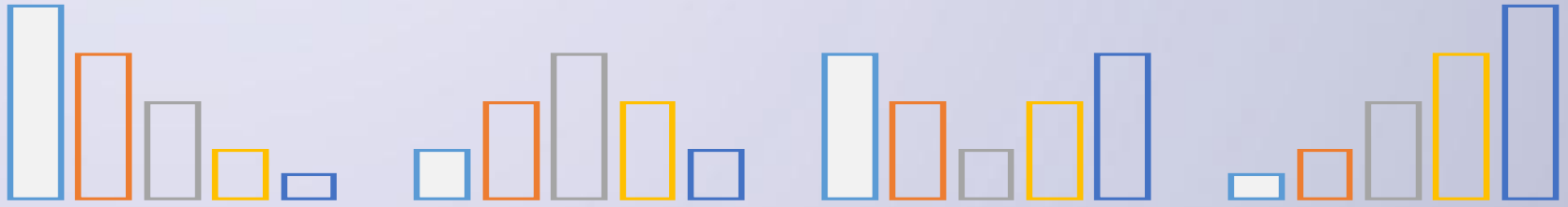
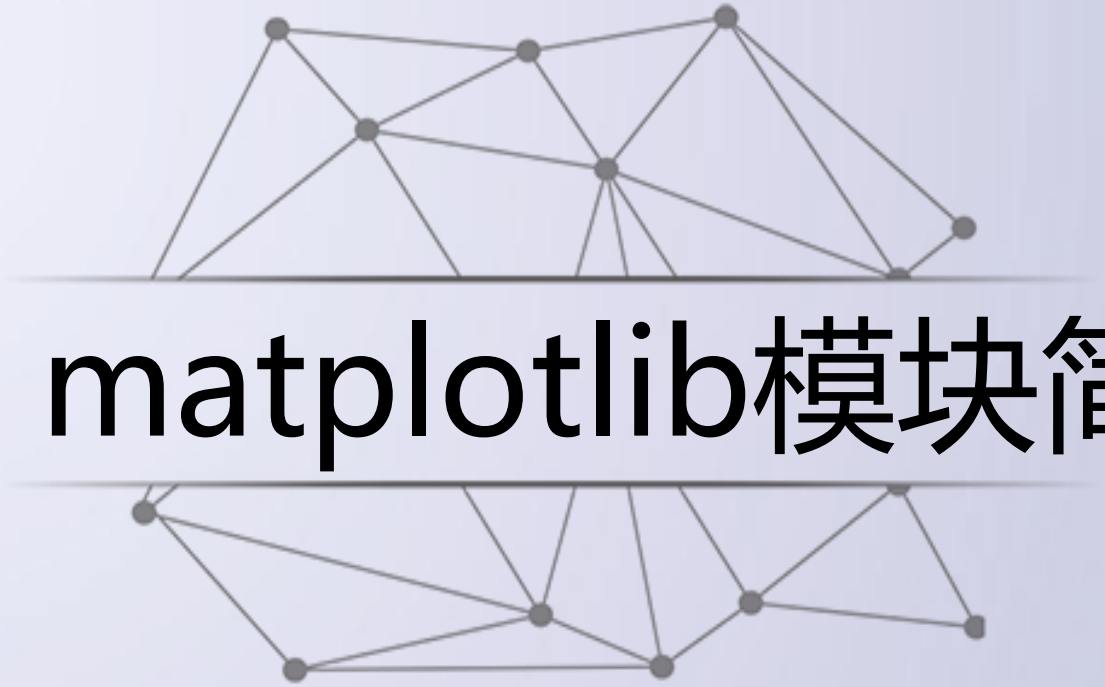
第7章 matplotlib与数据可视化



内容

- matplotlib模块简介
- matplotlib模块绘图要点
- 实例绘图分析



matplotlib模块简介

matplotlib模块简介

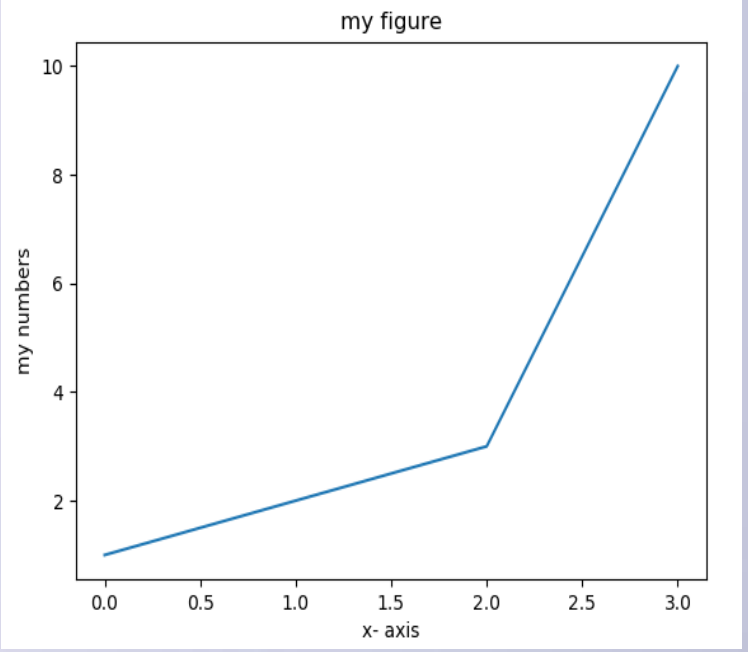
- 基本可视化功能
- 强大、可靠的可视化工具，标杆地位
- 对于标准的绘图工作，易理解
- 进行更复杂的绘图和自定义，很灵活
- 与Numpy及其提供的数据结构紧密相连
- 2D绘图：简单情形到具有两种刻度或不同子图的较高级图形；
- 典型的金融图表
- 3D绘图

简单情形（一维数据）

主要的绘图函数在子库matplotlib.pyplot中

```
>>>from matplotlib.pyplot import *
>>>plot([1,2,3,10])
>>>xlabel(“x- axis”)
>>>ylabel(“my numbers”)
>>>title(“my figure”)
>>>show()
```

```
>>>import matplotlib.pyplot as plt
```



绘图要点

绘图要点

- pyplot子库中的plot函数是最基础的绘图函数，原则上，需要两组数据（长度相等）
  - x值：包含x坐标（横坐标）的列表或数组
  - y值：包含y坐标（纵坐标）的列表或数组
- 提供大量函数和自定义绘图样式
  - 增加网格plt.grid
  - 操纵坐标轴plt.axis,
  - 坐标轴范围plt.xlim,plt.ylim,
  - 坐标轴标签plt.xlable,plt.ylabel
  - 标题 plt.title
  - 设置图表实例的属性plt.setp

