IRAN       745611
ITALY      423280
SYRIA      417724
HAITI      323782
AZERBAIJAN 319251
JAPAN      318441
INDONESIA   282517
ARMENIA    189000
Name: Total Deaths, dtype: int64
```

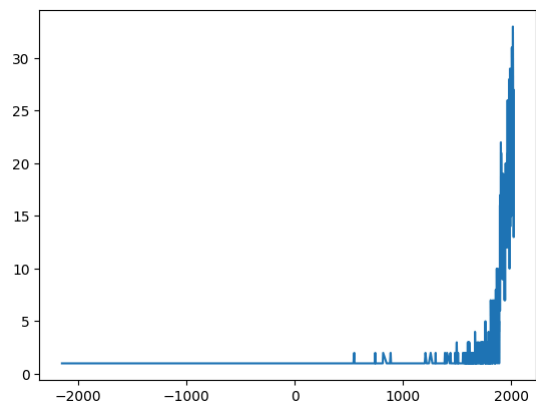
```
In [42]: # 1.2

# 确保用到的列为数值
Sig_Eqs["Mag"] = pd.to_numeric(Sig_Eqs["Mag"], errors="coerce")
Sig_Eqs["Year"] = pd.to_numeric(Sig_Eqs["Year"], errors="coerce")

# 震级 > 6.0
mag_6 = Sig_Eqs.loc[Sig_Eqs["Year"].notna() & Sig_Eqs["Mag"].notna() & (Sig_Eqs["Mag"] > 6.0)].copy()

# 按“年份”计数，并按年份升序排列
mag_6["Year_int"] = mag_6["Year"].astype(int)
mag_6_per_year = mag_6.groupby("Year_int").size().sort_index() # 每年条数 = size()

#画折线图
plt.plot(mag_6_per_year.index, mag_6_per_year.values)
plt.show()
print("趋势和解释：从图看来，近几百年的6级以上地震“显著变多”，而千年之前几乎没有，但这更可能是几百年前数据缺失或者无法记录所致")
# 趋势和解释：从图看来，近几百年的6级以上地震“显著变多”，而千年之前几乎没有，但这更可能是几百年前数据缺失或者无法记录所致
```



趋势和解释：从图看来，近几百年的6级以上地震“显著变多”，而千年之前几乎没有，但这更可能是几百年前数据缺失或者无法记录所致

```
In [44]: #1.3

# 把 Year/Mo/Dy 组合成“尽量精确”的日期：无月日则回退到年份
def _best_date_string(y, m, d):
    if pd.isna(y):
        return "N/A (无年份)"
    y = int(y)
    if y <= 0: # BCE
        if pd.notna(m) and pd.notna(d):
            m, d = int(m), int(d)
            if 1 <= m <= 12 and 1 <= d <= 31:
                return f"{abs(y):04d}-{m:02d}-{d:02d} BCE"
            return f"{abs(y)} BCE"
        # CE
        if pd.notna(m) and pd.notna(d):
            m, d = int(m), int(d)
            ts = pd.to_datetime({"year": [y], "month": [m], "day": [d]}, errors="coerce")
            if ts.notna().iloc[0]:
                return ts.dt.strftime("%Y-%m-%d").iloc[0]
            return str(y)

# 返回（该国总地震次数，史上最大震级地震的日期）
def CountEq_LargestEq(country_name, data):
    sub = data.loc[data["Country"] == country_name]
    total_count = len(sub)
    if total_count == 0:
        return 0, "N/A (无记录)"
    sub_valid = sub.dropna(subset=["Mag"])
    if sub_valid.empty:
        return total_count, "N/A (无震级数据)"
    row = sub_valid.loc[sub_valid["Mag"].idxmax()]
    return total_count, _best_date_string(row.get("Year"), row.get("Mo"), row.get("Dy"))

# 应用到每个国家
rows = []
for c in sorted(Sig_Eqs["Country"].dropna().unique()):
    total, date_str = CountEq_LargestEq(c, Sig_Eqs)
    rows.append({"Country": c, "TotalEarthquakes": total, "LargestEarthquakeDate": date_str})

results_sorted = (
    pd.DataFrame(rows)
    .sort_values(["TotalEarthquakes", "Country"], ascending=[False, True])
    .reset_index(drop=True)
)

