R 语言金融工程中文教程
Financial Engineering with R, Chinese Manual

李智 Li, Zhi

February 4, 2014

Contents

List of Figures

前言

在上小学的时候，有一年暑假回济南，去看望姥爷和姨，姨夫们。那天，三姨和三姨夫在家里招待了我们所有的人。他们家小区的后面是一个没有树的小山包。吃完晚饭，趁着明媚的夕阳，我们又一起去爬山。表哥，我，和表弟，小哥三个，自然地排成了一个小队，向山顶走。旁边路过，有一个老人，他用早上刚升起来的太阳形容我们。转眼二十多年的时间过去了，感谢老人的鼓励，我们兄弟三个都有了属于自己的事业，为祖国的发展做着贡献。

我非常喜欢走过北京机场T3航站楼的出站口。因为那里众多急切接飞机的人们，使我感觉像是一个明星，走在火红的地毯上。但是这次寒假回国，只有老迈的妈妈来接我，爸爸再也不会来了。爸爸是一个老党员，去世前大量写作，他起草完成过几千个技术专利申请的文件。我也要写作，作为一个传统，实现人生价值的传统。

这主要还是一部关于R语言编程应用的著作，最好的学习方法是把所有R语言程序的例子挑出来，运行后知道如何使用就可以了。知识本应该是免费的，我只是没有精力都写出来。本书的每一章都独立成文，分开阅读完全可行。重要的最后关于对冲的一章，是对这个概念的精确接触。如果有对本书的意见提出，请加入与本书同名的QQ群进行讨论，也可以联系作者的QQ2476784986。对冲是一个非常广泛应用，种类繁多的金融方法。再继续进行举例，多写几百页，也不可能结束。基于点到为止，深入浅出的原则，只给读者一点尝试，同时又练习了R语言。

符号注释

r - 净回报率
R - 毛回报率
S - 方差协方差矩阵
S - 股票价格
X - 期权履约价
μ - 期望值
σ^2 - 方差
σ - 标准差，波动性
t - 某一时间点
Δt - 某一个时间段time to maturity
$N()$ - 正态分布概率的函数
log - 对数函数计算
exp - 指数函数计算
Σ - 总和计算

Chapter 1

线性代数(Linear Algebra)

1.1 关于函数

函数(Function)是一个大学一年级的概念，虽然不是金融的有关部分，但它是数学的基础，为了要是这本书能够做到自圆其说，在这里必须要介绍清楚函数。对函数的的精确又简便的描述是：**数据入数据出(Data In Data Out)**。例如有一个名叫$f(x)$的函数：

$$f(x) = 3x^2 \tag{1.1}$$

R code:

```
x=7
f=3*x^2
f
```

在这个函数中，x只是一个用来占位子的符号，也是数据输入的位置。如果输入$x = 7$，这个函数就会根据其带有的表达式进行计算，3×7^2，然后输出$f(7) = 147$的结果。值得注意的是，函数是不可以有模棱两可的输出的，对于已经定义的函数(1.1)，如果输入的数值是7，那么输出的数值只可能是147，不可能是其他的数值。这样就生成了一个从7到147的指向关系，所以函数也被称作映象(Mapping)：

$$7 \to 147$$

但是换一个角度来看，-7也可以通过函数(1.1)指向147(ﬁ也就是说一个函数的输入可以是多个的，却都可以指向同一个输出：

$$\left.\begin{array}{c} 7 \\ -7 \end{array}\right\} \to 147$$

更多的情况下，需要使用函数建立一一对应(Bijection)的映象关系，也就是说，一个输入值只对应一个输出值，而且一个输出值也只能对于一个输入值。例如函数(1.1)，可以取消-7对应147的可能性，只允许7对应147。这就需要定义x的有效范围$x \geq 0$，在这个有效范围之内，所有的x和$f(x)$之间就建立起了一一对应的映象关系：

$$f(x) = 3x^2 \quad x \geq 0 \tag{1.2}$$

数学写法(Notation)十分重要，它是记录数学语言的方式。比方上面用$f(x)$指代函数(1.1)，和使用x作为占位子的符合，甚至使用0来代表什么都没有，都是数学写法。这些写法使困难的事情变得简单，使不可能完成的事变得可能。
复合函数(Composite Function)被记为$f \circ g$，是把一个函数的输出作为另一个函数的输入，然后输出最终结果。具体来讲$g()$是一个函数，它的输入是x。然后$f()$也是一个函数，它的输入是$g(x)$：

$$f \circ g(x) = f(g(x)) \tag{1.3}$$

使用上面函数(1.1)，$f(x) = 3x^2$，然后定义一个新函数$g(x) = x - 8$，只需要把$g()$的结果输入到$f()$的里面，就生成了这个复合函数：

$$f \circ g(x) = f(g(x)) = f(x-8) = 3(x-8)^2$$

R code:

```
x<-7
g<-function(x){x-8}
f<-function(x){3*x^2}
f(g(x))
```

最大值函数(Maximum Function)输出最大值的函数，写法max()。比方说有组数值被存放在了电脑里，需要找出其中最大的数值，就需要使用最大值函数。例如这组数值为： 3.14 1000 0.5 999 -9999 。要找出其中最大值，使用max()可以得出结果：

$$\max(3.14, 1000, 0.5, 999, -9999) = 1000 \tag{1.4}$$

R code:

```
max(3.14 , 1000 , 0.5 , 999 , -9999)
```

在这里等于号$=$可以被理解成输出的意思。

1.2 标量，向量和矩阵

数学家和统计学家之间总存在着一种争论，向量应该是横着的还是竖着的。数学家们喜欢把向量竖着放，统计学家喜欢把向量横着放。他们从来都是不同意对方的做法，认为对方的做法很傻。这两种不同的做法，都有着其中不同的历史和传承。在这里我们不会去关心统计学，所以这里的向量都是竖着放的。

标量(Scalar)只是一个数值，可以用任何符号来代表，大家可以把它理解成只有一行一列的1×1矩阵。

向量(Vector)指的是一列数字，也被称为数列，它是只有一列，但可以有很多行的$1\times n$矩阵。

R code:

```
x=10 #Scalar
y=c(3.14 , 1000 , 0.5 , 999 , -9999) #Vector
```

矩阵(Matrix)是把多个向量放在了一起，这就使矩阵可以是多行多列$m \times n$，这个英文词汇原本和环境、富饶多产的土壤有关系。可以使用R语言，生成几个向量，放在一起，并输出生成的矩阵：

R code:

```
x=c(1,2,3)    #vector
y=c(4,5,6)    #vector
z=c(7,8,9)    #vector
A=cbind(x,y,z)#matrix
A
```

矩阵转置(Transpose)是把矩阵里每一列的数字写在行里，写法是在代表矩阵的符号右上角画上一个小T。例如上面语句中生成的矩阵A，它的转置就是A^T：

$$A=\begin{vmatrix}1&4&7\\2&5&8\\3&6&9\end{vmatrix}\qquad A^T=\begin{vmatrix}1&2&3\\4&5&6\\7&8&9\end{vmatrix}\tag{1.5}$$

R code:

```
x=c(1,2,3)   #vector
y=c(4,5,6)   #vector
z=c(7,8,9)   #vector

A=cbind(x,y,z)#matrix
t(A)    #traspose
```

向量和常量都属于矩阵，所以向量和常量都可以被转置。竖着的向量转置后就成了横着的向量，横着的向量被转置后就成了竖着的向量。常量的转置还是常量，没有变化。

$$x = \begin{vmatrix} 1 \\ 2 \\ 3 \end{vmatrix} \qquad x^T = \begin{vmatrix} 1 & 2 & 3 \end{vmatrix}$$

$$b = \begin{vmatrix} 1 & 3 & 5 \end{vmatrix} \qquad b^T = \begin{vmatrix} 1 \\ 3 \\ 5 \end{vmatrix}$$

$$a = |10| \qquad a^T = |10|$$

对称(Symmetry)是说，如果一个转置后的矩阵$m \times m$，必须是正方形的矩阵，与其没有转置的时候相同，被称为对称矩阵。下面就是一个用S代表的对称矩阵，无论如何转置，都不会有变化。

$$S = \begin{vmatrix} 1 & 2 & 3 \\ 2 & 5 & 4 \\ 3 & 4 & 6 \end{vmatrix}$$

加法(Addition)，矩阵的加法，是把矩阵中有相同位置的数值，对应相加，这也就使相加的矩阵之间必须有相同的行数和列数，例如：

$$x + y = \begin{vmatrix} 1 \\ 2 \\ 3 \end{vmatrix} + \begin{vmatrix} 4 \\ 5 \\ 6 \end{vmatrix} = \begin{vmatrix} 1+4 \\ 2+5 \\ 3+6 \end{vmatrix} = \begin{vmatrix} 5 \\ 7 \\ 9 \end{vmatrix}$$

R code:

```
x=c(1,2,3)   #vector
y=c(4,5,6)   #vector
x+y
```

$$A+B = \begin{vmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{vmatrix} + \begin{vmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{vmatrix} = \begin{vmatrix} 1+1 & 4+1 & 7+1 \\ 2+2 & 5+2 & 8+2 \\ 3+3 & 6+3 & 9+3 \end{vmatrix} = \begin{vmatrix} 2 & 5 & 8 \\ 4 & 7 & 10 \\ 6 & 9 & 12 \end{vmatrix}$$

R code:

```
x=c(1,2,3)   #vector
y=c(4,5,6)   #vector
z=c(7,8,9)   #vector
A=cbind(x,y,z)#matrix
B=cbind(x,x,x)#matrix
A+B
```

一个常量可以和任何行数和列数的矩阵相加，尽管常量会与矩阵的行数和列数不同，只需要把矩阵中的所有数字，与这个常量相加，例如：

$$a + A = 10 + \begin{vmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{vmatrix} = \begin{vmatrix} 10+1 & 10+4 & 10+7 \\ 10+2 & 10+5 & 10+8 \\ 10+3 & 10+6 & 10+9 \end{vmatrix} = \begin{vmatrix} 11 & 14 & 17 \\ 12 & 15 & 18 \\ 13 & 16 & 19 \end{vmatrix}$$

R code:

```
a=10
A=cbind(x,y,z)#matrix
a+A
```

1.3 矩阵乘法和矩阵逆

点乘(Dot Product)，指的是两个相同长度的向量，先把相同位置的数值相乘，然后把所有的乘积加起来做和。这种向量之间的乘法的写法是一个圆点，所以叫做点乘，例如：

$$x \cdot y = \sum \begin{vmatrix} 1 \\ 2 \\ 3 \end{vmatrix} \cdot \begin{vmatrix} 4 \\ 5 \\ 6 \end{vmatrix} = \sum \begin{vmatrix} 1 \times 4 \\ 2 \times 5 \\ 3 \times 6 \end{vmatrix} = \sum \begin{vmatrix} 4 \\ 10 \\ 18 \end{vmatrix} = 4 + 10 + 18 = 32$$

R code:

```
x=c(1,2,3)   #vector
y=c(4,5,6)   #vector
x%*%y
```

但是，平时书写点乘的时候，需要把两个向量里，处在前面的向量横过来，这是从解决线性方程中来的传统，例如使用传统方法的点乘：

$$x \cdot y = \begin{vmatrix} 1 & 2 & 3 \end{vmatrix} \cdot \begin{vmatrix} 4 \\ 5 \\ 6 \end{vmatrix} = 1 \times 4 + 2 \times 5 + 3 \times 6 = 32$$

使用传统方法的点乘，行与列之间的相乘相加，还可以定义矩阵乘以向量，做法如下：

$$A \cdot x = \begin{vmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{vmatrix} \cdot \begin{vmatrix} 1 \\ 2 \\ 3 \end{vmatrix} = \begin{vmatrix} 1 \times 1 + 4 \times 2 + 7 \times 3 \\ 2 \times 1 + 5 \times 2 + 8 \times 3 \\ 3 \times 1 + 6 \times 2 + 9 \times 3 \end{vmatrix} = \begin{vmatrix} 30 \\ 36 \\ 42 \end{vmatrix}$$

R code:

```
A%*%x
```

矩阵与矩阵之间的相乘，也是用行乘以列来定义，结果是一个矩阵。矩阵的相乘是复合函数(1.3)在矩阵上的应用:

$$A \cdot A = \begin{vmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{vmatrix} \cdot \begin{vmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{vmatrix} = \begin{vmatrix} 30 & 66 & 102 \\ 36 & 81 & 126 \\ 42 & 96 & 150 \end{vmatrix}$$

R code:

```
A%*%A
```

从上面一系列的R代码中可以清楚的看到，点乘的R语言命令，使用$\% * \%$来代表，并完成运算的。

矩阵逆(Matrix Inverse)，指的是使用一个矩阵的逆，和这个矩阵相乘，乘积是一个只在左右对角线上是1，其他位置是0的矩阵。矩阵逆的写法，是把减一符号放在表示矩阵符号的右上角。找到一个矩阵的逆不是一件容易的事，但R语言提供了$solve()$函数来解决这个问题。在下面的例子里，可以看到，一个矩阵乘以它的逆，结果的确是一个0,1矩阵:

$$A \cdot A^{-1} = \begin{vmatrix} 1 & 4 & 7 \\ 2 & 0 & 8 \\ 3 & 6 & 9 \end{vmatrix} \cdot \begin{vmatrix} -0.8 & 0.1 & 0.5333 \\ 0.1 & -0.2 & 0.1 \\ 0.2 & 0.1 & -0.1333 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

R code:

```
A=matrix(c(1,4,7,2,0,8,3,6,9),nrow=3,ncol=3,byrow = TRUE)
solve(A)
A%*%solve(A)
```

满秩矩阵(Full Rank Matrix)，并不是所有的正方形矩阵都有逆，这也是把0放入上面A矩阵的原因。只有满秩矩阵才有逆。因为有时候在矩阵中，有些行或列可以使用其它的行和列表达出来，能被其它行和列表达出来的，就不带有比其它行和列更多的信息。所以矩阵中所有的行、列，都不能被其它行、列表达，就说明这个矩阵有足够的信息，也就是满秩，还可以被称为非奇(Non-singular)。

笛卡尔乘积(Cartesian Product)，指的是两个向量相乘，用一个竖直的向量，乘以另一个横躺的向量。不是相同位置的数值相乘再作和。而是每一个数值，都与另一个向量里的数值相乘:

$$x \times y = \begin{vmatrix} 1 \\ 2 \\ 3 \end{vmatrix} \times \begin{vmatrix} 4 & 5 & 6 \end{vmatrix} = \begin{vmatrix} 1 \times 4 & 1 \times 5 & 1 \times 6 \\ 2 \times 4 & 2 \times 5 & 2 \times 6 \\ 3 \times 4 & 3 \times 5 & 3 \times 6 \end{vmatrix}$$

笛卡尔乘积的符号是一个小差号，笛卡尔乘积实际上表示的是计算机程序算法中的for-loop循环。

1.4 线性方程求解

多元的线性方程，是每个高中学生都有能力解决的问题。请看下面的例子：

$$
\begin{aligned}
3x + 6y + 2z &= 5 \\
2x + 5y + 4z &= 11 \\
26x + 59y + 36z &= 7
\end{aligned}
\tag{1.6}
$$

可以看出这组线性方程是没有解的。因为第一个方程中的系数乘以4，再加上第二个方程的系数乘以7，就是第三个方程的系数。也就是说第三个方程可以用第一、二个方程来代表。所以这组线性方程是奇异的，是不满秩的。

再举一个满秩例子，来学习使用R语言解线性方程：

$$
\begin{aligned}
x + 4y + 7z &= 11 \\
2x + 8z &= 9 \\
3x + 6y + 9z &= 5
\end{aligned}
\tag{1.7}
$$

把这组线性方程写成矩阵形式：

$$
\begin{vmatrix} 1 & 4 & 7 \\ 2 & 0 & 8 \\ 3 & 6 & 9 \end{vmatrix} \cdot \begin{vmatrix} x \\ y \\ x \end{vmatrix} = \begin{vmatrix} 11 \\ 9 \\ 5 \end{vmatrix}
$$

等号左右两边都乘以系数矩阵的逆：

$$
\begin{vmatrix} 1 & 4 & 7 \\ 2 & 0 & 8 \\ 3 & 6 & 9 \end{vmatrix}^{-1} \cdot \begin{vmatrix} 1 & 4 & 7 \\ 2 & 0 & 8 \\ 3 & 6 & 9 \end{vmatrix} \cdot \begin{vmatrix} x \\ y \\ x \end{vmatrix} = \begin{vmatrix} 1 & 4 & 7 \\ 2 & 0 & 8 \\ 3 & 6 & 9 \end{vmatrix}^{-1} \cdot \begin{vmatrix} 11 \\ 9 \\ 5 \end{vmatrix}
$$

$$
\begin{vmatrix} x \\ y \\ z \end{vmatrix} = \begin{vmatrix} 1 & 4 & 7 \\ 2 & 0 & 8 \\ 3 & 6 & 9 \end{vmatrix}^{-1} \cdot \begin{vmatrix} 11 \\ 9 \\ 5 \end{vmatrix} = \begin{vmatrix} -5.23 \\ -0.2 \\ 2.43 \end{vmatrix}
$$

R code:

```
A=matrix(c(1,4,7,2,0,8,3,6,9),nrow=3,ncol=3,byrow = TRUE)
f=c(11,9,5)
solve(A)%*%f
```

1.5 二次形式

二次形式(Quadratic Form)是把多元二次方程用矩阵的写法表达出来，多元二次方程和二次形式实际上是同一个东西，只是它们的书写形式有所区别。例如，这里是最普通的二元二次方程：

$$x^2 + 2xy + y^2$$

二元指的是两个变量x、y，二次指的是每一项的次方为二。表达式(1.8)可以被写成矩阵形式：

$$\begin{vmatrix} x & y \end{vmatrix} \cdot \begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix} \cdot \begin{vmatrix} x \\ y \end{vmatrix}$$

