Pandas的DataFrame和Series，在matplotlib基础上封装了一个简易的绘图函数, 使得我们在数据处理过程中方便可视化查看结果。

- 好处: 方便快捷的可视化的方式洞察数据， 覆盖常用图标类型
- 不足: 不如Matplotlib灵活，仅仅看下分布情况，基本是足满足日常使用

Pandas 对 Matplotlib 绘图软件包的基础上单独封装了一个plot()接口，通过调用该接口可以实现常用的绘图操作

Pandas 之所以能够实现了数据可视化，主要利用了 Matplotlib 库的 plot() 方法，它对 plot() 方法做了简单的封装，因此您可以直接调用该接口。

下面看一组简单的示例：

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

```
#创建包含时间序列的数据
data = np.random.randn(8,4)
data
```

```
array([[-1.65752269, -0.44660288,  1.16489096,  0.91876119],
       [-0.2693576 ,  0.24650992,  0.07003693, -0.58011073],
       [-0.82313854,  1.12339681,  0.06295753, -0.08929852],
       [ 0.46882122,  0.82677928,  0.34750223, -1.18911339],
       [ 1.33212735,  1.35856265,  1.10181229,  0.56089559],
       [ 1.16586359,  0.37455372, -0.09985598, -0.6449436 ],
       [-0.04778479,  0.16956326,  1.70147546,  1.32642704],
       [ 1.84492414,  0.77921013,  1.18669448, -0.26725438]])
```

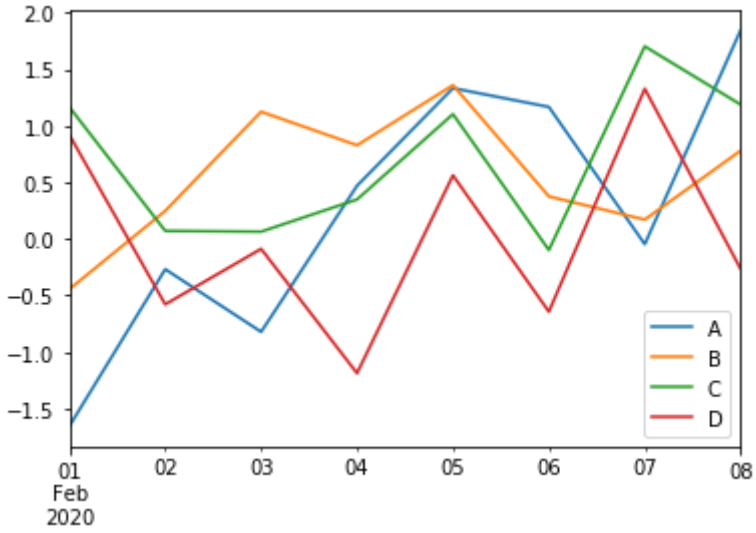
```
df = pd.DataFrame(data,index=pd.date_range('2/1/2020',periods=8),
columns=list('ABCD'))
df
```

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

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

	A	B	C	D
2020-02-01	-1.657523	-0.446603	1.164891	0.918761
2020-02-02	-0.269358	0.246510	0.070037	-0.580111
2020-02-03	-0.823139	1.123397	0.062958	-0.089299
2020-02-04	0.468821	0.826779	0.347502	-1.189113
2020-02-05	1.332127	1.358563	1.101812	0.560896
2020-02-06	1.165864	0.374554	-0.099856	-0.644944
2020-02-07	-0.047785	0.169563	1.701475	1.326427
2020-02-08	1.844924	0.779210	1.186694	-0.267254

```
df.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e1834e6648>
```