print("所有国家地震总数及最大地震日期（按总数降序排列）:")
print(results_sorted.to_string(index=False))
```

所有国家地震总数及最大地震日期 (按总数降序排列):

Country	TotalEarthquakes	LargestEarthquakeDate
CHINA	626	1668
INDONESIA	404	2004-12-26
IRAN	388	856
JAPAN	364	2011-03-11
TURKEY	344	1939-12-26
ITALY	332	1915-01-13
GREECE	275	365
PHILIPPINES	226	1897-09-21
MEXICO	204	1787-03-28
CHILE	200	1960-05-22
PERU	185	1716-02-06
RUSSIA	157	1952-11-04
BALKANS NW	118	1667
PAPUA NEW GUINEA	109	1919-05-06
CALIFORNIA	104	1857-01-09
INDIA	101	1897-06-12
TAIWAN	98	1920-06-05
ALASKA	80	1964-03-28
COLOMBIA	75	1826-06-18
ECUADOR	67	1906-01-31
VENEZUELA	66	1530
NEW ZEALAND	65	1826
SOLOMON ISLANDS	62	1977-04-21
AFGHANISTAN	61	1909-07-07
VANUATU ISLANDS	59	1913-10-14
ALBANIA	57	1893-06-14
ALGERIA	57	1980-10-10
PAKISTAN	52	1945-11-27
FRANCE	44	1817-03-11
GUATEMALA	44	1942-08-06
ARGENTINA	39	1894-10-27
NICARAGUA	37	1898-04-29
EL SALVADOR	36	1776-05-30
SPAIN	34	1954-03-29
SWITZERLAND	31	1601
MYANMAR (BURMA)	30	1912-05-23
AZORES	28	1968-02-28
COSTA RICA	28	1822-05-07
NEW CALEDONIA	28	1875-03-28
SYRIA	28	1202
PORTUGAL	27	60 BCE
AUSTRALIA	25	2004-12-23
KERMADEC ISLANDS	25	1986-10-20
TAJIKISTAN	25	1907-10-21
ISRAEL	24	0031-09-02 BCE
IRAQ	22	1864-12-02
HAITI	21	1842-05-07
MOROCCO	21	2023-09-08
SOUTH KOREA	20	1700-09-12
JAMAICA	19	1899-06-14
DOMINICAN REPUBLIC	18	1946-08-04
FIJI ISLANDS	18	1919-01-01
TONGA ISLANDS	18	1919-04-30
AZERBAIJAN	17	1667
BULGARIA	17	1904-04-04
ICELAND	17	1912-05-06
BANGLADESH	16	1918-07-08
BOLIVIA	16	1957-11-29
CANADA	16	1949-08-22
EGYPT	16	1995-11-22
PANAMA	16	1882-09-07
CUBA	15	2020-01-28
GEORGIA	15	1905-10-21
KYRGYZSTAN	15	1946-11-02
BRAZIL	14	1963-11-09
ETHIOPIA	14	1906-08-25
PUERTO RICO	14	1943-07-29
ROMANIA	14	1977-03-04
UNITED KINGDOM	14	1580
UZBEKISTAN	14	1976-04-08
NEPAL	13	2015-04-25
SANRIKU, JAPAN	13	1257
SOUTH AFRICA	13	1942-11-10
GUAM	12	1902-09-22
LEBANON	12	551
UKRAINE	12	103
ARMENIA	11	1988-12-07
HAWAII	11	1868-04-03
HONDURAS	11	1856-08-04
TURKMENISTAN	11	1895-07-08
KAZAKHSTAN	10	1889-07-11
YEMEN	10	1982-12-13
GERMANY	9	1978-09-03
GREECE-ALBANIA	9	1911-02-18
MARTINIQUE	9	1906-12-03
TUNISIA	9	1957-02-20
AUSTRIA	8	1590
CYPRUS	8	1953-09-10
GUADELOUPE	8	1843-02-08
POLAND	8	2004-09-21
TANZANIA	8	1910-12-13
UTAH	8	1934-03-12
CONGO	7	1992-09-11
COSTA RICA-PANAMA	7	1941-12-05
MONGOLIA	7	1905-07-09
NEVADA	7	1915-10-03
OKLAHOMA	7	2011-11-06
SOUTH SANDWICH ISLANDS	7	1929-06-27
ALASKA PENINSULA	6	2021-07-29
MONTANA	6	1959-08-18
NEW YORK	6	1944-09-05
NORTH KOREA	6	1518
