蒙芳秀, 12331305

1. Significant earthquakes since 2150 B.C.

1.1 **[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.

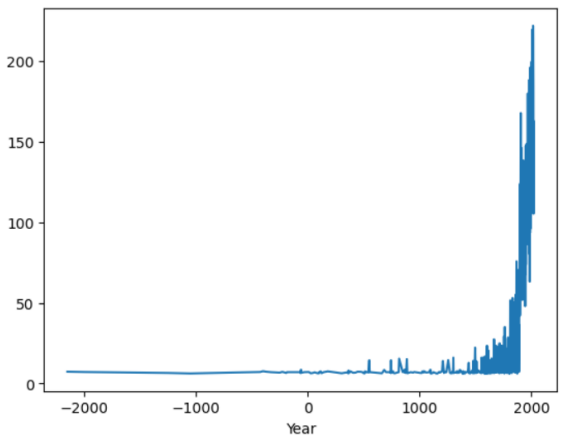
```
In [14]: # Compute the total number of deaths caused by earthquakes since 2150 B.C. in each country
         Sig_Eqs1 = Sig_Eqs.groupby('Country')['Total Injuries'].sum().sort_values(ascending=False)[0:10]
         Sig_Eqs1

Out[14]: Country
         CHINA         1319998.0
         TURKEY         317585.0
         HAITI          313431.0
         IRAN           206769.0
         INDIA          201239.0
         PAKISTAN       168795.0
         JAPAN          150110.0
         INDONESIA      100171.0
         GUATEMALA       76626.0
         PERU            73728.0
         Name: Total Injuries, dtype: float64
```

1.2 **[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?

```
In [18]: # Compute the total number of earthquakes with magnitude larger than 6.0 (use column Mag as the magnitude) worldwide each year
         Sig_Eqs2=Sig_Eqs[Sig_Eqs['Mag']>6.0]
         Sig_Eqs22 = Sig_Eqs2.groupby('Year')['Mag'].sum().plot()

Out[18]: <Axes: xlabel='Year'>
```



Due to the influence of human activities and natural changes, the total number of earthquakes with magnitudes greater than 6.0 (listed as Mag magnitudes) worldwide each year shows an increasing trend year by year.

1.3 [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

Note: I got inspired by reading https://blog.51cto.com/u_16213327/7239823 python

In [71]:
```
#https://blog.51cto.com/u_16213327/7239823
def CountEq_LargestEq(Sig_Eqs,country):
    #the total number of earthquakes since 2150 B.C. in a given country
    Sig_Eqs3 = Sig_Eqs[Sig_Eqs['Country']==country]['Mag'].sum()
    #the date of the largest earthquake ever happened in this country
    Sig_Eqs33 = Sig_Eqs[Sig_Eqs['Country']==country].sort_values('Mag',ascending=False)[0:1]
    columns = ['Year', 'Mo','Dy','Hr','Mn','Sec']
    Sig_Eqs33set = Sig_Eqs33.loc[:, columns]
    Sig_Eqs33set['Total Mag']=Sig_Eqs3
    return Sig_Eqs33set
#CountEq_LargestEq(Sig_Eqs,'CHINA')
Country=list(set(Sig_Eqs['Country'].tolist()))
result_df = pd.DataFrame()
for country in Country:
    CountEq_LargestEq1=CountEq_LargestEq(Sig_Eqs,country)
    columns = CountEq_LargestEq1.columns
    result_df = pd.concat([result_df, CountEq_LargestEq1[columns]], axis=0) #按列合并
result_df.sort_values('Total Mag',ascending=False)
```

Out[71]:

	Year	Mo	Dy	Hr	Mn	Sec	Total Mag
982	1668.0	7.0	25.0	NaN	NaN	NaN	3532.6
5743	2011.0	3.0	11.0	5.0	46.0	24.1	2447.2
5341	2004.0	12.0	26.0	0.0	58.0	53.4	2217.1
238	856.0	12.0	22.0	NaN	NaN	NaN	1524.3
3957	1964.0	3.0	28.0	3.0	36.0	0.0	1458.1
...	...	...	...	...	...	...	...
1469	1800.0	NaN	NaN	NaN	NaN	NaN	0.0
588	1490.0	NaN	NaN	NaN	NaN	NaN	0.0
2273	1882.0	1.0	NaN	NaN	NaN	NaN	0.0
2462	1897.0	4.0	25.0	NaN	NaN	NaN	0.0
2126	1871.0	9.0	NaN	NaN	NaN	NaN	0.0

156 rows × 7 columns

2. Wind speed in Shenzhen during the past 10 years

[10 points] Plot monthly averaged wind speed as a function of the observation time. Is there a trend in monthly averaged wind speed within the past `10` years?

Note: “Huiran Feng explained to me what is asked in problem set 2”

In [74]:
```
#2-1
##Wind speed in Shenzhen during the past 10 years
import pandas as pd
import numpy as np
# Read another csv
Wind_speed_data = pd.read_csv('2281305.csv')
Wind_speed_data.info()
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_4644\1251023456.py:5: DtypeWarni
Specify dtype option on import or set low_memory=False.
  Wind_speed_data = pd.read_csv('2281305.csv')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 111984 entries, 0 to 111983
Data columns (total 43 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   STATION           111984 non-null  int64
 1   DATE              111984 non-null  object
 2   SOURCE            111984 non-null  int64
 3   REPORT_TYPE       111984 non-null  object
 4   CALL_SIGN         111984 non-null  object
 5   QUALITY_CONTROL   111984 non-null  object
 6   AA1               6493 non-null    object
```

In [89]:
```
#拆分数据框
#将WND拆分
splitdata=Wind_speed_data['WND'].str.split(',',expand=True)
splitdata.columns=['direction angle','direction quality code','type code','speed rate','speed quality code']
Wind_speed_data1=pd.concat([Wind_speed_data,splitdata],axis=1)
Wind_speed_data1=Wind_speed_data1.drop('WND',axis=1)
```

Out[89]:

	STATION	DATE	SOURCE	REPORT_TYPE	CALL_SIGN	QUALITY_CONTROL	AA1	AA2	AA3	AJ1	...	REP
0	59493099999	2010-01-02T00:00:00	4	SY-MT	ZGSZ	V020	06,0000,2,1	24,0000,2,1	NaN	NaN	...	
1	59493099999	2010-01-02T01:00:00	4	FM-15	ZGSZ	V020	NaN	NaN	NaN	NaN	...	
2	59493099999	2010-01-02T02:00:00	4	FM-15	ZGSZ	V020	NaN	NaN	NaN	NaN	...	

In [103]:
```
#将DATE拆分
splitdata1=Wind_speed_data1['DATE'].str.split('-',expand=True)
splitdata1.columns=['year','month','hour']
Wind_speed_data2=pd.concat([Wind_speed_data1,splitdata1],axis=1)
Wind_speed_data2=Wind_speed_data2.drop('DATE',axis=1)
Wind_speed_data2['year_month'] = Wind_speed_data2['year'] + '-' + Wind_speed_data2['month']
Wind_speed_data2
```

Out[103]:

	STATION	SOURCE	REPORT_TYPE	CALL_SIGN	QUALITY_CONTROL	AA1	AA2	AA3	AJ1	AY1	...	TMP	VIS	di
0	59493099999	4	SY-MT	ZGSZ	V020	06,0000,2,1	24,0000,2,1	NaN	NaN	6,1,06,1	...	+0161,1	004000,1,N,1	
1	59493099999	4	FM-15	ZGSZ	V020	NaN	NaN	NaN	NaN	NaN	...	+0170,1	002600,1,N,1	
2	59493099999	4	FM-15	ZGSZ	V020	NaN	NaN	NaN	NaN	NaN	...	+0180,1	002600,1,N,1	
3	59493099999	4	SY-MT	ZGSZ	V020	NaN	NaN	NaN	NaN	6,1,03,1	...	+0192,1	005000,1,N,1	
4	59493099999	4	FM-15	ZGSZ	V020	NaN	NaN	NaN	NaN	NaN	...	+0180,1	002100,1,N,1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
111979	59493099999	4	FM-15	99999	V020	NaN	NaN	NaN	NaN	NaN	...	+0290,1	009999,1,9,9	
111980	59493099999	4	FM-15	99999	V020	NaN	NaN	NaN	NaN	NaN	...	+0290,1	009999,1,9,9	
111981	59493099999	4	FM-15	99999	V020	NaN	NaN	NaN	NaN	NaN	...	+0290,1	009999,1,9,9	
111982	59493099999	4	FM-15	99999	V020	NaN	NaN	NaN	NaN	NaN	...	+0290,1	009999,1,9,9	
111983	59493099999	4	FM-15	99999	V020	NaN	NaN	NaN	NaN	NaN	...	+0290,1	009999,1,9,9	

111984 rows × 49 columns

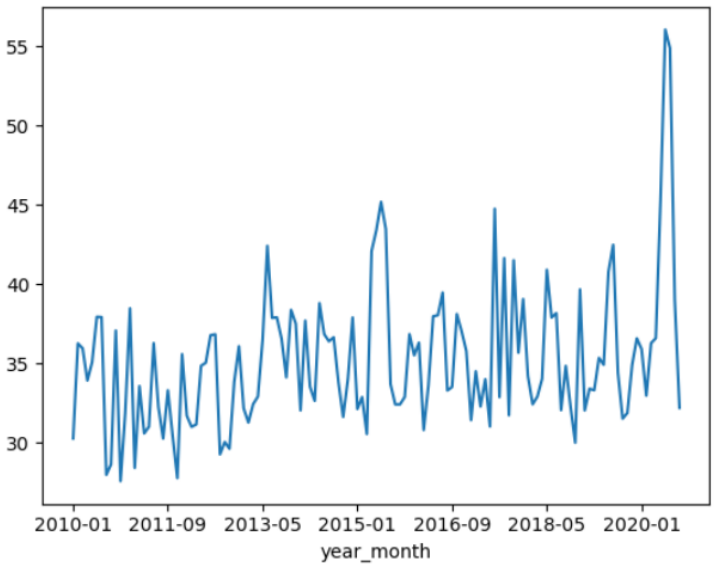
In [126]:
```
# Data clean
#学习冯汇然同学
# 1. Drop direction angle == 999
Wind_speed_data2= Wind_speed_data2[Wind_speed_data2['direction angle'] != '999']
# 2. Drop direction quality code in [2,3,6,7]
Wind_speed_data2 = Wind_speed_data2[~Wind_speed_data2['direction quality code'].isin(['2', '3', '6', '7'])]
# 3. Drop type code == 9
Wind_speed_data2= Wind_speed_data2[Wind_speed_data2['type code'] != '9']
# 4. Drop speed rate == 9999
Wind_speed_data2 = Wind_speed_data2[Wind_speed_data2['speed rate'] != '9999']
Wind_speed_data2['speed rate'] = Wind_speed_data2['speed rate'].astype(float)
# 5. Drop speed quality code in [2,3,6,7]
Wind_speed_data2= Wind_speed_data2[~Wind_speed_data2['speed quality code'].isin(['2', '3', '6', '7'])]
# Show
Wind_speed_data2
```

Out[126]:

ALITY_CONTROL	AA1	AA2	AA3	AJ1	AY1	...	VIS	direction angle	direction quality code	type code	speed rate	speed quality code	year	month	hour	year_month
V020	06,0000,2,1	24,0000,2,1	NaN	NaN	6,1,06,1	...	004000,1,N,1	040	1	N	20.0	1	2010	01	02T00:00:00	2010-01
V020	NaN	NaN	NaN	NaN	6,1,03,1	...	005000,1,N,1	140	1	N	10.0	1	2010	01	02T03:00:00	2010-01
V020	NaN	NaN	NaN	NaN	NaN	...	002100,1,N,1	300	1	N	40.0	1	2010	01	02T04:00:00	2010-01
V020	NaN	NaN	NaN	NaN	NaN	...	001800,1,N,1	320	1	N	50.0	1	2010	01	02T05:00:00	2010-01
V020	06,0000,2,1	NaN	NaN	NaN	6,1,06,1	...	003000,1,N,1	270	1	N	10.0	1	2010	01	02T06:00:00	2010-01
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
V020	NaN	NaN	NaN	NaN	NaN	...	009999,1,9,9	170	1	N	30.0	1	2020	09	11T17:00:00	2020-09
V020	NaN	NaN	NaN	NaN	NaN	...	009999,1,9,9	180	1	N	40.0	1	2020	09	11T18:00:00	2020-09

In [128]:
```
Wind_speed_data2.groupby('year_month')['speed rate'].mean().plot( )
```

Out[128]: <Axes: xlabel='year_month'>



In the past 10 years, the monthly averaged wind speed had an increasing trend.

3. Explore a data set

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

Note: I got inspired by reading https://www.qb5200.com/article/593749.html

In [4]:
```
import pandas as pd
global_mean=pd.read_excel('monthly global mean of baseline data from AGAGE GC-MD data.xlsx')
#https://www.qb5200.com/article/593749.html
#3-1
# using dropna() method 删除空行
global_mean1= global_mean.dropna()
# 删除全为0的列
global_mean2 = global_mean1.loc[:, (global_mean1!= 0).any(axis=0)]
#删除有0的列
global_mean3=global_mean2.loc[:,global_mean2.all(axis=0)] #: axis=0对列操作
#删除全0的行
global_mean4=global_mean3.loc[(global_mean3!= 0).any(axis=1),:]
global_mean4
```

Out[4]:

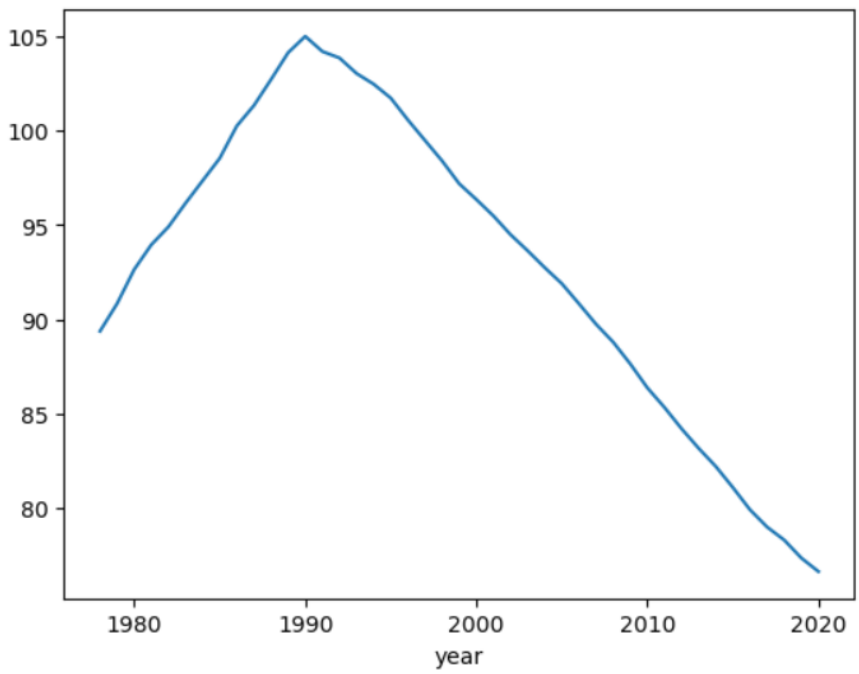
	time	month	year	CFC-11	sigama11	CFC-12	sigama12	CCl4	sigama14	N2O	sigama15
0	1978.542	7	1978	147.067	6.349	268.552	11.092	88.972	1.348	299.316	0.823
1	1978.625	8	1978	148.527	5.784	269.862	11.241	89.489	1.784	299.441	0.561
2	1978.708	9	1978	148.925	5.412	271.445	9.476	89.393	1.479	299.889	0.490
3	1978.792	10	1978	149.670	5.189	273.775	9.159	88.964	1.188	300.557	0.582
4	1978.875	11	1978	150.647	4.904	276.780	7.185	88.941	0.657	300.598	0.467
...	...	...	...	...	...	...	...	...	...	...	...
496	2019.875	11	2019	225.271	0.649	503.022	0.240	76.919	0.427	332.712	0.373
497	2019.958	12	2019	225.030	0.685	502.679	0.374	76.854	0.422	332.740	0.413
498	2020.042	1	2020	224.758	0.717	502.339	0.486	76.749	0.414	332.753	0.434
499	2020.125	2	2020	224.479	0.743	501.955	0.613	76.639	0.402	332.849	0.543
500	2020.208	3	2020	224.183	0.726	501.469	0.629	76.541	0.403	332.854	0.575