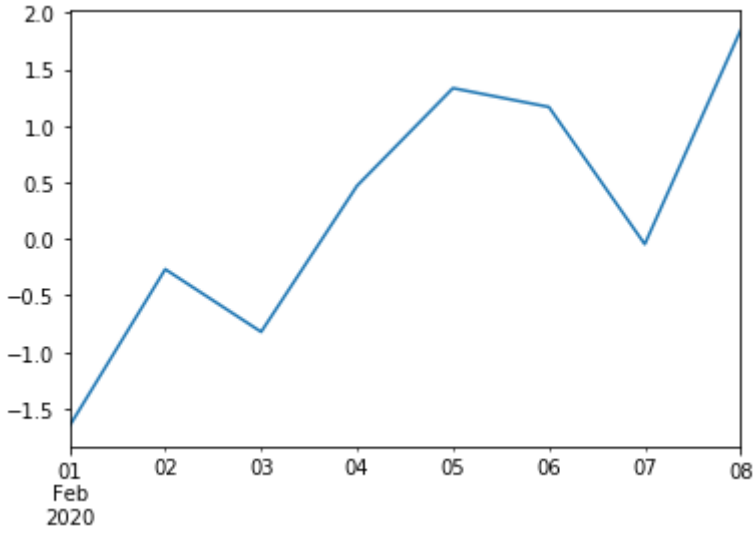



```
df["A"]
```

```
2020-02-01   -1.657523
2020-02-02   -0.269358
2020-02-03   -0.823139
2020-02-04    0.468821
2020-02-05    1.332127
2020-02-06    1.165864
2020-02-07   -0.047785
2020-02-08    1.844924
Freq: D, Name: A, dtype: float64
```

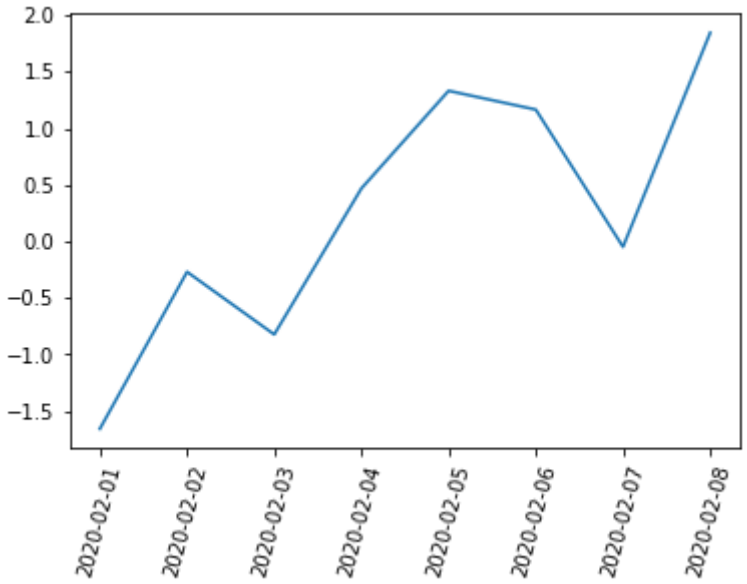
```
df["A"].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e183ce35c8>
```




```
plt.plot(df.index,df["A"])
plt.xticks(rotation=75)
```

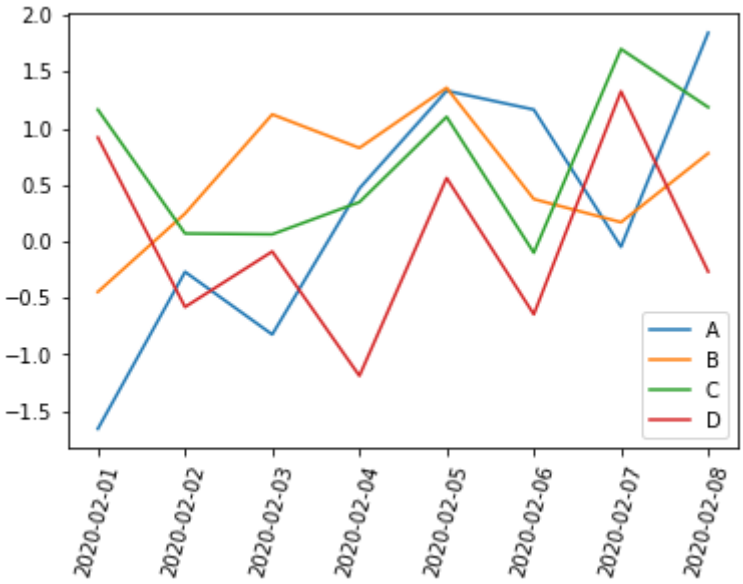
```
([0, 1, 2, 3, 4, 5, 6, 7], <a list of 8 Text xticklabel objects>)
```




```
df.index = df.index.astype(str)
```

```
#rot参数就是rotation缩写
df.plot(rot=75)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e183e88348>
```