SAMOA ISLANDS	6	1917-06-26
TOKAIDO, JAPAN	6	1782-08-22
TONGA TRENCH	6	1982-12-19
VIETNAM	6	1935-11-01
BHUTAN	5	2009-09-21
GHANA	5	1862-07-10
HUNGARY	5	1834-10-15
JORDAN	5	2150 BCE
MICRONESIA, FED. STATES OF	5	1911-08-16
NEPAL-INDIA	5	1505
PANAMA-COSTA RICA	5	1934-07-18
RWANDA	5	2015-08-07
S. MEXICO	5	1925-11-16
SW. SUMATRA	5	1797-02-10
TRINIDAD	5	2006-09-29
VANUATU	5	1878-01-10

WASHINGTON	5	1949-04-13
ATLANTIC OCEAN	4	1941-11-25
COLOMBIA-ECUADOR	4	1958-01-19
HAWAIIAN ISLANDS	4	1871-02-20
INDIAN OCEAN	4	1928-03-09
MALAWI	4	1989-03-10
PERU-ECUADOR	4	1953-12-12
SEIKAI DO, JAPAN	4	1970-07-25
SOUTH SUDAN	4	1990-05-20
THAILAND	4	2014-05-05
TURKEY; ARMENIA	4	1679-06-04
UGANDA	4	1912-07-09
VIRGIN ISLANDS	4	1867-11-18
AFGHANISTAN; PAKISTAN	3	2015-12-25
EL SALVADOR-GUATEMALA	3	1902-02-26
KASHIMA, JAPAN	3	1927-08-18
MALAYSIA	3	1976-07-26
MISSOURI	3	1812-02-07
NORTHERN MARIANA ISLANDS	3	2016-07-29
S. JAVA SEA	3	1930-07-19
SAUDI ARABIA	3	2009-05-19
SCOTIA SEA	3	2013-11-17
SE. HOKKAIDO ISLAND, JAPAN	3	1839-05-01
SLOVAKIA	3	2004-01-10
TAIWAN REGION	3	1996-09-05
THE NETHERLANDS	3	1992-04-13
TURKEY; SYRIA	3	2023-02-06
AFGHANISTAN-TAJIKISTAN	2	1998-02-20
ARKANSAS	2	1811-12-16
BOLIVIA-NORTHERN CHILE	2	1916-08-25
BOSO, JAPAN	2	1642
BRITISH COLUMBIA	2	1946-06-23
CAMEROON	2	1945-09-12
CANARY ISLANDS	2	N/A (无震级数据)
CHILE-ARGENTINA	2	1950-12-09
COLORADO	2	2011-08-23
COTE D'IVOIRE	2	1879-02-11
EL SALVADOR; GUATEMALA	2	2001-01-13
ERITREA	2	1884-07-20
FIJI	2	N/A (无震级数据)
FLORES SEA	2	1836-11-28
GREECE-BULGARIA	2	N/A (无震级数据)
IDAHO	2	1983-10-28
INDIA-BANGLADESH	2	1997-11-21
INDIA-CHINA	2	1950-08-15
IRAN-IRAQ	2	1008
IWATE, JAPAN	2	1979-02-20
JAPAN TRENCH	2	1971-08-02
KENTUCKY	2	1980-07-27
KENYA	2	1928-01-06
LEBANON-SYRIA	2	1759-11-25
LIBYA	2	1963-02-21
MARMARA SEA	2	1332
MEXICO-GUATEMALA	2	1921-02-04
MOZAMBIQUE	2	2006-02-22
MYANMAR (BURMA); INDIA	2	1988-08-06
MYANMAR; INDIA	2	1954-03-21
N. MEXICO	2	1852-11-29
N. NEW ZEALAND	2	1950-03-14
OMAN	2	1570
PANAMA-COLOMBIA	2	1904-01-20
PENNSYLVANIA	2	1840-11-11
PERU-BRAZIL	2	2015-11-24
PERU-CHILE	2	1833-09-18
SAINT LUCIA	2	N/A (无震级数据)
SAMOA	2	1981-09-01
SEIONAIKAI, JAPAN	2	1510
SYRIAN COASTS	2	N/A (无震级数据)
TASMAN SEA	2	1913-02-22
TOGO	2	1788
URUGUAY	2	N/A (无震级数据)
VIRGIN ISLANDS; PUERTO RICO	2	N/A (无震级数据)
W. HOKKAIDO ISLAND, JAPAN	2	1947-11-04
W. LUZON ISLAND, PHILIPPINES	2	1925-05-05
W. SOLOMON SEA	2	1895-03-06
WYOMING	2	1994-02-03
AFGHANISTAN; INDIA	1	818
AFGHANISTAN; TAJIKISTAN	1	1976-11-27
ALABAMA	1	2003-04-29