501 rows × 11 columns

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

In [5]:
```
#3-2
global_mean4.groupby('year')['CCl4'].mean().plot()
```

Out[5]: <Axes: xlabel='year'>



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

Note: I got inspired by reading and https://blog.51cto.com/u_16213393/7071739

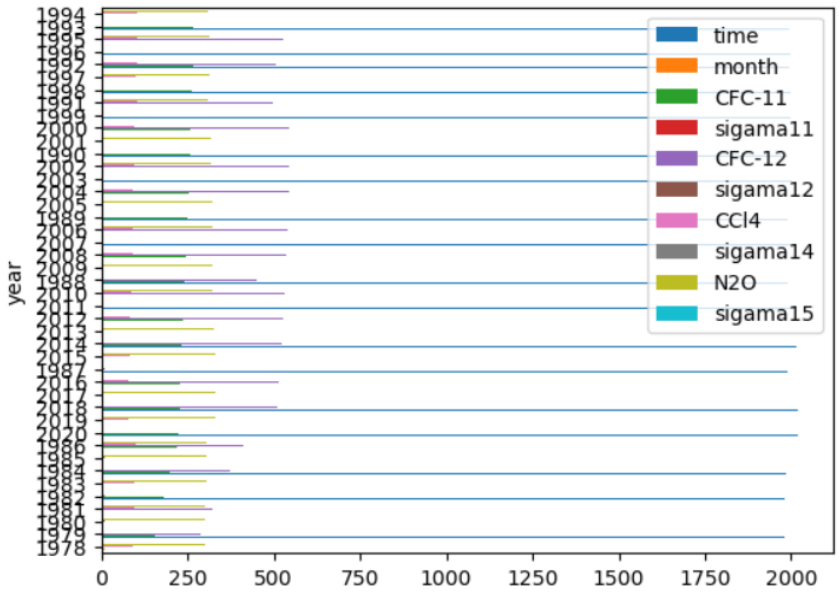
In [11]:
```
#3-3
global_mean5=global_mean4.groupby('year').mean()
global_mean5.sort_values(by='CFC-11')
```

Out[11]:

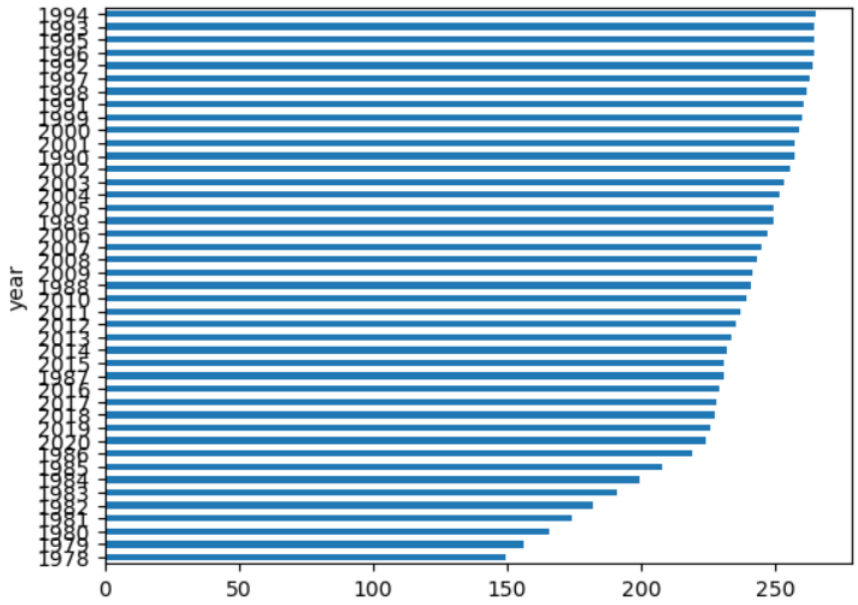
year	time	month	CFC-11	sigama11	CFC-12	sigama12	CCl4	sigama14	N2O	sigama15
1978	1978.750	9.5	149.573833	5.543500	273.446500	9.445500	89.364667	1.321167	300.009333	0.575167
1979	1979.500	6.5	156.145417	5.187417	289.108917	8.562333	90.826000	1.770417	300.794500	0.311500
1980	1980.500	6.5	165.689250	4.737500	306.952667	8.309833	92.631083	1.866083	301.083333	0.316417
1981	1981.500	6.5	174.205583	4.416833	321.762583	8.248750	93.942333	2.158000	301.365083	0.355167
1982	1982.500	6.5	182.177167	4.356417	338.954167	7.957083	94.889333	1.551000	303.444250	0.395083
1983	1983.500	6.5	190.881667	4.610583	355.958833	7.743667	96.141333	1.662917	303.635333	0.254917
1984	1984.500	6.5	199.263500	4.284167	372.582167	7.612167	97.334417	1.544917	304.019083	0.300083
1985	1985.500	6.5	208.019167	4.353667	390.789417	7.472167	98.520750	1.286500	304.443500	0.378833
1986	1986.500	6.5	219.007083	5.053000	410.324500	7.879250	100.234500	1.529750	305.334917	0.422917
2020	2020.125	2.0	224.473333	0.728667	501.921000	0.576000	76.643000	0.406333	332.818667	0.517333
2019	2019.500	6.5	225.957667	0.745000	504.560333	0.301250	77.359250	0.428833	332.321083	0.389417
2018	2018.500	6.5	227.446750	0.873833	508.465417	0.315500	78.311250	0.430417	331.542167	0.345667
2017	2017.500	6.5	228.326250	1.024750	511.417750	0.487250	78.999083	0.448083	330.360500	0.412000

In [13]:
```
#3-3
global_mean5=global_mean4.groupby('year').mean()
global_mean5.sort_values(by='CFC-11').plot.barh()
```