所有的二元二次方程都可以把变量提出来，写成一个向量，用x代表。并且把系数提出来写成一个对称的正方形矩阵，用A代表。二次形式就成为了向量和矩阵之间的点乘：

$$x^T Ax \tag{1.8}$$

R code:

```
t(x)%*%A%*%x
```

表达式(1.8)是二次形式的准确表述。可以看出，二次形式只有3个部分，一个向量横过来，乘以一个正方形的矩阵，再乘以这个向量竖着放。表达式(1.8)并没有要求正方形的矩阵A是对称的，但是为了要使二次形式整齐好看，所有的教科书都会额外规定矩阵A是对称的。矩阵A装载的是多元二次方程里系数的信息，不改变多元二次方程的系数，矩阵A是完全可以被写为对称的。

Chapter 2

回报率（Returns）

2.1 回报率的计算

净回报率(Net Return)，可以被想象成银行的存款利息。比方说银行的存款年利息是6%，存入本金(Deposit)$10000，一年后再取出来，存款就变成了$10600。如果说这$10000是一笔投资，6%就是这笔投资的回报率。这一正本书都是关于回报率的描述，为了体现回报率的重要性，在编程和写作中，我们使用一个专门的英文字母，小写的r指代回报率。这个银行存款的例子里$r = 0.06$。

收益(Revenue)，指的是经过投资，增长出的，以前本金中没有的数额。收益可以是正值或负值，正值说明投资挣到了钱，负值说明投资赔了钱。在上面存款的例子里，收益是$600。写成数学语言：$Revenue = r \times Deposit = 0.06 \times 10000 = 600$。

总回报率(Gross Return)，也被称为毛回报率，指的是经过投资，结果的总数与本金之间的比率。总回报率的计算十分简便，只需用1+净回报率。我们也使用一个专门的英文字母，大写的R来代表总回报率。在这个银行存款的例子里，总回报率是1.06。写成数学语言：$R = 1 + r = 1 + 0.06 = 1.06$。

复利(compound return)，比方说本金仍旧是10000，银行年利息0.06，经过一年的存款，把全部的金额10600再次存入银行，去挣0.06的年利。又过了一年，再把当年的本金加利息，统统存入银行，又可以有0.06的利息。这就叫做计复利，或者复利存款。在这里，每一年都有1.06的总回报率。最终的总回报率，是把每一年的总回报率都相乘起来。用本金乘以最终的总回报率，就是最终存款的总结果。用n代表复利存款的年数，复利最终的总回报率可以写成:

$$R = (1 + r) \times (1 + r) \times (1 + r) \times \cdots = (1 + r)^n \tag{2.1}$$

代入例子里的数值，年利0.06，存了10年，最终总回报率：

$$R = (1 + 0.06) \times (1 + 0.06) \times (1 + 0.06) \times \cdots = (1 + 0.06)^{10} = 1.790848$$

10年后复利存款的总数为：

$$Total = Deposit \times R = 10000 \times 1.790848 = 17908.48$$

进一步观察公式(2.1)可以发现，$\log(R) = \log(1.790848) = 0.5826891$，而且$10\log(1+0.06) = 0.5826891$，两种算法得到了相同的结果。因为exp是log的反函数，把exp用在$10\log(1+0.06)$上，也可以得到1.790848的结果。我们就有了另外一种计算复利最终总回报率的公式：

$$R = \exp\sum_{i=1}^{n}\log(1 + r) \tag{2.2}$$

连续复利(continuous return)可以被想象成，存款的期限被切成了无数尽可能小的时间段，回报率也随着这些小的时间段被切成了小块。用这些小的时间段，和小块的利息，来计算复利。还是使用上面年利0.06的例子，这一年被切成了无数非常小的时间段，每一个小时间段都有一个小块的利息。每一个小时间段过后，就把本金和利息全部复利再存到下一个小时间段，直到一年结束。问像这样进行了无数次小复利，一年终总回报率是多少。可以用连续复利公式计算：

$$R = \exp(rt), \quad 0 \le t \le 1 \tag{2.3}$$

公式(2.3)里的t代表的不是一般的时间，而是在总时间期限中的比例。如果总期限是一年，存了半年，只占总期限的0.5，那么$t = 0.5$。如果总期限是一年，存了9个月，占总期限的0.75，那么$t = 0.75$。当然，如果总期限是一年，存了12个月，$t = 1$。还有r是总期限到期，不计复利的回报率。我们使用$r = 0.06$，$t = 1$，计算一年期连续复利：

$$R = \exp(rt) = e^(0.06 \times 1) = 1.061837 \tag{2.4}$$

回报率的计算R代码

```
r=.06
deposite=10000

# Revenue
r*10000
```

```
# Gross return
R=1+r

# Total
(1+r)*10000

# Compound Interest for 10 Years
year=1:10
deposit=10000
gross_return=(1+.06)^year
total=gross_return*10000
cbind(year,gross_return,total)

#Compound Interest by Sum
log(1.790848 )
10*log(1+0.06)
exp(10*log(1+0.06))

#Continuous Return
r=0.06
dt=1
exp(r*dt)
```

2.2 房产的价格

市场价格(Market price)，一个资产会随着市场的变化而改变价格，在这里我们用房地产为例，来展示资产的市场价格和回报率是如何计算的。比方说，有个人买了一幢住宅，买入的价格为30万美元。他持有这所房产6年，第一年房产市场上涨了13%，第二年房产市场又上涨了5%，第三年房产市场没有任何变化，第四年房产市场下跌了1%，第五年房产市场又下跌了2.5%，第六年房产市场突然大涨19%。我们用一个数列$r = (.13, .05, 0, -.01, -.025, .19)$代表房产市场的变化，因为这些变化都是我们所称的回报率。使用$R_yr = 1 + r$代表每一年的总回报率，在银行存款的例子里每一年的总回报率是不变的1.06，但R_yr是随着市场变化的。我们使用作log和exp的公式(2.2)把R_yr乘起来作复利。

代码中有使用到cumsum()函数，这是作积累和的函数。比方说，数列中第一个数字是1，它自己的积累是它本身。然后2出现了，2要和1积累到一起，结果是3。然后3出现了，3要和1,2积累到一起，结果是6。然后4出现了，4要和1,2,3积累到一起，结果是10。然后5出现了，5要和1,2,3,4积累到

```
> cbind(year,r,R_yr,log_R_yr,sum,R,price)
     year      r  R_yr    log_R_yr       sum        R    price
[1,]    1  0.130 1.130  0.12221763 0.1222176 1.130000 339000.0
[2,]    2  0.050 1.050  0.04879016 0.1710078 1.186500 355950.0
[3,]    3  0.000 1.000  0.00000000 0.1710078 1.186500 355950.0
[4,]    4 -0.010 0.990 -0.01005034 0.1609575 1.174635 352390.5
[5,]    5 -0.025 0.975 -0.02531781 0.1356397 1.145269 343580.7
[6,]    6  0.190 1.190  0.17395331 0.3095930 1.362870 408861.1
> 
```

Figure 2.1: 房产的市场价格

一起，结果是15。使用下面的R代码，可以看到这样的结果：

```
x=c(1,2,3,4,5)
cumsum(x)
```

我们使用大写的R，代表计复利后每年的总回报率。市场价格计算为$price = 300000 \times R$。来看用数据表格方式输出的结果，截图(2.2)一共6行数据，每一行代表一年。小r列是房产市场的每年回报率，大R列是房产市场计算复利的每年总回报率。房产市场在第一年上升了0.13，第一年的总回报率是1.13，这个房产在第一年达到价值\$339000。到了第六年，通过复利计算，房产市场的总回报率比较刚购入房产的时候，上升至1.36，这个房产的价值也达到了\$408861.1。

收房租就是分红(Renting AKA Dividends)，房产出租可以增加这个房产投资的回报。仍旧是这个购入价格\$300000的房产，在购买者的手里持有了6年，房产市场每一年的回报还是承袭上面的计算。这个人将这所房产以\$1200每月的价格出租了出去，每年可以得到\$14400的房租。要求计算6年后这所房产投资的回报率。很明显，房租要给这个投资积累回报，6年的房租总数是\$86400。再加上这所房产第六年的市价\$408861.1，在这个投资上积累的总财富就达到了\$408861.1 + \$86400 = \$495261.1，第六年的总回报率，算上房租，为\$495261.1/300000 = 1.6509。这个投资者很幸运，房产买入时价值\$300000，到第六年房产价值就上升到了\$408861.1，是原价的1.36倍。六年来累积的房租价值为\$86400，所以总回报率是1.65，一个不小的数字。请参看截图(2.2)。

房价计算R代码

```
# Price of a Townhouse
price0=300000 #US Dollars
year=1:6
```

```
> cbind(price,rent,value,R)
        price  rent    value        R
[1,] 339000.0 14400 353400.0 1.178000
[2,] 355950.0 28800 384750.0 1.282500
[3,] 355950.0 43200 399150.0 1.330500
[4,] 352390.5 57600 409990.5 1.366635
[5,] 343580.7 72000 415580.7 1.385269
[6,] 408861.1 86400 495261.1 1.650870
> 
```

Figure 2.2: 收房租就是分红

```
r=c(.13,.05,0,-.01,-.025,.19)
R_yr=1+r
log_R_yr=log(R_yr)
sum=cumsum(log_R_yr)
R=exp(sum)
price=R*price0
cbind(year,r,R_yr,log_R_yr,sum,R,price)


#Renting the House AKA Dividends
1200 #monthly rent
12*1200 #yearly rent
rent=cumsum(rep(14400,6))
value=price+rent
R=value/price0
cbind(price,rent,value,R)
```

2.3 杠杆原理

金融杠杆(Leverage)可以被想象成贷款买房。比方说，购买一所价值$300000的房产，需要20%的保证金(Margin)，也就是首付$60000，剩下的$240000是从银行里借的。这样买房的人就用$60000的资金，撬动了$300000的房产，比例是1:5。

如果房产市场上升了0.1，这个房产的价值就涨到$330000，带来的收益是$30000。这个收益是保证金的0.5。房产市场只上涨了0.1，使用杠杆使得收益放大了5倍。如果房产市场下降了-0.1，杠杆还是会使收益放大5倍，使

用保证金进行投资的回报率成了-0.5。

所以，我们用大写的L代表杠杆比例，用小r代表市场的回报率，用r_L代表凭借保证金投资的回报率：

$$r_L = L \times r \tag{2.5}$$

这个例子里，$L = 5$ $r = 0.1$，计算r_L：

$$r_L = L \times r = 5 \times 0.1 = 0.5$$

值得注意的是，如果房产市场下降了-0.2，$r_L = 5 \times (-0.2) = -1$，就是说买房产的保证金在市场小幅变化中全部损失掉了。银行就会为了保证其资产的安全，将这个房产根据市场价格$240000收归银行所有。

杠杆原理R代码

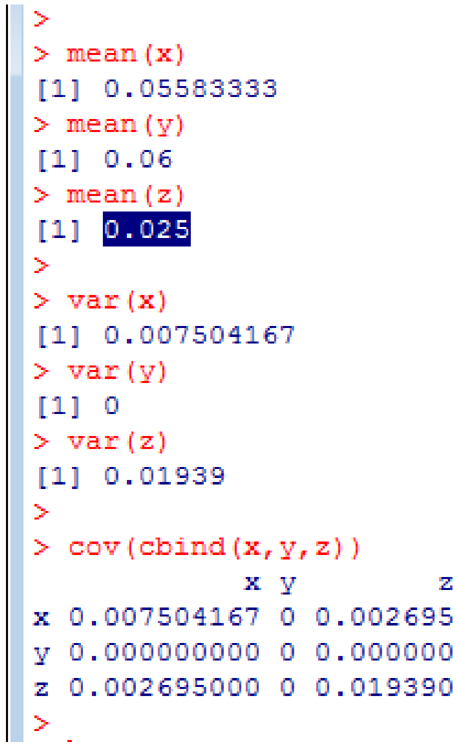
```
#Leverage
0.20*300000
0.80*300000

L=1/0.2 #ratio
r=0.1   #market change

300000*r
300000*r/60000

r_L=L*r #return on leverage
```

2.4 回报率的估计

我们的投资者有3个不同的资产。第一个是他的房产，我们用小x代表房产每年的市场回报率，在上面已经给出了。他的第二个资产是在银行里的存款，每年0.06的利息，6年没有变化，用y来代表。他的第三个资产是购买的一只股票，用z来代表，这只股票先涨后跌，走势十分险峻，下面给出了股票六年中每年的回报率：

$$\begin{aligned} x &= (.13, .05, 0, -.01, -.025, .19) \qquad (2.6) \\ y &= (.06, .06, .06, .06, .06, .06) \\ z &= (.09, .1, .2, -.2, -.05, .01) \end{aligned}$$

平均值(Average),在这里我们使用的平均回报率，是每一个回报率数列的平均值。房产的平均回报是x的平均值0.05583333。存款的平均回报是y的



```
>
> mean(x)
[1] 0.05583333
> mean(y)
[1] 0.06
> mean(z)
[1] 0.025
>
> var(x)
[1] 0.007504167
> var(y)
[1] 0
> var(z)
[1] 0.01939
>
> cov(cbind(x,y,z))
            x y        z
x 0.007504167 0 0.002695
y 0.000000000 0 0.000000
z 0.002695000 0 0.019390
>
```

Figure 2.3: 回报率的估计

平均值0.06。股票的平均回报是z的平均值0.025。在R中可以使用mean()函数计算。

方差（Variance）的概念用来描述一组数据偏离平均值的范围大小。我们这里的数据是回报率，所以计算方差，就是计算回报率变化范围的大小。方差越大，回报率的变化范围也越大。而且方差永远是大于0的，不可能为负值。在R中方差用var()函数计算。

方差-协方差矩阵（Variance-Covariance Matrix），在二次形式的介绍中，我们提到了一个对称矩阵A。这里的方差-协方差矩阵就是二次形式中的对称矩阵A。而且在后面内容中用到二次形式的时候，就必须使用方差-协方差矩阵。方差-协方差矩阵是对称的，而且上面计算的方差数值，全部都放在了方差-协方差矩阵的对角线上。在R中这个矩阵使用cover()函数计算。

回报率的估计R代码

```
#return estimations
x=c(.13,.05,0,-.01,-.025,.19)
```

```
y=c(.06,.06,.06,.06,.06,.06)
z=c(.09,.1,.2,-.2,-.05,.01)

mean(x)
mean(y)
mean(z)

var(x)
var(y)
var(z)

cov(cbind(x,y,z))
```

2.5 预测回报率

做预测有很多种方法，可以使用历史数据，听取专家的建议，或者是自己的凭空猜测，都可以用来预测未来。但预测的结果是否精确，就要根据使用的方法，大相径庭了。总而言之预测是一个非常复杂的应用数学问题。这里我们只有位置介绍一种叫做Garch的模型，并且提供相关的R语言代码。

这一小节的描述有三个目的，使用tseries程序包读取欧元对美金(EURUSD)的外汇牌价，通过牌价计算回报率数列，解读Garch模型输出的截图。

这小节的代码，需要调用fgarch和tsereis两个程序包，安装和调用R程序包请参看其它教程，这里不再介绍。R函数get.hist.quote()可以从因特网上抓取众多的股票，外汇之类的金融牌价，这里只是取得了2011-2-1到2012-2-1的欧元对美金的外汇价格。然后做了一个简单的价格走势图(2.5)。

因为这列牌价数据一共有366个，是每日的牌价，我们计算的回报率，也是相对前一天的每日回报率。计算中采取(2.2)中log的策略，因为log后的牌价，再相减就相当于相除。所以我们把所有的牌价先做log，然后用后一天的log牌价减去前一天的log牌价，再用exp函数做回来，就成了每日的总回报率，再减一得到回报率。计算中的相减是用$diff()$函数完成的，它的作用就是将后一个前去前一个的意思。用p_{i+1}表示后一个牌价，用p_i表示前一个牌价，每日的回报率就可以表示为：

$$r_i = \exp\{\log(p_{i+1}) - \log(p_i)\} - 1 \tag{2.7}$$

在建立Garch模型之前必须要检查数据的质量，就是要查看回报率QQ图(2.5)和Garch模型QQ图(2.5)。主要是看图上的数据点是否处在一条直线上，如果在一条直线上就说明正态分布。

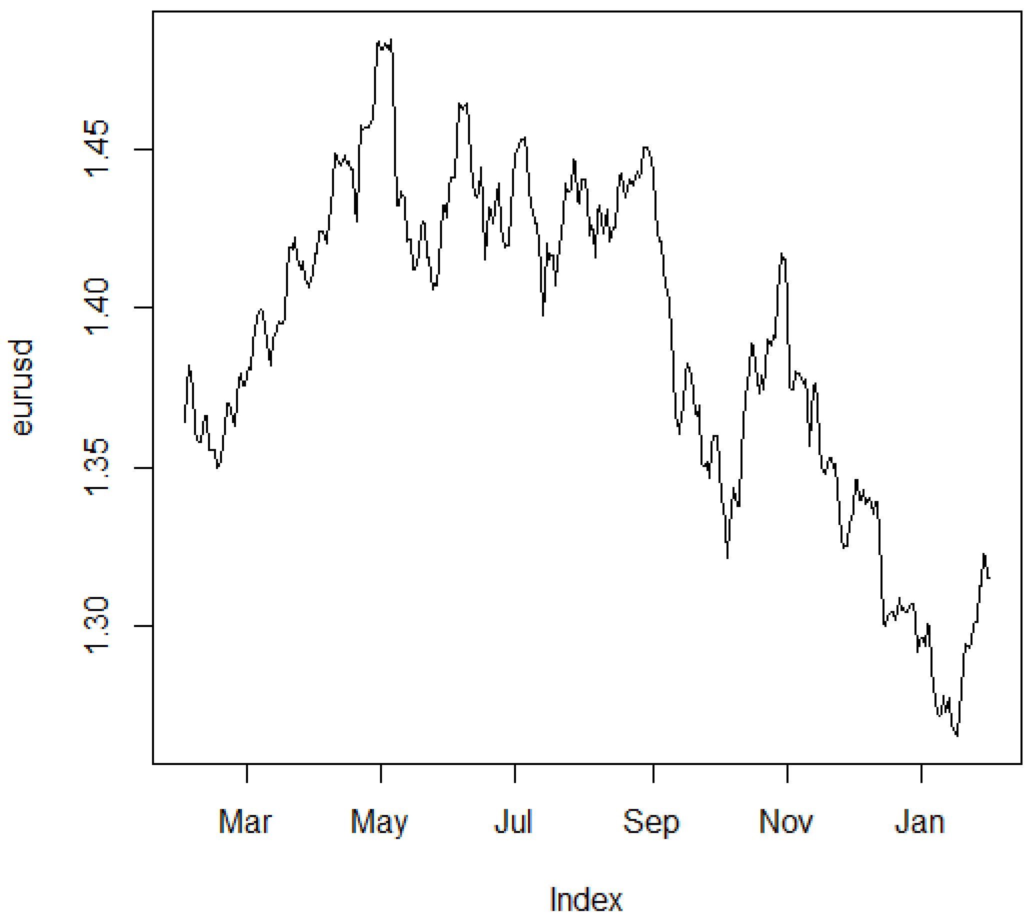


Figure 2.4: 欧元/美金牌价2011-2012

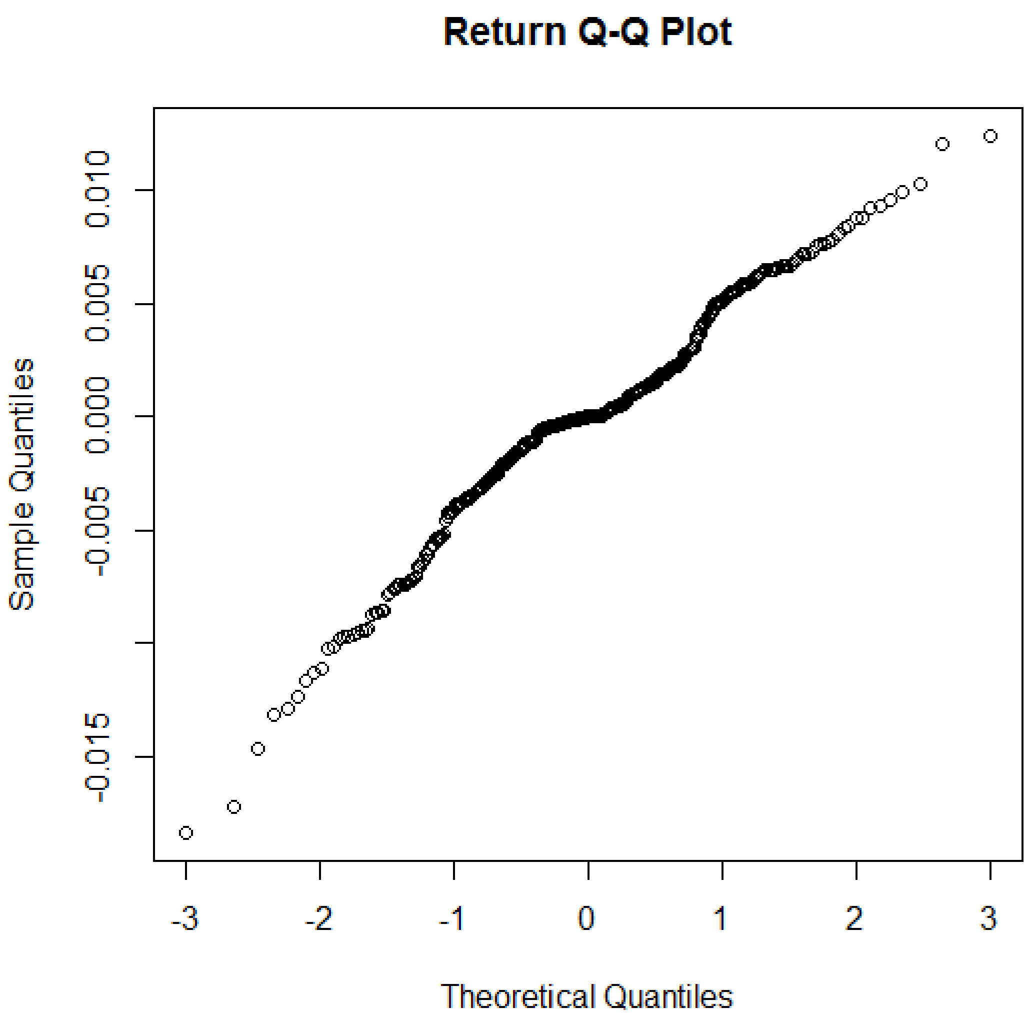


Figure 2.5: EURUSD回报率QQ图

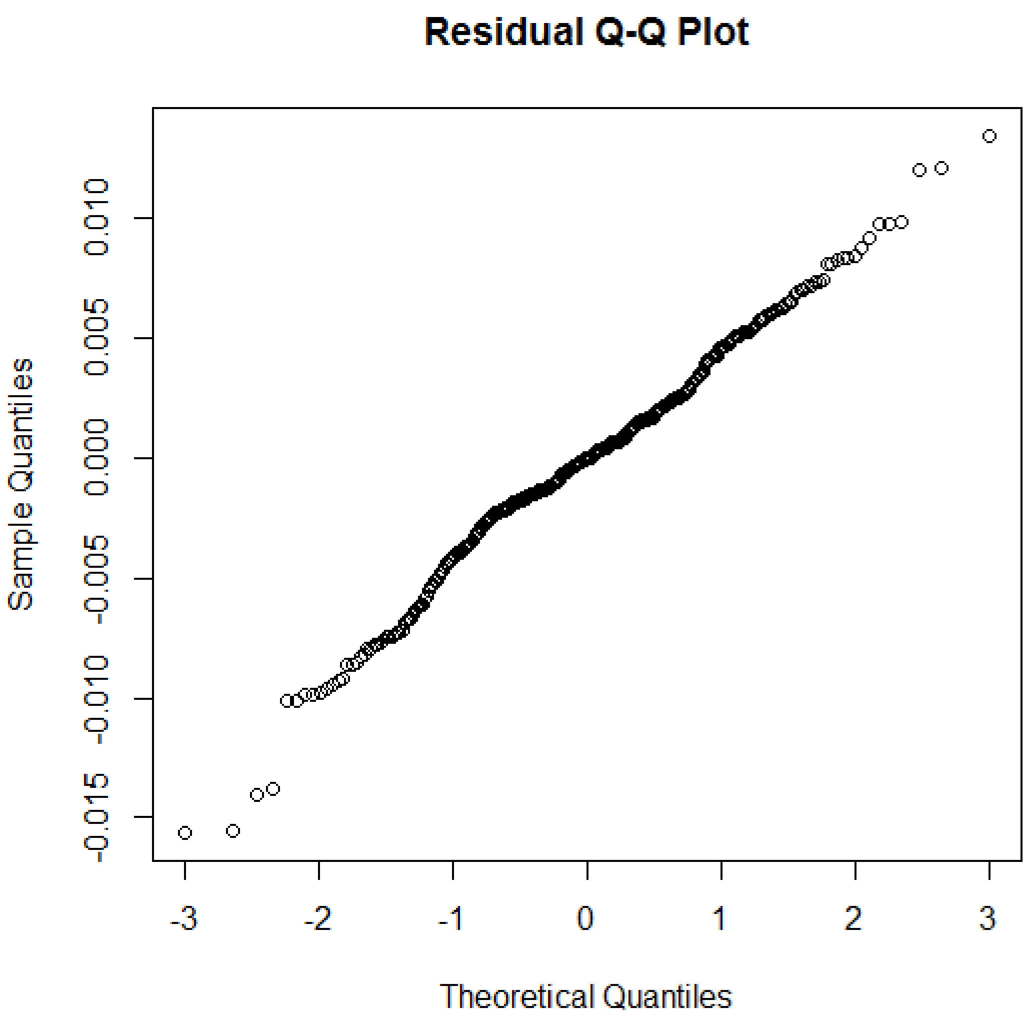


Figure 2.6: EURUSD Garch 模型QQ图

最后是Garch模型画出的预测图(2.5)，用信心区间的形式预测2012-2-1以后的20天的回报率。在r数列中存在着365个日回报率数据，但在预测图上用黑线部分只画出了最后的90个日回报率。可以看到日回报率在水平0线的上下跳动，非常接近一个随机的过程。图中红线的部分是对回报率均值的预测，未来20天回报率均值是处在0的位置上的。图中蓝色线划出了一个上限范围，绿线划出了一个下限范围，是指未来的日回报率将会随机的出现在这两个范围之间，我们对这个推论有95%信心。

Garch模型预测回报率R代码

```
library(fGarch)
library(tseries)

eurusd <- get.hist.quote(instrument = "EUR/USD", provider =
 "oanda",end="2012-02-01",
start = "2011-02-01",)
plot(eurusd)

eurusd=as.numeric(eurusd)
r=exp(diff(log(eurusd)))-1

qqnorm(r,main="Return Q-Q Plot")
plot(density(r))
acf(r)

model = garchFit(~arma(1,0)+garch(1,1),cond.dist="std",data=r,
 prediction.interval=T)
qqnorm(model@residuals,main="Residual Q-Q Plot")

predict(model,n.ahead=20,plot=TRUE) # T 95% interval
```

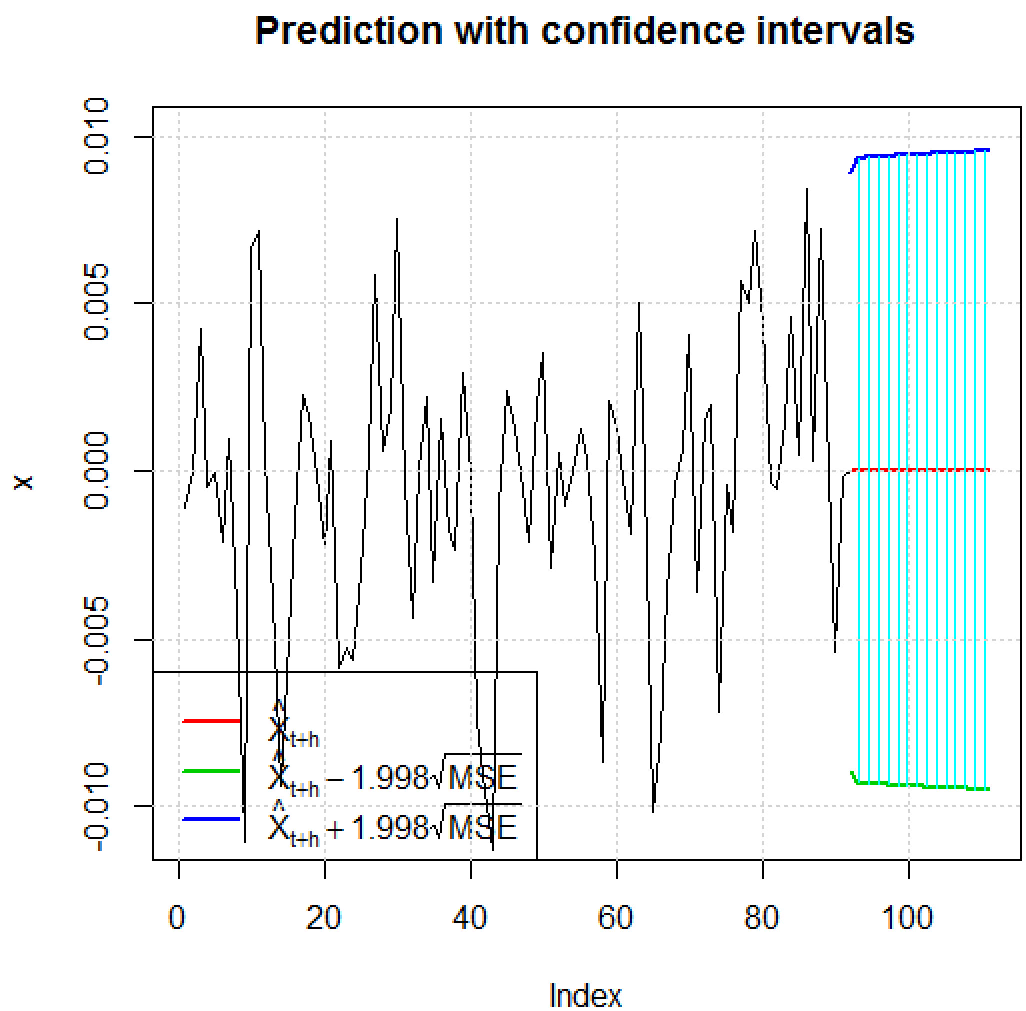


Figure 2.7: 欧元/美金回报率预测

Chapter 3

债券种类(Bonds)

什么是债券（Bond）？债券其实就是银行存款和贷款，数值为正就是存款，数值为负就是贷款，关键在于利率是固定的。零息债券是计复利的存款和贷款，息票债券是不计复利的存款和贷款。

3.1 零息(Zero)息票（Coupon）债券

在计算机网络存在以前，定期债券都是用纸张印刷的凭证。上面会印有证书的签发日期，和到期日期，还会清楚的印有债券的面值。如果证券到了期，持有人就可以去金融机构把证券兑换成相等数量的钞票。在金融界，恒定不二的法则，必须花钱买收益。债券的定价，就成为了，不断被讨论和更新的问题。我们先讨论一个简单的例子，叫零息债券。

比方说，有人向你兜售一张面值$1万元的零息债券，10年后到期，换句话说，凭着这张债券，10年后可以到金融机构里去换1万元的钱。问你会用多少钱来买这张零息债券。你回答说，这完全取决于我的银行存款模式:

1. 现在的银行年利率是0.06，使用每年计算复利，存款10年，只需要$5583.948就可以拿到1万元。

$$\frac{10000}{(1+0.06)^{10}} = 5583.948$$

```
10000/1.06^10
```

2. 仍旧使用年利率0.06，如果银行允许每半年计算复利，存款10年，只需要$5536.758就可以拿到1万元。

$$\frac{10000}{(1+0.03)^{20}} = 5536.758$$

```
10000/1.03^20
```

3. 仍旧使用年利率0.06，如果银行允许计算连续复利，存款10年，只需要\$5488.116就可以拿到1万元。

$$\frac{10000}{\exp\{0.06 \times 10\}} = 5488.116$$

```
10000/exp(0.06*10)
```

值得注意的是，零息债券本质上是银行的复利存款，因为它们的算法都是相同的。而且零息债券也向我们揭示了贬值的概念，如果有1万元，不存在银行里保值，10年以后，只相当于10年前的\$5583.948，按第一个存款模式计算。

比方说，一年后银行的存款利息上升到0.07，问如果希望卖出这个债券，但还有9年到期，按第一种银行存款模式计算，价格是多少：

$$\frac{10000}{(1+0.07)^9} = 5439.337$$

```
10000/(1.07)^9
```

由于银行利率的提高，零息债券的价格下降了。

下面再来解释息票债券。与零息债券不同的是，息票债券会在票面上印有每年付给持有者的利息金额。一张1万元的息票债券，10年到期，票面上印有每年付给持有者\$600的利息。是说这个票面利率为0.06，到期后还可以向金融机构兑换1万元的钞票。你会说，这不就是不计复利的银行存款吗？如果银行现在的利率为0.06，用不计复利的方式计算，这张息票债券的价格是1万元。但问题是，现在银行年利率为0.07，高于息票债券的票面利率。如何计算这张息票债券的价格。

在银行里存款1年拿到\$600，和存款2年拿到\$600，所需的钱数是不一样的。这张债券允许我们，连续10年，每年有\$600的收入。从银行存款的视角来看，我们在同一时间存了11笔钱。第一笔存了1年，取出来是\$600元。第二笔存了2年，取出来还是\$600。直到第十笔，存了10年，取出的时候还是\$600。最后还有一笔钱，存了10年，取出来的时候正好是1万元。所以这张息票债券的价格是这11笔钱的总和：

$$\sum_{t=1}^{10} \frac{600}{(1+0.07)^t} + \frac{10000}{(1+0.07)^{10}} = 9297.642$$

```
t=1:10
a=600/1.07^t
b=10000/(1.07)^10
sum(a,b)
```

由于银行存款利率高于债券票面利率，这张债券的价格低于债券面值。

向大家提供一个统一的公式来计算这两种债券的价格：

$$\sum_{t=1}^{T}\frac{C}{(1+r)^t}+\frac{PAR}{(1+r)^T} \tag{3.1}$$

$$=\frac{C}{r}\{1-(1+r)^{-T}\}+\frac{PAR}{(1+r)^T} \tag{3.2}$$

$$=\frac{C}{r}+\{PAR-\frac{C}{r}\}(1+r)^{-T} \tag{3.3}$$

C代表息票债券的年利息，指的是上面例子里的600。PAR代表债券的面值，是例子里的10000。小写的r代表银行利率，不是债券的票面利率。大写的T代表债券的总时间，例子里给的是10年。小写的t代表，第一年，第二年，第三年$\cdots$ 公式（3.1）是我们在例子里使用的计算方法，与公式（3.2,3.3）划等号。可以用几何级数的方法证明，请参看相关教课书。