ANATAHAN REGION, N. MARIANA ISLANDS	1	2010-05-29
ANTARCTICA	1	1973-10-06
ANTIGUA AND BARBUDA; ST KITTS	1	1974-10-08
ANTIGUA ISLAND & ST. CHRISTOPHER	1	N/A (无震级数据)
ANTIGUA; SAINT KITTS AND NEVIS	1	1690-04-16
ARMENIA-AZERBAIJAN-IRAN	1	1998-07-09
ARMENIA; AZERBAIJAN	1	N/A (无震级数据)
BALKANS	1	N/A (无震级数据)
BALLENY ISLANDS	1	1998-03-25
BARBADOS, SAINT VINCENT, DOMINICA, ANTIGUA	1	N/A (无震级数据)
BELGIUM	1	1983-11-08
BERING SEA	1	1991-02-21
BOLIVIA-PERU	1	1994-06-09
BOSNIA-HERZEGOVINA	1	1984-05-13
BOSTON AND SALEM, MASSACHUSETTS	1	N/A (无震级数据)
BRAZIL; FRENCH GUIANA; SURINAME; GUYANA	1	1885-08-04
BRITISH VIRGIN ISLANDS	1	N/A (无震级数据)
BURUNDI	1	2004-02-24
CALIFORNIA, MEXICO	1	1927-01-01
CALIFORNIA-NEVADA	1	1986-07-21
CALIFORNIA; MEXICO	1	1940-05-19
CANADA; MAINE	1	1982-01-11
CANO ISLAND	1	N/A (无震级数据)
CASCADIA SUBDUCTION ZONE	1	1700-01-27
CAYMAN IS; HONDURAS	1	2025-02-08
CENTRAL AFRICAN REPUBLIC	1	1921-09-16
COLOMBIA-PANAMA	1	2015-07-29
COLOMBIA; SAN CRISTOBAL, VENEZUELA	1	1973-04-24
COMOROS	1	2018-05-15
CONNECTICUT	1	N/A (无震级数据)
COSTA RICA-NICARAGUA	1	1950-10-05
CZECH REPUBLIC	1	2008-11-22
DJIBOUTI	1	1989-08-20
E. AWA, TOKUSHIMA PREFECTURE	1	1808-08-08
E. HOKKAIDO ISLAND, JAPAN	1	1938-05-28
E. LUZON ISLAND, PHILIPPINES	1	N/A (无震级数据)
E. SPORADES ISLANDS, AEGEAN ISLANDS	1	N/A (无震级数据)
EAST MEDITERRANEAN SEA	1	N/A (无震级数据)
EAST PACIFIC RIDGE	1	1958-11-04
EL SALVADOR; GUATEMALA; NICARAGUA	1	1859-12-09
EL SALVADOR; HONDURAS	1	1915-12-29

EL SALVADOR; HONDURAS; NICARAGUA	1	1932-05-21
ENSHUNADA	1	1855-11-07
FRENCH GUIANA	1	N/A (无震级数据)
FRENCH POLYNESIA	1	1848-07-12
FUTUNA ISLAND	1	1993-03-12
GABON	1	1974-09-23
GISBORNE	1	1880-09-08
GREECE, TURKEY	1	2014-05-24
GREECE-BULGARIA BORDER REGION	1	2006-02-20
GREECE-TURKEY	1	N/A (无震级数据)
GRENADA	1	N/A (无震级数据)
GUADELOUPE; ANTIGUA; ST KITTS	1	N/A (无震级数据)
GUAM, NORTHERN MARIANA ISLANDS	1	2010-08-13
GUAM; NORTHERN MARIANA ISLANDS	1	N/A (无震级数据)
GUATEMALA; EL SALVADOR	1	1862-12-20
GUINEA	1	1983-12-22
GULF OF MEXICO	1	2006-09-10
HONDURAS-GUATEMALA-EL SALVADOR	1	1982-09-29
HONDURAS; N GUATEMALA	1	1980-08-09
HONDURAS; GUATEMALA	1	N/A (无震级数据)
HONSHU ISLAND, JAPAN	1	1996-09-04
IBARAKI, JAPAN	1	1982-07-23
ILLINOIS	1	2008-04-18
INDIA-BANGLADESH BORDER	1	2000-01-03
INDONESIA-MALAYSIA	1	1967-04-12
INDONESIA; MALAYSIA	1	N/A (无震级数据)
INSTANBUL (CONSTANTINOPLE)	1	N/A (无震级数据)
IRAN; PAKISTAN	1	2013-04-16
IRAQ-SYRIA	1	N/A (无震级数据)
IRELAND	1	N/A (无震级数据)
ISRAEL; JORDAN	1	1927-07-11
ITALY-BALKANS NW	1	1976-09-15
JAVA-S. JAVA SEA	1	1930-06-19