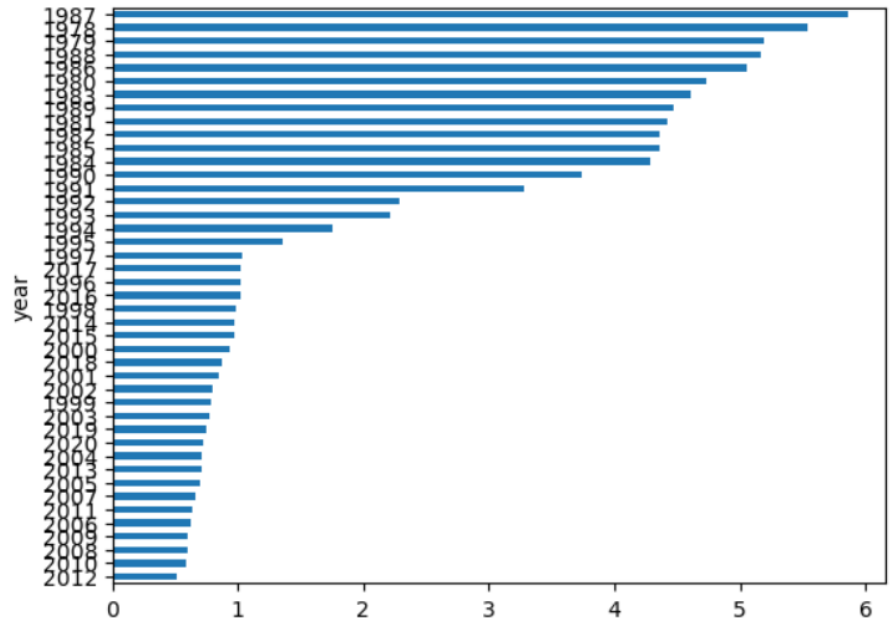
Out[13]: <Axes: ylabel='year'>



```
In [19]: global1_mean4.groupby('year')['CFC-11'].mean().sort_values().plot.barh( )
Out[19]: <Axes: ylabel='year'>
```



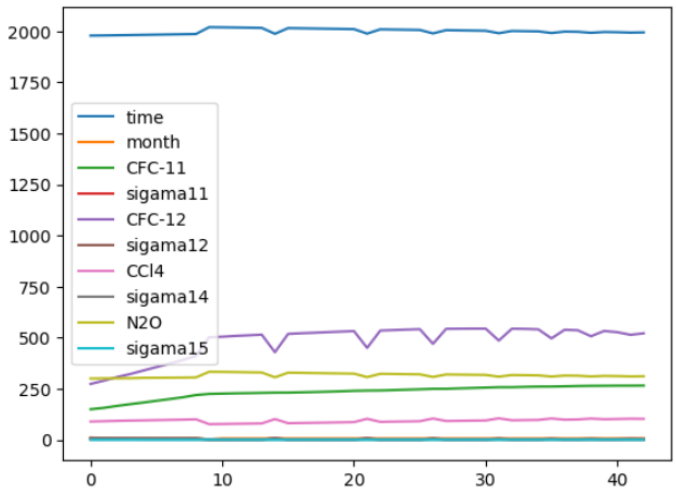

```
In [20]: global1_mean4.groupby('year')['sigama11'].mean().sort_values().plot.barh( )
Out[20]: <Axes: ylabel='year'>
```



Combining the above two figures, it can be seen that the benchmark value measured by CFC-11 was the highest in 1994 and the lowest in 1978, but the corresponding standard deviation in 1978 was also large.

In [29]:
```
#https://blog.51cto.com/u_16213393/7071739
import matplotlib.pyplot as plt
global_mean6=global_mean5.sort_values(by= 'CFC-11') #对所有数据排序
columns = global_mean6.columns.tolist()  # 获取所有列的名称
data_values = [global_mean6[col].tolist() for col in columns]  # 将每一列的数据存储到列表中
for i in range(len(columns)):
    plt.plot(data_values[i], label=columns[i])  # 画出每一列的数据，并为每一列添加标签
plt.legend()  # 添加图例
```

Out[29]: <matplotlib.legend.Legend at 0x2a919a572d0>



We can find that the reference value measured by CFC-12 is the largest, and the reference value measured by CC14 is the smallest, but the error between both is very small.