除了使用默认的线条绘图外，您还可以使用其他绘图方式，如下所示：

```
柱状图：bar() 或 barh()
直方图：hist()
箱状箱：box()
区域图：area()
散点图：scatter()
```

通过关键字参数kind可以把上述方法传递给 plot()。

柱状图

创建一个柱状图，如下所示：

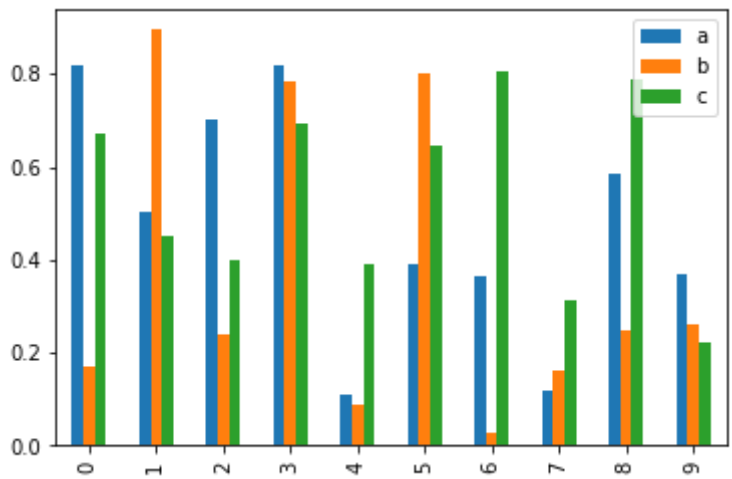
```
df = pd.DataFrame(np.random.rand(10,3),columns=['a','b','c'])
df
```

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

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

	a	b	c
0	0.820065	0.172410	0.672101
1	0.505018	0.894733	0.451726
2	0.701287	0.239377	0.398238
3	0.816350	0.785743	0.695153
4	0.108015	0.087210	0.388533
5	0.389367	0.803135	0.645788
6	0.363083	0.027328	0.805431
7	0.118767	0.161020	0.313948
8	0.585794	0.247668	0.789531
9	0.367779	0.259998	0.220597

```

#或使用df.plot(kind="bar")
#df.plot.bar()
df.plot(kind="bar")
```

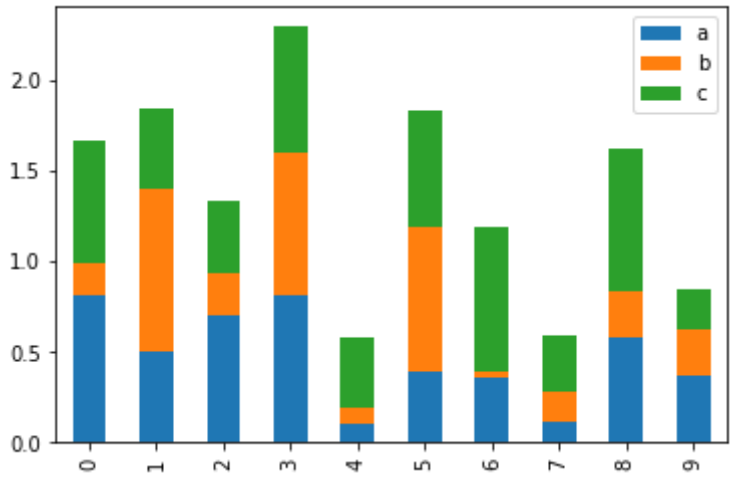
```
<matplotlib.axes._subplots.AxesSubplot at 0x1e184165748>
```



