全球区划数据集的读取和定矩形区域、定点索引

所需第三方库：geopandas 使用数据库：GAMD_410, 下载地址：https://gadm.org/download_world.html ，该数据集较为精细，唯一需要注意的是使用时的某些政治问题，谨慎对待

这种gpkg数据库也可以用开源的QGIS打开，方便快速可视化

geopandas是在pandas基础上开发的，很多基本的操作相通，也增加了一些方法和索引器之类

```
import geopandas as gpd
```

```
gamd_path = "H:\\Region\\GAMD_410\\gadm_410.gpkg"
gamd_file = gpd.read_file(gamd_path)
gamd_file
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	UID	GID_0	NAME_0	VARNAME_0	GID_1	NAME_1	VARNAME_1	NL_NAME_1	ISO_1	HASC_1	...
0	1	AFG	Afghanistan		AFG.1_1	Badakhshan	Badahšan			AF.BD	...
1	2	AFG	Afghanistan		AFG.1_1	Badakhshan	Badahšan			AF.BD	...
2	3	AFG	Afghanistan		AFG.1_1	Badakhshan	Badahšan			AF.BD	...
3	4	AFG	Afghanistan		AFG.1_1	Badakhshan	Badahšan			AF.BD	...
4	5	AFG	Afghanistan		AFG.1_1	Badakhshan	Badahšan			AF.BD	...
...	...	...	...	...	...	...	...	...	...	...	...
356503	356504	ZWE	Zimbabwe		ZWE.10_1	Midlands			ZW-MI	ZW.MI	...
356504	356505	ZWE	Zimbabwe		ZWE.10_1	Midlands			ZW-MI	ZW.MI	...
356505	356506	ZWE	Zimbabwe		ZWE.10_1	Midlands			ZW-MI	ZW.MI	...
356506	356507	ZWE	Zimbabwe		ZWE.10_1	Midlands			ZW-MI	ZW.MI	...
356507	356508	ZWE	Zimbabwe		ZWE.10_1	Midlands			ZW-MI	ZW.MI	...

356508 rows × 53 columns

数据库中每条记录都是一个最小区划的多边形（最后一列geometry中的MULTIPOLYGON类就是通过大量经纬度点描绘的多边形），包含多层区划，如NAME_0一般代表国家（并没有对应说明type，不绝对代表国家），而NAME_1常代表省（ENGTYPE: Province）。

```
china = gamd_file[gamd_file["NAME_0"] == "China"]
del gamd_file
china
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

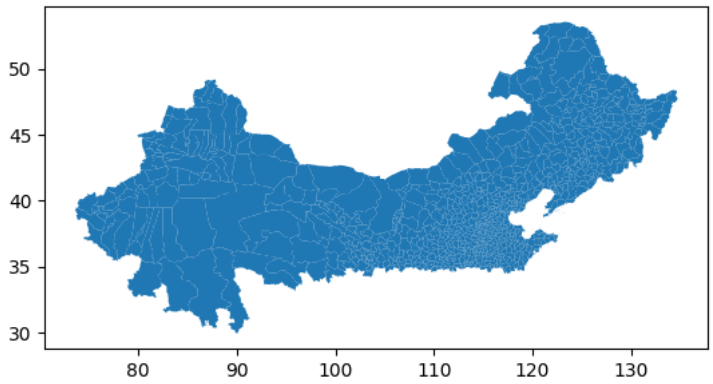
	UID	GID_0	NAME_0	VARNAME_0	GID_1	NAME_1	VARNAME_1	NL_NAME_1	ISO_1	HASC_1	...	ENGTYPE_5	GOVERNEDBY
42435	42436	CHN	China		CHN.1_1	Anhui	Ānhuī	安徽|安徽	CN-AH	CN.AH	...		
42436	42437	CHN	China		CHN.1_1	Anhui	Ānhuī	安徽|安徽	CN-AH	CN.AH	...		
42437	42438	CHN	China		CHN.1_1	Anhui	Ānhuī	安徽|安徽	CN-AH	CN.AH	...		
42438	42439	CHN	China		CHN.1_1	Anhui	Ānhuī	安徽|安徽	CN-AH	CN.AH	...		
42439	42440	CHN	China		CHN.1_1	Anhui	Ānhuī	安徽|安徽	CN-AH	CN.AH	...		
...	...	...	...	...	...	...	...	...	...	...	...	...	...
44865	44866	CHN	China		CHN.31_1	Zhejiang	Zhèjiāng	浙江	CN-ZJ	CN.ZJ	...		
44866	44867	CHN	China		CHN.31_1	Zhejiang	Zhèjiāng	浙江	CN-ZJ	CN.ZJ	...		
44867	44868	CHN	China		CHN.31_1	Zhejiang	Zhèjiāng	浙江	CN-ZJ	CN.ZJ	...		
44868	44869	CHN	China		CHN.31_1	Zhejiang	Zhèjiāng	浙江	CN-ZJ	CN.ZJ	...		
44869	44870	CHN	China		CHN.31_1	Zhejiang	Zhèjiāng	浙江	CN-ZJ	CN.ZJ	...		

2435 rows × 53 columns

cx索引器可以也只可以用于索引经纬度范围，只要该区划选取的矩形区域有重叠就会被选取，我觉得可能判断某个经纬度点属于哪个区划会更重要一点（?

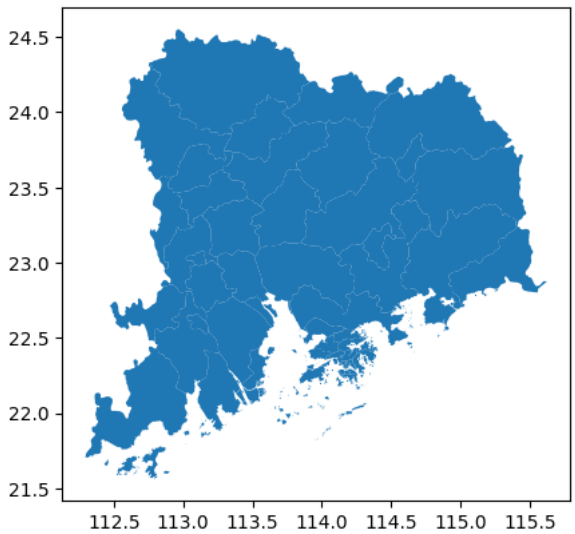
```
china.cx[:, 35:].plot()
```

```
<AxesSubplot:>
```




```
china.cx[113: 115, 22: 24].plot()
```

```
<AxesSubplot:>
```




```
# 东莞区划的多边形，顺便一提东莞最细就到整个东莞的区划，广州的话还分为了几个区
china[china["NAME_2"] == "Dongguan"]["geometry"]
```

```
42706    MULTIPOLYGON (((114.12109 23.05014, 114.12822 ...
Name: geometry, dtype: geometry
```

```
china[china["NAME_2"] == "Dongguan"]
```

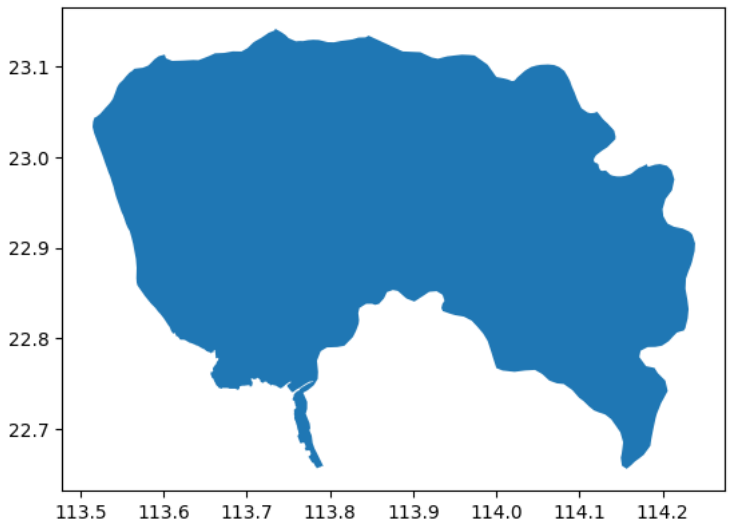
```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	UID	GID_0	NAME_0	VARNAME_0	GID_1	NAME_1	VARNAME_1	NL_NAME_1	ISO_1	HASC_1	...	ENGTYPE_5	GOVERNEDBY
42706	42707	CHN	China		CHN.6_1	Guangdong	Guǎngdōng	廣東|广东	CN-GD	CN.GD	...		

1 rows × 53 columns

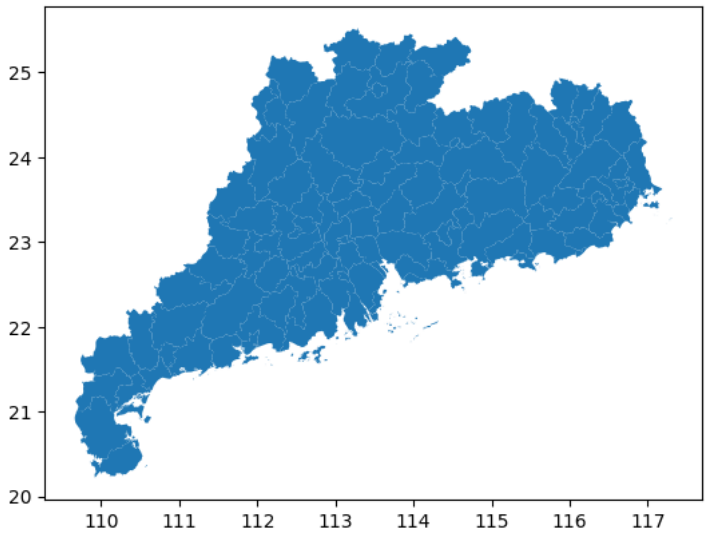
```
# 单点索引办法：contains，适用于单列，即GeoSeries

from shapely.geometry import Point
test = Point(113.9, 22.9)

# 单点索引到东莞市
china[china["NAME_2"] == china[china["geometry"].contains(test)]["NAME_2"].iloc[0]].plot()
```

```
<AxesSubplot:>
```




```
# 单点索引到整个广东
china[china["NAME_1"] == china[china["geometry"].contains(test)]["NAME_1"].iloc[0]].plot()
```

```
<AxesSubplot:>
```


25
24
23
22
21
20
110
111
112
113
114
115
116
117