KIRIBATI	1	1905-06-30
KOORI	1	1731-10-07
LAKE ERIE (GREAT LAKES)	1	N/A (无震级数据)
LAOS	1	2007-05-16
LAOS; VIETNAM	1	2001-02-19
LHOKNGA, ACEH	1	N/A (无震级数据)
MACQUARIE ISLAND	1	1989-05-23
MADAGASCAR	1	2017-01-11
MARTINIQUE & BARBADOS	1	N/A (无震级数据)
MASSACHUSETTS	1	1755-11-18
MASSACHUSETTS; NEW HAMPSHIRE	1	1727-11-09
MEXICO; GUATEMALA	1	2014-07-07
MICHIGAN; OHIO; INDIANA	1	N/A (无震级数据)
MINDANAO ISLAND, PHILIPPINES	1	1978-06-14
MIYAKOJIMA, JAPAN	1	1667
MONTENEGRO	1	1979-04-09
MONTSERRAT	1	N/A (无震级数据)
MOZAMBIQUE; ZIMBABWE	1	2018-12-22
MYANMAR	1	2011-03-24
MYANMAR (BURMA) COAST	1	N/A (无震级数据)
MYANMAR (BURMA); THAILAND	1	2025-03-28
MYNAMAR (BURMA)	1	N/A (无震级数据)
N. SANRIKU, JAPAN	1	1763-03-11
NE. HOKKAIDO ISLAND, JAPAN	1	N/A (无震级数据)
NEMURO, JAPAN	1	N/A (无震级数据)
NEPAL; INDIA	1	1934-01-15
NEPAL; TIBET (XIZANG PROVINCE)	1	1916-08-28
NEVADA-CALIFORNIA BORDER	1	1992-06-29
NEW HAMPSHIRE	1	N/A (无震级数据)
NEW HAMPSHIRE; MAINE	1	1940-12-24
NEW JERSEY	1	1895-09-01
NICARAGUA; EL SALVADOR	1	1921-03-28
NORTH ATLANTIC RIDGE	1	1925-10-13
NORTH CAROLINA	1	2020-08-09
NORTH CORINTH GULF	1	1965-07-06
NORWAY	1	1819-08-31
NOSHIRO, JAPAN	1	1983-06-09
NW. HOKKAIDO ISLAND, JAPAN	1	1863-09-20
NW. HONSHU ISLAND, JAPAN	1	N/A (无震级数据)
OFF COAST SW AVALON PENINSULA, NEWFOUNDLAND	1	N/A (无震级数据)
OREGON	1	1993-09-21
PACIFIC OCEAN	1	1932-11-02
PAKISTAN-NW AFGHANISTAN	1	1984-02-16
PAKISTAN; INDIA	1	2009-02-20
PERU-BOLIVIA	1	1952-02-26
PORTUGAL; MOROCCO	1	1969-02-28
PUERTO RICO; DOMINICAN REPUBLIC	1	N/A (无震级数据)
ROMANIA; MOLDOVA	1	1821-11-17
RUSSIA; MONGOLIA	1	1761-12-09
SAGAMI, JAPAN	1	1331
SAINT VINCENT	1	N/A (无震级数据)
SE. NEW ZEALAND	1	1922-12-25
SEA OF JAPAN	1	1643
SETKAIDO-NANKAIDO	1	1698-12-22
SHINANO	1	N/A (无震级数据)
SIERRA LEONE	1	1795-05-20
SLOVENIA	1	2004-07-12
SOLOMON ISLANDS; NEW CALEDONIA	1	1934-07-21
SOUTH AFRICA; SWAZILAND	1	2000-02-07
SOUTH CAROLINA	1	1886-09-01
SOUTH CAROLINA; GEORGIA; PENNSYLVANIA	1	N/A (无震级数据)
SOUTH COASTS OF ASIA MINOR	1	N/A (无震级数据)
SPAIN; ALGERIA; MOROCCO	1	881
SRI LANKA	1	N/A (无震级数据)
SUDAN	1	1993-08-01
SURUGA, JAPAN	1	1589
SW. KYUSHU ISLAND, JAPAN	1	1769-08-29
SYRIA; ISRAEL; ASIA	1	N/A (无震级数据)
SYRIA; LEBANON	1	2024-08-12
TAJIKISTAN; AFGHANISTAN	1	2002-04-12
TEXAS	1	2013-04-18
TIMOR SEA	1	1891-10-05
TOBAGO	1	1997-04-22
TONGA	1	2023-05-10
TOYAMA PREF., JAPAN	1	N/A (无震级数据)
TRINIDAD & ST. CHRISTOPHER	1	1831-12-03
TRINIDAD; GRENADA	1	1888-01-10
TURKEY-CIS	1	1940-05-07
UK	1	1931-06-07
VENEZUELA-N COLOMBIA	1	1981-10-18
VIRGINIA	1	2011-08-23
W. KYUSHU ISLAND, JAPAN	1	N/A (无震级数据)