3.2 到期收益率(Yield to Maturity)

购买债券的价格越低，就越有赚头。当购买债券时，面对卖方给出的债券价格，我们会想知道债券的回报率到底是多少。所以由已知债券价格，通过公式（3.1,3.2,3.3）计算出来的回报率r，叫做到期收益率。

比方说，一张面值1万元，10年到期，每年提供收益$600的息票债券，卖家要价$9900.00。所以，这张息票债券的到期收益率必须满足这个等式：

$$\frac{600}{r}+\{10000-\frac{600}{r}\}(1+r)^{-10}=9900 \tag{3.4}$$

只有$r=.0592386$才能满足上面的等式。

```
r=.0592386
600/r+(10000-600/r)*(1+r)^10-9900
```

银行的一年期利率经常会被称为即期汇率（Spot Rate），这是指使用复利的方式对未来的金额打折，得到当前相对应的价值。同时也被称为，用NPV函数打折。如果，X代表需要被打折的金额，y即期汇率，NPV函数为：

$$\frac{X}{(1+y)^T} \tag{3.5}$$

3.3 期限结构(Term Structure)

期限结构是指计算在不同期限上的到期收益率。使用t代表期限，PAR代表面值，P_t代表特定期限上的债券价格，到期收益率可以计算为：

$$y_t = (\frac{PAR}{P_t})^{1/t} - 1 \tag{3.6}$$

比方说面值1万元的零息债券，可以分为3种期限及其价格，表格所示：

Maturity	Price
1year	9500
2year	8100
3year	7300

这3个价格的到期收益率为：

$$y_1 = \{\frac{10000}{9500}\}^{1/1} - 1 = 0.05263158$$
$$y_2 = \{\frac{10000}{8100}\}^{1/2} - 1 = 0.11111111$$
$$y_3 = \{\frac{10000}{7300}\}^{1/3} - 1 = 0.11060352$$

```
t=1:3
PAR=10000
P=c(9500,8100,7300)
(PAR/P)^(1/t)-1
```

Chapter 4

投资组合理论(Portfolio)

投资组合理论（Portfolio Theory）是讲在一定程度的波动性上，如何组合两种或多种投资，来达到回报率的最大化。投资的回报率r是一个随机变量，比方说一支股票的回报率，今年可以是0.09，到明年就可能变成0.01。但我们相信，回报率r的均值是客观存在的，这个均值是回报率随机分布的中心。随机回报率出现在其均值左右的机会非常大，远离其均值出现的机会非常小。我们使用随机变量r代表回报率，用$E(r)$来代表回报率的均值。

有可能随机回报率在其均值左右出现的范围非常宽，也有可能随机回报率在其均值左右非常近的范围出现。我们就需要另外一个参数，波动性，来描述范围的宽度。波动性用小写的希腊字母σ代表，这个字母英文读作sigma，希腊文大写是Σ。波动性越高，说明出现的范围越宽，反之出现的范围越窄，或者说是越确定。均值$E(r)$，波动性σ他们都只是不随机的参数，是由随机变量r的特性决定的。如果你有机会去采访随机变量r，它可以确定地告诉你，这两个参数的具体数值是多少。但在现实中，我们只能随机抽取大量的样本，来估计这两个参数。由于样本是随机的，估计值也就变成随机的变量了。

这整个章节都是关于讨论，投资组合是否处在边缘位置，是否处在有效位置，还会有什么样的后果。

4.1 组合两种资产(Two Assets)

比方说，有两种投资资产，它们回报率的均值分别为$E(r_1) = 0.06, E(r_2) = 0.11$，它们回报率的波动性分别为$\sigma_1 = .012, \sigma_2 = 0.22$，我们想得到投资组合的回报率均值（Portfolio Return）用$E(r)$代表。而且这两个资产回报率之间有一个正面的统计相关性（Correlation）$\rho = 0.19$，也就是说，一个资产上涨，另一个资产也会上涨，如果是一个资产下跌，另一个资产也会很可能跟着下跌。根据资产的波动性σ和它们之间的关系性ρ，我们可以复原

二次形式中的矩阵，然后再计算投资组合的波动性。

协方差（Covariance）的计算公式：

$$covar = \rho\sigma_1\sigma_2 \tag{4.1}$$

投资组合方差（Variance）的计算公式，这已经就是二次形式了：

$$variance = p^2\sigma_1^2 + (1-p)^2\sigma_2^2 + 2p(1-p)covar \tag{4.2}$$

公式（4.2）中的p代表第一个资产在投资组合中的比重，那么第二个资产在投资组合中的比重就是$(1-p)$。比如第一个资产占总投资的0.1，第二个资产就占0.9。在组合两种资产的问题中，我们从0-1，每隔0.1取一个点做p，来观察对组合的回报率和波动性的影响。

投资组合的波动性（σ）是方差开平方根，的计算公式：

$$\sigma = \sqrt{variance} \tag{4.3}$$

最后还要回过头来，计算投资组合的回报率均值，使用资产所占的比例，进行加权（Weighted Average）：

$$E(r) = pE(r_1) + (1-p)E(r_2) \tag{4.4}$$

加权平均和前面介绍的向量点乘是完全相同的概念。这里的p和$(1-p)$是资产占总投资的比重。用所有的资产与其相应的比重相乘后在加在一起，就是加权平均。一个投资组合是一组的比重与可用的资产相对应，投资组合中所有的比重总和为1。加权平均，点乘，都是lesbegue积分的特殊形式，但是我们本着用到才学到的原则，不再对数学理论深入。

每一个投资组合都有相对应的回报率均值和波动性。有的投资组合会有相同的波动性，但他们的回报率均值却不相同。我们的目的是在相同的波动性下，找到能最大化回报率均值的投资组合。

运行本小节的R代码，可以得到与（4.1）截图相似的结果。这张图最重要的部分是曲线中由黑色粗线标出的部分，这段曲线叫做有效前沿（Efficient Frontier）。图中整条曲线，包含粗线和细线的部分，叫做边缘曲线（Envelope）。之所以被称为边缘，因为它是划分可行投资组合（Feasible Portfolio）与不可行投资组合（Infeasible Portfolio）的界限。在边缘曲线左侧的坐标点，代表了不可行组合。在边缘曲线上和边缘曲线右侧的坐标点，代表了可行组合。有效前沿上的组合之所以有效，是因为在给定的波动性的情况下，这些组合最大化回报率均值。在这个问题中，由p代表的投资组合比重全部处在边缘上。

组合两个资产问题R代码

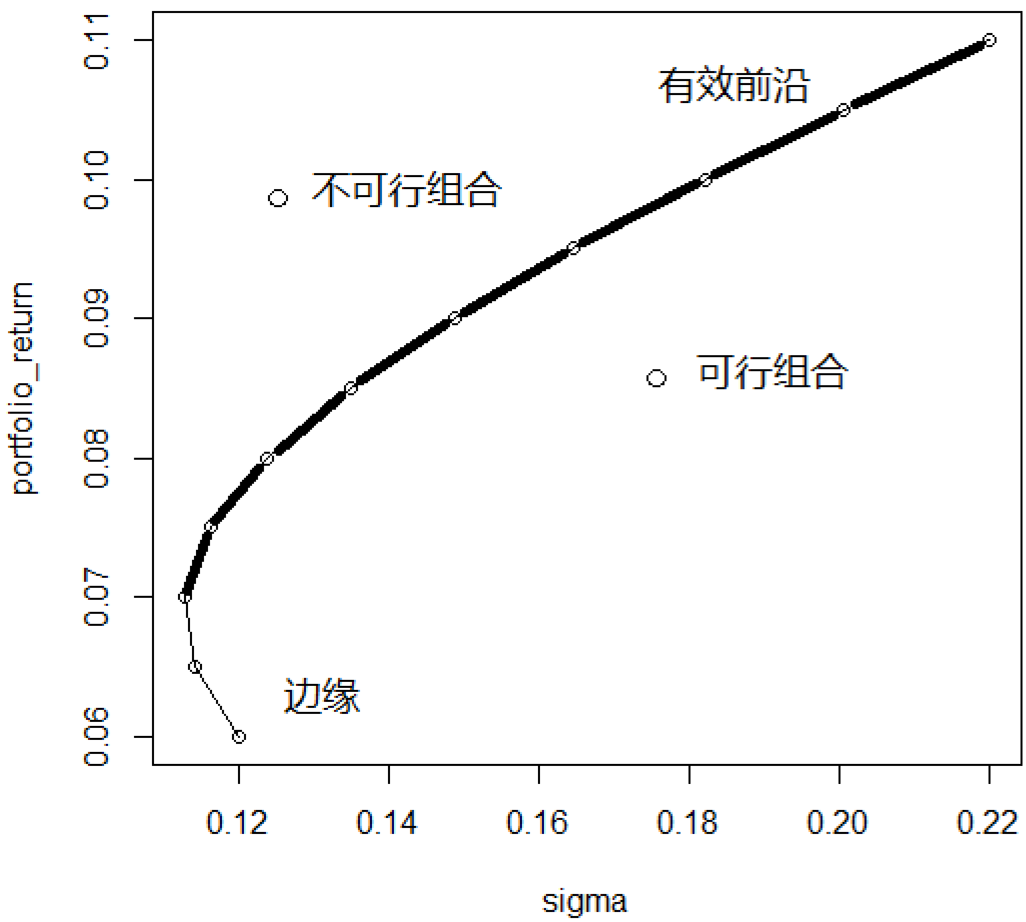


Figure 4.1: 两个资产投资组合作图-有效前沿

```
#Two-Asset Portfolio Problem

#Asset1
r1 =.06
sigma1=.12

#Asset2
r2 =.11
sigma2=.22

rho =.19
covar =rho*sigma1*sigma2
p =c(0,.1,.2,.3,.4,.5,.6,.7,.8,.9,1)# percent of Asset1

##Portfolio return as function of percent of Asset1##
variance=p^2*sigma1^2+(1-p)^2*sigma2^2+2*p*(1-p)*covar
sigma=sqrt(variance)
portfolio_return=r1*p+r2*(1-p)

cbind(p,sigma,return)
plot(sigma,portfolio_return,type='o')
```

4.2 基本定义

曾经有很长的一段时间，组合多种资产是一个困扰着数学家和经济学家，挥之不去，又无法解决的问题。与组合两种资产相同，我们仍旧希望计算处在边缘上的投资组合，并画出那条漂亮的边缘曲线，使用多个资产。终于在1973年，从麻省理工学院的默顿教授那里，传来了振奋人心的捷报。默顿教授告诉我们，只要在坐标系中，垂直的坐标轴上任意取一点c，然后以c为起点，画一条与边缘曲线相切的直线，切点就是一个边缘组合（4.2）。默顿教授还告诉我们，如果已知两个边缘组合，其它所有的边缘组合都可以被写成已知组合的加权平均。

在进一步介绍默顿的定理之前，我们先了解几个基本的定义和书写方法。如果有N个资产，每个资产的均值回报率由$E(r_1), E(r_2), \cdots, E(r_N)$代表。我们使用一个叫做$E(r)$的向量，来盛放每个资产的均值回报率：

$$E(r) = \begin{bmatrix} E(r_1) \\ E(r_2) \\ \vdots \\ E(r_N) \end{bmatrix} \tag{4.5}$$

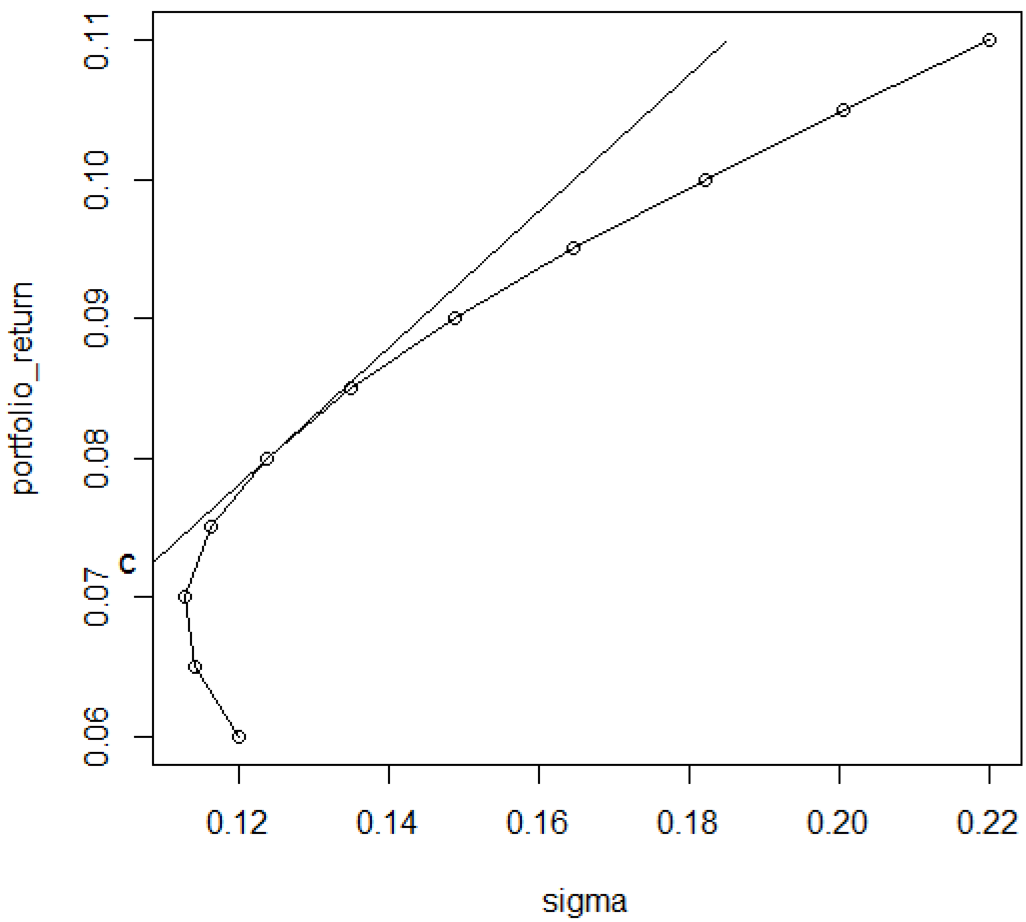


Figure 4.2: 默顿切线方法演示图

比方说，有4个资产，它们的均值回报率为$E(r_1) = 0.1, E(r_2) = 0.2, E(r_3) = 0.15, E(r_4 = 0.01)$，在R语言中由均值回报率组成的数列:

```
Er=c(0.1,0.2,0.15,0.01)
```

这N个资产的回报率之间，会存在有一个正方形$N \times N$，对称的，方差-协方差矩阵，表达的是资产回报率之间的相互关系。这个矩阵在书写是用大写的S代表。矩阵对角线上的σ^2都是方差，其余的都是协方差。

$$S = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \cdots & \sigma_{1N}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \cdots & \sigma_{2N}^2 \\ \vdots & \vdots & & \vdots \\ \sigma_{N1}^2 & \sigma_{N2}^2 & \cdots & \sigma_{NN}^2 \end{bmatrix} \tag{4.6}$$

以4个资产为例，在R语言中生成这个4×4回报率的方差-协方差矩阵:

```
S=matrix(c( .10, .01,.03, .05,
            .01, .30,.06,-.04,
            .03, .06,.40, .02,
            .05,-.04,.02, .50),nrow=4,ncol=4)
```

我们使用小写的x代表一个投资组合，如果其中有N个资产，用$x_1, x_2, \cdots x_N$代表每一个资产在这个组合中的比重。所以x是一个长度为N的向量，而且所有比重的总和为1:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \sum_{i=1}^{N} x_i = 1 \tag{4.7}$$

4个资产，比重分别为$x_1 = 0.2, x_2 = 0.1, x_3 = 0.6, x_4 = 0.1$，在R语言中演示投资组合比重的总和为1 (Sum to 1):

```
x=c(0.2,0.1,0.6,0.1)
sum(x)
```

使用x代表一个投资组合，$E(r_x)$就代表这个组合的均值回报率。把所有的资产以比重的方式放在一起，所得的回报率。用加权平均计算，或者叫做点乘，完全是先乘后加的计算方式:

$$E(r_x) = x^T \cdot E(r) = \begin{bmatrix} x_1 & x_2 & \ldots & x_N \end{bmatrix} \begin{bmatrix} E(r_1) \\ E(r_2) \\ \vdots \\ E(r_N) \end{bmatrix} = \sum_{i=1}^{N} x_i E(r_i) \tag{4.8}$$

还是以4个资产，4个比重，在R语言中计算，投资组合的回报均值:

```
x=c(0.2,0.1,0.6,0.1)
Er=c(0.1,0.2,0.15,0.01)
x%*%Er
```

所有资产的回报率都是随机的，由它们配搭得到的回报率也应该是随机的。所以投资组合也有方差，而且是把资产的方差-协方差矩阵，按照比重加起来计算的。计算方式是第一章提到的二次形式，把横着的比重乘以方差-协方差矩阵，再乘以竖着的比重：

$$\sigma_x^2 = x^T S x = \begin{bmatrix} x_1 & x_2 & \cdots & x_N \end{bmatrix} \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \cdots & \sigma_{1N}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \cdots & \sigma_{2N}^2 \\ \vdots & \vdots & & \vdots \\ \sigma_{N1}^2 & \sigma_{N2}^2 & \cdots & \sigma_{NN}^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j \sigma_{ij} \tag{4.9}$$

以4个资产为例，在R语言里计算投资组合的方差：

```
x=c(0.2,0.1,0.6,0.1)
S=matrix(c( .10, .01,.03, .05,
            .01, .30,.06,-.04,
            .03, .06,.40, .02,
            .05,-.04,.02, .50),nrow=4,ncol=4)
t(x)%*%S%*%x
x%*%S%*%x
```

4.3 组合多种资产(N Assets)

默顿定理1 (Merton 1973)已知所有资产回报率向量为$E(r)$，资产间的方差-协方差矩阵为S，选取一个数值为c，与其对应的边缘投资组合可以计算为：

$$x = \frac{S^{-1}\{E(r) - c\}}{\sum S^{-1}\{E(r) - c\}} \tag{4.10}$$

x是一个边缘投资组合。

我们选择5个c点，用4个资产，看能不能在R语言里画出边缘曲线。定义好已知条件后，在把相同的计算重复5遍，在画一个图。资产均值回报率向量$E(r) = (0.1, 0.2, 0.15, 0.01)$，五个c为$0.00000001, 0.021, 0.45, 0.8, 4$，方差-协方差矩阵：

$$S = \begin{vmatrix} 0.1 & 0.01 & 0.03 & 0.05 \\ 0.01 & 0.3 & 0.06 & -0.04 \\ 0.03 & 0.06 & 0.4 & 0.02 \\ 0.05 & -0.04 & 0.02 & 0.5 \end{vmatrix}$$

运行提供的R代码就可以得到以下结果（4.3）：

默顿第一定理演示图R代码

```
Er=c(0.1,0.2,0.15,0.01)
S=matrix(c( .10, .01,.03, .05,
            .01, .30,.06,-.04,
            .03, .06,.40, .02,
            .05,-.04,.02, .50),nrow=4,ncol=4)

c=0.00000001
a=solve(S)%*%(Er-c)
b=sum(a)
x1=a/b
Er_x1=t(x1)%*%Er
sigma1=sqrt(t(x1)%*%S%*%x1)

c=0.021
a=solve(S)%*%(Er-c)
b=sum(a)
x2=a/b
Er_x2=t(x2)%*%Er
sigma2=sqrt(t(x2)%*%S%*%x2)

c=.45
a=solve(S)%*%(Er-c)
b=sum(a)
x3=a/b
Er_x3=t(x3)%*%Er
sigma3=sqrt(t(x3)%*%S%*%x3)

c=.8
a=solve(S)%*%(Er-c)
b=sum(a)
x4=a/b
Er_x4=t(x4)%*%Er
sigma4=sqrt(t(x4)%*%S%*%x4)

c=4
a=solve(S)%*%(Er-c)
```

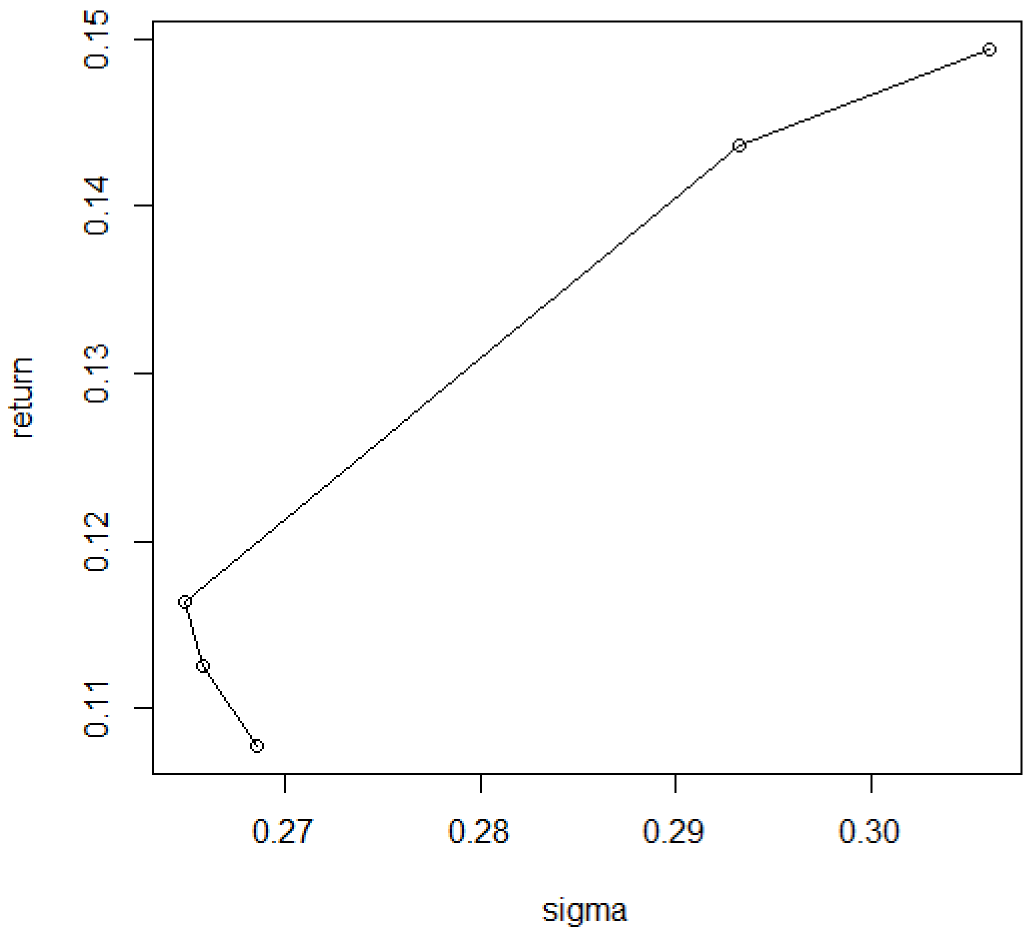


Figure 4.3: 默顿第一定理演示图

```
b=sum(a)
x5=a/b
Er_x5=t(x5)%*%Er
sigma5=sqrt(t(x5)%*%S%*%x5)

sigma=c(sigma3,sigma4,sigma5,sigma1,sigma2)
return=c(Er_x3,Er_x4,Er_x5,Er_x1,Er_x2)
plot(sigma,return,type='o')
cbind(sigma,return)
```

默顿定理2 (Merton 1973)已知两个边缘组合，其它的边缘组合都可以写成，两个已知边缘组合的加权平均。让x和y做为两个边缘组合，让λ和$(1-\lambda)$作为比重，任何边缘组合都可以写为：

$$\lambda x+(1-\lambda)y=\begin{bmatrix}\lambda x_1+(1-\lambda)y_1\\ ...\\ \lambda x_N+(1-\lambda)y_N\end{bmatrix} \tag{4.11}$$

在默顿定理1的例子里，我们已经计算了边缘组合。在下面的例子里，我们要把组合x_5写成组合x_3,x_4的加权平均：

$$x_3=\begin{vmatrix}0.63193682\\0.15591812\\0.08067196\\0.13147311\end{vmatrix} x_4=\begin{vmatrix}0.61876576\\0.18214624\\0.08760799\\0.11148000\end{vmatrix} x_5=\begin{vmatrix}0.60844141\\0.20270559\\0.09304492\\0.09580808\end{vmatrix}$$

```
x3=c(0.63193682,0.15591812,0.08067196,0.13147311)
x4=c(0.61876576,0.18214624,0.08760799,0.11148000)
x5=c(0.60844141,0.20270559,0.09304492,0.09580808)

(x3-x4)/(x5-x4)
lambda=-1.2757275
lambda*x3+(1-lambda)*x4
```

4.4 资本资产定价模型(CAPM)

资本资产定价模型(capital asset pricing model,CAPM)，有非常多不同的用途。这里我们分别举例进行详细介绍。

资本市场线（Capital Market Line，CML）是讲，首先我们要有一个处在边缘（envelope）上的投资组合，然后还要配搭一个没有风险（risk-free）的投资资产，如何搭配才能达到要求。这个边缘组合，一般被称为市

场投资组合（market portfolio），用M代表，所以它的回报率和标准差记做：r_M, σ_M。无风险资产的方差当然为0，回报率用r_f代表。我们所要达到的搭配记做x，所以它的回报率标准差记做：r_x, σ_x。CML的公式可以表达为：

$$r_x = r_f + \frac{r_M - r_f}{\sigma_M}\sigma_x \tag{4.12}$$

我们假设$r_M > r_f$，从公式（4.12）可以看出，σ_x越大可以得到的r_x越大。一直在讲σ_x代表的是波动性和风险，所以这个公式准确的告诉我们，高风险带来高回报，而且回报到底有多少。对于所需要的边缘组合，我们通常会用有效组合，例如$S\&P500$这样市场上公认的指数基金。

假设无风险投资回报率$r_f = 0.06$，市场投资组合的回报率和标准差为$r_M = 0.2, \sigma_M = 0.4$。资本市场线的斜率就是$(r_M - r_f)/\sigma_M = (0.2 - 0.06)/0.4 = 0.35$。如果我们希望尝试$0.2, 0.3$来做$\sigma_x$，所能达到的$r_x$为：

$$\begin{aligned} r_{x1} &= r_f + 0.35\sigma_{x1} = 0.06 + 0.35 \times 0.2 = 0.13 \\ r_{x2} &= r_f + 0.35\sigma_{x2} = 0.06 + 0.35 \times 0.3 = 0.165 \end{aligned}$$

把r_{x1}, r_{x2}和相应的σ_x画在图上（4.4）就可以看到这条资本市场线，而且斜率的确是0.35。 如果我们认为$\sigma_x = 0.3$的风险太高，不宜使用。我们选择$\sigma_x = 0.2$来搭配已有的有效组合，还有无风险投资。σ_x/σ_M是需要分配给有效组合的部分，$(1 - \sigma_x/\sigma_M)$是需要分配给无风险投资的部分。使用$\sigma = 0.2$，50%的资金要给这个有效组合，另外50%的资金要做无风险投资。

资本市场线R代码

```
r_f=0.06
r_M=0.2
sigma_M=0.4

slope=(r_M-r_f)/sigma_M
r_x1=r_f+slope*0.2
r_x2=r_f+slope*0.3

return=c(r_x1,r_x2)
sigma=c(0.2,0.3)
plot(sigma,return,type='o',ylim=c(0.06,.2),main="CML")

0.2/sigma_M
1-0.2/sigma_M
```

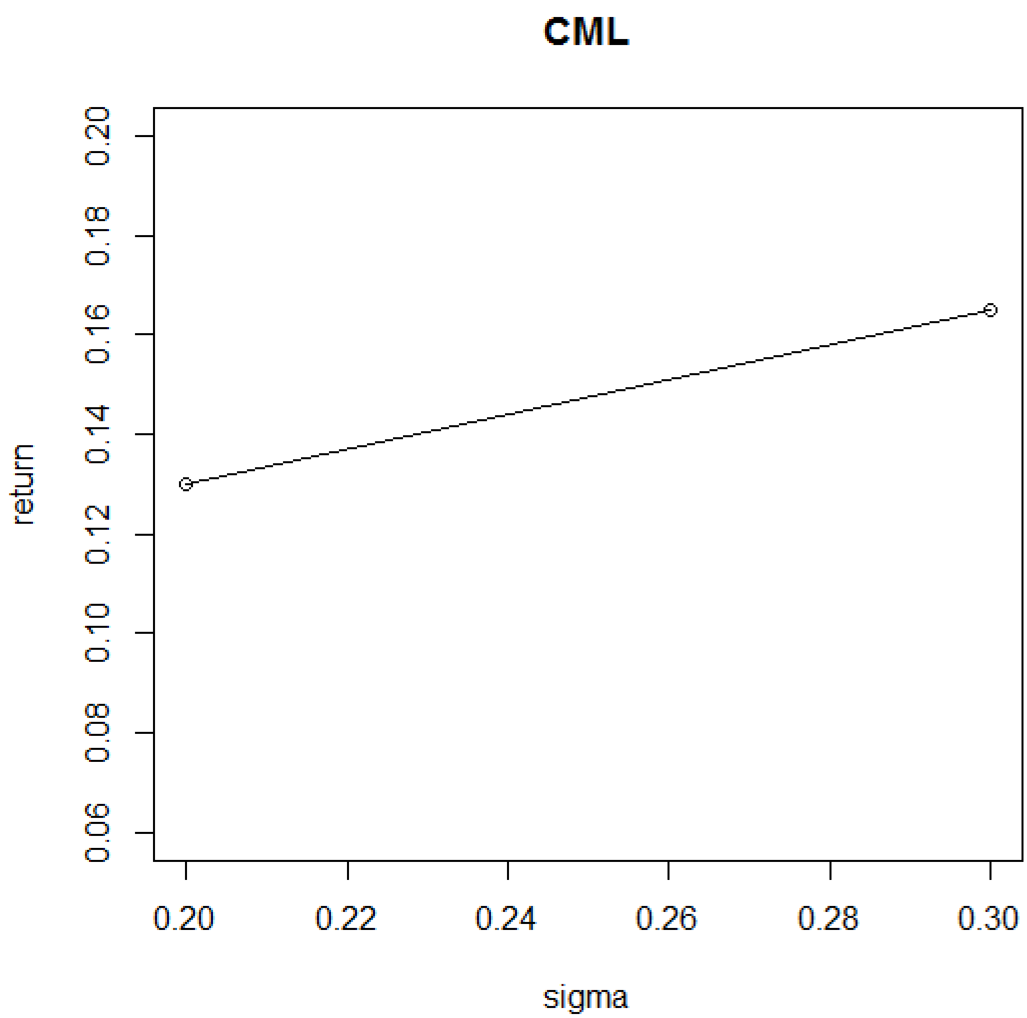


Figure 4.4: 资本市场线作图

证券市场线(Security Market Line, SML)是CAPM的另外一种用法。这是布莱克1972年出版的方法。如果我们有一个边缘组合叫做y，其它组合的回报率都可以通过线性回归的方式，从y的回报率计算出来。让β_x作为这个回归的斜率，$E(r_x), E(r_y)$作为两个组合回报率均值。r_f是市场上存在的无风险投资的回报率，例如银行存款。

$$E(r_x) = r_f + \beta_x \left[E(r_y) - r_f\right] \tag{4.13}$$

$$where$$

$$\beta_x = \frac{Cov(x,y)}{\sigma_y^2},\ and \tag{4.14}$$

$$Cov(x,y) = x^T S y \tag{4.15}$$

这个线性回归的斜率β_x的定义理论值是公式（4.14,4.15）来计算。但是在日常的实行中，β_x是通过线性回归方法估计出来的。SML的关键是，如果我们有一个组合x，不知道它的回报率是多少。但市面上存在着非常知名的组合y，例如$S\&P500$，我们可以通过y来计算x的回报率。

举例，银行存款利率$r_f = 0.04$，市场组合y的回报率$E(r_y) = 0.12$，市场组合的方差$\sigma_y^2 = 0.0008$。有一只股票，NEWX，用x代表，x, y之间的协方差为$Cov(x,y) = 0.0001$。问NEWX的回报率是多少。先算β_x:

$$\beta_x = 0.001/0.0008 = 1.25$$

代入公式（4.13）：

$$E(r_x) = 0.04 + 1.25(0.12 - 0.04) = 0.14$$

股票NEWX的回报率为0.14。如果今年NEWX的价格是\$50，问明年定价\$57合适吗?

$$50 \times (1 + 0.14) = 57$$

所以\$57在明年是一个合适的公平价格。

资本市场线CML R代码

```
r_f=0.04
r_y=0.12
cov=0.001
var=0.0008

beta=cov/var
r_f+beta*(r_y-r_f)

50*(1+0.14)
```

CAPM的检验

使用CAPM模型的关键在于，估计斜率β。因为在现实中，回报率方差和协方差的准确数值根本是不可能知道的，通过方差和协方差来计算斜率，是行不通的。还好这个斜率β，可以作为线性回归的参数来估计。并且可以通过参数估计来检验CAPM模型的合理性。

这个应用实例，我们用$S\&P500$作为市场投资组合，考虑与微软（Microsoft）公司股票回报率的线性关系。股票的牌价是通过R语言tseries程序包，在线读取的1993-11-01到2002-04-03，一共2374天的数据。将牌价转换成功回报率十分简便，只要使用公式：

$$r_i = \exp\{\log P_{i+1} - \log P_i\} - 1 \tag{4.16}$$

这里P_i代表第i天的股票牌价。相应的R代码也只有一行：

```
sp500=exp(diff(as.numeric(log(sp500))))-1
```

线性回归模型是通过R语言自带的lm()函数计算的，线性回归的结果是截图（4.4）。可以看到，数值0.0008413是模型截距α的估计值，数值1.3314185由红色方框表明，是我们所需的模型斜率β的估计值。从理论上讲，CAPM模型的截距α参数是等于0的。而且这里的截距估计值0.0008413也是非常的小，与理论相符。通常在统计学中需要使用p-value来检验参数的显著性，但是在这里我们使用了过多的2000多个样本，解释p-value已经毫无意义。

使用斜率β的估计值，可以帮助我们计算微软股票的回报率均值。截距α可以提供更多信息。如果α不为0，这从CAPM的角度说明，股票的定价不恰当。如果$\alpha > 0$，说明股票的定价太低，回报率相对较高。这是一个资产值得买入的标志。但我们也需要小心，当估计的α不接近0，也可能是因为我们选择的市场组合并不是处在边缘上。当CAPM模型得到了非常出色的结果，α, β的估计值全都与理论相符，这也只能说明我们选择的市场组合处在边缘上，我们根本不可能知道它是否是一个有效组合。

资本资产定价模型（CAPM）β估计R代码

```
library("tseries") # load the tseries library

sp500 = get.hist.quote(instrument = "^gspc", start =
 "1993-11-01", end="2003-04-03",quote="AdjClose")
ms = get.hist.quote(instrument = "msft", start =
"1993-11-01", end="2003-04-03",quote="AdjClose")

sp500=exp(diff(as.numeric(log(sp500))))-1 #prices into returns
ms=exp(diff(as.numeric(log(ms))))-1
```

```
> summary(fit)

Call:
lm(formula = ms ~ sp500)

Residuals:
     Min       1Q   Median       3Q      Max
-0.15273 -0.01080 -0.00081  0.01048  0.14870

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0008413  0.0003935   2.138   0.0326 *
sp500       1.3314185  0.0341320  39.008   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01916 on 2371 degrees of freedom
Multiple R-squared: 0.3909,     Adjusted R-squared: 0.3906
F-statistic:  1522 on 1 and 2371 DF,  p-value: < 2.2e-16

>
```

Figure 4.5: 资本资产定价模型（CAPM）β估计

```
fit<-lm(ms ~ sp500)
summary(fit)
```

4.5 Black-Litterman模型（1991）

有一天，一个朋友来办公室找我，让我帮忙计算两支股票的回报率均值。这已经不是第一次了，他知道我懂得如何计算，我也乐得让他帮我搜集一些市面上的消息他说，市面上有一种名叫x的投资组合，配搭两支股票，达到了0.09的年回报率。他还说，这个组合中，第一支股票占0.246，第二支股票占0.754。他还知道，两支股票的方差-协方差矩阵。我们用银行定期一年存款利率0.05，作为无风险投资回报率。。和所有的应用题一样，我们在草纸上罗列了这些前提条件：

$$\begin{aligned} x &= |0.246, 0.754| \\ r_x &= 0.09016 \\ r_f &= .05 \\ S &= \begin{vmatrix} 0.1100 & 0.0044 \\ 0.0044 & 0.2000 \end{vmatrix} \end{aligned}$$

先计算投资组合x的方差，这是一个二次形式：

$$Var_x = x^T S x = |0.246, 0.754| \cdot \begin{vmatrix} 0.1100 & 0.0044 \\ 0.0044 & 0.2000 \end{vmatrix} \cdot \begin{vmatrix} 0.246 \\ 0.754 \end{vmatrix} = 0.1210$$

再计算在上面提到的，Merton公式里的λ，使用公式(4.20)：

$$\lambda = \frac{r_x - r_f}{Var_x} = \frac{0.09 - 0.05}{0.1210} = 0.3292$$

再计算两支股票的回报率均值，输出的是一个向量，用r代表：

$$r = \lambda S x + r_f = 0.3292 \times \begin{vmatrix} 0.1100 & 0.0044 \\ 0.0044 & 0.2000 \end{vmatrix} \cdot \begin{vmatrix} 0.246 \\ 0.754 \end{vmatrix} + 0.05 = \begin{vmatrix} 0.0600 \\ 0.1000 \end{vmatrix} \tag{4.17}$$

所以说，第一只股票的年回报率均值为0.06，第二支股票的年回报率均值为0.10。

Blac-Litterman Model, R 代码

```
#Black-Litterman Model
S=matrix(c(  .1100, .0044,
```

```
       .0044, .2000 ),nrow=2,ncol=2)
r_f=.05 #risk free rate
x=c(.246,.754)#envelope portfolio
r_x=0.09016     #x%*%Er

var_x=t(x)%*%S%*%x
lambda=as.numeric((r_x-r_f)/var_x)
r=lambda*S%*%x+r_f
r
```

从上面的(Merton1973)命题1公式里的分母部分可以用λ来标记，这只是一个定义，并不能用来计算λ:

$$\lambda = \sum S^{-1}\{E(r) - c\} \tag{4.18}$$

这样(Merton1973)公式就可以被简单的写成:

$$x = \frac{S^{-1}\{E(r) - c\}}{\lambda}$$

公式等号两边都乘以λ:

$$\lambda x = S^{-1}\{E(r) - c\}$$

然后再两边都乘以S:

$$S\lambda x = E(r) - c$$

最后两边再加上c:

$$S\lambda x + c = E(r)$$

从上面的命题5知道，可以用r_f代替c，得到如下:

$$E(r) = \lambda S x + r_f \tag{4.19}$$

但是这里的λ还需要用下面的公式计算:

$$\lambda = \frac{r_x - r_f}{\sigma_x^2} \tag{4.20}$$

Chapter 5

布莱克-肖尔斯模型（Black-Scholes）

5.1 零回报

比方说，我持有一支的股票（Stock），现在的市场价格\$30，用$S_0 = 30$来代表。我对你说，我可以把这支股票200个工作日后的认购期权（Option）卖给你，履约价（Strike Price）为\$25，用$X = 25$代表。换而言之，200个工作日后，如果这支股票的价格仍旧在\$25之上，你可以付给我\$25履行合约，这支股票就成了你的，你可以保留所有的盈余。如果200个工作日后，这支股票的价格低于了\$25履约价的水平，你可以选择放弃从我这里买入这支股票。这是一个认购期权。

但你现在要买这个认购期权，才能在200工作日后用\$25履行合约。认购期权（Call Option）是在未来某时间点以一定价格买入股票的权利。我告诉你，认购期权的价格是\$6.19，记做$C = 6.19$。

这支股票的回报率均值，方差分别记为$r = 0.06$，$\sigma = 0.12$。你知道，200工作日后股票的盈余，减去现在付出的买期权的费用\$6.19，才是这次投资的盈余。你还知道，股票的价格都是随机出现的，所以你想用计算机随机生成10万个，这支股票在第200个工作日的价格，计算平均盈余。如果平均盈余¡0，说明期权\$6.19的要价太高。如果平均盈余¿0，说明期权\$6.19的要价太低。

你使用R语言书写了下面的程序。程序的第一部分是用来生成10万个一年内，250个工作日，的股票价格。这部分程序的结果是一个250行，10万列的矩阵，每行代表一个工作日，每列代表一个股票价格走势。这个矩阵的第200行就是需要的10万个第200天的股票价格。程序的第二部分，输入履约价，和买入期权的价格，计算最终盈余。

程序中的n代表生成的随机股价的数量，我们可以先设置$n = 10$，然后

使用$ts.plot()$函数吧这10个250天的股票价格画出图来看看样子截图(5.1)。

使用S_{200}代表第200天的股票价格，先生成10万个股价序列，从中提出所有的S_{200}与\$25的履约价相比较。如果$S_{200} > 25$，股票上的收益就是$S_{200} - 25$。如果$S_{200} < 25$，就放弃期权，股票上的收益就是0。数学表达式写为：

$$\max(S_{200} - X, 0) \tag{5.1}$$

还要对这10万个股票收益做平均值，然后再对平均后的收益乘以$\exp\{-r\Delta t\}$进行打折。因为这支股票有一个稳定0.06的上升趋势，我们预期200工作日后，经过连续复利，股票的总收益率是$\exp\{0.06 \times 200/250\} = 1.0492$，所以必须除以这个总收益率，才能把未来200天的价值，转化成与现在同等的价值，也就是乘以$1/1.0492 = 0.9531$。

由于需要花\$6.19才能有权利在200天后作出选择，打折后的股票收益必须减去\$6.19的认购期权价格（Call Price）。可以看到，经过程序的计算，剩下的投资结余，刚好为0。

除了可以花钱买在未来以一定价格买入股票的权利，还可以买在未来以一定价格卖出股票的权利，认沽期权（Put Option）。当然认沽期权的价格（Put Price）也可以巧妙的计算出来，使预期的最终受益为0。

收益为0的期权价格R代码

```
#Lognormal Stock Price Simulation.
days <- 250 #working days per year
n <- 10  #number stock prices
S0=30  #Price at t=0
dt<-1/days
r<-.06
sigma<-.12

S <- matrix((r-0.5*sigma^2)*dt+sigma*sqrt(dt)*rnorm(days*n),
ncol=days, nrow=n)
S <- S0*exp(apply(S,1,cumsum))
ts.plot(S)

#Compute option call price
C=6.187974  #Call price
X=25  #Strike price

C_t=pmax(S[200,]-X,0)
exp(-r*200*dt)*mean(C_t)-C
```

股票价格的lognormal随机生成公式：

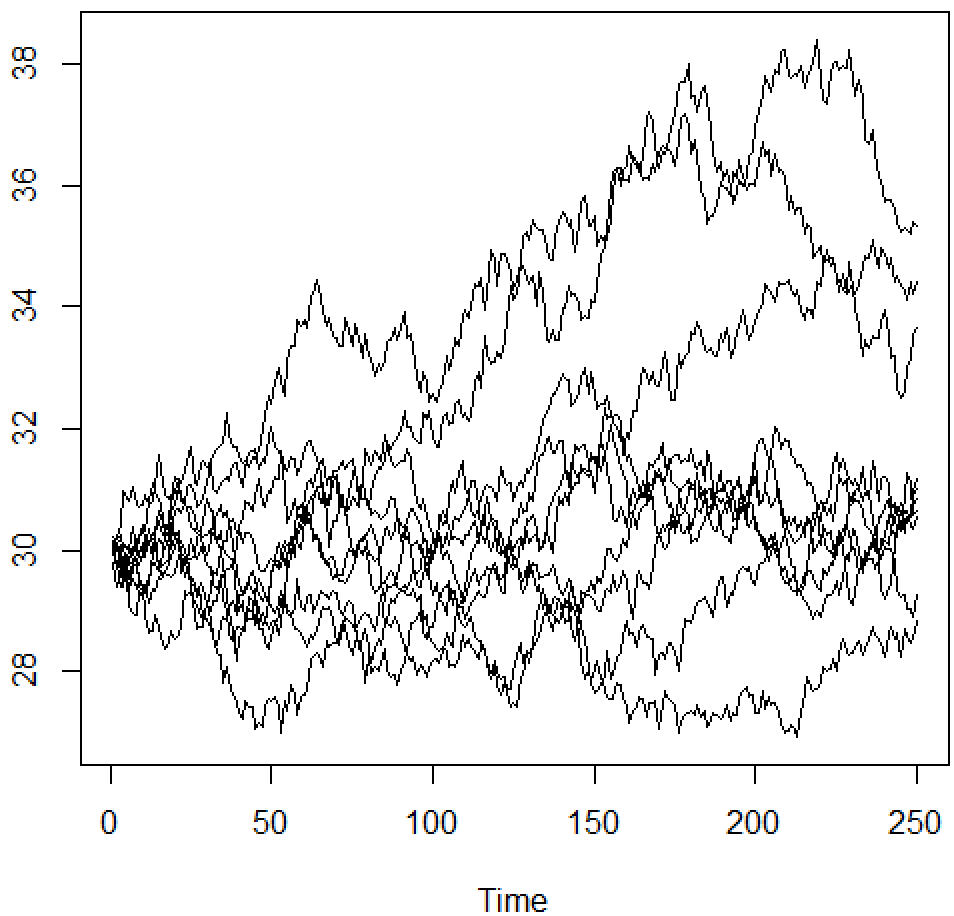


Figure 5.1: 10个随机生成的股票价格序列

$$S_T = S_t e^{(r-0.5\sigma^2)(T-t)+\sigma\sqrt{T-t}Z_T} \tag{5.2}$$

5.2 相关的定义

期权（Options）有两种，认购和认沽。认购期权（Call Option）是指在未来某天或某天之前，允许以特定价格，购买某股票的权利。认沽期权（Put Option）是指在未来某天或某天之前，允许以特定价格，卖出某股票的权利。期权交易可以涉及到其它的投资领域，在这里我们只使用股票作为讨论的例子。

欧式期权（European Option）是指其买卖的权利只能在未来某天的当天行使。美式期权（American Option）是指其买卖的权利可以再未来某天当天或之前行使。

我们用大写的C（Call Price）来代表认购期权的价格，用大写的P（Put Price）来代表认沽期权的价格。并且用大写X代表期权履约价格（Strike Price）。股票的价格用大写S代表，用右下角的小标来代表时间，例如S_0，S_t。

我们还是使用r代表回报率，银行存款利息也是回报用r_f代表。如果一个投资组合名叫小写的x，这个投资组合的回报率用r_x代表。当然股票的回报率都是随机出现的，用σ来代表回报率的标准差。标准差σ是用来衡量回报率波动性（volatility）的指标，标准差越大，波动性就越大。

认购期权是买未来的股票价格上涨，认购期权是用来延迟股票的购买，认购期权购买的是未来可以用便宜价格购入股票的权利。认沽期权买的是未来的股票下跌，认沽期权是用来延迟股票的购买，认沽期权购买的是未来可以用高价卖出股票的权利。履约价越高，认购期权价就越低。履约价越低，认沽期权价也越低。

5.3 套利命题

套利（Arbitragee）是指用价格冲突赚取利益。比方说，我持有一只股票价格是$30，你想从我这里以$35的价格购买，我就可以赚取$5的利益。套利命题(Arbitrage Propositions)指的是如果违反就能产生价格冲突，并被赚取利益的命题。

1. 考虑一个美式认购期权，其价格为C，履约价为X，在其定价的时候股票价格为S_0，所以：

$$C > \max(S_0 - X, 0) \tag{5.3}$$

   举例，美式期权不一定要等到规定的未来某天才能履约（买卖股票），可以买了期权立即履约。如果我有一支股票，股票价格是$30，

认购期权的履约价是\$25，我的认购期权价格肯定要大于\$5。如果认购期权价小于\$5，你可以马上用\$25买入这个股票，同时用\$30卖出，抛去花掉的认购期权价，总共花费还不到\$30，就产生了盈余。

2. 考虑一个欧式认购期权，其花费为C，履约价为X，在其定价的时候股票价格为S_0，所以：

$$C > \max(S_0 - PV(X), 0) \tag{5.4}$$
$$PV(X) = \exp\{-r\Delta t\}X \tag{5.5}$$

这个命题的意思是，因为是欧式期权，不可能在约定好的日期之前履约，但可以把履约费用X存到银行里去生利息，从而又产生了盈余。所以在期权定价的时候，还要避免这一方面策略的套利。做法就是在计算C的时候，用$PV()$函数把X打折扣。X被看做在履约时，经过银行连续复利储蓄的结果。

3. 考虑一个美式认购期权，履约价为X，在期权到期履约之前有一个股票价格为S_t，并且这支股票没有分红在履约期限之前没有分红，所以：

$$\max(S_t - PV(X), 0) > \max(S_t - X, 0) \tag{5.6}$$

这个命题是在讲，在美式认购期权到期之前，在任何股价买入股票都是不值得的。把认购期权作价转让给别人，可以得到更多的价值，如果这支股票没有分红。把这个认购期权卖出去，不要履约，你可以挣更多的钱。

举例：一个美式认购期权的履约价$X = 25$，3三个月后$\Delta t = 0.25$后需要决定是否履约购入股票或放弃期权。股票当前价格$S = 40$，银行年利率$r = 0.06$，如果买入股票可收益$S - X = 40 - 25 = 15$。但是根据套利命题2，这个美式期权的市场价格至少是在$\max(S - PV(X), 0) = 40 - \exp(-0.25 \times 0.06)25 = 40 - 0.9851 \times 25 = 15.37$。

美式认购期权的各种特色，没有人会频繁的拿出来讨论，因为经常的，美式认购期权的价值和欧式认购期权的价格无异。但是，美式欧式认沽期权的价值却是不同的，欧式认沽期权的价值要少于美式认沽期权的价值。

4. 买权卖权等价命题（Put-call parity）. 考虑一支股票上同时有欧式认购和认沽期权，认购和认沽有相同的履约价X，股票在履约期限T之前没有分红，所以：

$$P + S_0 = C + PV(X) \tag{5.7}$$

等号的左边是认沽期权的价格P加股票价格S0，等于号的右边是认购期权的价格C加履约价的现价值PV（X）（present value）。要做认沽，必须先买入股票，再付清认沽期权价：$P + S_0$ 要做认购，必须准备相当的履约价值$PV(X)$在付清履约价格：$C + PV(X)$ 同时进行认购和认沽，就相当于先持有这支股票，在履约价X卖出，同时又以相同的履约价X买入相同的股票，整个过程前后价值没有变化。所以上面的等号是一定成立的。整个命题只适用于欧式期权，美式期权可以有不同的结果。这个命题中有4个数值，已知其中的3个就可以算出第4个。

5. 认购价的凹性（Call price convexity）。考虑在一个股票上的3个同时到期履约的认购期权。假设，认购价格C_1的履约价为X_1，C_2的履约价为X_2，C_3的履约价为X_3。并且$X1 < X2 < X3$，X_1X_2的距离与X_2X_3的距离相同，$X_3 - X_2 = X_2 - X_1$，或者$X_2 = (X_1 + X_3)/2$。所以：

$$C_2 < \frac{C_1 + C_3}{2} \tag{5.8}$$

凹形∪（Convex），突形∩（Concave）指的是两种线条弯曲的方式。命题5是讲，这3个认购价格都是凹性曲线上的点。如图（5），X_1, X_2, X_3分别为20,50,80，所以50 = (20+80)/2。但是C_1C_3都为29，C_2只有20，所以：

$$20 < \frac{29 + 29}{2}$$

R code:

```
X=seq(from=0,to=100,by=.1)
C=((X-50)^2+(X-50)+2000)/100
plot(X,C,type='o')
```

5.4 计算布莱克-肖尔斯

如果股票价格是对数正态分布（lognormally distributed），如果股票在到期履约之前没有分红，如果期权是欧式。所以，认购和认沽期权的价格要用下面的公式来计算：

$$Call = SN(d_1) - Xe^{-r\Delta t}N(d_2) \tag{5.9}$$

$$Put = C - S + Xe^{-r\Delta t} \tag{5.10}$$

$$Put = Xe^{-r\Delta t}N(-d_2) - SN(-d_1) \tag{5.11}$$

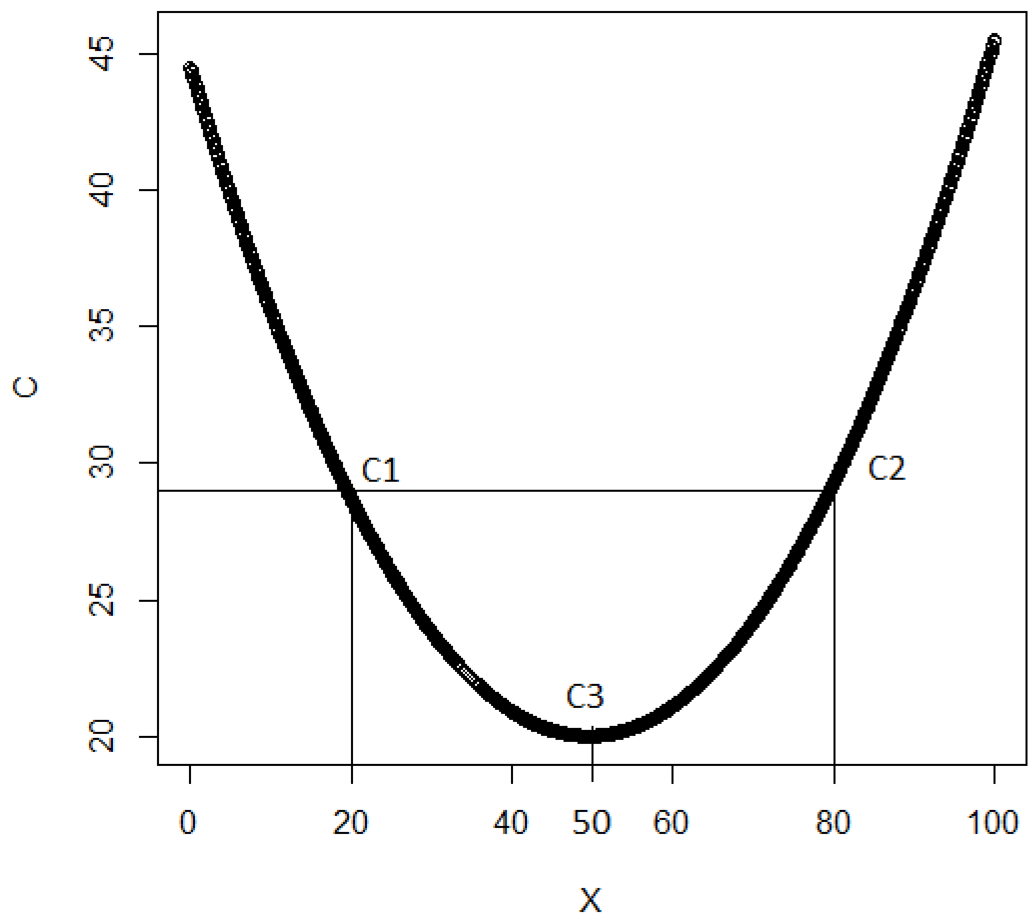


Figure 5.2: 认购价的凹性

而且：

$$d_1 = \frac{\log\frac{S}{X} + (r + 0.5\sigma^2)\Delta t}{(\sigma\sqrt{\Delta t})} \tag{5.12}$$

$$d_2 = d1 - \sigma\sqrt{\Delta t} \tag{5.13}$$

这五个公式告诉我们，要计算对数正态分布的股票价格的期权价格，认购和认沽，要知道5个前提条件：

1. S 当前股票价格
2. σ 股票回报率的波动性
3. X 期权履约价
4. r 银行利息，或无风险投资回报
5. Δt 距离未来履约期限还有的时间段

要先根据公式（5.12,5.13）计算d_1, d_2。在把d_1, d_2输入$N()$函数，计算$N(d_1)$, $N(d_2)$, $N(-d_1)$, $N(-d_2)$。$N()$函数输出的是正态分布的概率，R语言里把这个函数叫做pnorm()，例如：

```
pnorm(d1)
```

把$N(d_1), N(d_2), N(-d_1), N(-d_2)$代入公式（5.9）中结果就是认购期权价格。公式（5.10,5.11）结果是相同的，计算的都是认购期权的价格，但是（5.11）是直接进行计算，公式（5.10）根据的是前面介绍的等价关系得到的：

$$S + P = C + PV(X)$$
$$therefore$$
$$P = C + PV(X) - S$$

使用这些公式在实例中，比方说$S = 30, X = 25, r = .06, \Delta t = .8, \sigma = .12$，计算：

$$d_1 = \frac{\log(30/25) + (0.06 + 0.5 \times 0.12^2) \times 0.8}{0.12 \times \sqrt{0.8}} = 2.19956$$
$$d_2 = 2.19956 - 0.12 \times \sqrt{0.8} = 2.092229$$
$$C = 30 \times N(2.19956) - 25 \times \exp(-0.06 \times 0.8) \times N(2.092229) = 6.187974$$
$$P = 6.187974 - 30 + 25 \times \exp(-0.06 \times 0.8) = 0.01631911$$
$$P = 25 \times \exp(-0.06 \times 0.8) \times N(-2.092229) - 30 \times N(-2.19956) = 0.01631911$$

进行期权价格计算的R代码已经附加在本小节的后面，可以运行，并检查计算结果。可以试着升高银行存款利率r，可以观察到这对两种期权价格的影响不大。但是股票价格S的变化对期权价格的影响非常的大。这表明期权交易相当于使用保证金（Margin）做杠杆的交易。认购期权本质是在做长股票，做短无风险的存款。认沽期权本质是做短股票，做长无风险的存款。最后值得注意的是，在上面套利命题中提到，不值得提前认购履约，但是提前认沽履约比卖出持有的期权更值得。

布莱克-肖尔斯期权价格计算R代码

```
#Black-Scholes Option Pricing Calculation
S =30  #Current stock price
X =25  #Exercise price
r =.06 #Risk-free of interest
dt =.8 #Time to maturity of option(in years)
sigma =.12  #Stock volatility, standard deviation

d1 =(log(S/X)+(r+0.5*sigma^2)*dt)/(sigma*sqrt(dt))
d2 =d1-sigma*sqrt(dt)

C =S*pnorm(d1)-X*exp(-r*dt)*pnorm(d2)
P1 =C-S+X*exp(-r*dt) #by Put-Call parity
P2 =X*exp(-r*dt)*pnorm(-d2)-S*pnorm(-d1)#direct formula
```

布莱克-肖尔斯期权价格计算的第二部分还告诉我们，可以用已知的认购价格C，计算未知的股票波动性σ。而且进行这个计算的函数是没有任何形式存在的（No Closed Form），也就是说，不可能用公式的形式写在纸上。我们能做的是提前生成一系列数值作为候选人，选择符合要求的数值作为最终的结果。

同样需要五个前提条件计算股票的波动性σ，它们是$S, X, r, \Delta t, C$。然后要生成一系列的数值做σ的候选人，每一个σ候选人都要使用公式（5.12,5.13）计算d_1, d_2。代入公式（5.14），每一个σ候选人都会有其相应的$Error$数值:

$$Error = N(d_1) - X/S \exp\{-r\Delta t\} N(d_2) - C/S \tag{5.14}$$

能使$Error$值等于0的σ候选人，就是需要得出的股票波动性。

这里的计算实例，我们使用$S = 30, X = 25, r = .06, \Delta t = .8, C = 6.19$。并且，$\sigma$数值候选人是在0-0.2上，每隔0.02取一个点。请看输出结果作图（5.4）。对应$Error = 0$的σ值是0.12，这与上面计算期权价格的例子的前提条件相同。

布莱克-肖尔斯，波动性计算R代码

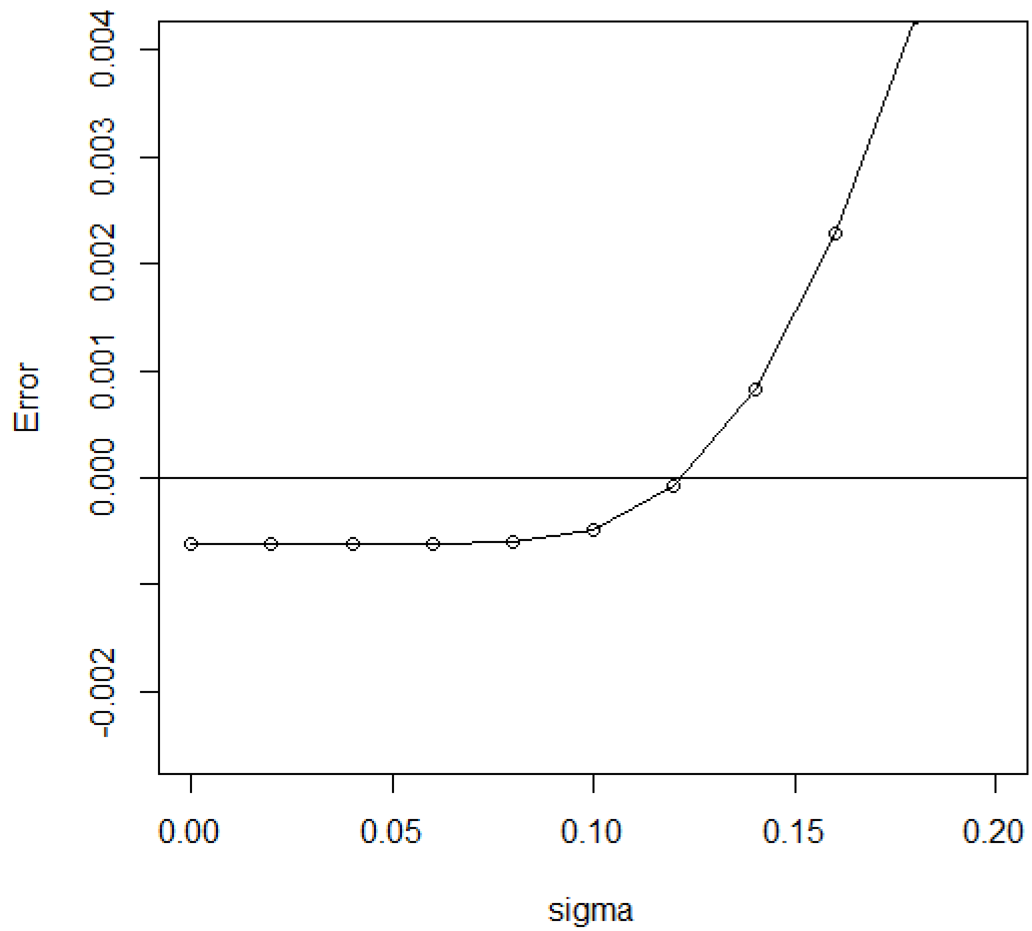


Figure 5.3: 布莱克-肖尔斯，波动性计算作图

```
#Black-Scholes Implied volatility Calculation
S =30   #Current stock price
X =25   #Exercise price
r =.06  #Risk-free of interest
dt =.8  #Time to maturity of option(in years)
C =6.19#Call Price

sigma =seq(0,.2,.02)

d1 =(log(S/X)+(r+0.5*sigma^2)*dt)/(sigma*sqrt(dt))
d2 =d1-sigma*sqrt(dt)

Error = pnorm(d1) - X/S*exp(-r*dt)*pnorm(d2) - C/S
cbind(sigma,Error)

plot(sigma,Error,type='o',ylim=c(-.0025,.004))
abline(h=0)
```

Chapter 6

对冲基金(Hedge Funds)

6.1 套期保值(Insured Portfolio)

在危机四伏的金融世界中，有一个被反复提到的专业名词叫“对冲（Hedging）”。对冲的意思是指能够降低金融投资对市场变化的敏感性的一切做法。对冲可以减小市场变化对投资的影响。对冲可以分为两种，一种是套期保值，另外一种是德尔塔避险。接下来的两个小节，会对这两种对冲概念进行详细的介绍。

比方说，你买入了一支股票，价格$56。你希望不管怎么样，在一年后都可以用至少$50的价格将这一支股票卖出去。凭着你对金融市场的了解，你会去购买这支股票的认沽期权，履约期是一年，履约价是$50。在一年到期后，如果这支股票的价格低于$50，你仍旧可以凭着这个认沽期权，在$50的价位上把这支股票处理掉。如果一年后的股票价格高于$50，你可以不执行认沽期权，保留这支股票，或在高于$50的价位将这支股票卖出。所以认沽期权保证股票的最低价值，并且不妨碍股票向上升值的空间。

套期保值（Insured Portfolio）指的是使用购买期权来保障所持股票价值的做法。更重要的是，我们可以制作自己的期权，没有必要去购买期权。这才是套期保值类对冲的核心。制作自己的期权，在上世纪70年代，是一个轰动一时的数学大发现。作为一种新方法，它曾经被大量投资者购买，并学习，而且引发了金融市场大盘的多次共鸣。至于为什么期权可以被制作，很多的概率论教科书已经将其作为定理收录，并证明。有兴趣了解细节的读者可以翻阅158页，Probability with Martingales，David Williams，1991。

具体的做法是，将一部分的资金存为无风险的银行存款赚取利息，其余的资金买成股票。根据当前的股票价格，距离到期日的时间，和预定的履约价格，来决定划分资金的比例。比方说，股票买入价为$56，希望在一年后以不低于$50的价格卖出，所以距离到期日的时间是1，履约价为$50。经

过计算，应该把75.5%的资金换成股票，把24.5%的资金作为存款。我们只考虑每周的股票价格。

一周后股票价格变为$60，距离到期日51周，换算成时间是51/52，履约价格仍旧是$50，根据这些新的信息，经过计算应该把82.5%的资金换成股票，把17.5%的资金作为存款。这样，每一个时间点上都会有新的市场信息，每一个时间点都要对资金的比例进行再平衡（Rebalance）。持续这样再平衡，一直到预定的到期日，结果就相当于复制（Replicate/Duplicate）了一个认沽期权。保证投资不低于$50的价位，而且还保留了升值的空间。

在下面的R语言代码里，我们要使用一个叫omega()的R函数来计算投资股票资金的比例。这个函数需要的输入是：股票价格S，履约价X，距离到期时间Δt，银行存款净回报r，和股票标准差σ。值得注意的是，这个函数不需要输入股票净回报的均值，因为布莱克-肖尔斯理论告诉我们，股票净回报率均值对这些计算没有意义。在omega()函数中，同样要计算d_1, d_2，然后是$N(d_1), N(d_2)$，还有认购Call，认沽Put价格。如果用小写p代表投资股票的比例，计算投资股票的比例公式如下：

$$p = \frac{S \times N(d_1)}{S + put} \tag{6.1}$$

这个公式（6.1）的R语言代码表达式：

```
p <-S*Nd1/(S+put) #proportion in shares
```

当然omega()函数中的其它数量都是前面已经详细介绍过的，这里不再重复。

这个例子里，我们使用银行存款的年利率0.08，一年有52周，所以存款周利率为0.08/52=0.001538，每周的毛回报率就是1.001538。在股票价格$56上，0.7545016的资金投入股票，0.2454984的资金投入存款。当新的股票价格出现，价位$60，股票带来的毛回报率是60/56=1.0714286。所以在已过的一周75.5%的资金有1.0714286的毛回报率，另外24.5%的资金有1.001538毛回报率。总体来讲，所有投入的资金，混在一起的毛回报率，只是一个加权平均$0.7545016 \times 1.0714286 + 0.2454984 \times 1.001538 = 1.0542707$。如果在股票$56的时候，只投入了$1000的资金，到了股票价格$60的时候，最初的投资就变成了$1054.271。

在股票价格$60，我们要再次决定如何分配股票和存款的投资。经过omega()函数的计算，0.8252678的资金投入股票，0.1747322的资金投入存款。就这样再平衡资金，一周后，股票价格$62.50152，在股票上的毛回报率是62.50152/60=1.0416920。在存款的毛回报率仍旧是1.001538。所以这一周的总投资回报是$0.8252678 \times 1.0416920 + 0.1747322 \times 1.001538 = 1.0346759$。

当观测到了$62.50152，这是第3个股票价格，所以时间已经过去了两周。第一周得到毛回报率1.0542707，第二周得到毛回报率1.0346759。像

这样我们可以计算每一周相对前一周的毛回报率，但当前相对于投资刚刚开始的毛回报率应该是计算复利。相对于投资开始的时候，现在过了两周，投资的毛回报率应该达到了$1.0542707 \times 1.0346759 = 1.090828$。所以如果投资开始的时候，总资金是\$1000，到现在两周后，投资的价值应该变成了\$1090.828。以此类推，在每一周的新股价上用布莱克-肖尔斯的omega()函数计算再平衡投资组合，等到设定的到期日，无论股票价格如何，其结果都和买了一个认沽期权一样。

用来实现这个过程的R代码很简单，除了omega()函数和最后的作图部分，剩下的代码就没有几行了。但其中的回报率计算都使用了log，exp函数。主要是log使相乘的运算关系变成了相加的运算关系，然后exp又可以把相加关系变成相乘关系。使代码更简单明了。下面是这个过程的完整R代码：

```
omega <- function(S,X,dt,r,sigma ){
    d1 <-(log(S/X)+(r+0.5*sigma^2)*dt)/(sigma*sqrt(dt))
    d2 <-d1-sigma*sqrt(dt)
    Nd1 <-pnorm(d1)
    Nd2 <-pnorm(d2)
    call  <-S*Nd1-X*exp(-r*dt)*Nd2
    put <-call-S+X*exp(-r*dt)
    p <-S*Nd1/(S+put) #proportion in shares
return(p)
}
S=c(56,60,62.50152,63.15873,62.79999,57.0989,57.86864,
57.51877,58.61156,59.01615,61.63971,64.76013,67.41739,
67.65814,68.38517,70.92214,73.10424,68.94892,65.1327,
65.816,65.63953,64.20456,64.63979,64.50028,66.8621,
69.02875,66.03646,67.10408,68.14783,68.16633,65.77052,
65.73707,63.7284,66.93607,68.31863,70.1023,75.2122,
70.24492,72.50172,74.03201,70.50749,71.59684,71.62407,
70.63535,68.13668,71.6675,71.34427,77.05522,74.58645,
76.39313,81.2814,85.82948,92.45092)

week =0:52
deposit =1000
X =rep(50,53)
dt =1-week/52
r =0.08
sigma =.30
```

```
p =omega(S,X,dt,r,sigma)
stock.R =c(exp(diff(log(S))),1)
bond.R =c(rep(1+r/52,52),1)
R =p*stock.R+(1-p)*bond.R
cum.R =exp(cumsum(log(R)))
value =deposit*cum.R
cbind(week,S,p,1-p,stock.R,bond.R,R,cum.R,value)

par(mar = c(5, 4, 4, 4) + 0.3)  # Leave space for z axis
plot(week, S,type="o",col="red") # first plot stock price
par(new = TRUE)
plot(week, value, type = "l", axes = FALSE, bty = "n", xlab =
 "", ylab = "")
axis(side=4, at = pretty(range(value)))
```

这个omega()函数告诉我们，如果股票价格在升高，就把比较多的资金投入股票，如果股票的价格在下降，就把比较多的资金放在银行里。显而易见，如果大部分的投资者都使用这个策略，当股票上升就把资金投入股市，股票下降就把资金撤出股市，结果是整个股市的暴涨暴跌。所以套期保值式的对冲策略，曾经在世界范围，多次导致各地投资市场的崩盘。

截图(6.1)演示了这种对冲策略可以追随股票价格的变化，非常紧密。黑线是股票的价格线，红线是投资价值变化线。这个插图左侧的垂直坐标表示的是股票价格的刻度，插图右侧的垂直坐标表示的是投资价值的刻度。水平的坐标表示52周。

从截图(6.1)可以看到，使用这个策略的结果是把\$1000的本金变成了\$1590.314。可以想一想，如果我们把\$1000在股票价格\$56的时候全部买成股票，到最后股票价格\$92.45092的时候，价值应该是$1000/56\times 92.45092 = \$1650.909$。使用这个策略，要比不用策略少了\$60.59529。这是因为复制认沽期权和购买认沽期权一样，都是要花代价的。如果我们在起初，股票价格\$56，使用$S = 56, X = 50, r = 0.08, \sigma = .30, \Delta t = 1$的输入，计算这个认沽价格应该是\$2.376976。所以为这些股票买认沽的花费为$1000/56 \times 2.376976 = \$42.446$。可以看出，这个策略把认沽的费用都复制出来了。但数值上的不同是因为，我们只做了52次离散的再平衡。如果我们可以做连续不停的再平衡，数值就可以对应上了。

6.2 德尔塔避险(Delta hedging)

这是另外一个类型的对冲做法，又被称作**自筹投资组合**（**Self Financing Portfolio**）。所谓的自筹投资组合是说，当组合中一些资产的价值发生变化，就会有另外一些在相反方向的资产产生相应的价值变化，从而这个投

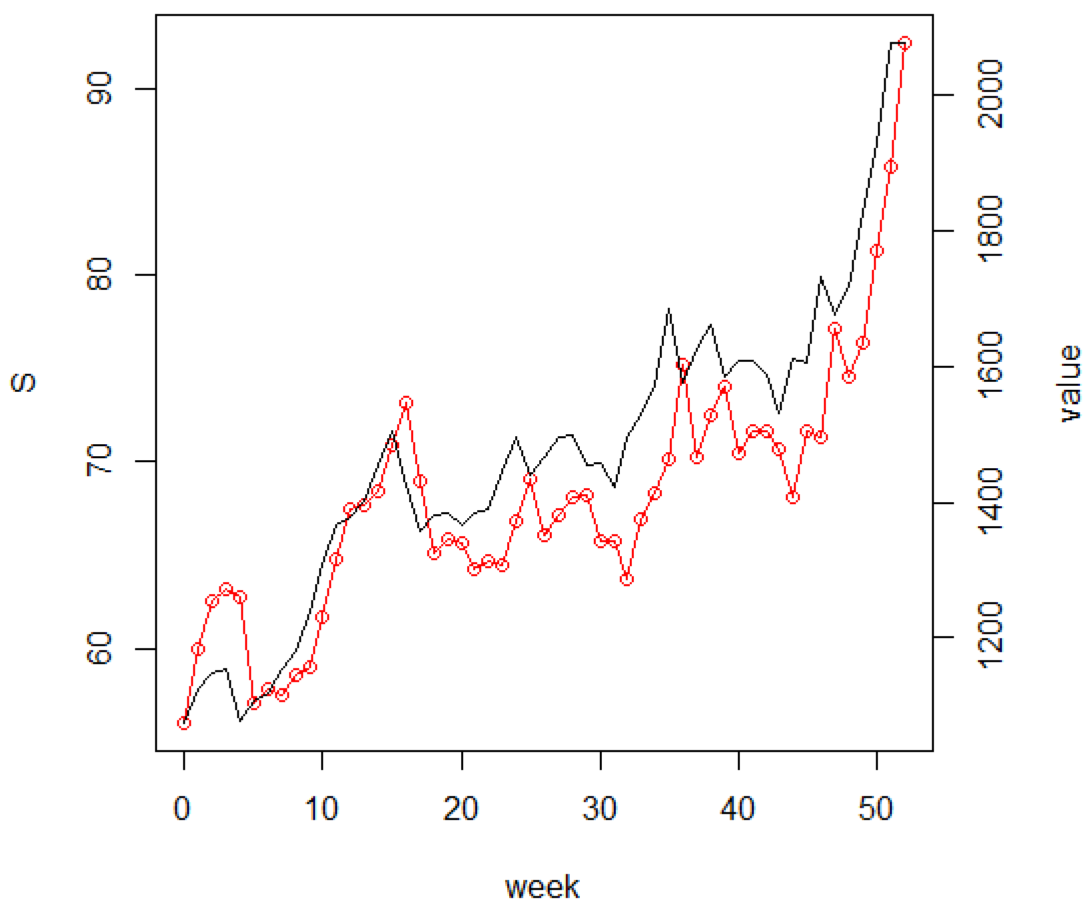


Figure 6.1: 套期保值再平衡策略投资价值走势举例插图

资组合的价值总体上不发生变化，这个组合被设立后就不需要注入更多的资金进行维护。更好的是这种对冲的做法也可以为我们制作自己的期权。

比方说，我们在做长方向上的资产有了价值变化，德尔塔避险的做法，会使在做短方向上的资产产生刚好等价并相反的价值变化，用来抵消做长方向的价值变化。在某个方向上的价值变化被称作德尔塔（Delta/δ），价值变化相互抵消被称作中性德尔塔（Delta Neutral)。

有一家银行，向一个投资者签发（write）了10000支股票的认购期权，也就是说在到期日，如何股票价格合适，这个投资者会向这个银行购买这10000支股票。这样银行向投资者出售了认购期权，收了钱，然后再考虑需要储备多少支这个股票，等履约日投资者来购买。问题是如果银行储备了10000支股票，但是履约日股票价格太低，投资者拒绝购买股票，这10000支股票就砸在了银行手里。如果银行没有储备10000支股票，到履约日股票价格高于履约价，投资者会选择购入股票，银行就无法兑现这个期权。

有趣的是，银行签发认购期权，是做短这个期权；储备股票，是买入股票，是做长这支股票。从而银行就拥有了，一长一短，两个方向上的两个资产。使用C来代表这个认购期权的价值，使用N代表储备股票的数量，S代表股票价格，所以$N \times S$代表储备股票的价值。使用V代表这个由短期权和长股票形成的投资组合的价值，用减号代表做短，就有了这个等式：

$$V = NS - C \tag{6.2}$$

广大的读者们不是银行，但只要将等式（6.2）中$-C$当做借来的钱，同样可以使用德尔塔避险。借的钱数要与签发的期权的价值相等，等式中的负号既可以代表做短，又可以代表向银行贷款。如果银行现在就把10000支股票准备好，等到履约日，再由投资者决定是否从银行买入股票，这种做法叫做全覆盖策略（Covered Strategy）。如果银行一直等到履约日，再决定是否给投资者准备股票，这种做法叫做裸奔策略（Naked Strategy）。

注意，银行已经把期权卖给了投资者，投资者持有期权的价值会随着股票价格的变化而变化。银行要决定持有多少这个股票，原则是，银行要使用它持有的股票，抵消掉投资者持有期权所有价值变化。投资者期权贬值，银行就没有必要准备太多的股票。这样就使得投资组合等式（6.2）的价值变化为0。当然，我们还是使用在每一个股票价格上进行再平衡的方法。从理论上讲，我们要使用无限多次再平衡，也就是连续的再平衡，才能使等式（6.2）的价值变化为0。还好在股票价格变化很平稳的市场状态下，每天或每周做一次再平衡，是足够的。

比方说，银行向投资者卖出了10000个认购期权，银行必须考虑需要储备多少股票才能抵消1个单位的认购期权，然后乘以10000就是需要准备股票的总数。其实答案很简单，就是我们一直在计算的数量$N(d_1)$。在布莱克-肖尔斯的理论框架中，使用$N(d_1)$个股票，就可以抵消一个认购期权的

价值变化。具体来讲，在卖这个期权的时候，股票价格$S = 100$，银行存款年利率$r = 0.04$，股票的标准差$\sigma = 0.23$，履约价$X = 105$，而且13星期后期权到期$\Delta t = 0.25$。通过这些信息我们可以计算$N(d_1)$:

$$\begin{aligned}
d_1 &= \frac{\log(S/X) + (r + 0.5\sigma^2)\Delta t}{\sigma\sqrt{\Delta t}} \\
&= \frac{\log(100/105) + (0.04 + 0.5 \times 0.23^2) \times 0.25}{0.23 \times \sqrt{0.25}} \\
&= -0.279806 \\
N(d_1) &= pnorm(-0.279806) = 0.389813
\end{aligned}$$

这个期权只能被持有13周，也就是$\frac{1}{4}$年，因为存款利率是年利率，要把时间转化成年的单位，刚购入期权的时候还有0.25年到期，所以当时的Δt是0.25。通过这些计算得到$N(d_1)$，在刚刚签约期权的日子，银行需要储备0.389813个股票来抵消1个期权的价值变化。

当然还需要计算出这个欧式期权的价格，前一章节已经介绍过方法，$C = 2.96155$。所以，10000个期权卖给投资者，银行收取\$29615.50的现金。然后银行需要在\$100的价位，购买价值$\$389813 = 10000 \times 0.389813 \times 100$的股票来进行对冲。

过了一周，股票价格变为S=98.79，距离到期日还有12星期，所以$\Delta t = 12/52$，重新计算$N(d_1) = 0.339811$。银行只需要3398.11个股票进行对冲，需要在$S = 98.79$的价位处理掉多余的股票。出售多余股票后获得现金:

$$10000(0.389813 - 0.339811)98.79 = 49396.9758$$

一开始的时候，银行为了买股票花费（Cost）了\$389813的现金，因为银行使用的钱都是客户的存款，总有一天使用连本带利还给客户，所以这些资金也需要计算利息。方法是计算每一周的连续复利。年利率是0.04，每年有52周，每一周换算$\Delta t = 1/52$:

$$Cost \times e^{r\Delta t} = 389813 \times e^{0.04 \times 1/52} = 390112.9715$$

因为处理多余股票得到\$49396.9758的现金，上一周的花费经过计算利息后为\$390112.9715。这周，在股票\$98.79价位上，设立对冲的花费就是:

$$390112.9715 - 49396.9758 = 340716$$

我们要计算每一周的花费，这也是本R代码for-loop中实现的功能。

德尔塔避险举例R代码

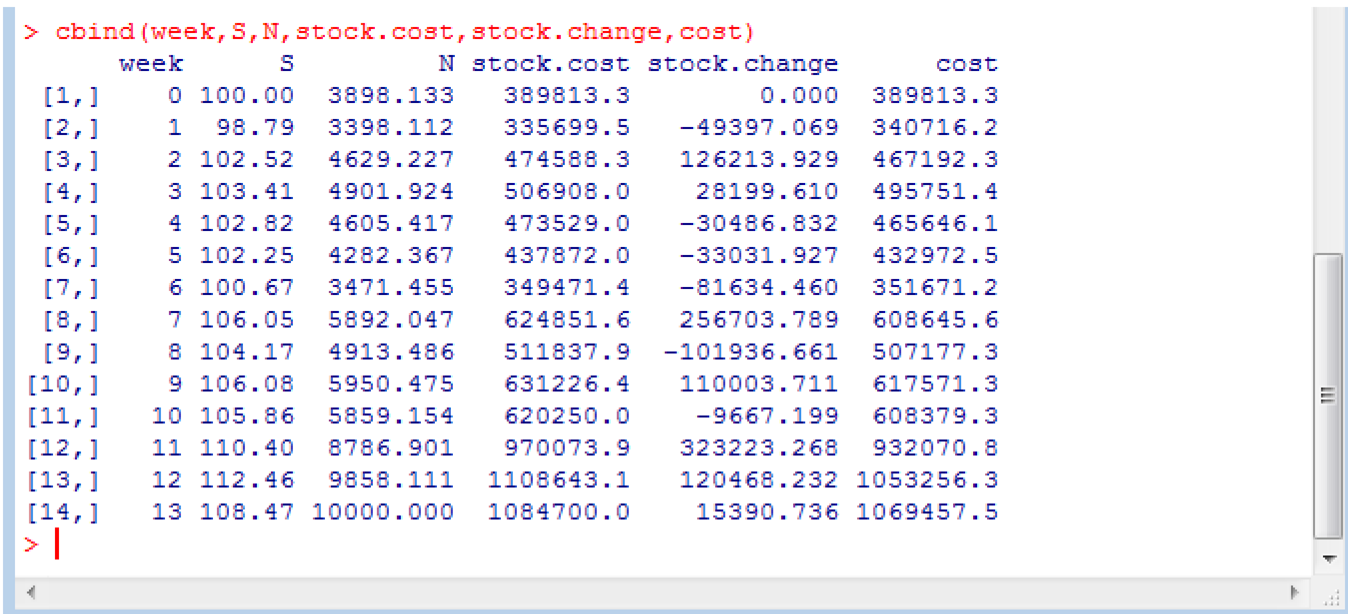
```
delta <- function(S,X,dt,r,sigma ){
```

```
    d1 <-(log(S/X)+(r+0.5*sigma^2)*dt)/(sigma*sqrt(dt))
    Nd1 <-pnorm(d1)
return(Nd1)
}

S=c( 100,98.79,102.52,103.41,102.82,102.25,100.67,
106.05,104.17,106.08,105.86,110.4,112.46,108.47)
X=105
week=0:13
dt=1/4-week/52
r=0.04
sigma=.23

N=delta(S,X,dt,r,sigma)*10000
stock.cost=N*S
stock.change=c(0,diff(N))*S
R=c(1,rep(exp(r/52),13))

cost=stock.cost
intrest.cost=rep(0,14)
for(i in 2:14){
cost[i]=cost[i-1]*R[i]+stock.change[i]
}
#interest.cost=cost*(exp(r/52)-1)
#interest.cost=c(0,interest.cost[-14])

cbind(week,S,N,stock.cost,stock.change,cost)
```

过了13个星期，最后的股票价格是$S = 108.47$，高于105的履约价格，投资者决定向银行购买股票。经过这段时间的对冲，10000股票已经准备好，而且总共的花费是\$1069457.5。股票交易，银行又从投资者收取了\$1050000现金，出售期权时收取的现金\$29615.50经过计算连续复利，应该是$29615.50e^{0.04\times 13/52}$。然后银行可以计算着整个过程的净收入:

$$1050000 + 29615.50e^{0.04\times 13/52} - 1069457.5 = 10455.60$$

代码中还计算了每周花费的利息，只是这两行命令用#符号省略了。下面是输出结果的截图（6.2），S列是股票价格序列，N列是储备股票数量的序列，stock.cost是储备股票需要的资金，stock.change列代表由于股票储备变化而产生的资金，cost列代表每周累积的总花费。



```
> cbind(week,S,N,stock.cost,stock.change,cost)
      week      S         N stock.cost stock.change      cost
 [1,]    0 100.00  3898.133   389813.3        0.000  389813.3
 [2,]    1  98.79  3398.112   335699.5   -49397.069  340716.2
 [3,]    2 102.52  4629.227   474588.3   126213.929  467192.3
 [4,]    3 103.41  4901.924   506908.0    28199.610  495751.4
 [5,]    4 102.82  4605.417   473529.0   -30486.832  465646.1
 [6,]    5 102.25  4282.367   437872.0   -33031.927  432972.5
 [7,]    6 100.67  3471.455   349471.4   -81634.460  351671.2
 [8,]    7 106.05  5892.047   624851.6   256703.789  608645.6
 [9,]    8 104.17  4913.486   511837.9  -101936.661  507177.3
[10,]    9 106.08  5950.475   631226.4   110003.711  617571.3
[11,]   10 105.86  5859.154   620250.0    -9667.199  608379.3
[12,]   11 110.40  8786.901   970073.9   323223.268  932070.8
[13,]   12 112.46  9858.111  1108643.1   120468.232 1053256.3
[14,]   13 108.47 10000.000  1084700.0    15390.736 1069457.5
>
```

Figure 6.2: 德尔塔避险再平衡策略投资花费举例输出插图