通过设置参数stacked=True可以生成柱状堆叠图，示例如下：

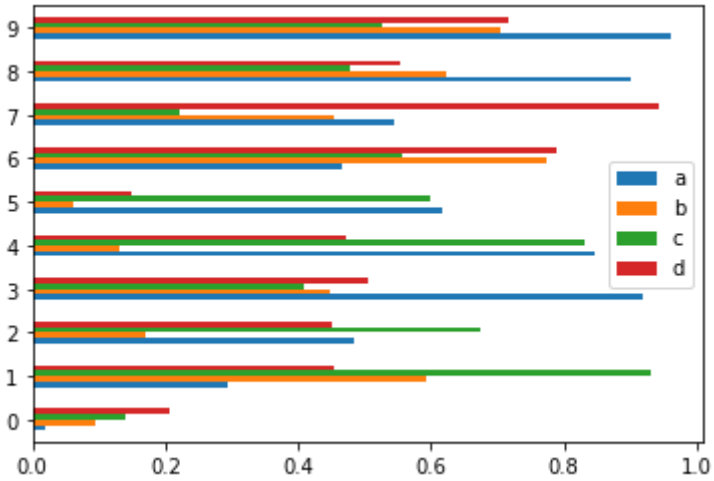
```
#df.plot.bar()
#df.plot(kind="bar",stacked=True)
#或者使用df.plot.bar(stacked="True")
df.plot.bar(stacked="True")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e18415ff48>
```



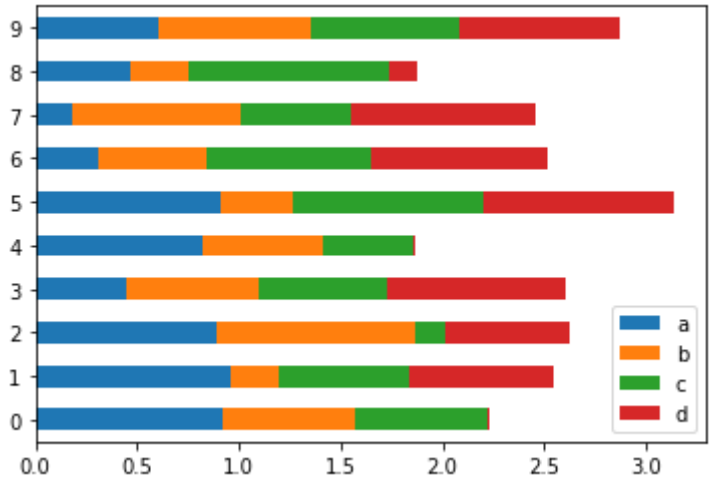
如果要绘制水平柱状图，您可以使用以下方法：

```
df = pd.DataFrame(np.random.rand(10,4),columns=['a','b','c','d'])
#print(df)
df.plot.barh()
#df.plot.barh(stacked=True)
plt.show()
```




```
df.plot.barh(stacked=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e185399c48>
```



直方图

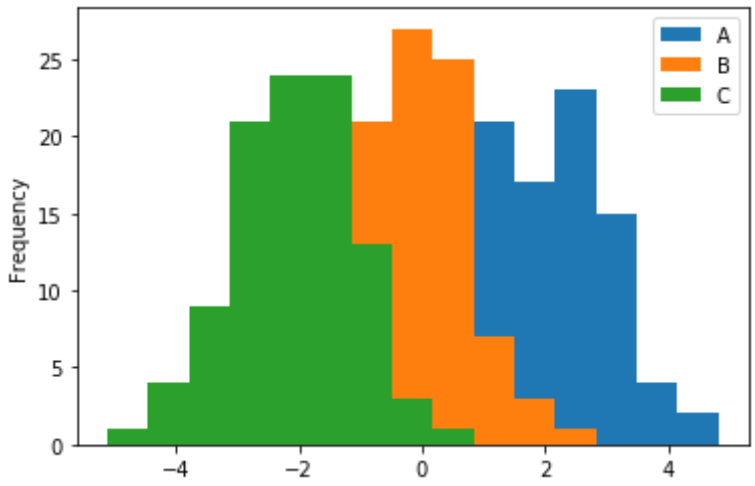
plot.hist() 可以实现绘制直方图，并且它还可以指定 bins（构成直方图的箱数）。

```
df = pd.DataFrame({'A':np.random.randn(100)+2,
                   'B':np.random.randn(100),
                   'C': np.random.randn(100)-2},
                  )
print(df)