WASHINGTON-OREGON BORDER	1	1993-03-25
YATSUSHIRO, JAPAN	1	1619
YELLOW SEA	1	1649
ZAMBIA	1	2017-02-24

In [64]: # 2

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

# 文件
file = "D:/桌面/ESE5023_Assignments_12431305/PS2/2281305.csv"

# 读需要的 column
wind_data = pd.read_csv(file, usecols=['DATE', 'WIND'], dtype=str, low_memory=False)

# 时间与风速解析
# 日期转换
wind_data['DATE'] = pd.to_datetime(wind_data['DATE'], errors='coerce')

# 解析 WIND: 第4个字段为风速, 第5个为质量码
def parse_wind_speed(wnd_string: str):
    try:
        parts = str(wnd_string).split(',')
        spd = parts[3] # 0.1 m/s
        qc = parts[4] # 质量码
        if qc not in {'0', '1', '4', '5', '9'}:
            return np.nan
        if spd == '9999':
            return np.nan
        return float(spd) / 10.0
    except Exception:
        return np.nan

wind_data['Wind_Speed'] = wind_data['WIND'].apply(parse_wind_speed)

# 丢弃解析失败的记录
wind_data = wind_data.dropna(subset=['DATE', 'Wind_Speed'])

# 过滤到 2010-2020
shenzhen_2010_2020 = (wind_data['DATE'] >= '2010-01-01') & (wind_data['DATE'] <= '2020-12-31')
wind_data = wind_data.loc[shenzhen_2010_2020].copy()

print(f"[OK] 2010-2020 有效行数: {len(wind_data)}")

# 计算“月平均风速”
monthly_avg = (
    wind_data.set_index('DATE')['Wind_Speed']
    .resample('MS') # 月初频率
    .mean()
    .dropna()
)

# 12 个月居中滚动均值
rolling_12m = monthly_avg.rolling(window=12, center=True).mean()

# 线性回归趋势
x = np.arange(len(monthly_avg), dtype=float)
y = monthly_avg.values.astype(float)

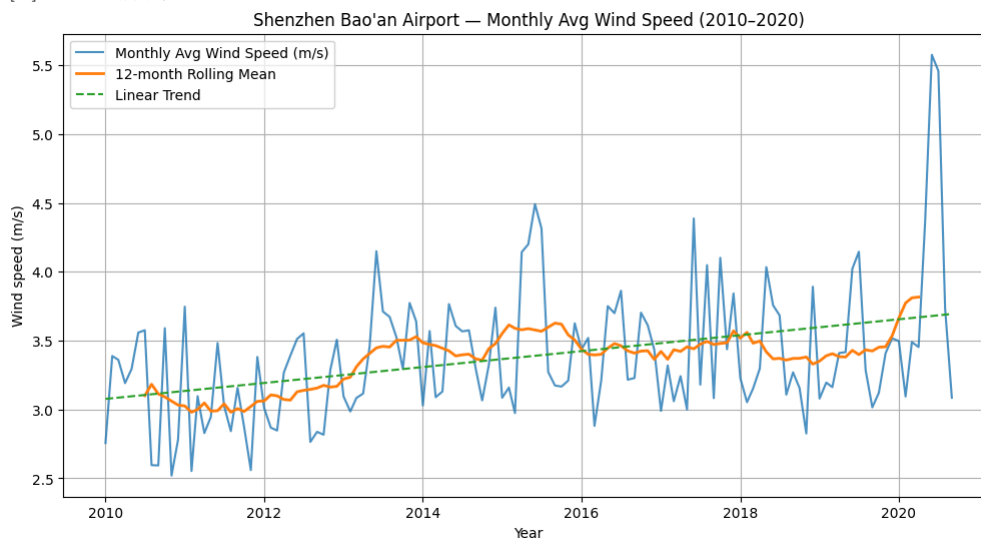
# 拟合直线
slope, intercept = np.polyfit(x, y, 1)
y_fit = slope * x + intercept

# R^2
ss_res = np.sum((y - y_fit) ** 2)
ss_tot = np.sum((y - y.mean()) ** 2)
r2 = 1 - ss_res / ss_tot if ss_tot > 0 else np.nan

# 绘图
plt.figure(figsize=(12, 6))
plt.plot(monthly_avg.index, monthly_avg.values, label='Monthly Avg Wind Speed (m/s)', alpha=0.8)
plt.plot(rolling_12m.index, rolling_12m.values, label='12-month Rolling Mean', linewidth=2.0)
plt.plot(monthly_avg.index, y_fit, linestyle='--', label='Linear Trend') # 不指定颜色
plt.title("Shenzhen Bao'an Airport - Monthly Avg Wind Speed (2010-2020)")
plt.xlabel("Year")
plt.ylabel("Wind speed (m/s)")
plt.grid(True)
plt.legend(loc="best")
plt.show()

# 统计数据
per_month = slope
per_year = per_month * 12.0
per_decade = per_year * 10.0
print(f"有效月份数: {len(monthly_avg)}")
print(f"线性回归斜率: {per_month:.4f} m/s 每月 = {per_year:.4f} m/s 每年 = {per_decade:.4f} m/s 每十年")
print(f"R^2: {r2:.4f}")
print("平均风速呈现上升趋势")
```