二维数据集绘图

数据集包含多个单独的子集，不同于一维

- 数据集有不同刻度(2个y轴)
- 用不同的方式可视化两组不同的数据（两个单独的子图）

  plt.subplots
- 其他绘图样式（散点图、直方图、箱线图）

  plt.scatter

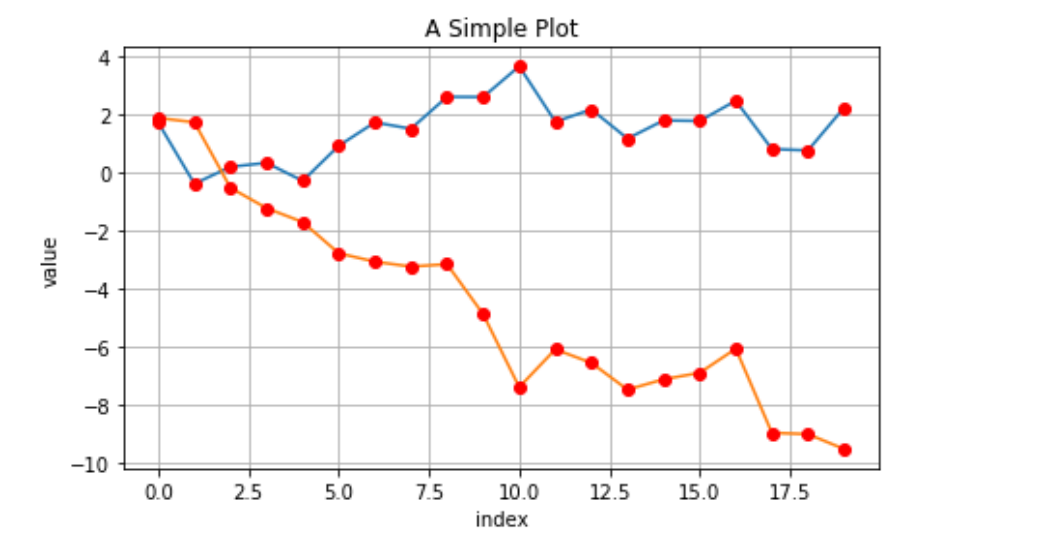
  plt.hist

  plt.boxplot

简单情形

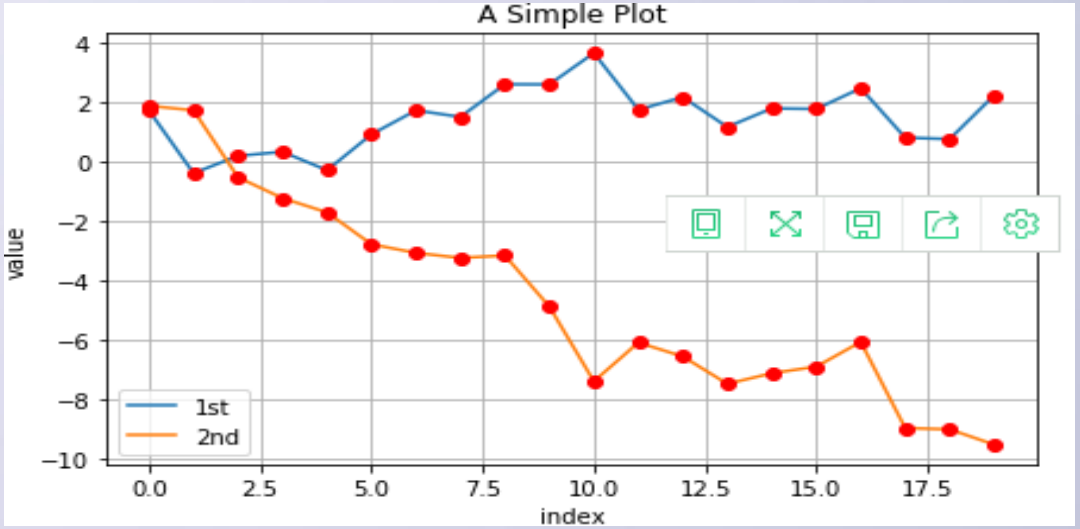
```
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(2000)
y = np.random.standard_normal((20, 2)).cumsum(axis=0)

plt.figure(figsize=(7, 4))
plt.plot(y, lw=1.5)
# plots two lines
plt.plot(y, 'ro')
# plots two dotted lines
plt.grid(True)
plt.axis('tight')
plt.xlabel('index')
plt.ylabel('value')
plt.title('A Simple Plot')
```



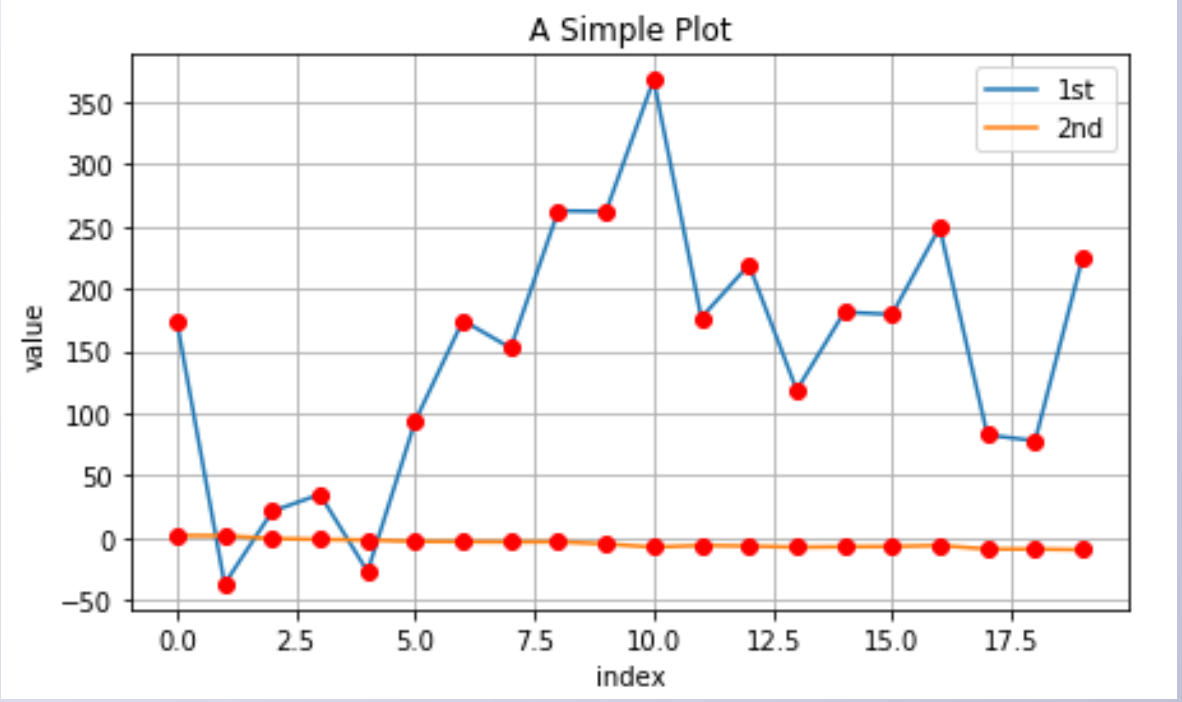
“装饰”

```
plt.figure(figsize=(7, 4))
plt.plot(y[:, 0], lw=1.5, label='1st')
plt.plot(y[:, 1], lw=1.5, label='2nd')
plt.plot(y, 'ro')
plt.grid(True)
plt.legend(loc=0)
plt.axis('tight')
plt.xlabel('index')
plt.ylabel('value')
plt.title('A Simple Plot')
```



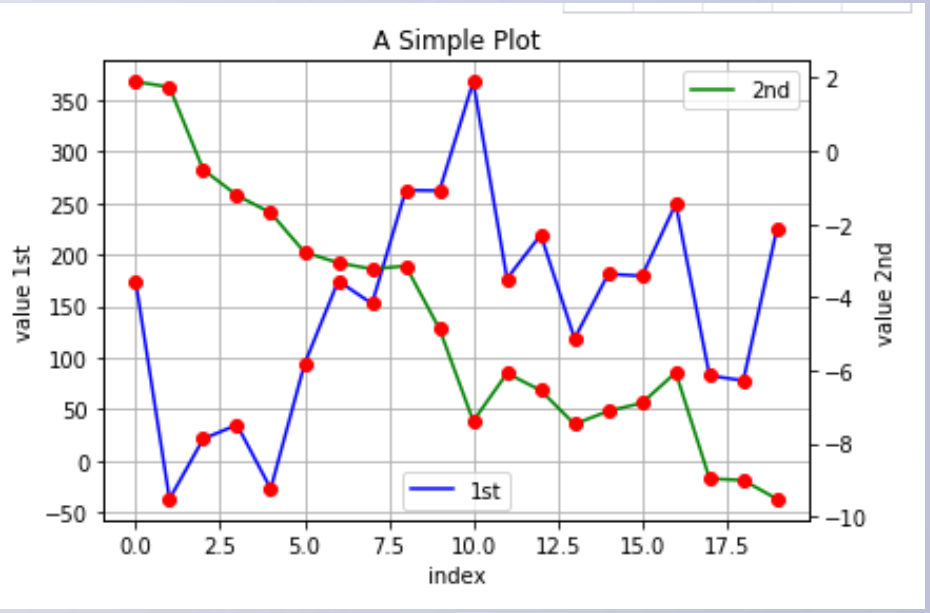
不同的刻度(原始)

```
# different scales
y[:, 0] = y[:, 0] * 100
plt.figure(figsize=(7, 4))
plt.plot(y[:, 0], lw=1.5, label='1st')
plt.plot(y[:, 1], lw=1.5, label='2nd')
plt.plot(y, 'ro')
plt.grid(True)
plt.legend(loc=0)
plt.axis('tight')
plt.xlabel('index')
plt.ylabel('value')
plt.title('A Simple Plot')
```



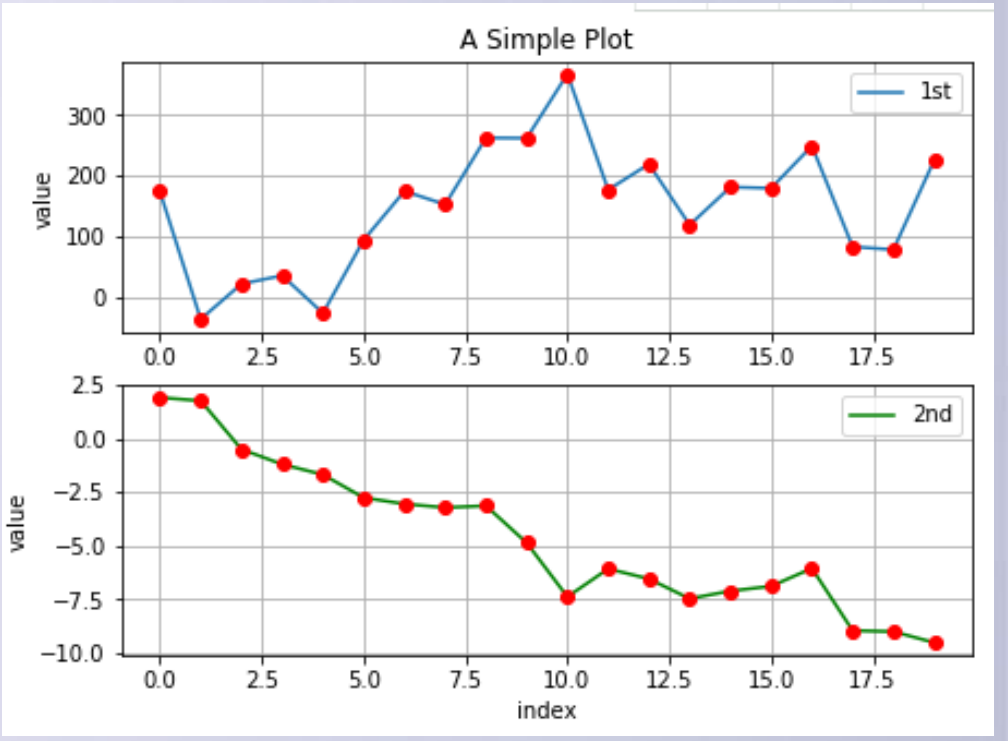
不同的刻度(两个y轴)

```
fig, ax1 = plt.subplots()
plt.plot(y[:, 0], 'b', lw=1.5, label='1st')
plt.plot(y[:, 0], 'ro')
plt.grid(True)
plt.legend(loc=8)
plt.axis('tight')
plt.xlabel('index')
plt.ylabel('value 1st')
plt.title('A Simple Plot')
ax2 = ax1.twinx()
plt.plot(y[:, 1], 'g', lw=1.5, label='2nd')
plt.plot(y[:, 1], 'ro')
plt.legend(loc=0)
plt.ylabel('value 2nd')
```



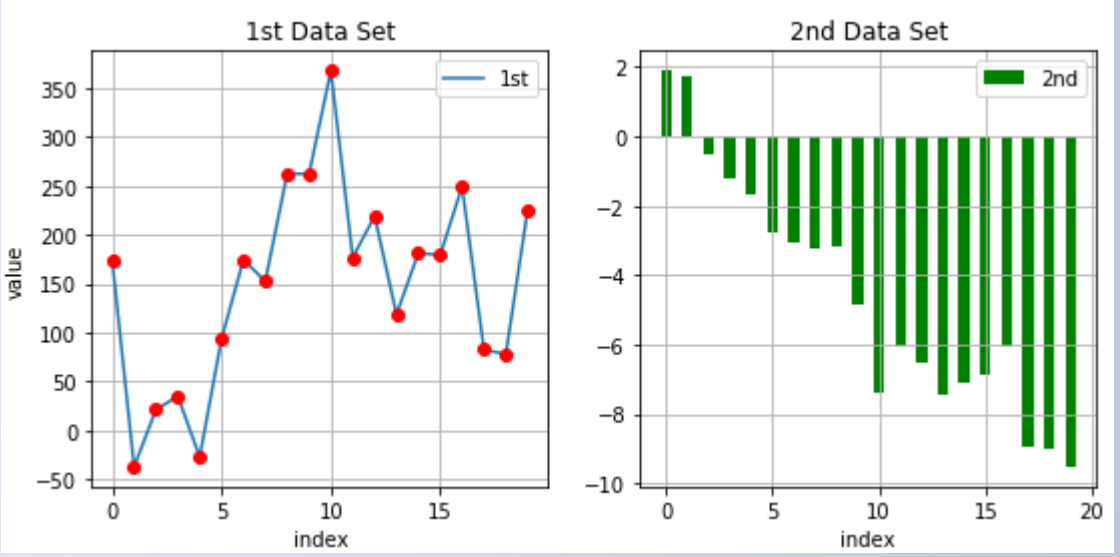
两个单独子图

```
plt.figure(figsize=(7, 5))
plt.subplot(211)
plt.plot(y[:, 0], lw=1.5, label='1st')
plt.plot(y[:, 0], 'ro')
plt.grid(True)
plt.legend(loc=0)
plt.axis('tight')
plt.ylabel('value')
plt.title('A Simple Plot')
plt.subplot(212)
plt.plot(y[:, 1], 'g', lw=1.5, label='2nd')
plt.plot(y[:, 1], 'ro')
plt.grid(True)
plt.legend(loc=0)
plt.axis('tight')
plt.xlabel('index')
plt.ylabel('value')
```



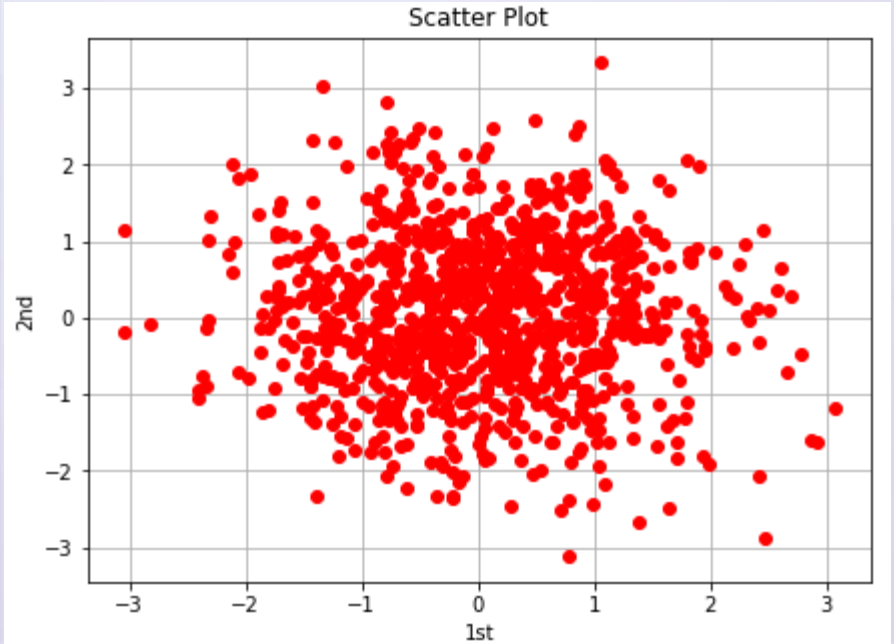
组合线/点子图和柱状子图

```
plt.figure(figsize=(9, 4))
plt.subplot(121)
plt.plot(y[:, 0], lw=1.5, label='1st')
plt.plot(y[:, 0], 'ro')
plt.grid(True)
plt.legend(loc=0)
plt.axis('tight')
plt.xlabel('index')
plt.ylabel('value')
plt.title('1st Data Set')
plt.subplot(122)
plt.bar(np.arange(len(y)), y[:, 1], width=0.5,
        color='g', label='2nd')
plt.grid(True)
plt.legend(loc=0)
plt.axis('tight')
plt.xlabel('index')
plt.title('2nd Data Set')
```

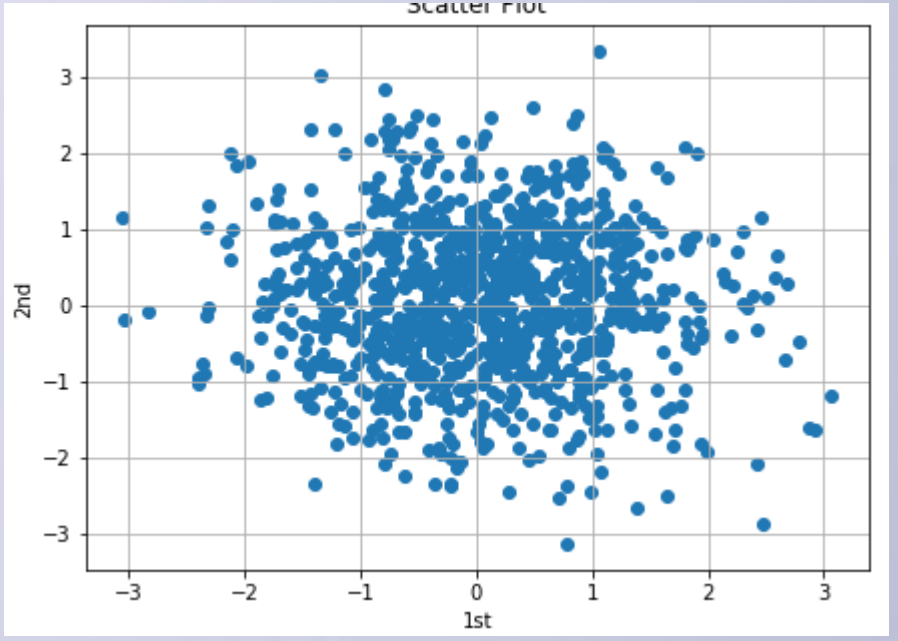


散点图

```
y = np.random.standard_normal((1000, 2))
plt.figure(figsize=(7, 5))
plt.plot(y[:, 0], y[:, 1], 'ro')
plt.grid(True)
plt.xlabel('1st')
plt.ylabel('2nd')
plt.title('Scatter Plot')
```

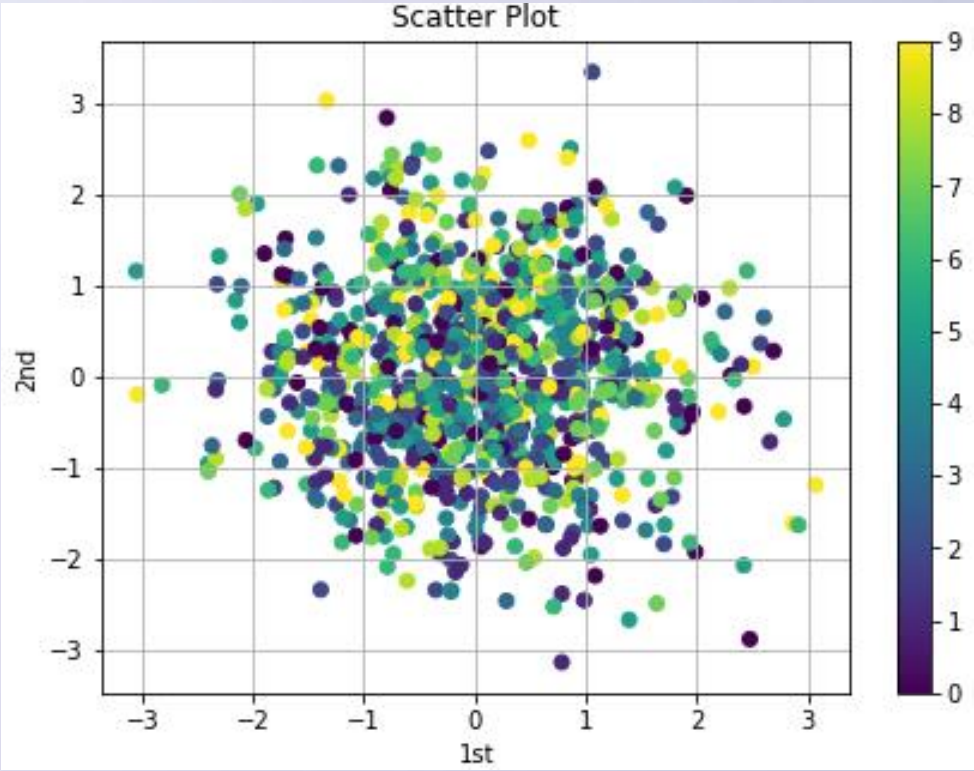



```
plt.figure(figsize=(7, 5))
plt.scatter(y[:, 0], y[:, 1], marker='o')
plt.grid(True)
plt.xlabel('1st')
plt.ylabel('2nd')
plt.title('Scatter Plot')
```



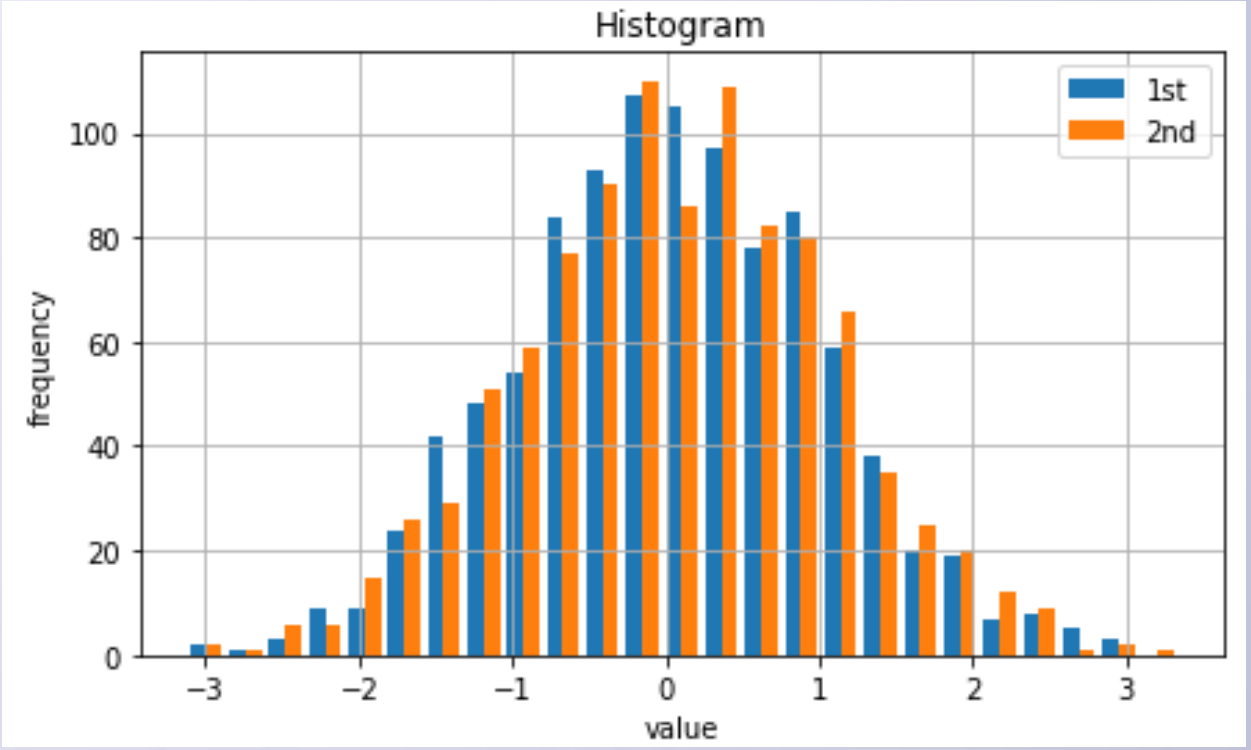
加入第三维的散点图

```
c = np.random.randint(0, 10, len(y))
plt.figure(figsize=(7, 5))
plt.scatter(y[:, 0], y[:, 1], c=c, marker='o')
plt.colorbar()
plt.grid(True)
plt.xlabel('1st')
plt.ylabel('2nd')
plt.title('Scatter Plot')
```



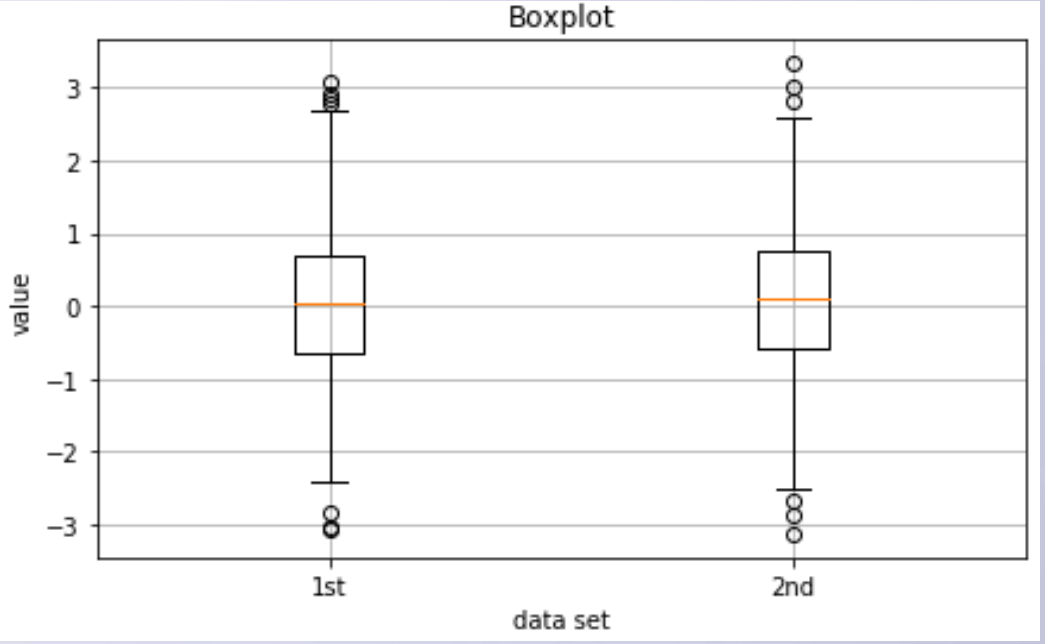
直方图

```
plt.figure(figsize=(7, 4))
plt.hist(y, label=['1st', '2nd'], bins=25)
plt.grid(True)
plt.legend(loc=0)
plt.xlabel('value')
plt.ylabel('frequency')
plt.title('Histogram')
```



箱线图

```
fig, ax = plt.subplots(figsize=(7, 4))
plt.boxplot(y)
plt.grid(True)
plt.setp(ax, xticklabels=['1st', '2nd'])
plt.xlabel('data set')
plt.ylabel('value')
plt.title('Boxplot')
```



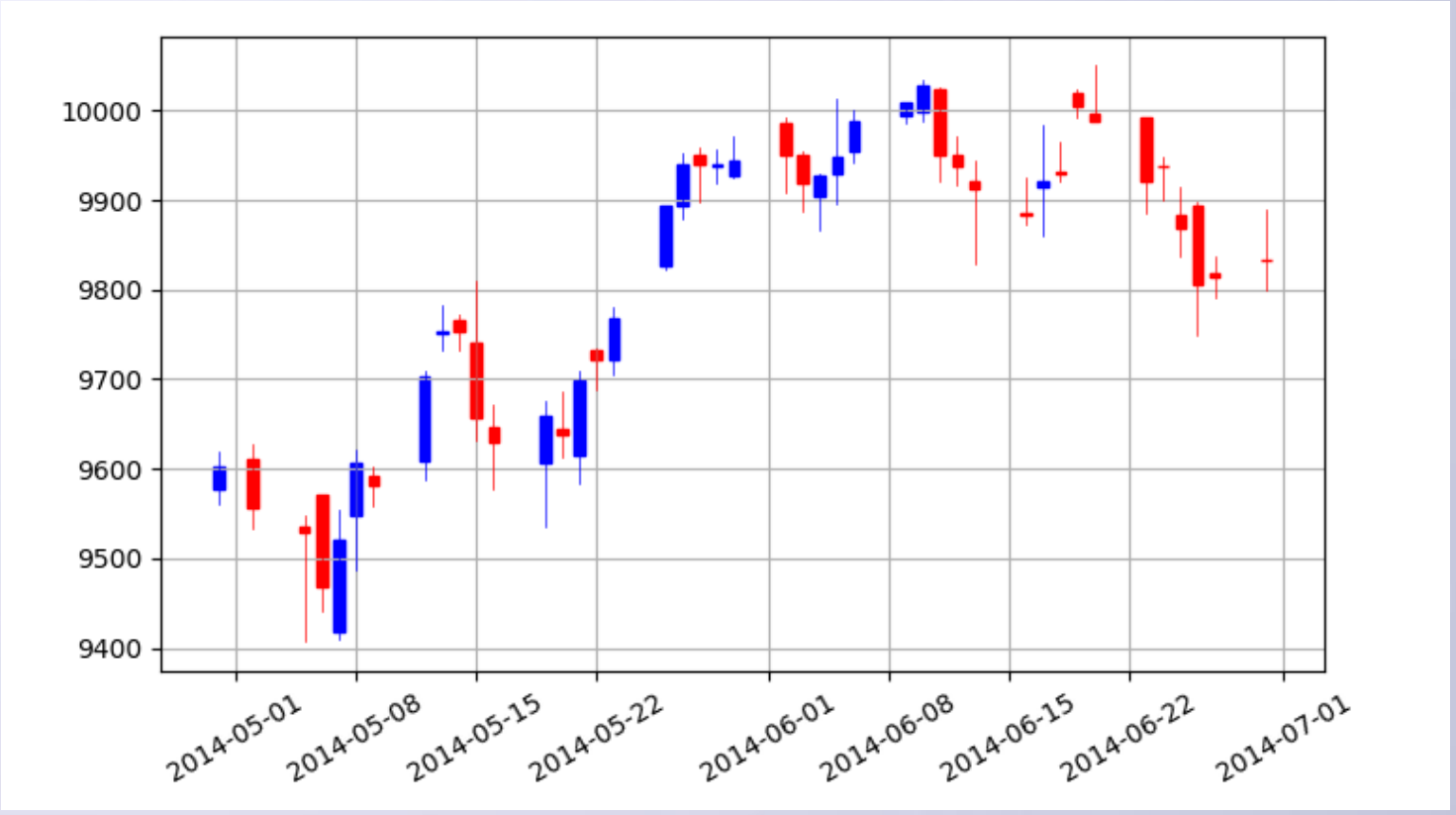
金融图表

- 提供了少数精选的特殊金融图表，这些图表(如烛状图)主要用于可视化历史价格数据或者类似的金融时间序列数据
- 下载数据模块（需要安装pandas_datareader）
- 绘图模块（需要安装mpl_finance）

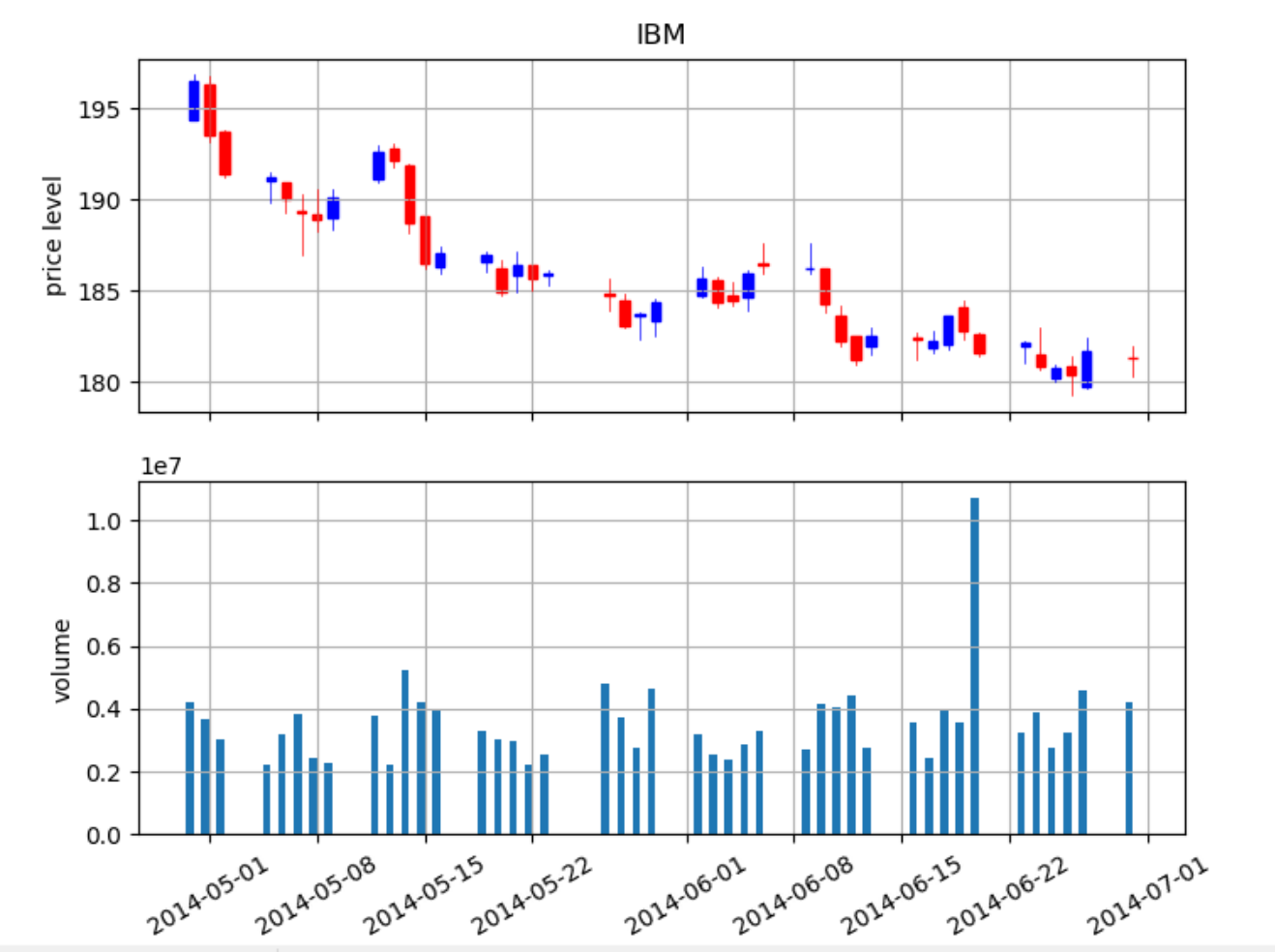
```
import pandas as pd
import mpl_finance as mpf
import matplotlib.dates as mpd
from datetime import datetime
start = datetime(2014, 5, 1)
end = datetime(2014, 6, 30)
import pandas_datareader.data as web
DAX = web.DataReader(name='^GDAXI', data_source='yahoo',
                     start='2014-05-01',end='2014-06-30')
#DAX1=web.get_data_yahoo('^GDAXI',start,end) 与上面等价
#time format
DAX.info()
DAX.tail()
DAX['time']=mpd.date2num(DAX.index)
DAX[:2]
```

```
quotes=DAX[['time','Open','High','Low','Close']].values
fig, ax = plt.subplots(figsize=(8, 5))
fig.subplots_adjust(bottom=0.2)
mpf.candlestick_ohlc(ax, quotes, width=0.6,
                     colorup='b', colordown='r')
plt.grid(True)
ax.xaxis_date()
# dates on the x-axis
ax.autoscale_view()
plt.setp(plt.gca().get_xticklabels(), rotation=30)
```

金融图表：每日正收益由蓝色矩形表示，负收益由红色矩形表示，日期信息在x轴



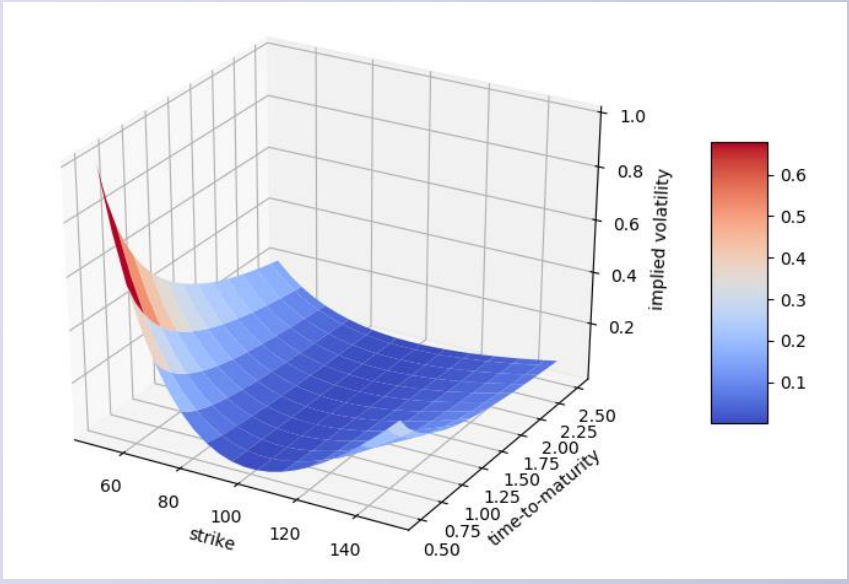
股价数据往往和成交量数据结合。在一张图表中同时提供市场活动相关的信息。基于IBM股价历史数据



```
import matplotlib.pyplot as plt
import matplotlib.dates as mpd
import pandas as pd
import mpl_finance as mpf
from datetime import datetime
start = datetime(2014, 5, 1)
end = datetime(2014, 6, 30)
import pandas_datareader.data as web
data = web.DataReader(name='IBM', data_source='yahoo',
                      start=start,end=end)
data['time']=mpd.date2num(data.index)
quotes = data[['time','Open','High','Low','Close','Volume']].values
fig, (ax1, ax2) = plt.subplots(2, sharex=True, figsize=(8, 6))
mpf.candlestick_ohlc(ax1, quotes, width=0.6, colorup='b', colordown='r')
ax1.set_title('IBM')
ax1.set_ylabel('price level')
ax1.grid(True)
ax1.xaxis_date()
plt.bar(quotes[:, 0] - 0.25, quotes[:, 5], width=0.5)
ax2.set_ylabel('volume')
ax2.grid(True)
ax2.autoscale_view()
plt.setp(plt.gca().get_xticklabels(), rotation=30)
```

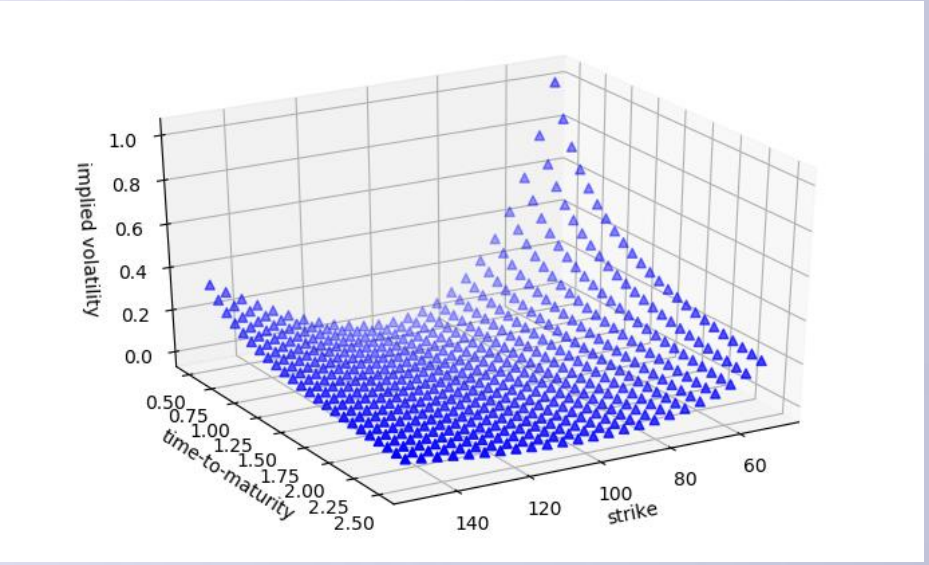
3D绘图

```
strike = np.linspace(50, 150, 24)
ttm = np.linspace(0.5, 2.5, 24)
strike, ttm = np.meshgrid(strike, ttm)
strike[:2]
iv = (strike - 100) ** 2 / (100 * strike) / ttm
# generate fake implied volatilities
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(figsize=(9, 6))
ax = fig.gca(projection='3d')
surf = ax.plot_surface(strike, ttm, iv, rstride=2,
cstride=2,
cmap=plt.cm.coolwarm, linewidth=0.5,antialiased=True)
ax.set_xlabel('strike')
ax.set_ylabel('time-to-maturity')
ax.set_zlabel('implied volatility')
fig.colorbar(surf, shrink=0.5, aspect=5)
```



3D绘图

```
fig = plt.figure(figsize=(8, 5))
ax = fig.add_subplot(111, projection='3d')
ax.view_init(30, 60)
ax.scatter(strike, ttm, iv, zdir='z', s=25,
c='b', marker='^')
ax.set_xlabel('strike')
ax.set_ylabel('time-to-maturity')
ax.set_zlabel('implied volatility')
```

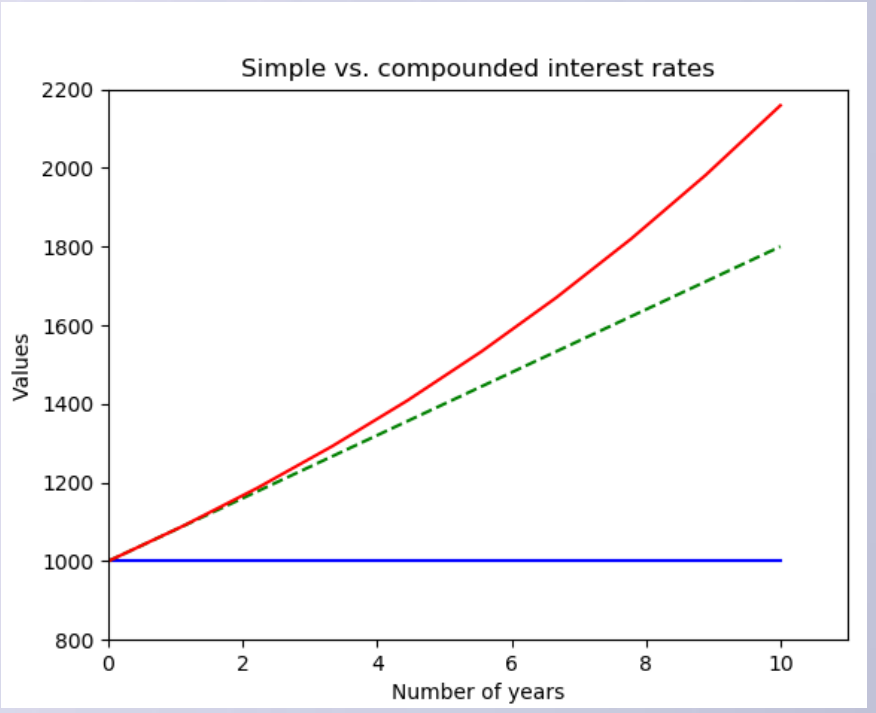


实例绘图分析

例：简单利率和复利利率

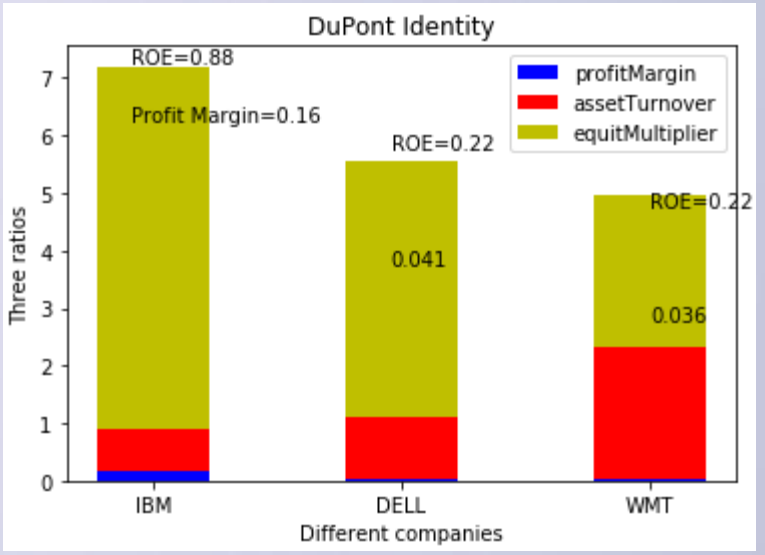
- FV(简单利率)=PV(1+R × n)　　PV:当前值，n:周期数
- FV(复利利率)=PV$(1+R)^n$　　R:单位周期利率
- -------代码区---------

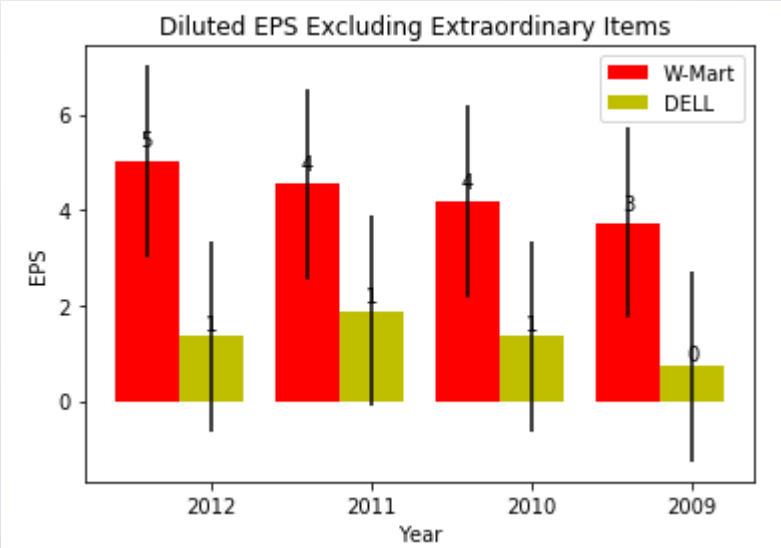
```
import numpy as np
from matplotlib.pyplot import *
from pylab import *
pv=1000
r=0.08
n=10
t=linspace(0,n,n)
y1=np.ones(len(t))*pv # this is a horizontal line
y2=pv*(1+r*t)
y3=pv*(1+r)**t
title('Simple vs. compounded interest rates')
xlabel('Number of years')
ylabel('Values')
xlim(0,11)
ylim(800,2200)
plot(t, y1, 'b-'), plot(t, y2, 'g--'), plot(t, y3, 'r-')
show()
```



其他要素

- 为图形添加文字 figtext(x,y,'文字')
- 杜邦等式（比率分析）毛利率、资产周转率、权益乘数
- 净现值图示曲线
- 有效使用颜色
- 使用不同形状





例题：分散投资

可以把不同股票包含在投资组合里来降低公司特有风险，如下面有两只股票5年里的收益：

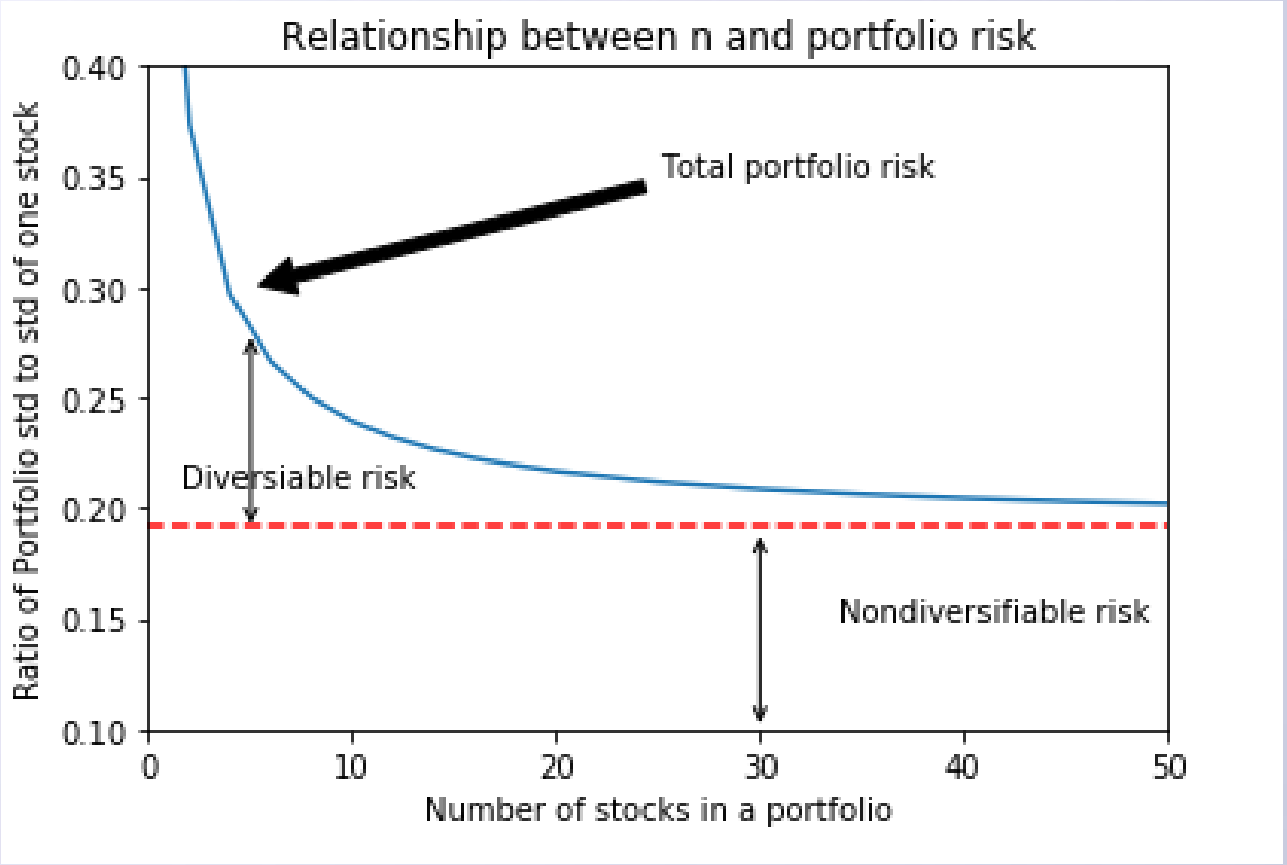
Year	Stock A	Stock B
2009	0.102	0.1062
2010	-0.02	0.23
2011	0.213	0.045
2012	0.12	0.234
2013	0.13	0.113

考虑这两只股票以等权重构成的投资组合，计算这个组合的均值和标准差，与个股比较，并绘图，计算两只股票的相关系数。

多元化的投资组合需要的个股数量分析：

其中:n是投资组合中的股票数，$\overline{\sigma_p}$是投资组合的年收益率标准差

n	$\bar{\sigma}_p$	$\frac{\bar{\sigma}_p}{\bar{\sigma}_1}$	n	$\bar{\sigma}_p$	$\frac{\bar{\sigma}_p}{\bar{\sigma}_1}$
1	49.236	1.00	45	20.316	0.41
2	37.358	0.76	50	20.203	0.41
4	29.687	0.60	75	19.860	0.40
6	26.643	0.54	100	19.686	0.40
8	24.983	0.51	200	19.432	0.39
10	23.932	0.49	300	19.336	0.39
12	23.204	0.47	400	19.292	0.39
14	22.670	0.46	500	19.265	0.39
16	22.261	0.45	600	19.347	0.39
18	21.939	0.45	700	19.233	0.39
20	21.677	0.44	800	19.224	0.39
25	21.196	0.43	900	19.217	0.39
30	20.870	0.42	1000	19.211	0.39
35	20.634	0.42	∞	19.158	0.39
40	20.456	0.42			


Relationship between n and portfolio risk
Ratio of Portfolio std to std of one stock
0.40
0.35
0.30
0.25
0.20
0.15
0.10
Total portfolio risk
Diversiable risk
Nondiversifiable risk
0
10
20
30
40
50
Number of stocks in a portfolio

小结

- matplotlib模块简介
- matplotlib模块绘图要点
- 实例绘图