```

```
           A         B         C
0   1.857554 -0.171816 -3.873587
1   0.206073  0.289319 -2.866064
2   2.681510  0.390182 -2.749068
3   1.747893  0.627839 -2.221596
4   0.394566 -1.182312 -1.695850
..       ...       ...       ...
95  0.588466 -1.726702 -3.516482
96  1.470252  0.094586 -2.671825
97  1.094508  1.216310 -3.433163
98  2.095755 -0.458329 -0.880253
99  0.605104  1.671117 -0.631358

[100 rows x 3 columns]
```

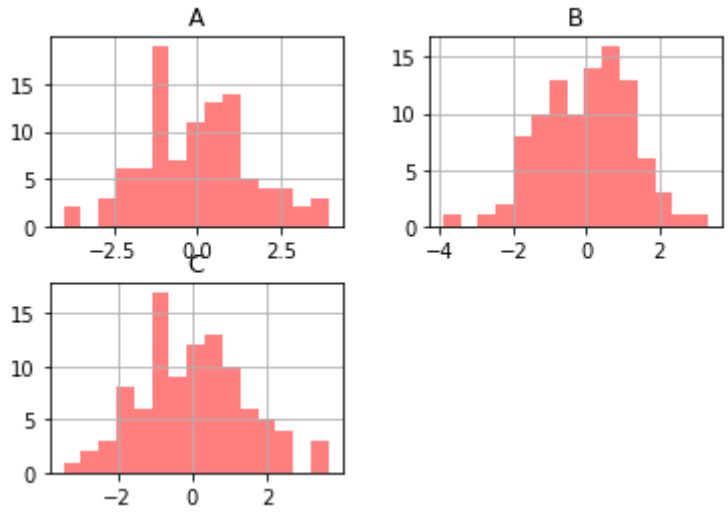
```
#指定箱数为15
df.plot.hist(bins=15)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e1857766c8>
```



给每一列数据都绘制一个直方图，需要使以下方法

```
df.diff().hist(color="r",alpha=0.5,bins=15)
plt.show()
```

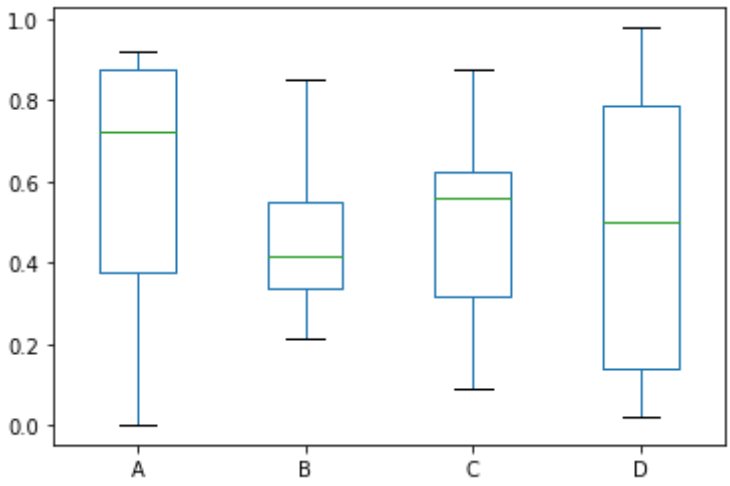


箱型图

通过调用 Series.box.plot() 、DataFrame.box.plot() 或者 DataFrame.boxplot() 方法来绘制箱型图，它将每一列数据的分布情况，以可视化的图像展现出来。

```
df = pd.DataFrame(np.random.rand(10, 4), columns=['A', 'B', 'C', 'D'])
df.plot.box()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e185aeb448>
```