[OK] 2010-2020 有效行数: 111346



有效月份数: 129
线性回归斜率: 0.0048 m/s 每月 = 0.0578 m/s 每年 = 0.5782 m/s 每十年
R²: 0.1354
平均风速呈现上升趋势

```
In [7]: #3-1 对地震中心深度的无效数据进行过滤

# 将"Focal Depth (km)"转为数值、丢弃无法转换的行，并按深度从大到小排序
Sig_Eqs = (
    Sig_Eqs.assign(**{"Focal Depth (km)": pd.to_numeric(Sig_Eqs["Focal Depth (km)"], errors="coerce")})
    .dropna(subset=["Focal Depth (km)"]).sort_values(by="Focal Depth (km)", ascending=False)
    .reset_index(drop=True)
)
Sig_Eqs
```

Out[7]:

	Search Parameters	Year	Mo	Dy	Hr	Mn	Sec	Tsu	Vol	Location Name	...	Total Missing Description	Total Injuries	Total Injuries Description	Total Damage (\$Mil)	Total Damage Description	Total Houses Destroyed	Total Houses Destroyed Description	Total Houses Damaged	H Dar Descr
0	NaN	2002.0	8.0	19.0	11.0	8.0	24.3	NaN	NaN	FIJI ISLANDS	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	NaN	2018.0	9.0	6.0	15.0	49.0	18.7	NaN	NaN	FIJI ISLANDS	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	NaN	2015.0	5.0	30.0	11.0	23.0	2.1	NaN	NaN	JAPAN: BONIN ISLANDS [CHICHUJIMA ISLAND]	...	NaN	12.0	1.0	NaN	NaN	NaN	NaN	NaN	
3	NaN	1970.0	7.0	31.0	17.0	8.0	5.4	NaN	NaN	PERU: JUIN PROVINCE FELT ARGENTINA TO MEXICO...	...	NaN	NaN	2.0	NaN	2.0	NaN	NaN	NaN	
4	NaN	1921.0	12.0	18.0	15.0	29.0	0.0	NaN	NaN	PERU	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
...
3569	NaN	2013.0	8.0	7.0	9.0	6.0	52.2	NaN	NaN	GREECE: CENTRAL: PHTHIOTIS, REGGINI	...	NaN	NaN	NaN	NaN	2.0	100.0	2.0	NaN	
3570	NaN	1988.0	9.0	6.0	0.0	42.0	33.7	NaN	NaN	PAPUA NEW GUINEA: KAIAPIT (LANDSLIDE GENERATED)	...	NaN	NaN	NaN	NaN	2.0	NaN	3.0	NaN	
3571	NaN	2014.0	2.0	26.0	0.0	0.0	7.0	NaN	NaN	AUSTRALIA: KALGOORLIE	...	NaN	NaN	NaN	NaN	1.0	NaN	NaN	NaN	
3572	NaN	1938.0	11.0	6.0	8.0	53.0	53.0	1718.0	NaN	JAPAN: OFF EAST COAST HONSHU	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3573	NaN	1938.0	11.0	6.0	21.0	38.0	47.0	1719.0	NaN	JAPAN: NEAR E COAST HONSHU	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

3574 rows × 40 columns

```
In [5]: #3-2新中国以来每年的地震次数

# 读数据
file_path = r"D:/桌面/ESE5023_Assignments_12431305/PS2/earthquakes.tsv"
Sig_Eqs = pd.read_csv(file_path, sep="\t")

# 抽取国家、清洗年份
# 国家: 从 Location Name 的第一个冒号(支持中英文)左侧提取: 统一大写
Sig_Eqs["Country"] = (
    Sig_Eqs["Location Name"].astype("string")
    .str.extract(r"^\s*([^\s:]+)", expand=False)
    .str.strip()
    .replace("", pd.NA)
    .str.upper()
)

# 年份转为数值
Sig_Eqs["Year"] = pd.to_numeric(Sig_Eqs["Year"], errors="coerce")

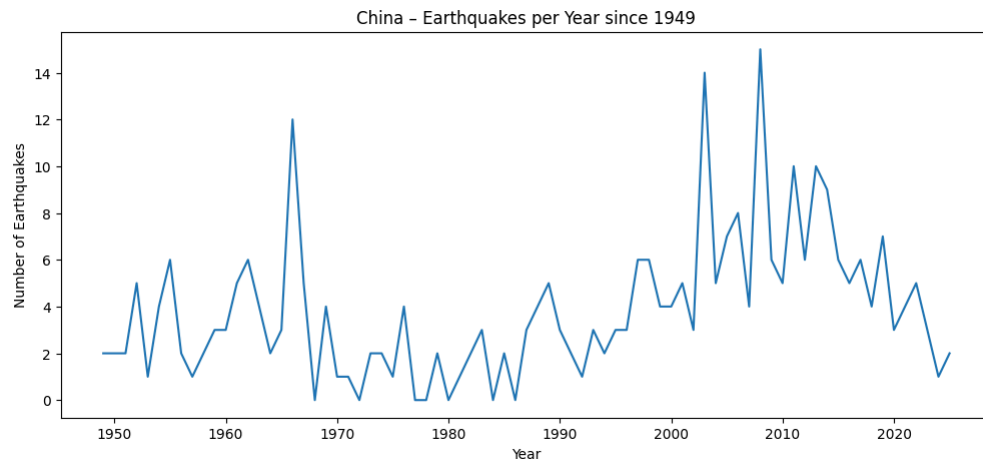
# 统计中国1949年以来每年的地震次数
china = Sig_Eqs[(Sig_Eqs["Country"] == "CHINA") & (Sig_Eqs["Year"] >= 1949)].dropna(subset=["Year"]).copy()
china["Year"] = china["Year"].astype(int)

# 计数
counts = china.groupby("Year").size().rename("Earthquake_Count")

# 补齐年份 (1949 ~ 数据中中国地震的最大年份), 空缺填0
year_min, year_max = 1949, (int(china["Year"].max()) if not china.empty else 1949)
counts = counts.reindex(range(year_min, year_max + 1), fill_value=0)

# 绘图: 折线图
plt.figure(figsize=(12, 5))
plt.plot(counts.index, counts.values)
plt.title("China - Earthquakes per Year since 1949")
plt.xlabel("Year")
plt.ylabel("Number of Earthquakes")
```

