

# Chapter 8 – Python股票分析

Frank Ziwei Zhang  
School of Finance



上海對外經貿大學  
SHANGHAI UNIVERSITY OF INTERNATIONAL BUSINESS AND ECONOMICS

# Contents

Q1

股票市场

Q2

股票组合

Q3

CPAM 模型

Q4

股价随机过程

Q5

组合绩效评估

## 8.1 股票市场

## 8.1.1 多层次股票市场

---

主板市场

中小企业板市场

创业板市场

科创板市场

新三板市场

四板市场

# 8.1.1 多层次股票市场

## 科创板市场：

2019年1月28日，中国证监会发布了《关于在上海证券交易所设立科创板并试点注册制的实施意见》，标志着科创板进入最后倒计时。科创板将服务于符合国家战略、突破关键核心技术、市场认可度高的科技创新企业，重点支持高新技术产业和战略性新兴产业。同时，科创板在审批机制、上市要求以及交易机制等方面均有重大突破。

- 发行人申请在科创板上市，市值及财务指标只需要符合下列标准中的一项，具体如下：
- 预计市值不低于10亿元，最近两年净利润均为正且累计净利润不低于5000万元，或者预计市值不低于10亿元，最近一年净利润为正且营业收入不低于1亿元；
- 预计市值不低于15亿元，最近一年营业收入不低于2亿元，且最近3年累计研发投入占最近3年累计营业收入的比例不低于15%；
- 预计市值不低于20亿元，最近一年营业收入不低于3亿元，且最近3年经营活动产生的现金流量净额累计不低于1亿元；
- 预计市值不低于30亿元，且最近一年营业收入不低于3亿元；
- 预计市值不低于40亿元，主要业务或产品需经国家有关部门批准，市场空间大，目前已取得阶段性成果；医药行业企业需至少有一项核心产品获准开展二期临床试验；其他符合科创板定位的企业需具备明显的技术优势并满足相应条件。

此外，针对申请的公司是红筹企业或存在表决权差异安排的公司，市值及财务指标则至少需要符合下列标准中的一项：（一）预计市值不低于100亿元；（二）预计市值不低于50亿元，且最近一年营业收入不低于5亿元。

## 8.1.2 主要的股票指数

- 1、上证综合指数
- 2、深证成份股指数
- 3、上证180指数
- 4、上证50指数
- 5、沪深300指数
- 6、中证500指数
- 7、中证800指数
- 8、中证1000指数
- 9、国证2000指数

## 8.1.3 指数的数据下载与可视化

中证指数有限公司的官方网站提供了关于A股股指历史走势数据的下载服务。下面，通过导入已下载的上证综指、深圳成指、中证500和中证800指数在2014年至2018年的日收盘价数据并且通过Python进行可视化（见图8-1），具体的代码如下：

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from pylab import mpl
mpl.rcParams['font.sans-serif'] = ['SimHei']
mpl.rcParams['axes.unicode_minus'] = False
```



```
index_data = pd.read_excel('D:\Zhangzw\Python\Python金融数据分析\RawData\第8章\国内A股主要股指的日收盘数据 (2014-2018) .xlsx', sheet_name='Sheet1', header=0, index_col=0) #导入外部数据
index_data.plot(subplots=True, layout=(2,2), figsize=(10,10), fontsize=13, grid=True)
```

## 8.2 股票组合



## 8.2.1 投资组合的主要变量

### 1、投资组合的预期收益率

关于组合收益率的计算，曾经在第3章讨论NumPy模块的时候简单提到过，现在针对投资组合的预期收益率，给出一般化的计算公式，具体如下：

$$E(R_p) = E\left(\sum_{i=1}^N w_i R_i\right) = \sum_{i=1}^N w_i E(R_i) = [w_1, w_2, \dots, w_N] [E(R_1), E(R_2), \dots, E(R_N)]^T$$

在计算股票收益率的时候，针对第*i*只股票在第*t*个交易日的收益率用式子（8-2）表示，可以将收益率变为连续复利的收益率：

$$R_{it} = \ln \frac{P_{it}}{P_{it-1}}$$

## 8.2.1 投资组合的主要变量

### 2、投资组合的波动率（风险）

在计算投资组合的波动率之前，需要首先计算得到每只股票收益率之间的协方差和相关系数。

首先，考虑由两只股票组成的投资组合收益率的波动率（下文简称“收益波动率”或“波动率”）具体的表达式如下：

$$\sigma_p^2 = w_1^2 \sigma_1^2 + w_2^2 \sigma_2^2 + 2w_1 w_2 \text{Cov}(R_1, R_2) = w_1^2 \sigma_1^2 + w_2^2 \sigma_2^2 + 2w_1 w_2 \rho_{12} \sigma_1 \sigma_2$$

$$\sigma_p = \sqrt{w_1^2 \sigma_1^2 + w_2^2 \sigma_2^2 + 2w_1 w_2 \text{Cov}(R_1, R_2)} = \sqrt{w_1^2 \sigma_1^2 + w_2^2 \sigma_2^2 + 2w_1 w_2 \rho_{12} \sigma_1 \sigma_2}$$

$$w = [w_1, w_2, \dots, w_N], \quad \Sigma = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1N} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{N1} & \sigma_{N2} & \dots & \sigma_N^2 \end{bmatrix}, \quad \sigma_{ij} = \text{Cov}(R_i, R_j)$$

$$\sigma_p = \sqrt{w \Sigma w^T}$$

$$\text{年波动率} = \sqrt{252} \times \text{日动率}$$

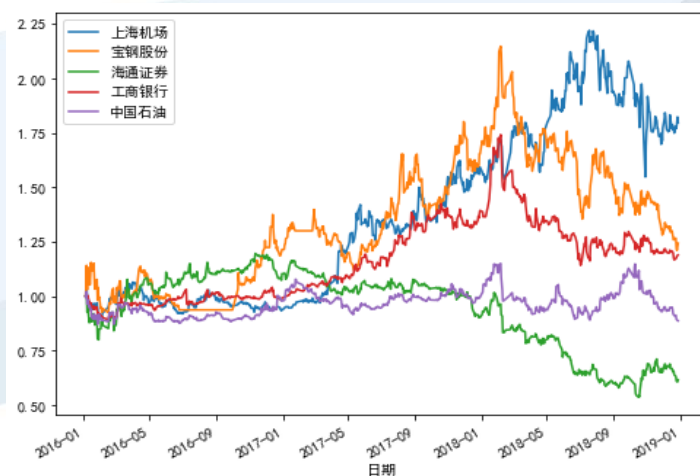
## 8.2.1 投资组合的主要变量

【例8-1】假定投资组合配置了5只A股股票（具体信息见表8-1），数据是2016年至2018年期间的每个2018年期间的每个交易日收盘价格。下面就通过Python计算投资组合的预期收益率和年波动率，具体分为4个步骤。

股票代码	600009	600019	600837	601398	601857
证券简称	上海机场	宝钢股份	海通证券	工商银行	中国石油

第1步：导入股票的收盘价格数据并且进行可视化（见图8-2），具体的代码如下：

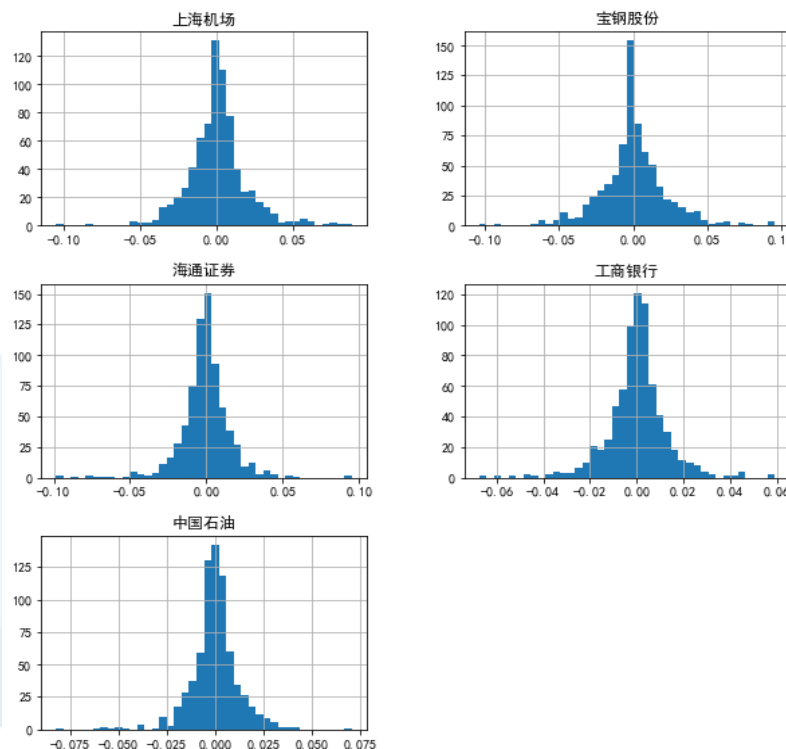
```
data = pd.read_excel('D:\Zhangzw\Python\Python金融数据分析\RawData\第8章\构建投资组合的五只股票数据（2016-2018）.xlsx',sheet_name='Sheet1',header=0,index_col=0) #导入外部数据
(data/data.iloc[0]).plot(figsize=(8,6)) #将股价按首个交易日进行归一处理并可视化
```



## 8.2.1 投资组合的主要变量

第2步：按照前面介绍的式子（8-2）构建这5只股票日收益率的时间序列，同时进行可视化（见图8-3），代码如下：

```
R=np.log(data/data.shift(1)) #按照对数收益率的计算公式得到股票收益率  
R=R.dropna()                #删除缺省的数据  
R.describe()  
R.hist(bins=40,figsize=(10,10)) #将股票收益率按照直方图方式展示
```



## 8.2.1 投资组合的主要变量

第3步：计算每只股票的平均收益率、波动率以及协方差，由于运用的是日数据，因此需要进行年化处理，代码如下：

```
R_mean=R.mean()*252      #计算股票的年化平均收益率
print(R_mean)
```

```
上海机场  0.202051
宝钢股份  0.075045
海通证券  -0.167117
工商银行  0.059691
中国石油  -0.041456
dtype: float64
```

```
R_cov=R.cov()*252      #计算股票的协方差矩阵并且年化处理
print(R_cov)
```

	上海机场	宝钢股份	海通证券	工商银行	中国石油
上海机场	0.091724	0.022705	0.022375	0.014478	0.017292
宝钢股份	0.022705	0.119489	0.042816	0.023992	0.030719
海通证券	0.022375	0.042816	0.072361	0.021051	0.028913
工商银行	0.014478	0.023992	0.021051	0.040094	0.016621
中国石油	0.017292	0.030719	0.028913	0.016621	0.041939

## 8.2.1 投资组合的主要变量

```
R_corr=R.corr()    #计算股票的相关系数矩阵  
print(R_corr)
```

	上海机场	宝钢股份	海通证券	工商银行	中国石油
上海机场	1.000000	0.216880	0.274644	0.238733	0.278804
宝钢股份	0.216880	1.000000	0.460463	0.346625	0.433946
海通证券	0.274644	0.460463	1.000000	0.390828	0.524851
工商银行	0.238733	0.346625	0.390828	1.000000	0.405316
中国石油	0.278804	0.433946	0.524851	0.405316	1.000000

```
R_vol=R.std()*np.sqrt(252)    #计算股票收益率的年化波动率  
print(R_vol)
```

上海机场	0.302861
宝钢股份	0.345671
海通证券	0.269000
工商银行	0.200236
中国石油	0.204791

dtype: float64

## 8.2.1 投资组合的主要变量

第4步：运用前面生成的随机权重数计算投资组合的预期收益率和收益波动率，代码如下：

```
R_port=np.sum(weights*R_mean) #计算投资组合的预期收益率  
print('投资组合的预期收益率: ',round(R_port,4))
```

投资组合的预期收益率： 0.0427

```
vol_port=np.sqrt(np.dot(weights,np.dot(R_cov,weights.T))) #计算投资组合收益波动率  
print('投资组合收益波动率',round(vol_port,4))
```

投资组合收益波动率 0.1957

## 8.2.2 投资组合的有效前沿

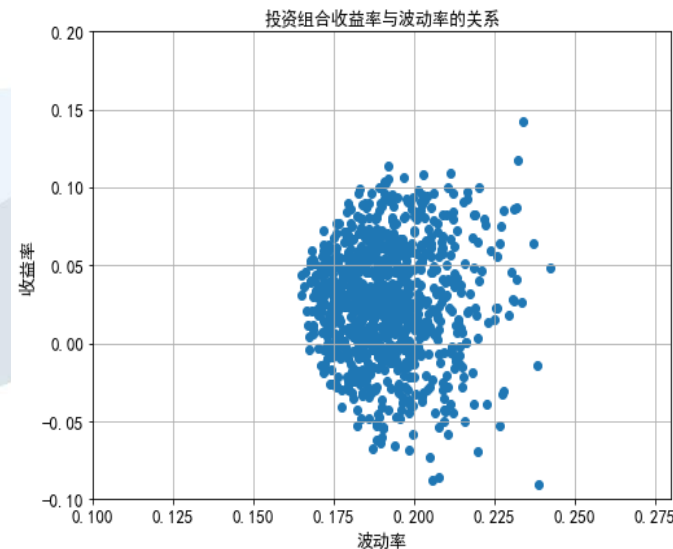
### 1、通过 Python绘制可行集

Python可以非常方便地生成投资组合的权重随机数，然后再根据权重数生成相对应的投资组合预期收益率与收益波动率，并且进行可视化。

【例8-2】沿用前面例8-1的信息，针对投资组合配置的5只股票，运用Python随机生成1000组权重的数组，进而绘制投资组合的可行集（见图84），具体的代码如下：

```
Rp_list=[] #建立一个初始的投资组合收益率数列
Vp_list=[] #建立一个初始的投资组合收益波动率数列
for i in np.arange(1000): #生成1000个不同权重的预期收益率与收益波动率
    x=np.random.random(5)
    weights=x/sum(x)
    Rp_list.append(np.sum(weights*R_mean))
    Vp_list.append(np.sqrt(np.dot(weights,np.dot(R_cov,weights.T))))

plt.figure(figsize=(8,6))
plt.scatter(Vp_list,Rp_list)
plt.xlabel(u'波动率',fontsize =13)
plt.ylabel(u'收益率',fontsize =13,rotation=90)
plt.xticks(fontsize=13)
plt.yticks(fontsize=13)
plt.xlim(0.1,0.28)
plt.ylim(-0.1,0.2)
plt.title(u'投资组合收益率与波动率的关系',fontsize=13)
plt.grid('True')
plt.show()
```





## 8.2.2 投资组合的有效前沿

### 2、通过 Python构建有效前沿

其实，有效前沿就是求解以下这个最优方程式：

$$\min_{w_i} \sigma_p = \min_{w_i} \sqrt{\sum_{i=1}^N \sum_{j=1}^N w_i w_j \text{Cov}(R_i, R_j)}$$

约束条件分别是：

$$\sum_{i=1}^N w_i = 1$$

$$w_i > 0$$

$$E(R_p) = E\left(\sum_{i=1}^N w_i R_i\right) = \text{给定常数}$$

## 8.2.2 投资组合的有效前沿

【例8-3】沿用前面例8-1的信息，同时给定投资组合的预期收益率等于10%，运用Python计算使得投资组合收益波动率最小情况下的每只股票的配置权重，具体的代码如下：

```
import scipy.optimize as sco    #导入Scipy的子模块optimize
def f(w):                      #定义一个需要求解最优化的函数
    w=np.array(w)              #设置投资组合中每只股票的权重
    Rp_opt=np.sum(w*R_mean)    #计算最优投资组合的预期收益率
    Vp_opt=np.sqrt(np.dot(w,np.dot(R_cov,w.T)))    #计算最优投资组合的收益波动率
    return np.array([Rp_opt,Vp_opt])    #以数组的格式输出结果

def Vmin_f(w):                #定义一个得到最小波动率的权重函数
    return f(w)[1]            #输出前面定义的函数f(w)结果的第二个元素

cons=({'type':'eq','fun':lambda x:np.sum(x)-1},{ 'type':'eq','fun':lambda x:f(x)[0]-0.1})
#以字典格式一次输入权重的约束条件，预测收益率等于10%
bnds=tuple((0,1) for x in range(len(R_mean)))    #以元组格式输入权重的边界条件
len(R_mean)*[1.0/len(R_mean),]    #用于生成一个权重相等的数组

result=sco.minimize(Vmin_f,len(R_mean)*[1.0/len(R_mean),],method='SLSQP',bounds=bnds,constraints=cons)
print('投资组合预期收益率10%时上海机场的权重',round(result['x'][0],4))
print('投资组合预期收益率10%时宝钢股份的权重',round(result['x'][1],4))
print('投资组合预期收益率10%时海通证券的权重',round(result['x'][2],4))
print('投资组合预期收益率10%时工商银行的权重',round(result['x'][3],4))
print('投资组合预期收益率10%时中国石油的权重',round(result['x'][4],4))
```

## 8.2.2 投资组合的有效前沿

【例8-4】依然沿用例8-1的信息，计算该投资组合收益波动率的全局最小值，以及与该最小波动率相对应的预期收益率，具体的代码如下：

```
cons_vmin=({'type':'eq','fun':lambda x:np.sum(x)-1}) #仅设置权重和等于1的约束条件

result_vmin=sco.minimize(Vmin_f,len(R_mean)*[1.0/len(R_mean),],method='SLSQP',bounds=
bnds,constraints=cons_vmin)
Rp_vmin=np.sum(R_mean*result_vmin['x'])
Vp_vmin=result_vmin['fun']
print('波动率在可行集是全局最小值时的投资组合预期收益率',round(Rp_vmin,4))
print('在可行集是全局最小的波动率',round(Vp_vmin,4))
```

波动率在可行集是全局最小值时的投资组合预期收益率 0.0306  
在可行集是全局最小的波动率 0.1636

## 8.2.2 投资组合的有效前沿

【例8-5】依然沿用例8-1的信息，最终完成对有效前沿的创建并可视化（见图8-5），具体的代码如下：

```
Rp_target=np.linspace(Rp_vmin,0.25,100) #生成投资组合的目标收益率数组
```

```
Vp_target=[]
```

```
for r in Rp_target:
```

```
    cons_new=({'type':'eq','fun':lambda x:np.sum(x)-1},{ 'type':'eq','fun':lambda x:f(x)[0]-r})
```

```
    #以字典格式输入预期收益率等于目标收益率的约束条件和权重的约束条件
```

```
result_new=sco.minimize(Vmin_f,len(R_mean)*[1.0/len(R_mean)],,method='SLSQP',bounds=bnds,constraints=cons_new)
```

```
Vp_target.append(result_new['fun'])
```

```
plt.figure(figsize=(8,6))
```

```
plt.scatter(Vp_list,Rp_list)
```

```
plt.plot(Vp_target,Rp_target,'r-',label=u'有效前沿',lw=2.5)
```

```
plt.plot(Vp_vmin,Rp_vmin,'y*',label=u'全局最小波动率',markersize=14)
```

```
plt.xlabel(u'波动率',fontsize =13)
```

```
plt.ylabel(u'收益率',fontsize =13,rotation=90)
```

```
plt.xticks(fontsize=13)
```

```
plt.yticks(fontsize=13)
```

```
plt.xlim(0.15,0.28)
```

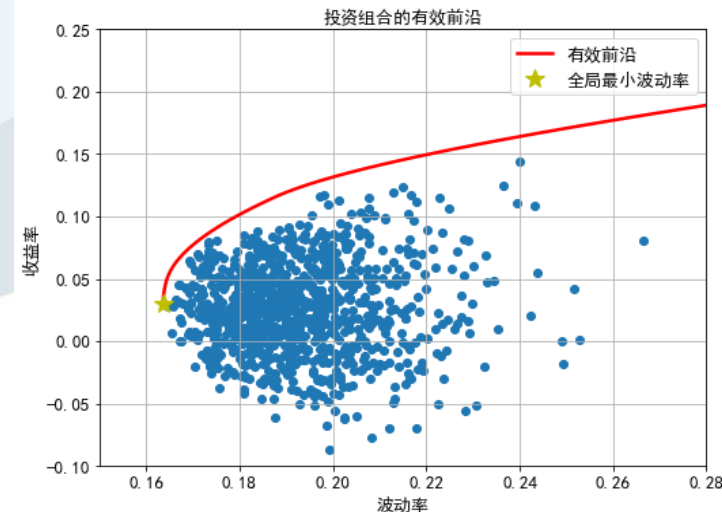
```
plt.ylim(-0.1,0.25)
```

```
plt.title(u'投资组合的有效前沿',fontsize=13)
```

```
plt.legend(fontsize=13)
```

```
plt.grid('True')
```

```
plt.show()
```



## 8.2.3 资本市场线

现在将无风险资产纳入投资组合中，也就意味着投资者可以按照无风险利率借入任何数量的资金，这样就引出了资产市场线。

资本市场线（**Capital Market Line, CML**）是一条从无风险收益率引出的与有效前沿相切的一条切线，并且该切线有且仅有一条。资本市场线的数学表达式是：

$$E(R_p) = R_F + \left[ \frac{E(R_M) - R_F}{\sigma_M} \right] \sigma_p$$

通常而言无风险利率是已知的，因此计算资本市场线的斜率就是求解如下的最大值：

$$\max_{w_i} \frac{E(R_p) - R_F}{\sigma_p}$$

约束条件分别是：

$$\sum_{i=1}^N w_i = 1$$

$$w_i > 0$$

## 8.2.3 资本市场线

【例8-6】依然沿用例8-1的信息，同时假定无风险利率是2%年，通过Python模拟资本市场线并且可视化，具体分为两个步骤。

第1步：计算得到资本市场线的斜率、市场组合的预期收益率和收益波动率，具体的代码如下：

```
def F(w):                #定义一个新的需要求解最优化的函数
    Rf=0.02               #无风险利率为2%
    w=np.array(w)         #设置投资组合中每只股票的权重
    Rp_opt=np.sum(w*R_mean) #计算最优投资组合的预期收益率
    Vp_opt=np.sqrt(np.dot(w,np.dot(R_cov,w.T))) #计算最优投资组合的收益波动率
    SR=(Rp_opt-Rf)/Vp_opt  #定义投资组合的夏普比率
    return np.array([Rp_opt,Vp_opt,SR])          #以数组的格式输出结果

def SRmin_F(w):          #定义一个使负的夏普比率最小化的函数，也就是夏普比率最大
    return -F(w)[2]

cons_SR=({'type':'eq','fun':lambda x:np.sum(x)-1}) #设置权重的约束条件
result_SR=sco.minimize(SRmin_F,len(R_mean)*[1.0/len(R_mean),],method='SLSQP',bounds=bnds,constraints=cons_SR)
Rf=0.02
slope=-result_SR['fun']  #资本市场线的斜率
Rm=np.sum(R_mean*result_SR['x']) #市场组合的预期收益率
Vm=(Rm-Rf)/slope        #市场组合收益波动率
print('市场组合的预期收益率',round(Rm,4))
print('市场组合的波动率',round(Vm,4))
```

市场组合的预期收益率 0.1827

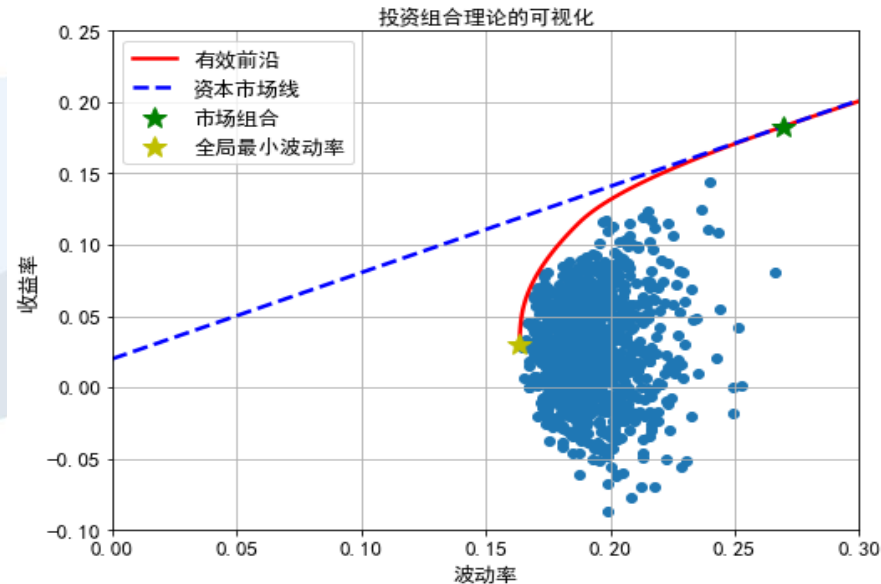
市场组合的波动率 0.2695

## 8.2.3 资本市场线

第2步：模拟资本市场线并且可视化（见图8-6），具体的代码如下

```
Rp_cml=np.linspace(0.02,0.25)    #刻画资本市场线的投资组合预期收益率数组  
Vp_cml=(Rp_cml-Rf)/slope         #刻画资本市场线的投资组合收益波动率数组
```

```
plt.figure(figsize=(8,6))  
plt.scatter(Vp_list,Rp_list)  
plt.plot(Vp_target,Rp_target,'r-',label=u'有效前沿',lw=2.5)  
plt.plot(Vp_cml,Rp_cml,'b--',label=u'资本市场线',lw=2.5)  
plt.plot(Vm,Rm,'g*',label=u'市场组合',markersize=14)  
plt.plot(Vp_vmin,Rp_vmin,'y*',label=u'全局最小波动率',markersize=14)  
plt.xlabel(u'波动率',fontsize =13)  
plt.ylabel(u'收益率',fontsize =13,rotation=90)  
plt.xticks(fontsize=13)  
plt.yticks(fontsize=13)  
plt.xlim(0.0,0.3)  
plt.ylim(-0.1,0.25)  
plt.title(u'投资组合理论的可视化',fontsize=13)  
plt.legend(fontsize=13)  
plt.grid('True')  
plt.show()
```



## 8.3 CAPM模型



## 8.3.1 系统风险与非系统风险

### 1、系统风险

系统风险（**systematic risk**），也称为不可分散的风险（**undiversifiable risk**），是所有股票共同承担的风险（即市场风险），一般是指由于上市公司外部、不为公司所预计和控制的因素造成的风险。这些风险因素通常表现为全球性或区域性的恐慌或者贸易争端，国民经济严重衰退或不景气，政府出台紧缩的宏观经济调控政策等。

### 2、非系统风险

非系统风险（**nonsystematic risk**），也称为特殊风险（**idiosyncratic risk**）或者可分散风险（**diversifiable risk**），是指发生于公司内部特定事件所造成的风险，纯粹由公司自身的因素而引发，与整个证券市场不存在系统性的、全面的联系。

非系统风险是股票的特定性风险，这一部分风险可以通过选取一个由不同股票构成的充分分散化的投资组合进行消除。

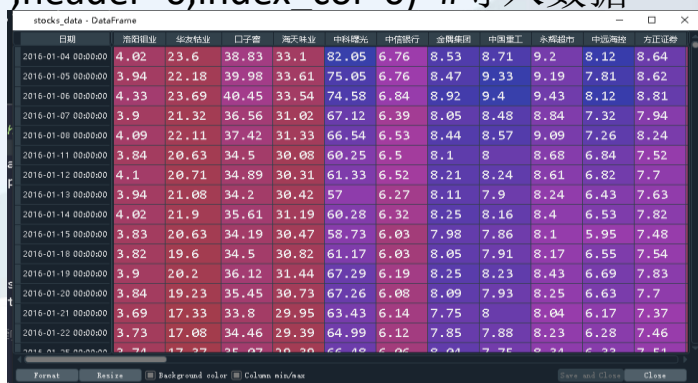
## 8.3.1 系统风险与非系统风险

### 3、案例

【例8-7】用上证180指数的成分股模拟投资组合，考察投资组合中不同股票数量与整个投资组合波动率（风险）之间的关系。假定选择2016年至2018年作为观测期间，观测频率是交易日，鉴于成分股中的部分上市公司是在观测期内才上市，因此将观测期内（即2016至2018年）上市的公司股票剔除，最终保留155只股票。在投资组合中逐次增加成分股股票数量（N）并且确保投资组合中的不同股票是相等权重（ $1/N$ ），比如，第1次仅配置1只股票，权重为100%；第2次配置2只股票，每只股票权重降至50%；第3次配置3只股票，票权重降至 $1/3$ ，以此类推，一直到最后第155次配置全部155只股票，每只股票权数为 $1/155$ 。整个模拟的过程运用 Python完成，并且分为3个步骤。

第1步：从外部导入155只股票2016年至2018年的收盘价数据，具体的代码如下：

```
stocks_data=pd.read_excel('D:\Zhangzw\Python\Python金融数据分析\RawData\第8章\上证180指数成分股日收盘价（2016-2018年（剔除该期间上市的股票））.xlsx',sheet_name='Sheet1',header=0,index_col=0) #导入数据
```



日期	宁德时代	华友钴业	口子窖	海天味业	中科曙光	中信银行	金隅集团	中国重工	永辉超市	中远海控	方正证券
2016-01-04 00:00:00	4.02	23.6	38.83	33.1	82.05	6.76	8.53	8.71	9.2	8.12	8.64
2016-01-05 00:00:00	3.94	22.18	39.98	33.61	75.05	6.76	8.47	9.33	9.19	7.81	8.62
2016-01-06 00:00:00	4.33	23.69	40.45	33.54	74.58	6.84	8.92	9.4	9.43	8.12	8.81
2016-01-07 00:00:00	3.9	21.32	36.56	31.02	67.12	6.39	8.05	8.48	8.84	7.32	7.94
2016-01-08 00:00:00	4.09	22.11	37.42	31.33	66.54	6.53	8.44	8.57	9.09	7.26	8.24
2016-01-11 00:00:00	3.84	20.63	34.5	30.08	60.25	6.5	8.1	8	8.68	6.84	7.52
2016-01-12 00:00:00	4.1	20.71	34.89	30.31	61.33	6.52	8.21	8.24	8.61	6.82	7.7
2016-01-13 00:00:00	3.94	21.08	34.2	30.42	57	6.27	8.11	7.9	8.24	6.43	7.63
2016-01-14 00:00:00	4.02	21.9	35.61	31.19	60.28	6.32	8.25	8.16	8.4	6.53	7.82
2016-01-15 00:00:00	3.83	20.63	34.19	30.47	58.73	6.03	7.98	7.86	8.1	5.95	7.48
2016-01-18 00:00:00	3.82	19.6	34.5	30.82	61.17	6.03	8.05	7.91	8.17	6.55	7.54
2016-01-19 00:00:00	3.9	20.2	36.12	31.44	67.29	6.19	8.25	8.23	8.43	6.69	7.83
2016-01-20 00:00:00	3.84	19.23	35.45	30.73	67.26	6.08	8.09	7.93	8.25	6.63	7.7
2016-01-21 00:00:00	3.69	17.33	33.8	29.95	63.43	6.14	7.75	8	8.04	6.17	7.37
2016-01-22 00:00:00	3.73	17.08	34.46	29.39	64.99	6.12	7.85	7.88	8.23	6.28	7.46

## 8.3.1 系统风险与非系统风险

第2步：计算每只股票的日收益率数据，并且运用for语句快速生成不同股票数量对应的投资组合收益波动率，具体的代码如下：

```
return_stocks=np.log(stocks_data/stocks_data.shift(1)) #建立股票日收益率时间序列
return_stocks=return_stocks.dropna() #缺失数据的处理
n=len(return_stocks.columns) #计算全部股票的数量
vol_port=np.zeros(n) #生成存放投资组合收益波动率的初始数组

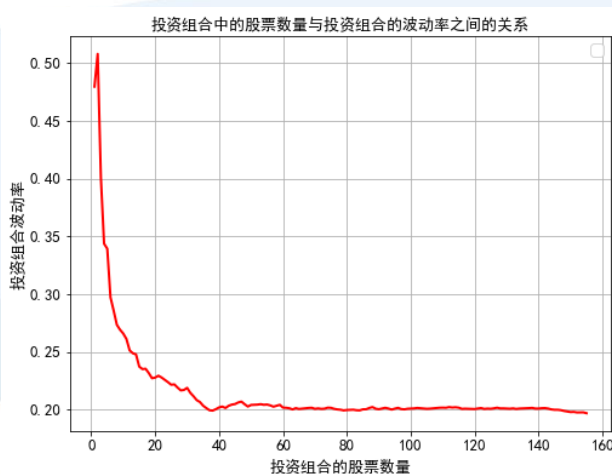
for i in range(1,n+1):
    weight=np.ones(i)/i #依次生成投资组合中每只股票的等权重数组
    return_cov=252*return_stocks.iloc[:, :i].cov() #依次计算投资组合中股票收益率的年化协方差
    vol_port[i-1]=np.sqrt(np.dot(weight,np.dot(return_cov,weight.T))) #依次计算并存放投资组合的年化波动率
```

## 8.3.1 系统风险与非系统风险

第3步：运用在第2步中计算得到的针对不同股票数量所对应的投资组合波动率，可视化股票数量与投资组合波动率之间的关系（见图8-7），具体的代码如下：

```
N_list=np.arange(n)+1 #生成从1到155的数组
```

```
plt.figure(figsize=(8,6))
plt.plot(N_list,vol_port,'r-',lw=2.0)
plt.xlabel(u'投资组合的股票数量',fontsize =13)
plt.ylabel(u'投资组合波动率',fontsize =13,rotation=90)
plt.xticks(fontsize=13)
plt.yticks(fontsize=13)
plt.title(u'投资组合中的股票数量与投资组合的波动率之间的关系',fontsize=13)
plt.legend(fontsize=13)
plt.grid('True')
plt.show()
```



## 8.3.2 模型数学表达式及运用

### 1、数学表达式

资本资产定价模型的数学表达式

$$E(R_i) = R_F + \beta_i[E(R_M) - R_F]$$

下面，通过 Python 自定义资本资产定价模型的表达式，具体的代码如下：

```
def Ri_CAPM(beta,Rm,Rf):  
    """构建资本资产定价模型来计算投资资产的预期收益  
    beta: 代表了单一股票或者某个股票组合的贝塔值:  
    Rm: 代表了市场收益率:  
    Rf: 代表了无风险利率。"""  
    return Rf+beta*(Rm-Rf)
```

## 8.3.2 模型数学表达式及运用

【例 8-8】假定有 4 只股票，分别是 A 股票、B 股票、C 股票和 D 股票，股票的贝塔值分别是  $\beta_A=0.5$ ， $\beta_B=0.8$ ， $\beta_C=1.2$  和  $\beta_D=1.5$ ，市场组合的预期收益率是 10%，无风险利率是 3%，运用资本资产定价模型计算这 4 只股票的预期收益率。↵

通过 Python 定义的资本资产定价模型函数 `Ri_CAPM` 求解例题中不同股票的贝塔值，具体的代码如下：

```
R_f=0.03 #无风险利率
R_m=0.1 #市场组合的预期收益率
beta_array=np.array([0.5,0.8,1.2,1.5]) #输入贝塔值的数组

Ri=Ri_CAPM(beta=beta_array,Rm=R_m,Rf=0.03)
print('贝塔等于0.5时投资资产的预期收益率: ',round(Ri[0],4))
print('贝塔等于0.8时投资资产的预期收益率: ',round(Ri[1],4))
print('贝塔等于1.2时投资资产的预期收益率: ',round(Ri[2],4))
print('贝塔等于1.5时投资资产的预期收益率: ',round(Ri[3],4))
```

贝塔等于0.5时投资资产的预期收益率： 0.065  
贝塔等于0.8时投资资产的预期收益率： 0.086  
贝塔等于1.2时投资资产的预期收益率： 0.114  
贝塔等于1.5时投资资产的预期收益率： 0.135

## 8.3.2 模型数学表达式及运用

【例8-9】利用8.2节例8-1中涉及的上海机场和宝钢股份这两只股票，以沪深300指数作为市场组合，基于2016年至2018年的日收盘价数据计算这2只股票的贝塔值。下面，直接运用 Python进行相应的计算，具体分为3个步骤。

第1步：导入沪深300指数2016年至2018年期间的日收盘价数据，并且生成日收益率的时间序列，具体的代码如下：

```
data_index=pd.read_excel('D:\Zhangzw\Python\Python金融数据分析\RawData\第8章\沪深300指数（2016-2018年）.xlsx',sheet_name='Sheet1',header=0,index_col=0) #导入外部数据
```

```
R_index=np.log(data_index/data_index.shift(1)) #按照对数收益率的计算公式得到沪深300值数的日收益率时间序列
```

```
R_index=R_index.dropna() #缺失数据的处理
```

```
R_index.describe()
```

沪深300指数

```
count 731.000000
mean -0.000194
std 0.011603
min -0.071853
25% -0.005131
50% 0.000350
75% 0.005062
max 0.042262
```



## 8.3.2 模型数学表达式及运用

第2步：计算上海机场股票的贝塔值，并且需要运用到6.2节介绍 Stats Models的子模块api，具体的代码如下：

```
import statsmodels.api as sm          #导入StatsModels的子模块api
R_index_addcons=sm.add_constant(R_index) #对自变量的样本值添加一列常数值
model_shjc=sm.OLS(endog=R.iloc[:,0],exog=R_index_addcons[1:]) #构建计算上海机场股票的普通最小二乘法的线性回归模型
result_shjc=model_shjc.fit()          #拟合线性回归模型
result_shjc.summary()
```

```
OLS Regression Results
=====
Dep. Variable:          上海机场      R-squared:          0.208
Model:                  OLS          Adj. R-squared:      0.207
Method:                 Least Squares  F-statistic:        191.3
Date:                  Tue, 24 May 2022  Prob (F-statistic):    8.34e-39
Time:                  20:20:28       Log-Likelihood:      1940.1
No. Observations:      730           AIC:                -3876.
Df Residuals:          728           BIC:                -3867.
Df Model:               1
Covariance Type:       nonrobust
=====

```

	coef	std err	t	P> t	[0.025	0.975]
const	0.0009	0.001	1.506	0.132	-0.000	0.002
沪深300指数	0.7496	0.054	13.832	0.000	0.643	0.856

```
=====
Omnibus:                93.029      Durbin-Watson:        2.128
Prob(Omnibus):           0.000      Jarque-Bera (JB):     450.812
Skew:                    0.459      Prob(JB):             1.28e-98
Kurtosis:                6.739      Cond. No.:            86.2
=====
```



## 8.3.2 模型数学表达式及运用

第3步：类似于第2步的做法，计算宝钢股份股票的贝塔值，具体的代码如下：

```
model_bggf=sm.OLS(endog=R.iloc[:,1],exog=R_index_addcons[1:]) #构建计算宝钢股份股票的普通最小二乘法的线性回归模型
result_bggf=model_bggf.fit()
result_bggf.summary()
```

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                OLS Regression Results
=====
Dep. Variable:                  宝钢股份      R-squared:                  0.325
Model:                          OLS          Adj. R-squared:              0.324
Method:                        Least Squares  F-statistic:                 350.8
Date:                          Tue, 24 May 2022 Prob (F-statistic):         3.50e-64
Time:                          20:26:34      Log-Likelihood:             1901.9
No. Observations:              730          AIC:                       -3800.
Df Residuals:                  728          BIC:                       -3791.
Df Model:                      1
Covariance Type:               nonrobust
=====
                                coef      std err          t      P>|t|      [0.025      0.975]
-----
const                0.0005      0.001      0.763      0.446      -0.001      0.002
沪深300指数          1.0694      0.057     18.729      0.000      0.957      1.182
=====
Omnibus:                  99.438      Durbin-Watson:              1.923
Prob(Omnibus):            0.000      Jarque-Bera (JB):           280.780
Skew:                    0.683      Prob(JB):                   1.07e-61
Kurtosis:                 5.714      Cond. No.                   86.2
=====
```

## 8.4 股价随机过程

# 8.4.1 马尔可夫过程与有效市场假说

## 1、有效市场假说

有效市场假说（Efficient Markets Hypothesis, EMH），又称有效市场理论（Efficient Markets Theory）。该假说正式的提出者是美国经济学家尤金·法玛（Eugene Fama）。在总结了前人的理论和实证的基础上，法玛于1970年系统性地提出了有效市场假说，该理论包含以下几个要点。

第一，市场上的每个投资者都是理性的经济人，金融市场中每只股票所代表的公司都处于这些理性经济人的严格监视之下，他们每天都在进行基本面的分析，用公司未来的盈利能力来评估公司的股票价格，把未来价值折算成今天的现值，并谨慎地在风险与收益之间进行权衡与取舍。

第二，股票的价格反映了这些理性经济人的供求平衡，想买入的人数正好等于想卖出的人数，也就是认为股价被高估的人数与认为股价被低估的人数正好相等，假如有人发现这两者不等（即存在套利的可能性），投资者立即会用买入或卖出股票的办法使股价迅速变动到能够使供求相等为止。

第三，股票的价格也能充分反映该资产的所有可获得的信息（即“信息有效”）。当信息变动时，股票的价格就一定会随之变动。一个利好消息或利空消息刚刚传出时，股票的价格就开始异动，当它已经路人皆知时，股票的价格也已经涨或跌到适当的价位了。

## 8.4.1 马尔可夫过程与有效市场假说

在金融理论中，根据信息集的大小，有效市场假说可以进一步分为3种。

第1种：弱式有效市场假说（**Weak-Form Market Efficiency**）。该假说认为在弱式有效的情况下，市场价格已充分反映了所有过去历史的证券价格信息

第2种：半强式有效市场假说（**Semi-Strong-Form Market Efficiency**）该假说认为价格已充分反映出所有已公开的有关公司营运前景的信息。

第3种：强式有效市场假说（**Strong-Form Market Efficiency**）。强式有效市场假说认为价格已充分地反映了所有关于公司营运的信息，这些信息包括已公开的或内部未公开的信息。

市场有效性程度	技术分析	基本面分析	组合管理
无效市场	有效	有效	积极进取
弱式有效	无效	有效	积极进取
半强式有效	无效	无效	积极进取
强式有效	无效	无效	消板保守

## 8.4.1 马尔可夫过程与有效市场假说

### 2、随机过程

如果一个变量的值以某种不确定的形式随时间变化，就称该变量服从某种**随机过程(Stochastic process)**。因此，随机过程就是对一连串随机事件动态关系的定量描述。众多投资者都热衷于对股价的预测，进而最大化财富效应，但是由于股价受许多随机因素的影响，它本身具有随机性，因此股价服从一个随机过程，对股价预测的努力往往是一种徒劳。

### 3、马尔可夫过程

**马尔可夫过程 (Markov process)** 是由俄罗斯数学家安德雷·马尔可夫提出，该过程是个特殊类型的随机过程，核心思想就是对标的变量未来的预测仅仅与标的变量的当前值有关，变量的历史值以及变量从过去到现在的演变方式与未来的预测均无关。

这种在已知“现在”的条件下，“未来”与“过去”彼此独立的特性就被称为马尔可夫性质，具有这种性质的随机过程就称为马尔可夫过程，其最原始的模型就是马尔可夫链。

由于对将来的预测存在着不确定性，因此预测必须以概率分布的形式表达。马尔可夫性质就意味着股票价格在将来的概率分布不依赖于股票价格在过去所遵循的特定路径。

## 8.4.2 维纳过程与广义维纳过程

### 1、维纳过程

维纳过程 (Wiener process) 由美国数学家、控制论创始人诺伯特·维纳 (Norbert Wiener) 提出, 维纳过程是期望值为0、方差为1的特殊马尔可夫过程。维纳过程曾在物理学中用来描述一个粒子受到大量小分子碰撞后所产生的运动, 该现象最早是由苏格兰植物学家罗特·布朗在观察花粉以及其他悬浮微小颗粒在水中不停地作不规则的曲线运动时发现的, 这种运动也被称为布朗运动 (Brownian motion)。下面就采用正规的数学表达方式描述:

性质 1: 变量  $x$  在瞬时 (极短时间)  $\Delta t$  的变化量  $\Delta x$  满足等式

$$\Delta x = \varepsilon \sqrt{\Delta t} \quad (8-16)$$

其中,  $\varepsilon$  服从标准正态分布 (期望值为 0、方差为 1 的正态分布), 同时  $\Delta t$  可以视为常

数, 因此  $\Delta x$  也服从正态分布,  $\Delta x$  的期望值为 0、标准差为  $\sqrt{\Delta t}$ 。

性质 2: 在任何两个不相重叠的  $\Delta t$  时间区间内, 变化量  $\Delta x$  之间是相互独立的。这个性质就说明变量  $x$  服从马尔可夫过程。

接着考虑在一段相对较长的时间区间  $T$  内变量  $x$  的变化, 这里将变化量表述为  $x_T - x_0$ 。该变化量可以看成是  $N$  个区间长度为  $\Delta t$  的  $x$  变化量总和, 具体如下:

$$x_T - x_0 = \sum_{i=1}^N \varepsilon_i \sqrt{\Delta t} \quad (8-17)$$

其中,  $N = T / \Delta t$ ,  $\varepsilon_i$  依然服从标准正态分布。

$x_T - x_0$  是一个服从均值 (漂移率) 为 0、方差 (方差率) 为  $T$  的正态分布。

## 8.4.2 维纳过程与广义维纳过程

### 2、广义维纳过程

如果变量 $y$ 服从广义维纳过程（generalized Wiener process）， $y$ 可以通过维纳过程 $dx$ 定义，具体如下：

$$dy = a dt + b dx \quad (8-18)$$

其中， $a$ 和 $b$ 都是常数。

同时，等式右边的 $a dt$ 这一项说明变量 $y$ 在单位时间 $\Delta t$ 内的漂移率为 $a$ 。如果忽略后面的 $b dx$ 这一项，式子（8-18）就退化成为 $dy = a dt$ ，即 $dy/dt = a$ 。对 $t$ 进行积分，就可以得出

$$y = y_0 + at \quad (8-19)$$

其中， $y_0$ 是 $y$ 在 $t=0$ 时刻的值（即变量 $y$ 的初始值）， $at$ 说明经过了时间 $T$ 以后变量 $y$ 的增量就是 $aT$ 。

接着讨论式子右边的第2项 $b dx$ ，这一项可被看成附加在变量 $y$ 路径上的噪声（noise），幅度为维纳过程的 $b$ 倍。由于维纳过程的标准差为1，因此 $b$ 倍维纳过程在单位时间内的方差就等于 $b^2$ 。因此，广义维纳过程的漂移率为 $a$ 、方差率为 $b^2$ 。

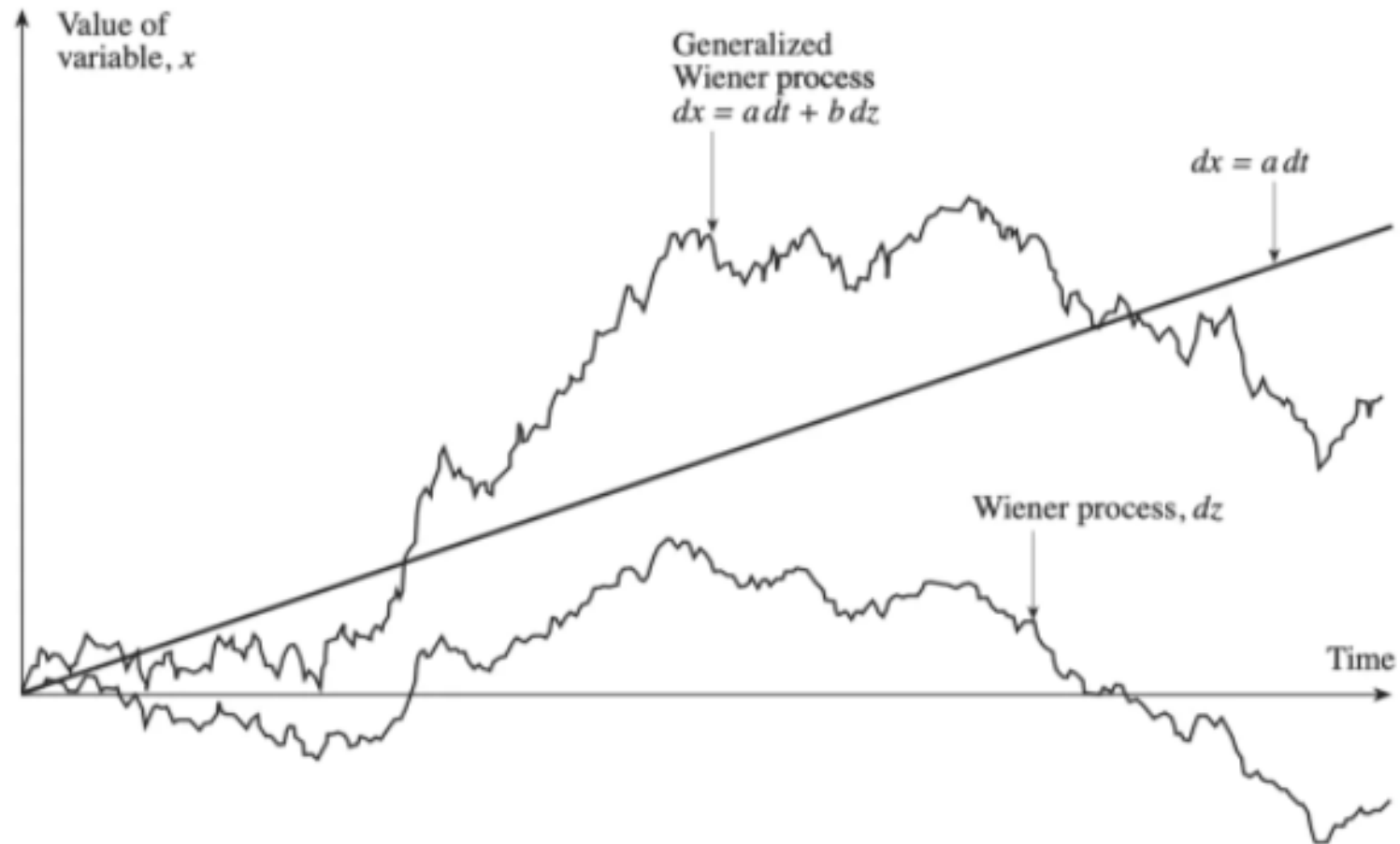
运用离散化的表达方式，在单位时间 $\Delta t$ 内，可以得到 $y$ 变化量 $\Delta y$ 的数学表达式

$$\Delta y = a \Delta t + b \varepsilon \sqrt{\Delta t} \quad (8-20)$$

## 8.4.2 维纳过程与广义维纳过程

### 2、广义维纳过程

**Figure 13.2** Generalized Wiener process with  $a = 0.3$  and  $b = 1.5$ .





## 8.4.3 几何布朗运动

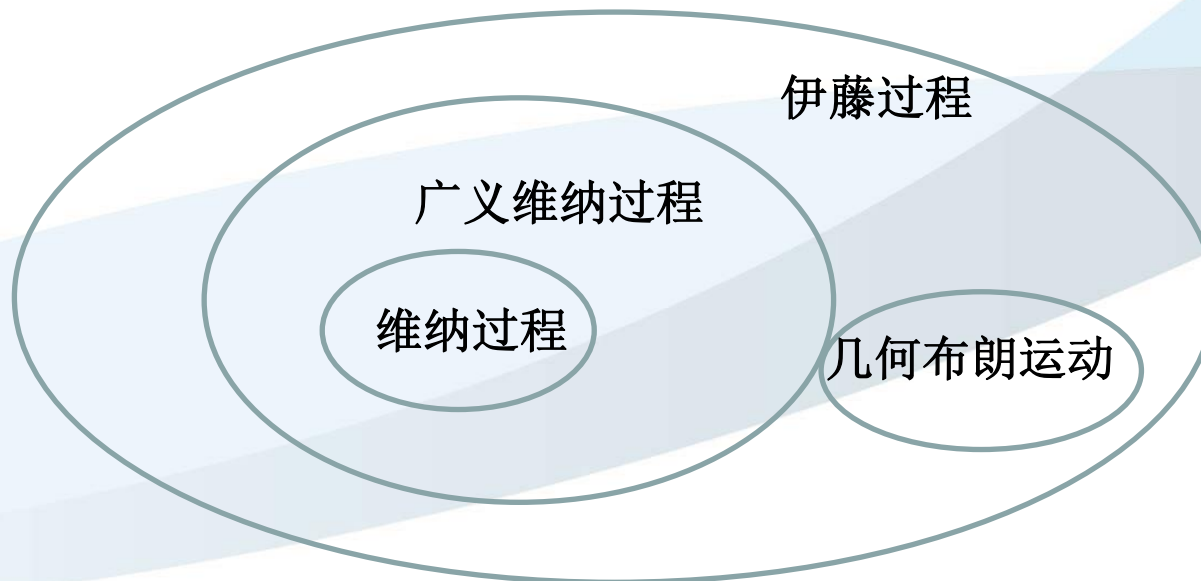
### 1、伊藤过程

当广义维纳过程中的  $a$ ,  $b$  不是常数, 而是关于  $(x, t)$  的函数时, 就变为一个伊藤过程:

$$dx = a(x, t) dt + b(x, t) dz$$

在一个很短的时间  $\Delta t$  内, 假设  $a$  和  $b$  在  $t$  到  $t + \Delta t$  中保持不变, 我们有:

$$\Delta x = a(x, t) \Delta t + b(x, t) \epsilon \sqrt{\Delta t}$$



## 8.4.3 几何布朗运动

### 2、股票价格具体服从怎样的随机过程？

金融学中，几何布朗运动在布莱克-斯科尔斯定价模型被用来定性股票价格，因而也是最常用的描述股票价格的模型。

因为投资者要求的预期收益率与股票价格无关，所以假定股票收益率的期望值是一个常数就更加合理。

如果股票价格为 $S$ ，股价的漂移率可以是 $uS$ ，其中 $u$ 是股票收益率的期望值并且是常数。在单位时间 $\Delta t$ ，股价 $S$ 的期望增量为 $uS\Delta t$ 。

同时，由于股票价格存在不确定性，一个相对合理的假设是无论股票价格是多少，在一个较短时间 $\Delta t$ 内股票收益率的波动率都是相同的。

可以得出股票价格服从的随机过程模型如下：

$$dS = uSdt + \sigma Sdx \quad (8-21)$$

等式两边同时除以 $S$ ，可以得到

$$\frac{dS}{S} = udt + \sigma dx \quad (8-22)$$

其中，参数 $u$ 是股票收益率的期望值，参数 $\sigma$ 是股票收益率的波动率。这个式子是用于上述股票价格最广泛的一种随机模型，该模型称为几何布朗运动（geometric Brownian motion）。

离散形式：

$$\frac{\Delta S}{S} = u\Delta t + \sigma\Delta x \quad (8-23)$$

由于前面提到 $\Delta x = \varepsilon\sqrt{\Delta t}$ ，因此可以得到

$$\frac{\Delta S}{S} = u\Delta t + \sigma\varepsilon\sqrt{\Delta t} \quad (8-24)$$

## 8.4.3 几何布朗运动

### 3、模拟服从几何布朗运动的股价：

由于几何布朗运动的离散形式无法接运用Python进行处理，需要运用欧拉离散方法变换为如下的差分方程：

$$S_t = S_{t-\Delta t} e^{(\mu - \frac{1}{2}\sigma^2) \Delta t + \sigma \varepsilon_t \sqrt{\Delta t}}$$

**【例8-10】**运用2014年至2018年东方航空A股股票（股票代码600115）的日收盘价格，模拟未来3年（2019年至2021年）该股票的每日价格走势，模拟路径共500条，同时在模拟过程中，将S0设定为2019年1月2日的收盘价4.73元/股。具体的操作分为4个步骤。

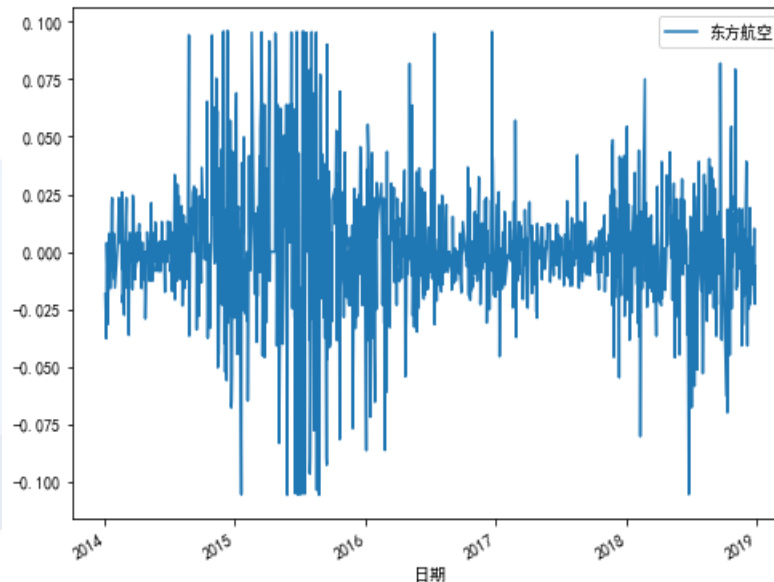
第1步：导入数据并且计算得到股票的平均年化收益率和年化波动率（见图8-8），具体的代码如下：

```
S=pd.read_excel('D:\Zhangzw\Python\Python金融数据分析\RawData\第8章\东方航空股票价格（2014-2018）.xlsx',sheet_name='Sheet1',header=0,index_col=0) #导入外部数据
```

```
S.describe()
```

## 8.4.3 几何布朗运动

```
R=np.log(S/S.shift(1)) #得到股票2014至2018年日收益率
R=R.dropna()
R.plot(figsize=(8,6))
mu=R.mean()*252 #股票的预期年化收益率
sigma=R.std()*np.sqrt(252)#股票收益的年化波动率
print('股票的预期年化收益率',round(mu,4))
print('股票收益的年化波动率',round(sigma,4))
股票的预期年化收益率 东方航空 0.113
dtype: float64
股票收益的年化波动率 东方航空 0.463
dtype: float64
```



## 8.4.3 几何布朗运动

第2步：输入需要进行模拟的相关参数，在这一步中将运用到Pandas模块中的时间戳索引函数 `DatetimeIndex`，用该函数生成从2019年1月2日至2021年12月31日并且是工作日的时间数据组，具体代码如下：

```
import numpy.random as npr #导入NumPy的子模块random
#date=pd.DatetimeIndex(start='2019-01-02',end='2021-12-31',freq='B')#生成2019至2021年的
#工作日数组
date=pd.date_range(start='2019-01-02',end='2021-12-31',freq='B')#生成2019至2021年的工
#作日数组
N=len(date)          #将N赋值为是date的长度
l=500                #模拟的路径数
dt=1.0/252           #单位时间的区间长度
mu=np.array(mu)       #将数据结构调整为数组
sigma=np.array(sigma) #将数据结构调整为数组
S_GBM=np.zeros((N,l)) #建立存放服从几何布朗运动的未来股价的初始数组
S_GBM[0]=4.73         #将模拟的起点设为2019年1月2日的收盘价
```

## 8.4.3 几何布朗运动

第3步：运用for语句并生成模拟的未来股价时间序列，具体的代码如下：

```
for t in range(1,N):
    epsilon=np.random.standard_normal(1)
    S_GBM[t]=S_GBM[t-1]*np.exp((mu-0.5*sigma**2)*dt+sigma*epsilon*np.sqrt(dt))
S_gbm=pd.DataFrame(S_GBM,index=date) #将模拟的数值转化为带有时间索引的数据框
S_gbm.describe()
```

Out[25]:

	0	1	2	...	497	498	499
count	783.000000	783.000000	783.000000	...	783.000000	783.000000	783.000000
mean	4.011887	4.272213	2.731853	...	2.359684	7.252345	4.204721
std	1.479978	1.455004	0.774881	...	0.740081	2.710029	0.938670
min	1.862478	2.248510	1.238602	...	0.940820	3.273791	2.226722
25%	2.702953	3.087261	2.061296	...	1.958295	5.038193	3.725624
50%	3.653801	3.677674	2.748009	...	2.440544	6.661354	4.244765
75%	5.615587	5.673121	3.339819	...	2.827259	8.692254	4.842133
max	6.966031	8.295143	4.731126	...	4.730000	15.936838	6.674749

[8 rows x 500 columns]

## 8.4.3 几何布朗运动

最后：将模拟的结果可视化（见图8-9），具体代码如下：

```
plt.figure(figsize=(8,6))
plt.plot(S_gbm)
plt.xlabel(u'日期',fontsize =13)
plt.ylabel(u'股价',fontsize =13,rotation=90)
plt.xticks(fontsize=13,rotation=30)
plt.yticks(fontsize=13)
plt.title(u'服从几何布朗运动的股价模拟全部路径（2019-2021年）',fontsize=13)
plt.grid('True')
plt.show()
```

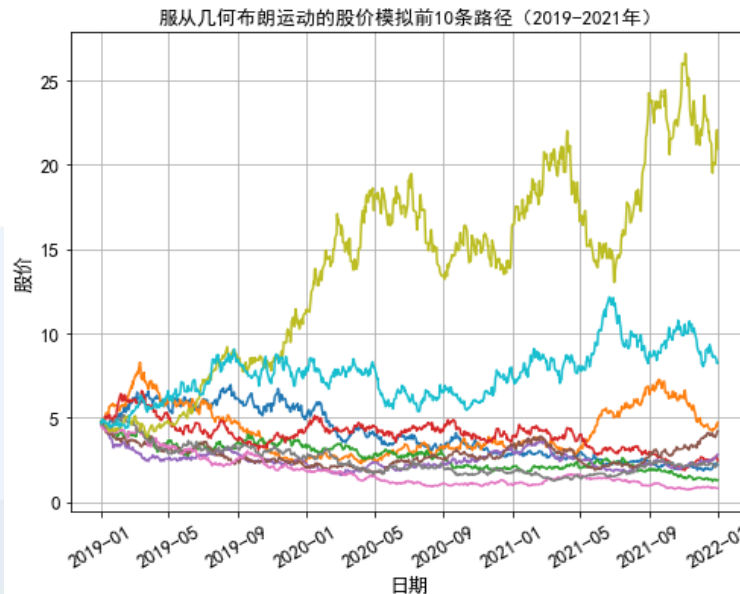




## 8.4.3 几何布朗运动

图8-9是将全部500条模拟路径进行了可视化，不难发现未来3年的股价绝大多数是处于0~20元的区间之中，下面就将模拟的前10条路径进行可视化（见图8-10）。

```
plt.figure(figsize=(8,6))
plt.plot(S_GBM.iloc[:,0:10]) #将模拟的前10条路径可视化
plt.xlabel(u'日期',fontsize =13)
plt.ylabel(u'股价',fontsize =13,rotation=0)
plt.xticks(fontsize=13,rotation=30)
plt.yticks(fontsize=13)
plt.title(u'服从几何布朗运动的股价模拟前10条路径（2019-2021年）',fontsize=13)
plt.grid('True')
plt.show()
```





## 8.5 投资组合的绩效评估

## 8.5.1 夏普比率

夏普比率（Sharpe Ratio, SR），也称为夏普指数，是由美国经济学家威廉·夏普（William Sharpe）提出，是当前评估公募基金、私募基金等投资组合绩效的标准化指标。夏普比率具体是指在某一时间区间内，投资组合获得的收益高于无风险利率的部分并除以投资组合收益波动率所得到的比值，反映了投资组合承担每一单位风险所带来的超额收益（相对于无风险收益率），在8.2节曾提到过夏普比率实际也是资本市场线的斜率。夏普比率的表达式如下：

$$SR = \frac{E(R_p) - R_F}{\sigma_F}$$

通过Python自定义一个用于计算夏普比率的函数，具体的代码如下：

```
def SR(Rp,Rf,Vp):  
    """计算夏普比率（Sharpe Ratio）  
    Rp: 表示投资组合的年化收益率  
    Rf: 表示年化无风险利率  
    Vp: 表示投资组合的年化收益波动率"""  
    return (Rp-Rf)/Vp
```

## 8.5.1 夏普比率

【例8-11】假定以表8-3中的4只公募基金作为评估对象，选择的观测期间是从2016年至2018年，观测频率是每个交易日；同时在计算夏普比率过程中，无风险利率假定选择银行一年期存款基准利率1.5%，具体的计算分3步进行。

基金代码	000411.0F	000697.0F	000867.0F	000996.0F
基金名称	景顺长城优质成长	汇添富移动互联	华宝品质生活	中银新动力

第1步：导入外部的数据并且绘制每个基金净值的日走势图（见图8-11），具体的代码如下：

```
fund=pd.read_excel('D:\Zhangzw\Python\Python金融数据分析\RawData\第8章\四只开放式股票型基金的净值（2016-2018年）.xlsx',sheet_name='Sheet1',header=0,index_col=0)
#导入外部数据
```

```
fund.plot(figsize=(8,6))
```



## 8.5.1 夏普比率

第2步：计算2016年至2018年这3年平均的基金年化收益率和波动率，同时，运用前面自定义计算夏普比率的函数SR对这4只基金进行计算，具体的代码如下：

```
R_fund=np.log(fund/fund.shift(1)) #生成基金日收益率的时间序列
R_fund=R_fund.dropna()
R_mean=R_fund.mean()*252          #计算全部3年的平均年化收益率
Sigma=R_fund.std()*np.sqrt(252)   #计算全部3年的年化波动率
R_f=0.015                         #一年期银行存款基准利率
SR_3years=SR(Rp=R_mean,Rf=R_f,Vp=Sigma)
print('2016至2018年平均3年的夏普比率\n',SR_3years) #输入\n表示输出时换行
```

2016至2018年平均3年的夏普比率

景顺长城优质成长基金 -0.492142

汇添富移动互联基金 -0.975046

华宝品质生活基金 -0.374546

中银新动力基金 -1.230769

dtype: float64

## 8.5.1 夏普比率

第3步：计算2016至2018年期间每一年的夏普比率，具体的代码如下

```
R_fund2016=R_fund.loc['2016-01-01':'2016-12-31'] #获取2016年的日收益率
R_fund2017=R_fund.loc['2017-01-01':'2017-12-31'] #获取2017年的日收益率
R_fund2018=R_fund.loc['2018-01-01':'2018-12-31'] #获取2018年的日收益率
R_mean_2016=R_fund2016.mean()*252 #计算2016年的年化收益率
Sigma_2016=R_fund2016.std()*np.sqrt(252) #计算2016年的年化波动率
R_mean_2017=R_fund2017.mean()*252 #计算2017年的年化收益率
Sigma_2017=R_fund2017.std()*np.sqrt(252) #计算2017年的年化波动率
R_mean_2018=R_fund2018.mean()*252 #计算2018年的年化收益率
Sigma_2018=R_fund2018.std()*np.sqrt(252) #计算2018年的年化波动率
SR_2016=SR(Rp=R_mean_2016,Rf=R_f,Vp=Sigma_2016) #计算2016年的夏普比率
SR_2017=SR(Rp=R_mean_2017,Rf=R_f,Vp=Sigma_2017) #计算2017年的夏普比率
SR_2018=SR(Rp=R_mean_2018,Rf=R_f,Vp=Sigma_2018) #计算2018年的夏普比率
print('2016年的夏普比率\n',SR_2016)
print('2017年的夏普比率\n',SR_2017)
print('2018年的夏普比率\n',SR_2018)
```

## 8.5.2 索提诺比率

索提诺比率（Sortino ratio, SOR）的分子与夏普比率相同，都是投资组合的期望收益率减去无风险利率，差异则在于分母，索提诺比率的分母是运用下偏标准差（Lower partial standard deviation）而不是总标准差，下偏标准差是只考虑亏损而不考虑盈利，隐含条件就是投资组合的上涨（正回报率）是符合投资者的需求而不应计入风险调整。该比率反映了投资组合承担每一单位下行风险所带来的超额收益（相对于无风险收益率）

索提诺比率的计算公式如下：

$$SOR = \frac{E(R_P) - R_F}{\sigma_{LP}}$$

就通过Python自定义一个用于计算索提诺比率的函数，具体代码如下：

```
def SOR(Rp,Rf,Vpl):  
    """计算索提诺比率（Sortino ratio）  
    Rp: 表示投资组合的年化收益率  
    Rf: 表示年化无风险利率  
    Vpl: 并表示投资组合收益率的年化下行标准差"""  
    return (Rp-Rf)/Vpl
```

## 8.5.2 索提诺比率

【例8-12】依然沿用前面例8-1的信息，运用Python计算4只基金的索提诺比率，具体分为两个步骤。

第1步：计算每只基金收益率的下行标准差，具体代码如下：

```
Vp_lower=np.zeros_like(R_mean) #生成防止基金收益率下行标准差的初始数组
for i in range(len(Vp_lower)):
    R_neg=R_fund.iloc[:,i][R_fund.iloc[:,i]<0] #生成基金收益率为负的时间序列
    Vp_lower[i]=np.sqrt(252)*np.sqrt(np.sum(R_neg**2)/len(R_neg)) #计算收益的年化下行标准差
    print(R_fund.columns[i],'收益率下行标准差',round(Vp_lower[i],4))
```

第2步：运用前面定义计算索提诺比率的函数SOR测算每只基金2016至2018年平均3年的索提诺比率，具体代码如下：

```
SOR_3years=SOR(Rp=R_mean_2016,Rf=R_f,Vpl=Vp_lower)
print('2016至2018年平均3年的索提诺比率\n',SOR_3years)
```

2016至2018年平均3年的索提诺比率

景顺长城优质成长基金 -0.754230

汇添富移动互联基金 -0.919355

华宝品质生活基金 -0.758294

中银新动力基金 -1.384996

dtype: float64

## 8.5.3 特雷诺比率

特雷诺比率（Treynor ratio, TR）由美国经济学家杰克·特雷诺（Jack Treynor）提出的用于评价投资组合业绩的指标。该比率与夏普比率相比，两个比率的分子均相同，只是分母改为了投资组合的贝塔值。该比率表示当投资组合每承受一单位系统风险时，会产生多少的风险溢价（相对于无风险收益率而言），该比率其实就是在5.2节提到的证券市场线（Security Market Line）的斜率。

特里诺比率的计算公式如下：

$$TR = \frac{E(R_P) - R_F}{\beta_P}$$

通过 Python 自定义一个用于计算特雷诺比率的函数，具体的代码如下：

```
def TR(Rp,Rf,beta):  
    """计算特雷诺比率（Treynor ratio）  
    Rp: 表示投资组合的年化收益率  
    Rf: 表示年化无风险利率  
    beta: 表示投资组合的贝塔值。"""  
    return (Rp-Rf)/beta
```



## 8.5.3 特雷诺比率

【例8-13】依然沿用前面例8-11的信息，计算4只基金的特雷诺比率，同时在计算投资组合的贝塔值是运用沪深300指数作为市场组合，具体的过程分为3个步骤。

第1步：导入外部沪深300指数的数据并且计算每个基金的贝塔值、具体的代码如下：

```
HS300=pd.read_excel('D:\Zhangzw\Python\Python金融数据分析\RawData\第8章\沪深300  
指数（2016-2018年）.xlsx',sheet_name='Sheet1',header=0,index_col=0) #导入外部数据  
R_HS300=np.log(HS300/HS300.shift(1)) #计算并生成沪深300指数的日收益率序列
```

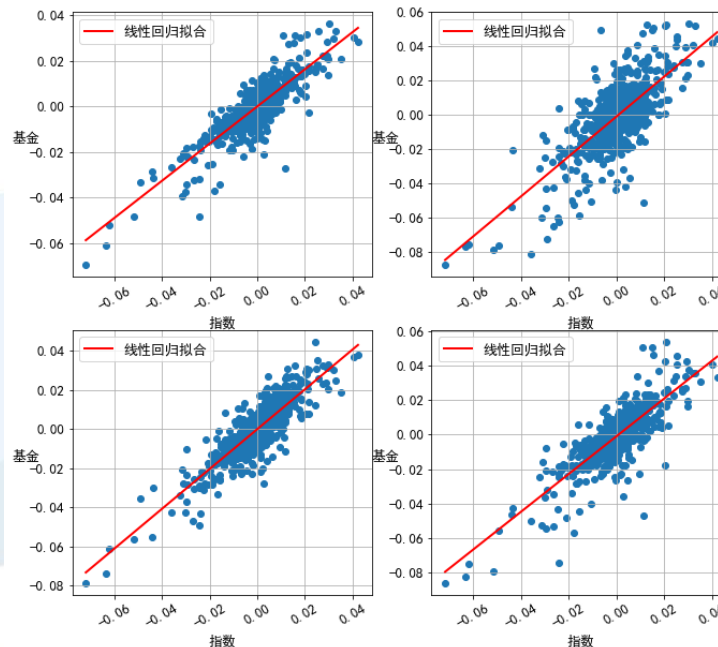
```
R_HS300=R_HS300.dropna()
```

```
import statsmodels.api as sm          #导入StatsModels的子模块api  
betas=np.zeros_like(R_mean)          #生成一个放置基金贝塔值的初始数组  
cons=np.zeros_like(R_mean)          #生成一个防止线性回归方程的常数项初始数组  
X=R_HS300                            #设置自变量的样本值  
X_addcons=sm.add_constant(X)         #对自变量的样本值增加一列常数项  
for i in range(len(R_mean)):  
    Y=R_fund.iloc[:,i]               #设定因变量的样本值  
    model=sm.OLS(endog=Y,exog=X_addcons) #构建普通最小二乘法线性回归模型  
    result=model.fit()               #生成一个线性回归的结果对象  
    cons[i]=result.params[0]         #生成线性回归方程的常数项数组  
    betas[i]=result.params[1]        #生成基金贝塔值的数组  
    print(R_fund.columns[i],'贝塔值',round(betas[i],4))
```

## 8.5.3 特雷诺比率

第2步：将线性回归的结果进行可视化（见图8-12），具体的代码如下：

```
X_list=np.linspace(np.min(R_HS300),np.max(R_HS300),200) #生成一个X轴的数组
plt.figure(figsize=(11,10))
for i in range(len(R_mean)):
    plt.subplot(2,2,i+1)
    plt.scatter(X,R_fund.iloc[:,i])
    plt.plot(X_list,cons[i]+betas[i]*X_list,'r-',label=u'线性回归拟合',lw=2.0)
    plt.xlabel(u'指数',fontsize=13)
    plt.ylabel(u'基金',fontsize=13,rotation=0)
    plt.xticks(fontsize=13,rotation=30)
    plt.yticks(fontsize=13)
    plt.legend(fontsize=13)
    plt.grid('True')
plt.show()
```



## 8.5.4 信息比率

跟踪误差（Tracking Error, TE）是指投资组合的收益率与基准组合（benchmark）收益率之间差异的标准差，反映了投资组合的主动管理风险。通常而言，跟踪误差是一个比较有效的衡量风险的标尺，可以对投资组合在实现投资者真实投资目标所承担的主动管理风险做出衡量。

投资组合的收益率与基准组合收益率之间的差异称为跟踪偏离度（Tracking Difference, TD），跟踪误差其实就是跟踪偏离度的标准差。

跟踪偏离度的数学表达式如下：

$$TD = E(R_p) - E(R_B)$$

其中，TD 表示跟踪偏离度， $E(R_p)$  含义与前面一致都是表示投资组合的预期收益率， $E(R_B)$  表示根据比较的需要而寻找的基准投资组合预期收益率，可以用指数的过往平均收益率替代。↵

跟踪误差的数学表达式如下：↵

$$TE = \sqrt{\frac{1}{N-1} \sum_{t=1}^N (TD_t - \overline{TD})^2} \quad (8-32) \quad \leftarrow$$

其中，TE 表示跟踪误差， $TD_t$  表示 t 时刻的跟踪偏离度，即  $TD_t = R_{Pt} - R_{Bt}$ ，

$\overline{TD}$  表示跟踪偏离度的均值。↵

## 8.5.4 信息比率

信息比率（Information ratio, IR）就是跟踪偏离度与跟踪误差的比率，该比率从主动管理的角度描述投资组合风险调整后的收益，这也是该比率不同于前面介绍的3个比率（即夏普比率、索提诺比率和特雷诺比率）的最典型特征。信息比率用来衡量投资组合承担主动管理风险所带来的超额收益（相对于基准投资组合收益率），它表示承担每一单位主动管理风险所带来的超额收益率，该比率越高说明每一单位主动风险带来的超额收益率越大，因此信息比率较大的基金业绩表现要优于信息比率较低的基金。

信息比率的数学表达式如下：

$$IR = \frac{TD}{TE} = \frac{E(R_p) - E(R_B)}{TE}$$

通过 Python 自定义一个用于计算信息比率的函数，具体的代码如下

```
def IR(Rp,Rb,TD):  
    """计算信息比率（Information ratio）  
    Rp: 表示投资组合的年化收益率；  
    Rb: 表示基准组合的年化收益率；  
    TD: 表示跟踪误差。"""  
    return (Rp-Rb)/TD
```

## 8.5.4 信息比率

【例8-14】依然沿用前面例8-11的开放式股票型基金的信息演示运用Python计算信息比率，同时在计算投资跟踪偏离度和跟踪误差过程中，运用沪深300指数作为基准组合，具体的过程分为两个步骤。

第1步：计算每个基金的跟踪误差，具体的代码如下

```
TE_fund=np.zeros_like(R_mean) #生成一个放置基金跟踪误差的初始数组
for i in range(len(R_mean)):
    TD=np.array(R_fund.iloc[:,i])-np.array(R_HS300.iloc[:,0]) #生成一个基金跟踪偏离率的数组
    TE_fund[i]=TD.std()*np.sqrt(252) #生成基金的变化跟踪误差数组
    print(R_fund.columns[i],'跟踪误差',round(TE_fund[i],4))
```

第2步：利用前面已经定义计算信息比率的函数IR测算得出每个基金的信息比率，具体的代码如下

```
R_mean_HS300=R_HS300.mean()*252 #计算沪深300指数的年化收益率
R_mean_HS300=np.array(R_mean_HS300) #将沪深300指数年化收益率变成数组
IR_3years=IR(Rp=R_mean,Rb=R_mean_HS300,TD=TE_fund)
print('2016至2018年平均的信息比率\n',round(IR_3years,4))
```

2016至2018年平均的信息比率

景顺长城优质成长基金 -0.2323

汇添富移动互联基金 -1.0925

华宝品质生活基金 -0.1588

中银新动力基金 -1.6901

**Your appreciation makes me a miracle.  
Thank you!**

**Frank Ziwei Zhang  
18117228563  
frank8027@163.com**



**上海對外經貿大學**  
SHANGHAI UNIVERSITY OF INTERNATIONAL BUSINESS AND ECONOMICS