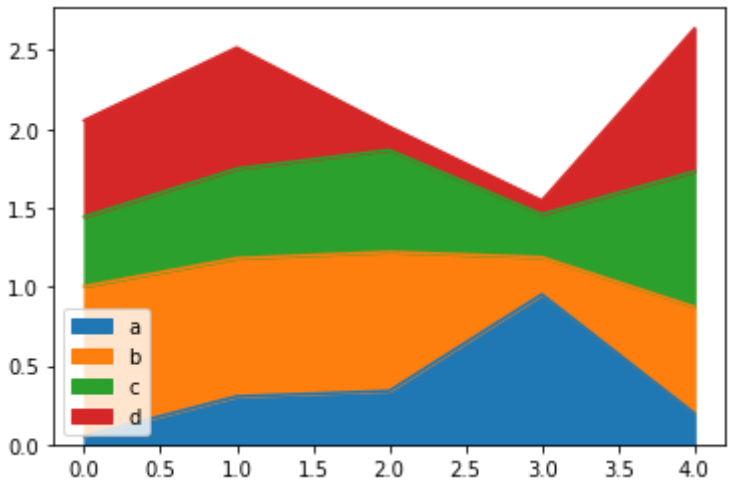


区域图

您可以使用 Series.plot.area() 或 DataFrame.plot.area() 方法来绘制区域图。

```
df = pd.DataFrame(np.random.rand(5, 4), columns=['a', 'b', 'c', 'd'])
df.plot.area()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e183074b88>
```

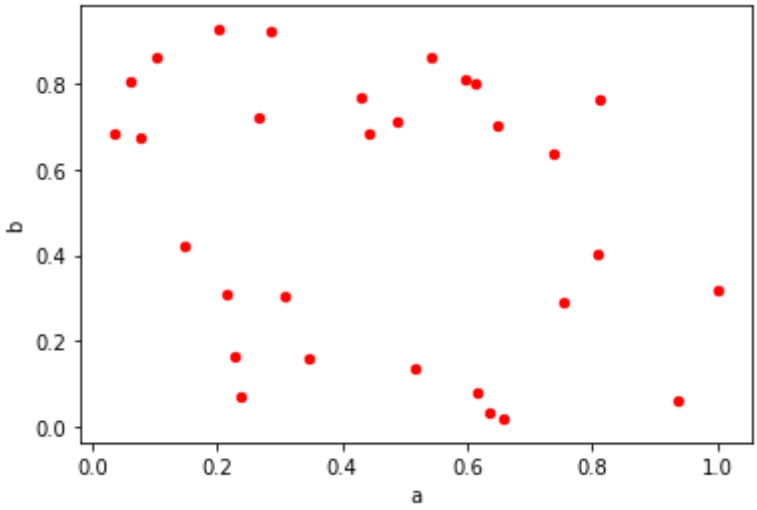


散点图

使用 DataFrame.plot.scatter() 方法来绘制散点图，如下所示：

```
data = np.random.rand(30, 4)
df = pd.DataFrame(data, columns=['a', 'b', 'c', 'd'])
#print(df)
df.plot.scatter(x='a',y='b',c='r')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e185c1fc48>
```



饼状图

饼状图可以通过 DataFrame.plot.pie() 方法来绘制。示例如下：

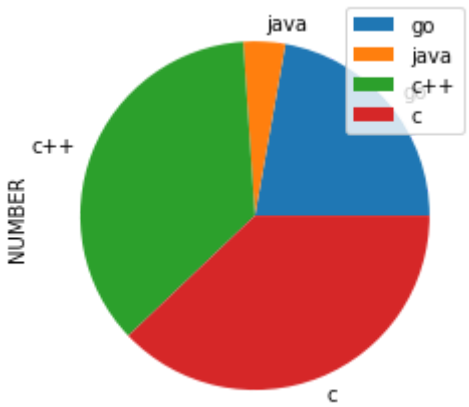
```
df = pd.DataFrame(np.random.rand(4), index=['go', 'java', 'c++', 'c'],
columns=['NUMBER'])
df
```

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

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

	NUMBER
go	0.524619
java	0.091596
c++	0.851322
c	0.895797

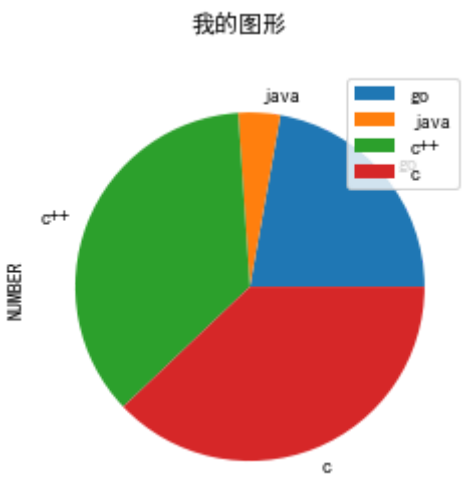
```
# 要么指定y对应的列,要么设置subplots=True
df.plot.pie(y='NUMBER')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e185860908>
```



```python
#需要显示中文,还需要设置字体
# 设置中文:
# 设置中文字体
plt.rcParams['font.sans-serif'] = ['SimHei']
# 中文负号
plt.rcParams['axes.unicode_minus'] = False

# # 设置分别率 为100
# plt.rcParams['figure.dpi'] = 100
# # 设置大小
# plt.rcParams['figure.figsize'] = (10,3)
df.plot.pie(subplots="True",title="我的图形")
```

```
array([<matplotlib.axes._subplots.AxesSubplot object at
0x000001E18600DB48>],
      dtype=object)
```



DataFrame.plot详解:

```
DataFrame.plot(x=None, y=None, kind='line', ax=None,
subplots=False,
               sharex=None, sharey=False,
layout=None,figsize=None,
               use_index=True, title=None, grid=None, legend=True,
```

```
                style=None, logx=False, logy=False, loglog=False,
                xticks=None, yticks=None, xlim=None, ylim=None,
rot=None,
                xerr=None,secondary_y=False, sort_columns=False,
**kwds)
```

- x : label or position, default None#指数据框列的标签或位置参数
- y : label or position, default None
- kind : str
  'line' : line plot (default)#折线图
  'bar' : vertical bar plot#条形图
  'barh' : horizontal bar plot#横向条形图
  'hist' : histogram#柱状图
  'box' : boxplot#箱线图
  'kde' : Kernel Density Estimation plot#Kernel 的密度估计图，主要对柱状图添加Kernel 概率密度线
  'area' : area plot#区域图
  'pie' : pie plot#饼图
  'scatter' : scatter plot#散点图 需要传入columns方向的索引
- ax : matplotlib axes object, default None#**子图(axes, 也可以理解成坐标轴) 要在其上进行绘制的matplotlib subplot对象。如果没有设置，则使用当前matplotlib subplot**其中，变量和函数通过改变figure和axes中的元素（例如：title,label,点和线等等）一起描述figure和axes，也就是在画布上绘图。
- subplots : boolean, default False#判断图片中是否有子图
- sharex : boolean, default True if ax is None else False#如果有子图，子图共x轴刻度，标签
- sharey : boolean, default False#如果有子图，子图共y轴刻度，标签
- layout : tuple (optional)#子图的行列布局 (rows, columns) for the layout of subplots
- figsize : a tuple (width, height) in inches#图片尺寸大小
- use_index : boolean, default True#默认用索引做x轴
- title : string#图片的标题用字符串
- grid : boolean, default None (matlab style default)#图片是否有网格
- legend : False/True/'reverse'#子图的图例，添加一个subplot图例(默认为True)
- style : list or dict#对每列折线图设置线的类型
- logx : boolean, default False#设置x轴刻度是否取对数
- xticks : sequence#设置x轴刻度值，序列形式（比如列表）
- yticks : sequence#设置y轴刻度，序列形式（比如列表）

- `xlim` : 2-tuple/list#设置坐标轴的范围，列表或元组形式
- `ylim` : 2-tuple/list
- `rot` : int, default None#设置轴标签（轴刻度）的显示旋转度数
- `fontsize` : int, default None#设置轴刻度的字体大小
- `colormap` : str or matplotlib colormap object, default None#设置图的区域颜色
- `position` : float
- `table` : boolean, Series or DataFrame, default False #如果为正，则选择DataFrame类型的数据并且转换匹配matplotlib的布局。
- `stacked` : boolean, default False in line and # 为True创建叠状图
- `sort_columns` : boolean, default False # 以字母表顺序绘制各列，默认使用前列顺序

```
1
```