Out[5]: Text(0, 0.5, 'Number of Earthquakes')



```
In [11]: #3-32000年以来、深度大于30km、地震等级大于5级、死亡人数大于100的地震总结

import pandas as pd

# 读取数据
file_path = r"D:/桌面/ESE5023_Assignments_12431305/PS2/earthquakes.tsv"
Sig_Eqs = pd.read_csv(file_path, sep="\t")

# 从 Location Name 抽取国家
Sig_Eqs["Country"] = (
    Sig_Eqs["Location Name"].astype("string")
    .str.extract(r"^\s*([^\s:]+)", expand=False) # 取第一个冒号前的文本
    .str.strip()
    .replace("", pd.NA)
    .str.upper()
)

# 死亡人数列（优先 Total_Deaths，否则 Deaths），并转为数值
Deaths_real = "Total_Deaths" if "Total_Deaths" in Sig_Eqs.columns else "Deaths"
Sig_Eqs[Deaths_real] = pd.to_numeric(Sig_Eqs[Deaths_real], errors="coerce")

# 把 Year、Mag、Focal Depth (km) 转为数值
Sig_Eqs["Year"] = pd.to_numeric(Sig_Eqs["Year"], errors="coerce")
Sig_Eqs["Mag"] = pd.to_numeric(Sig_Eqs["Mag"], errors="coerce")
Sig_Eqs["Focal Depth (km)"] = pd.to_numeric(Sig_Eqs["Focal Depth (km)"], errors="coerce")

# 筛选条件—Year >= 2000、深度 > 30、震级 > 5、死亡人数 > 100
cond = (
    (Sig_Eqs["Year"] >= 2000) &
    (Sig_Eqs["Focal Depth (km)"] > 30) &
    (Sig_Eqs["Mag"] > 5) &
    (Sig_Eqs[Deaths_real] > 100)
)

# 按国家排序：同一国家内按死亡人数降序
cols_show = ["Country", "Year", "Mag", "Focal Depth (km)", Deaths_real, "Location Name"]
filtered = (
    Sig_Eqs.loc[cond, cols_show]
    .sort_values(["Country", Deaths_real], ascending=[True, False])
    .reset_index(drop=True)
)

print("=== 2000年以来，深度>30km、M>5、死亡>100 的地震清单（按国家排序） ===")
if filtered.empty:
    print("无记录")
else:
    print(filtered.to_string(index=False))

# 按国家汇总：事件数、死亡总数（按国家字母序）
summary = (
    filtered.groupby("Country", as_index=False)
    .agg(Quake_Count=("Year", "count"),
        Total_Deaths=(Deaths_real, "sum"))
    .sort_values("Country")
    .reset_index(drop=True)
)

print("\n=== 按国家汇总（事件数 / 死亡总数） ===")
if summary.empty:
    print("无记录")
else:
    print(summary.to_string(index=False))
```

```
=== 2000年以来，深度>30km、M>5、死亡>100 的地震清单（按国家排序） ===
Country  Year  Mag  Focal Depth (km)  Total Deaths  Location Name
AFGHANISTAN  2015.0  7.5      231.0          399.0  AFGHANISTAN: HINDU KUSH
AFGHANISTAN  2002.0  7.4      226.0         166.0  AFGHANISTAN: HINDU KUSH; SAMANGHAN, KABUL, RUSTAQ
EL SALVADOR; GUATEMALA  2001.0  7.7       60.0          844.0  EL SALVADOR; GUATEMALA
INDIA  2011.0  6.9       50.0          127.0  INDIA: SIKKIM; NEPAL; CHINA; BHUTAN
INDONESIA  2009.0  7.5       81.0         1117.0  INDONESIA: SUMATRA: PADANG
INDONESIA  2018.0  6.9       34.0          560.0  INDONESIA: LOMBOK ISLAND
INDONESIA  2000.0  7.9       33.0          103.0  INDONESIA: SUMATRA: BENGKULU, ENGGANO
MEXICO  2017.0  7.1       48.0          369.0  MEXICO: MEXICO CITY, MORELOS, PUEBLA
PERU  2007.0  8.0       39.0          596.0  PERU: ICA, PISCO, LIMA
PERU  2001.0  8.4       33.0          103.0  PERU: AREQUIPA, MOQUEGUA, TACNA, AYACUCHO

=== 按国家汇总（事件数 / 死亡总数） ===
Country  Quake_Count  Total_Deaths
AFGHANISTAN      2         565.0
EL SALVADOR; GUATEMALA      1         844.0
INDIA             1         127.0
INDONESIA          3        1780.0
MEXICO            1         369.0
PERU              2         699.0
```

In []: