量化金融R语言第五讲——投资组合与对冲

（一）、投资组合优化

一、均值方差模型

1. 定义：

- 资产X_i意味着一个方差有限的随机变量。
- 投资组合意味着资产的组合：$P=\sum w_iX_i$，其中$\sum w_i=w\overrightarrow{1}$，$\overrightarrow{1}=(1,1,\ldots,1)$。这个组合可以是仿射的或者凸的。在凸组合的情况下，所有的资产权重都非负。
- 最优化意味着选择最佳的系数（权重）的过程，以便使投资组合满足我们的需要（也就是说，在给定的预期收益率水平上选择最小的风险，或者在给定的风险水平上选择最高的预期收益率，等等）。

2. 令$X_1,X_2,\ldots,X_n$是方差有限的随机收益率变量，$Q\in R^{n\times n}$是它们的协方差矩阵，$r=(EX_1,EX_2,\ldots,EX_n)$，$wr=\sum w_ir_i$

3. 关注的最优化问题：

$$min\left\{w^TQw\mid w\in R^n,w\overrightarrow{1}=1\right\}$$

$$max\left\{wr\mid w\in R^n,w^TQw=\sigma^2,w\overrightarrow{1}=1\right\}$$

$$min\left\{w^TQw\mid w\in R^n,w\overrightarrow{1}=1,wr=\mu\right\}$$

$$max\left\{wr-\lambda w^TQw\mid w\in R^n,w\overrightarrow{1}=1\right\}$$

$$min\left\{\sigma^2(w^TX-Y)\mid w\in R^n, w\overrightarrow{1}=1\right\}$$

w^TQw是投资组合的方差，wr是预期收益率。对于权重和，我们有$w\overrightarrow{1}=1$，这意味着我们愿意投资一个单位的资金。

4.

- 第一个问题是寻找风险最小的投资组合。如果不考虑无风险资产，答案可以是非平凡的。
- 第二个问题是在给定的方差水平上选择最大的预期收益率。
- 一个稍微不同的方法是在一个想要的预期收益率水平上寻找方差最小的投资组合。
- 第四个问题是最大化一个简单的效用函数，$return-\lambda\times risk$，其中λ是风险容忍的系数，它以一个任意数字表示我们对风险的态度。这个问题实际上等同于第一个问题。
- 在第五个问题中，Y 是第n+1个资产（比如说，一个指数），是一种我们不能买或者不想买，但需要复制的资产。其他类似的问题可以按同样的方式公式化。

第二个问题是一个带有二次约束的线性优化问题，而其他所有问题是带有线性约束的二次函数最优化问题。

二、解的概念

1. 在过去的 50 年中，人们开发了许多表现良好的伟大算法用于数值优化，尤其用于二次函数的情形。数值优化的细节讨论超出了范围。在线性约束和二次函数约束的特殊情况下，这些方法不是并不必要。我们可以使用 18 世纪提出的拉格朗日定理。

2. 如果$f:R^n\to R$和$g=(g_1,g_2,\dots,g_m):R^n\to R^m$有连续的偏导数，并且$a\in\{g(x)=0\}$是使得$g(x)=0$的$f(x)$的相对极值点，其中限定$rank(g^{'}(a))=m$。
存在系数$\lambda_1,\lambda_2,\dots,\lambda_m$使得$f^{'}(a)+\sum\lambda_i g^{'}(a)=0$。换句话说，这个函数$L:=f-\sum\lambda_i g_i:R^{m+n}\to R$所有的偏导数都是0。
在我们的例子里，这个条件也是充分的。二次函数的偏导数是线性的，所以最优化产生了求解一个线性系统方程的问题。

3. 运用它来解决第三个问题

$$min\left\{w^TQw \mid w \in R^n, w\overrightarrow{1} = 1, wr = \mu\right\}$$

等价于下面的线性方程组

$$\begin{bmatrix} Q & \vec{1} & \vec{r} \\ \vec{1} & 0 & 0 \\ \vec{r} & 0 & 0 \end{bmatrix}\begin{bmatrix} w \\ \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ \mu \end{bmatrix}$$

我们可以期望这个系统有唯一解。需要强调的是，我们利用拉格朗日定理得到的结果已经不再是一个最优化 问题。就像是在一维中，最小化二次函数需要求导数，但从复杂性的角度看，一个线性系统的方程是平凡的。

4.

$$max\left\{wr \mid w \in R^n, w^TQw = \sigma^2, w\overrightarrow{1} = 1\right\}$$

容易看出，拉格朗日函数对于λ的导数是约束本身。
为了看出这一点，我们取L的导数:

$$L := f + \sum \lambda_i g_i$$

$$\frac{\delta L}{\delta \lambda_i} = g_i$$

由此产生了一个非线性方程。

三、使用真实数据

```
minvariance <- function(assets, mu  =  0.005) {
    return <- log(tail(assets, -1) / head(assets, -1))
    Q <- rbind(cov(return), rep(1, ncol(assets)), colMeans(return))
    Q <- cbind(Q, rbind(t(tail(Q, 2)), matrix(0, 2, 2)))
    b <- c(rep(0, ncol(assets)), 1, mu)
    solve(Q, b)
}
```

获取数据：

```
library(quantmod)
getSymbols("MSFT", src = "yahoo", from = "2008-1-1", to = '2012-12-31')
```

```
## [1] "MSFT"
```

```
getSymbols("AAPL", src = "yahoo", from = "2008-1-1", to = '2012-12-31')
```

```
## [1] "AAPL"
```

```
getSymbols("GOOGL", src = "yahoo", from = "2008-1-1", to = '2012-12-31')
```

```
## [1] "GOOGL"
```

```
getSymbols("INTC", src = "yahoo", from = "2008-1-1", to = '2012-12-31')
```

```
## [1] "INTC"
```

```
IT <- data.frame(MSFT = MSFT$MSFT.Adjusted, AAPL = AAPL$AAPL.Adjusted,
                 GOOGL = GOOGL$GOOGL.Adjusted, INTC = INTC$INTC.Adjusted)
```

```
assets <- IT
return <- log(tail(assets, -1) / head(assets, -1))
head(return)
```

```
##            MSFT.Adjusted AAPL.Adjusted GOOGL.Adjusted INTC.Adjusted
## 2008-01-03   0.004250301  0.0004618833   0.0002042989  -0.027190941
## 2008-01-04  -0.028389477 -0.0794063873  -0.0422164262  -0.084545476
## 2008-01-07   0.006667774 -0.0134751915  -0.0118661591   0.009220968
## 2008-01-08  -0.034090714 -0.0366348613  -0.0274349111  -0.027472227
## 2008-01-09   0.029166264  0.0464937030   0.0335004615   0.021773862
## 2008-01-10  -0.003198446 -0.0077221421  -0.0099545034  -0.009273400
```

建立拉格朗日定理指定的线性等式系统

```
Q <- rbind(cov(return), rep(1, ncol(assets)), colMeans(return))
round(Q, 5)
```

```
##                MSFT.Adjusted AAPL.Adjusted GOOGL.Adjusted INTC.Adjusted
## MSFT.Adjusted        0.00041       0.00025        0.00025       0.00030
## AAPL.Adjusted        0.00025       0.00055        0.00032       0.00029
## GOOGL.Adjusted       0.00025       0.00032        0.00048       0.00025
## INTC.Adjusted        0.00030       0.00029        0.00025       0.00048
##                      1.00000       1.00000        1.00000       1.00000
##                     -0.00013       0.00077        0.00002      -0.00005
```

```
Q <- cbind(Q, rbind(t(tail(Q, 2)), matrix(0, 2, 2)))
round(Q, 5)
```

```
##                MSFT.Adjusted AAPL.Adjusted GOOGL.Adjusted INTC.Adjusted
## MSFT.Adjusted        0.00041       0.00025        0.00025       0.00030 1
## AAPL.Adjusted        0.00025       0.00055        0.00032       0.00029 1
## GOOGL.Adjusted       0.00025       0.00032        0.00048       0.00025 1
## INTC.Adjusted        0.00030       0.00029        0.00025       0.00048 1
##                      1.00000       1.00000        1.00000       1.00000 0
##                     -0.00013       0.00077        0.00002      -0.00005 0
##
## MSFT.Adjusted  -0.00013
## AAPL.Adjusted   0.00077
## GOOGL.Adjusted  0.00002
## INTC.Adjusted  -0.00005
##                 0.00000
##                 0.00000
```

```
mu <- 0.005
b <- c(rep(0, ncol(assets)), 1, mu)
b
```

```
## [1] 0.000 0.000 0.000 0.000 1.000 0.005
```

求解

```
solve(Q, b)
```

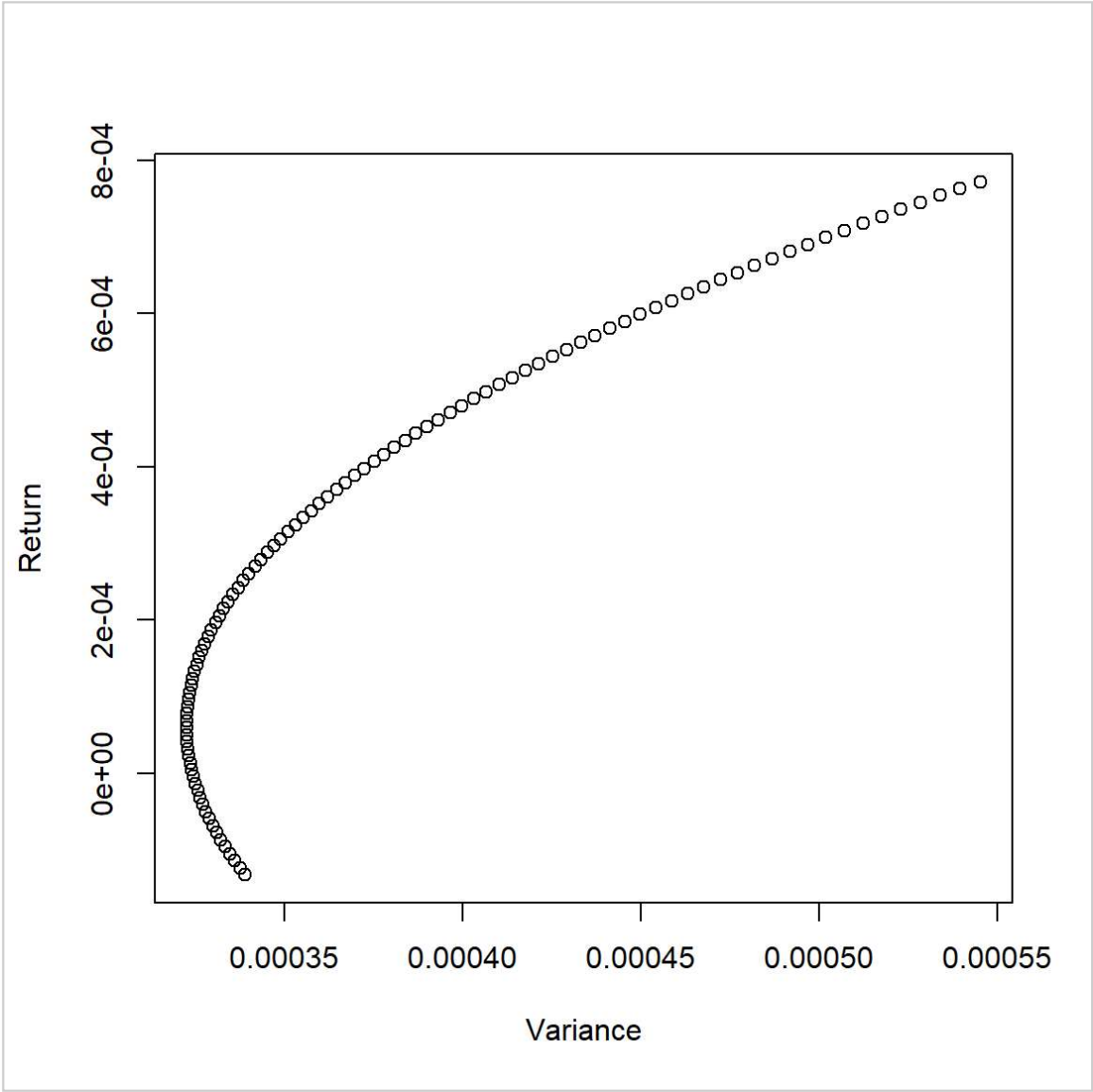
```
##  MSFT.Adjusted  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##  -1.7032635195   6.1254036936  -1.9138160774  -1.5083240967  -0.0001896878
##
##  -2.1766287364
```

```
minvariance(IT[, -1])
```

```
##  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##   6.352909e+00  -2.652009e+00  -2.700900e+00  -2.695386e-05  -2.344220e+00
```

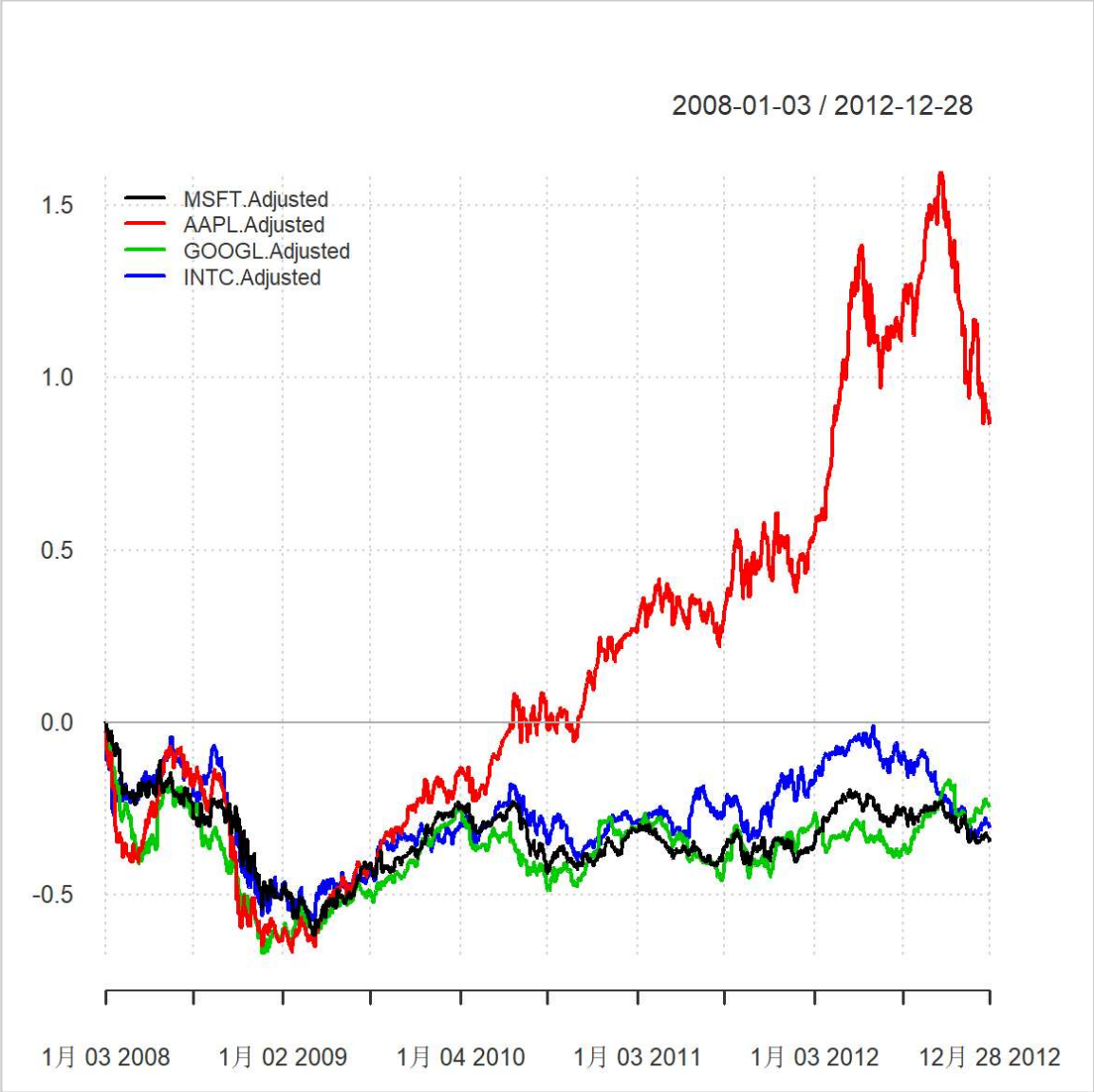
对更大的收益率范围求解最小访哈

```
frontier <- function(assets) {
  return <- log(tail(assets, -1) / head(assets, -1))
  Q <- cov(return)
  n <- ncol(assets)
  r <- colMeans(return)
  Q1 <- rbind(Q, rep(1, n), r)
  Q1 <- cbind(Q1, rbind(t(tail(Q1, 2)), matrix(0, 2, 2)))
  rbase <- seq(min(r), max(r), length = 100)
  s <- sapply(rbase, function(x) {
    y <- head(solve(Q1, c(rep(0, n), 1, x)), n)
    y %*% Q %*% y
  })
  plot(s, rbase, xlab = 'Variance', ylab = 'Return')
}
frontier(IT)
```



在方差—收益率平面上，理想的收益率—最小方差曲线叫作投资组合前沿（Portfolio Frontier）。忽略它向下方倾斜的部分（同样的方差可以用更高的预期收益率达到），我们得到了有效前沿（Efficient Frontier），毫无疑问必须选择有效前沿上的组合。
两个给定的收益率水平就足以计算投资组合前沿，把得到的投资组合连接起来就得到了整个前沿。

```
library(PerformanceAnalytics)
library(timeSeries)
IT <- timeSeries(IT)
IT_return <- returns(IT)
chart.CumReturns(IT_return, legend.loc = 'topleft', main = '')
```



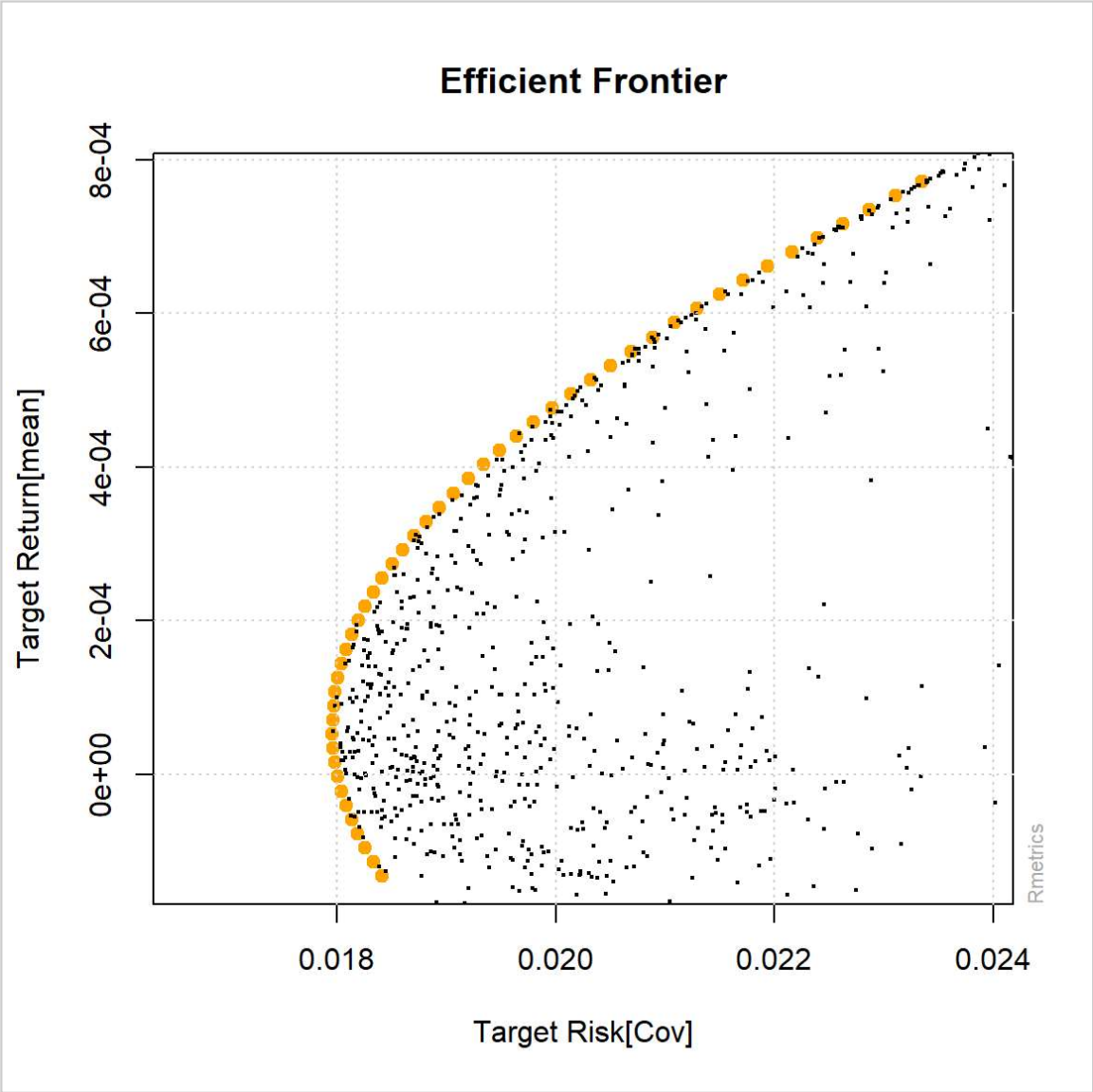

```
library(fPortfolio)
```

```
# 用交互方式绘图
plot(portfolioFrontier(IT_return))
```

```
Spec = portfolioSpec()
setSolver(Spec) = "solveRshortExact"
Frontier <- portfolioFrontier(as.timeSeries(IT_return), Spec,  constraints = "Short")
```

```
## Warning in as.vector(invSigma %*% one)/(one %*% invSigma %*% one): Recycling array of le
ngth 1 in vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
frontierPlot(Frontier, col = rep('orange', 2), pch = 19)
monteCarloPoints(Frontier, mcSteps = 1000, cex = 0.25, pch = 19)
grid()
```



四、切线组合和资本市场线

当组合中加入一个无风险资产R，会发生什么？如果$\sigma_R = 0$，并且X是任意的一个风险投资组合，那么$Var(\alpha R + (1-\alpha)X) = (1-\alpha)^2 Var(X)$，并且显然也有$E(\alpha R + (1-\alpha)X) = \alpha E(R) + (1-\alpha)E(X)$。

这意味着这些组合在均值——标准差平面上形成了一条直线。位于这条直线上的任何投资组合都可以通过投资于R和X来得到。

很明显，X的最佳选择位于这条直线与有效边界的切点。这个切点叫作市场组合或者切点组合，而风险资产的有效前沿在这个点的切线叫作资本市场线（Capital MarketLine），它包含了该情形下所有资产的有效投资组合。

五、协方差矩阵中的噪声

当我们优化投资组合时，其实我们并没有真实的协方差矩阵和预期收益率向量（它们是均值一方差模型的输入量）。我们使用观测来估计它们，因此Q、r以及模型的输出仍然是随机变量。 如果不深入细节，我们可以说模型中会产生惊人的巨大不确定性。尽管有强大数定律的保证，最优的投资组合权重会不时地在$\pm 200\%$之间变动。幸运的是，如果我们掌握有几年的数据（日收益率），测量风险的相对误差就仅为$20\% \sim 25\%$。

六、如果方差不够用

方差可以很容易地度量风险，但也存在一些缺陷。例如，在使用方差时，收益率中的正向变化也会被视为风险的增加。因此，人们开发了一些更复杂的风险度量方法。 比如下面的这个简例，主要关于多种方法运用于之前描述过的IT_return资产。

```
library(fPortfolio)
Spec <- portfolioSpec()
setSolver(Spec) <- "solveRshortExact"
setTargetReturn(Spec) <- mean(colMeans(IT_return))
efficientPortfolio(IT_return, Spec, 'Short')
```

```
##
## Title:
##  MV Efficient Portfolio
##  Estimator:         covEstimator
##  Solver:            solveRshortExact
##  Optimize:          minRisk
##  Constraints:       Short
##
## Portfolio Weights:
##  MSFT.Adjusted  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##         0.3466         0.2617         0.2224         0.1692
##
## Covariance Risk Budgets:
##  MSFT.Adjusted  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##         0.3347         0.2815         0.2188         0.1650
##
## Target Returns and Risks:
##   mean    Cov   CVaR    VaR
## 0.0002 0.0181 0.0433 0.0295
##
## Description:
##  Sun Jul 14 12:45:28 2019 by user: Xueyuanyuan
```

```
minvariancePortfolio(IT_return, Spec, 'Short')
```

```
##
## Title:
##  MV Minimum Variance Portfolio
##  Estimator:         covEstimator
##  Solver:            solveRshortExact
##  Optimize:          minRisk
##  Constraints:       Short
##
## Portfolio Weights:
##  MSFT.Adjusted  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##         0.3847         0.1528         0.2621         0.2004
##
## Covariance Risk Budgets:
##  MSFT.Adjusted  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##         0.3847         0.1528         0.2621         0.2004
##
## Target Returns and Risks:
##   mean    Cov   CVaR    VaR
## 0.0001 0.0180 0.0432 0.0290
##
## Description:
##  Sun Jul 14 12:45:28 2019 by user: Xueyuanyuan
```

```
minriskPortfolio(IT_return, Spec)
```

```
##
## Title:
##  MV Minimum Risk Portfolio
##  Estimator:         covEstimator
##  Solver:            solveRshortExact
##  Optimize:          minRisk
##  Constraints:       LongOnly
##
## Portfolio Weights:
##  MSFT.Adjusted  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##         0.3847         0.1528         0.2621         0.2004
##
## Covariance Risk Budgets:
##  MSFT.Adjusted  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##         0.3847         0.1528         0.2621         0.2004
##
## Target Returns and Risks:
##   mean    Cov   CVaR    VaR
## 0.0001 0.0180 0.0432 0.0290
##
## Description:
##  Sun Jul 14 12:45:28 2019 by user: Xueyuanyuan
```

```
maxreturnPortfolio(IT_return, Spec)
```

```
##
## Title:
##  MV Return Maximized Efficient Portfolio
##  Estimator:         covEstimator
##  Solver:            solveRshortExact
##  Optimize:          minRisk
##  Constraints:       LongOnly
##
## Portfolio Weights:
##  MSFT.Adjusted  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##         0.3466         0.2617         0.2224         0.1692
##
## Covariance Risk Budgets:
##  MSFT.Adjusted  AAPL.Adjusted GOOGL.Adjusted  INTC.Adjusted
##         0.3347         0.2815         0.2188         0.1650
##
## Target Returns and Risks:
##   mean    Cov   CVaR    VaR
## 0.0002 0.0181 0.0433 0.0295
##
## Description:
##  Sun Jul 14 12:45:28 2019 by user: Xueyuanyuan
```

（二）、最优对冲

一、衍生品的对冲

1. 对冲是同时进行两笔行情相关、方向相反、数量相当、盈亏相抵的交易。风险通常用未来现金流的波动来衡量，因此对冲的目的是减小全部组合价值的方差。对冲工具和对冲头寸不同，这通常会发生在商品敞口的对冲之中，因为商品可以在交易所交易，而交易所仅仅提供标准化（期限、数量和质量）的合约。

   最优对冲比是对冲工具占敞口的百分比，这个特定比率使整个头寸的波动率最小化。在本章中，我们会处理衍生品头寸的对冲，假设基础产品也在OTC市场上交易。因此，在敞口和对冲的衍生品之间没有错配，故基础风险不会上升。

2. 衍生品的市场风险
   远期或期货合约的价值取决于基础资产的即期价格、到期时间、无风险利率和敲定价格。在普通香草期权的情形下，基础资产的波动率也对期权价格有影响。成立的条件是，直到衍生品交易的时期，基础资产都不提供现金流（没有收入也没有成本）。否则，这种现金流也会影响价格。为了简化，这里我们先假设没有现金流（无红利支付

的股票），再讨论衍生品定价。尽管模型对其他基础资产（如货币或商品）的扩展需要稍微修正公式，但对基本逻辑没有影响。

因为敲定价格在整个期限内稳定，所以只有改变其他4个因素才会改变衍生品价值。衍生品对这几个变量的敏感性用希腊字母表示，它们是关于给定变量的一阶偏导数。

Black-Scholes-Merton模型假定了无风险利率和基础资产的波动率都是常数，因此，只要时间的改变是确定的，唯一影响衍生品价值的随机变量就是基础资产的即期价格。源于即期价格波动的风险，可以通过持有精确的delta数量来对冲，delta是衍生品价格对基础资产即期价格的敏感性：

$$\Delta = \frac{\delta c}{\delta S}$$

3. 静态delta对冲
   对冲一份远期协议对双方都是有约束力的债务，因而直截了当。在多头远期头寸中，我们确定会在其到期时买入，而空头头寸则意味着确定卖出基础资产。所以，通过按衍生品金额卖出（多头远期）或买入（空头远期）基础资产，我们可以完美地对冲远期头寸。通过对多头远期头寸的值求导，我们可以核查远期的delta:

$$LF = S - PV(K)$$

   LF表示多头远期，S代表即期价格，K是敲定价格，它是约定的远期价格。现值由PV表示。
   所以，delta等于1，而且依赖于真实的市场环境。
   然而，因为头寸每天结算，期货合约的价值是实际期货价格（F）和敲定价格（S）的差。因此，它的delta是F/S，而且会随时间而改变。

4. 动态delta对冲
   对期权来说，基础资产的支付是不确定的。这依赖于多头头寸一方的决策，这是买入期权的一方。这种或有要求权的对冲，不能由前文中买入并持有的静态策略实现。在二项式模型框架下，期权头寸总能在下一个时期对冲，而在Black-Scholes-Merton模型中，收敛到0。因此，头寸的对冲需要在每一个瞬间重新平衡。
   但在真实世界中，实际资产只能在离散的时间点上交易，所以对冲组合也在离散的时间点上调整。通过考察 一个不分红股票的普通香草平价（at-the-money）多头期权的例子，来看看对冲的结果。

```
library(fOptions)
GBSOption(TypeFlag = "c", S = 100, X = 100, Time = 1/2,
          r = 0.05, b = 0.05, sigma = 0.3)
```

```
##
## Title:
##  Black Scholes Option Valuation
##
## Call:
##  GBSOption(TypeFlag = "c", S = 100, X = 100, Time = 1/2, r = 0.05,
##      b = 0.05, sigma = 0.3)
##
## Parameters:
##           Value:
##  TypeFlag c
##  S        100
##  X        100
##  Time     0.5
##  r        0.05
##  b        0.05
##  sigma    0.3
##
## Option Price:
##  9.63487
##
## Description:
##  Sun Jul 14 12:45:28 2019
```

基于BS模型，这个看涨期权的价格是9．63487。
通常，在实践中，期权价格在标准化市场上报价，并且可以从BS公式推出隐含波动率。一个预期未来波动率低于隐含波动率的投资者，可以通过卖出期权，并同时对它进行delta对冲来获利。在下面的情境中，我们对前文中一个股票服从几何布朗运动（GBM）的期权给出空头头寸的delta对冲。我们保留了BSM模型的全部假设，除了连续时间交易。为了对冲空头头寸，我们需要有数量为delta的股票，而且随着delta的改变，需要定期地重新平衡组合。重新调整的频率应该根据基础资产的波动率和流动性调节。

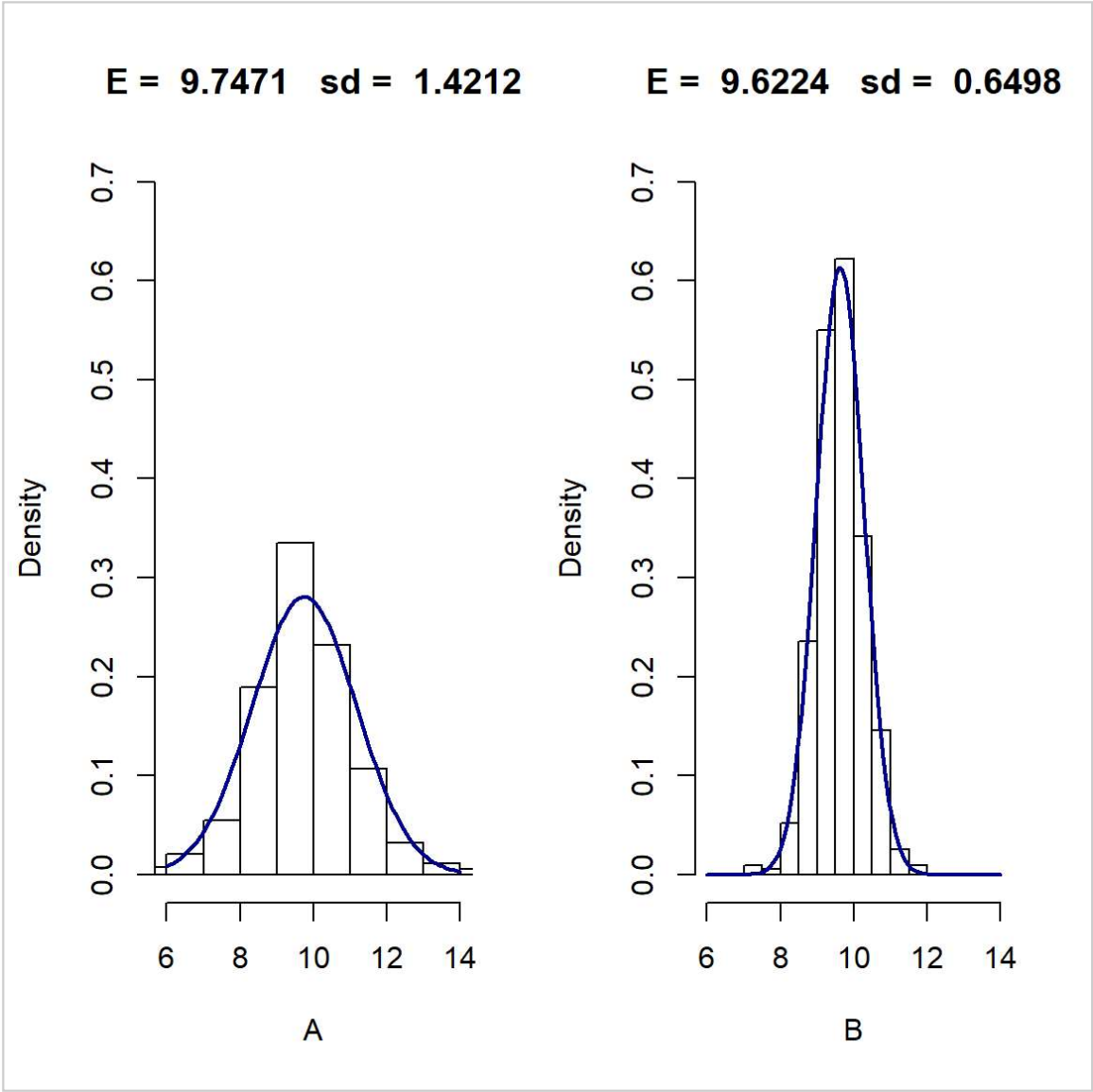
5. 来看一条股票价格的可能未来路径以及delta的变化。price_simulation函数通过给定参数生成价格过程：初始股票价格，GBM过程的漂移率和波动率以及看涨期权的保留参数和选定的重新平衡周期。

```
set.seed(2014)
Price_simulation = function(S0, mu, sigma, rf, K, Time,  dt, plots = FALSE ){

  t <- seq(0, Time, by = dt)
  N <- length(t)

  W <- c(0,cumsum(rnorm(N-1)))
  S <- S0*exp((mu-sigma^2/2)*t + sigma*sqrt(dt)*W)

  delta <- rep(0, N-1)
  call_ <- rep(0, N-1)

  for(i in 1:(N-1) ){
    delta[i] <- GBSGreeks("Delta", "c", S[i], K, Time-t[i], rf, rf, sigma)
    call_[i] <- GBSOption("c", S[i], K, Time-t[i], rf, rf, sigma)@price}

  if(plots){
    dev.new(width=30, height=10)
    par(mfrow = c(1,3))
    plot(t, S, type = "l", main = "Price of underlying")
    plot(t[-length(t)], delta, type = "l", main = "Delta", xlab = "t")
    plot(t[-length(t)], call_ , type = "l", main = "Price of option", xlab = "t")
  }
}

Price_simulation(100, 0.2, 0.3, 0.05, 100, 0.5, 1/250, plots = TRUE)
```

我们能看到一种可能的未来情景，据此即期价格上升并迅速达到价内的水平，所以期权在到期时执行。这个看涨期权的delta跟随股票价格的波动并收敛到1。如果即期价格走高，则执行看涨期权的概率上升。而为了复制这个看涨期权，我们需要买更多一些股票。同时，下降的股价导致更低的delta，标志着卖出。总之，如果股票价格昂贵，我们买入，如果价格低廉，则卖出。期权价格源于这个对冲成本。重新平衡的周期越短我们需要跟踪的价格运动越少。

6. 对冲成本定义为买入和卖出对冲头寸所需要的股票的累积净成本的现值。全部成本分成两部分，购买股份的支出数量以及头寸融资的利息。跟着BSM模型，我们使用无风险利率计算复利。我们会看到对冲成本取决于未来价格的运动，而且通过模拟一些股票的价格路径，我们可以画出成本分布。股票价格更高的波动率会引起对冲成本更高的波动率。

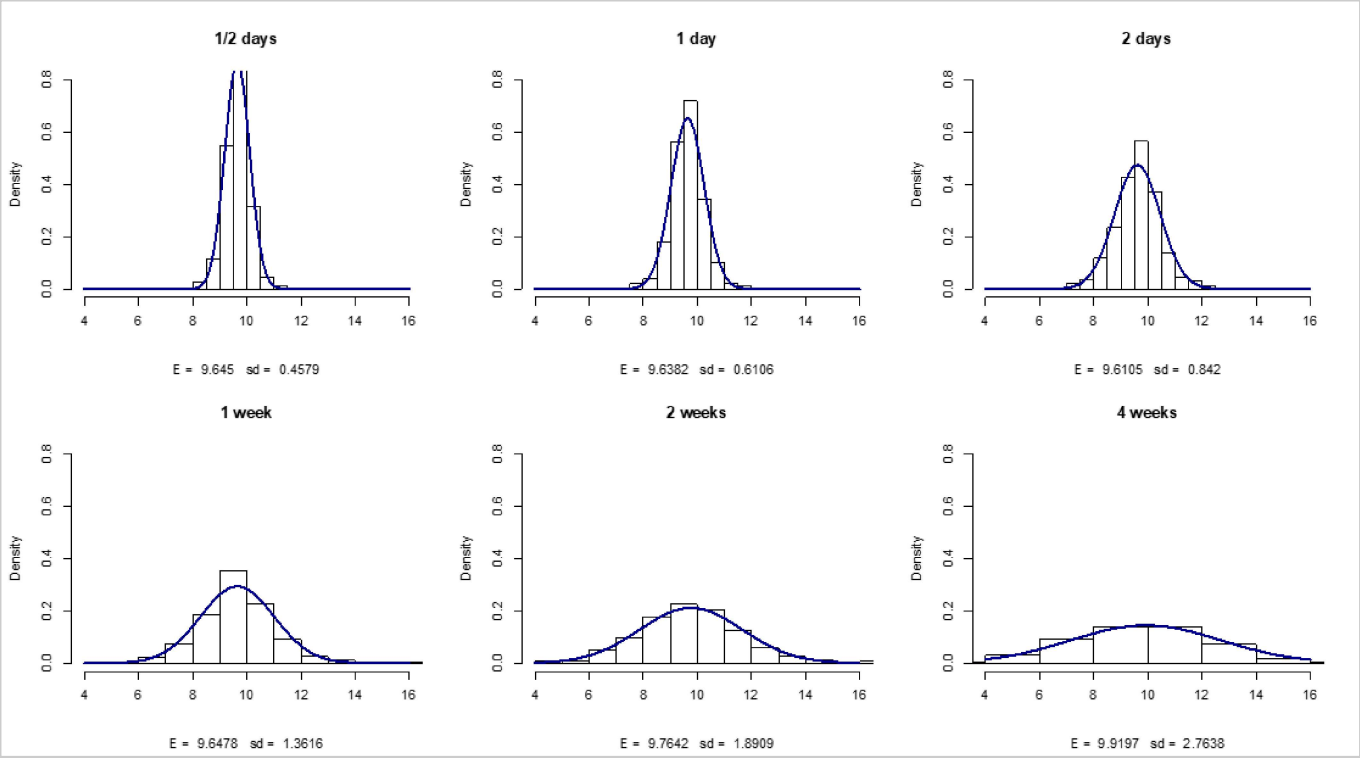
```
cost_simulation = function(S0, mu, sigma, rf, K, Time,  dt){
  t <- seq(0, Time, by = dt)
  N <- length(t)
  W <- c(0,cumsum(rnorm(N-1)))
  S <- S0*exp((mu-sigma^2/2)*t + sigma*sqrt(dt)*W)

  delta <- rep(0, N-1)
  call_ <- rep(0, N-1)

  for(i in 1:(N-1) ){
    delta[i] <- GBSGreeks("Delta", "c", S[i], K, Time-t[i], rf, rf, sigma)
    call_[i] <- GBSOption("c", S[i], K, Time-t[i], rf, rf, sigma)@price}

  share_cost <- rep(0,N-1)
  interest_cost <- rep(0,N-1)
  total_cost <- rep(0, N-1)

  share_cost[1] <- S[1]*delta[1]
  interest_cost[1] <- (exp(rf*dt)-1) * share_cost[1]
  total_cost[1] <- share_cost[1] + interest_cost[1]

  for(i in 2:(N-1)){
    share_cost[i] <- ( delta[i] - delta[i-1] ) * S[i]
    interest_cost[i] <- ( total_cost[i-1] + share_cost[i] ) * (exp(rf*dt)-1)
    total_cost[i] <- total_cost[i-1] + interest_cost[i] + share_cost[i]
  }

  c = max( S[N] - K , 0)
  cost = c - delta[N-1]*S[N] + total_cost[N-1]
  return(cost*exp(-Time*rf))
}
call_price = GBSOption("c", 100, 100, 0.5, 0.05, 0.05, 0.3)@price
A = rep(0, 1000)
for (i in 1:1000){A[i] = cost_simulation(100, .20, .30,.05, 100, 0.5, 1/52)}
B = rep(0, 1000)
for (i in 1:1000){B[i] = cost_simulation(100, .20, .30,.05, 100, 0.5, 1/250)}

par(mfrow=c(1,2))
hist(A, freq = F, main = paste("E = ",round(mean(A), 4) ,"  sd = ",round(sd(A), 4)), xlim =
  c(6,14), ylim = c(0,0.7))
curve(dnorm(x, mean=mean(A), sd=sd(A)), col="darkblue", lwd=2, add=TRUE, yaxt="n")
hist(B, freq = F, main = paste("E = ",round(mean(B), 4) ,"  sd = ",round(sd(B), 4)), xlim =
  c(6,14), ylim = c(0,0.7))
curve(dnorm(x, mean=mean(B), sd=sd(B)), col="darkblue", lwd=2, add=TRUE, yaxt="n")
```



左侧的直方图展示了周策略的成本分布，右侧的直方图属于重新平衡的日策略。
如我们所见，对冲成本的标准差可以通过缩短&而减小，这意味着对组合进行更频繁的再平衡调整，不仅对冲成本的波动率会随着变短的周期减小，而且它的期望值也会更低，接近BS价格。

7. 我们可以进一步研究重新平衡的周期，通过稍稍修正成本模拟函数来选择相同的未来路径。通过这种方式，我们可以比较不同的重新平衡的策略。
   Hull定义了delta对冲的表现度量，是卖空这个期权，并对冲它所需的成本标准差和期权理论价格的比例。
   cost_simulation函数需要修正，所以我们能同时计算几个重新平衡的周期：

```
# 仅展示部分 全部代码见附件
cost_simulation = function(S0, mu, sigma, rf, K, Time, dt, periods){

  t <- seq(0, Time, by = dt)
  N <- length(t)
  W = c(0,cumsum(rnorm(N-1)))
  S <- S0*exp((mu-sigma^2/2)*t + sigma*sqrt(dt)*W)
  SN = S[N]

  delta <- rep(0, N-1)
  call_ <- rep(0, N-1)

  for(i in 1:(N-1) ){
    delta[i] <- GBSGreeks("Delta", "c", S[i], K, Time-t[i], rf, rf, sigma)
    call_[i] <- GBSOption("c", S[i], K, Time-t[i], rf, rf, sigma)@price
  }

  S = S[seq(1, N-1, by = periods)]
  delta = delta[seq(1, N-1, by = periods)]

  m = length(S)

  share_cost <- rep(0,m)
  interest_cost <- rep(0,m)
  total_cost <- rep(0, m)

  share_cost[1] <- S[1]*delta[1]
  interest_cost[1] <- (exp(rf*dt*periods)-1) * share_cost[1]
  total_cost[1] <- share_cost[1] + interest_cost[1]


  for(i in 2:(m)){
    share_cost[i] <- ( delta[i] - delta[i-1] ) * S[i]
    interest_cost[i] <- ( total_cost[i-1] + share_cost[i] ) * (exp(rf*dt*periods)-1)
    total_cost[i] <- total_cost[i-1] + interest_cost[i] + share_cost[i]
  }

  c = max( SN - K , 0)

  cost = c - delta[m]*SN + total_cost[m]

  return(cost*exp(-Time*rf))

}
```



二、交易成本存在下的对冲

1. 增加组合调整的数目会引起对冲成本波动率的下降。因为Δt接近0，对冲成本接近通过BS公式推导的期权价格。之前，我们一直忽略了交易成本。在这里我们移除这个假设，并分析交易成本对期权对冲的效应。因为重新平衡变得更加频繁，交易成本增加了对冲的成本。但同时，更短的重新平衡周期减小了对冲成本的波动率。因此，需要在更多的细节上检验这种权衡，进而定义最优的重新平衡策略。当我们定义函数时，通过修正参数，可以把一个绝对的（对每个交易固定）或相对的（和交易规模成比例）的交易成本加入到代码中：

```
cost_simulation <- function(S0, mu, sigma, rf, K, Time, dt, periods, cost_per_trade){

  t <- seq(0, Time, by = dt)
  N <- length(t)
  W <- c(0,cumsum(rnorm(N-1)))
  S <- S0*exp((mu-sigma^2/2)*t + sigma*sqrt(dt)*W)
  SN <- S[N]

  delta <- mapply(GBSGreeks, S = S[1:(N-1)], Time = (Time-t)[1:(N-1)], Selection = "Delta",
  TypeFlag = "c", X = K, r = rf, b = rf, sigma = sigma)

  S <- S[seq(1, N-1, by = periods)]
  delta <- delta[seq(1, N-1, by = periods)]
  m <- length(S)

  share_cost <- rep(0,m)
  interest_cost <- rep(0,m)
  total_cost <- rep(0, m)

  share_cost[1] <- S[1]*delta[1] + cost_per_trade
  interest_cost[1] <- (exp(rf*dt*periods)-1) * share_cost[1]
  total_cost[1] <- share_cost[1] + interest_cost[1]

  for(i in 2:(m)){
    share_cost[i] <- ( delta[i] - delta[i-1] ) * S[i] + cost_per_trade
    interest_cost[i] <- ( total_cost[i-1] + share_cost[i] ) * (exp(rf*dt*periods)-1)
    total_cost[i] <- total_cost[i-1] + interest_cost[i] + share_cost[i]
  }

  c <- max( SN - K , 0)

  cost <- c - delta[m]*SN + total_cost[m]

  #call_price = GBSOption("c", 100, 100, 0.5, 0.05, 0.05, 0.3)@price

  return(cost*exp(-Time*rf))

}
```

交易成本的存在，抵消了频繁重新平衡引起的波动率缩减效应。所以，通过加权这些彼此联系的效应，决定了最优重新平衡周期。

2. 对冲最优化
   为了找到重新平衡周期的最优长度，我们必须定义**最优化准则**和**最大化或者最小化的度量**。一般来说，对冲通过对冲成本的方差度量来减少风险。因此，最优对冲需要最小化对冲成本的波动率。最优化的另一个目的是最小化成本的预期值。

如果没有交易成本，选择频率更高的重新平衡对冲组合，两个目标可以同时到达。

交易成本不仅提高成本的期望值，也提高波动率，当重新调整过于频繁时，波动率会大幅度上升。

3. 需要权衡期望值和波动率时，可以考虑在金融领域中普遍的一个方法，定义**效用函数**和**效用最大化的最优条**件。例如，在组合理论中假定的个体效应函数，回报率的预期值对它有正影响，回报率的方差对它有负影响。我们可以使用同样的技术，定义一个同时包含对冲成本的期望值和方差的效用函数。但是，在我们的例子里，两个因素都对交易者的效应有负影响。因此，两个函数都必须有正号，并且函数需要最小化，由此，目标函数是一个如下定义的效用函数：

$$U(x) = E(x) + \alpha Var(x)$$

x是表示对冲成本的随机变量，α是风险厌恶函数。

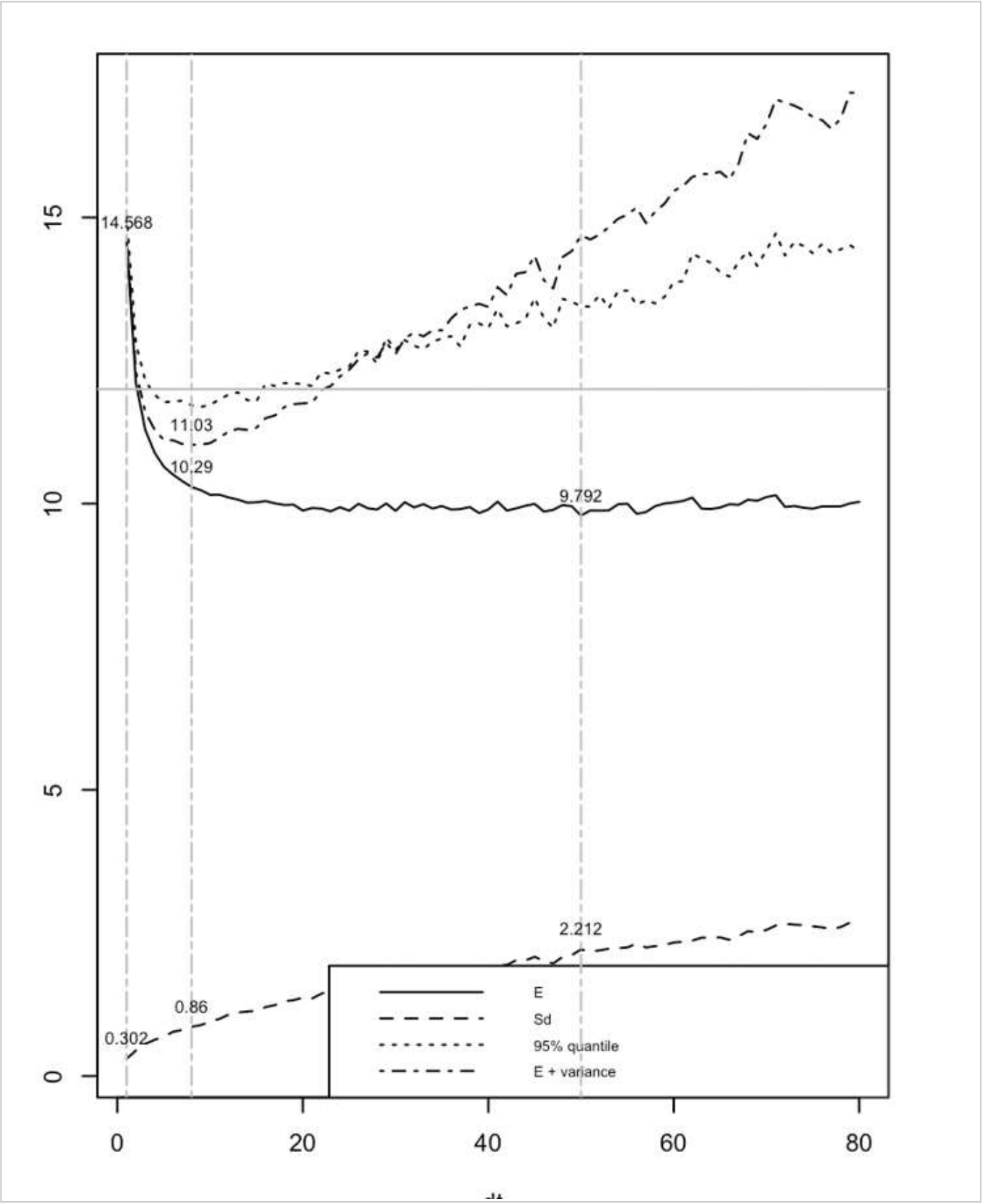
4. 还有一种方法可以替代均值-方差优化方法，把期望的（成本）值作为主要目标进行最小化，边界条件是选定风险度量为预定义值。在这里，我们选择VaR作为控制变量，这是一种下行风险度量，定义为在预定义的概率和选定时间区间上的最大损失或者最坏结果。

（此处模拟代码见附件）

5. 绝对交易成本情形下的最优对冲 对于存在交易成本和参数已经考察过的普通看涨期权来说，这个任务是找到重新平衡周期的最优长度。假设每笔交易的交易成本是0.01。之前函数的输出是一个矩阵，包含属于不同重新平衡周期的分布参数，以及对应不同准则的最优条件。

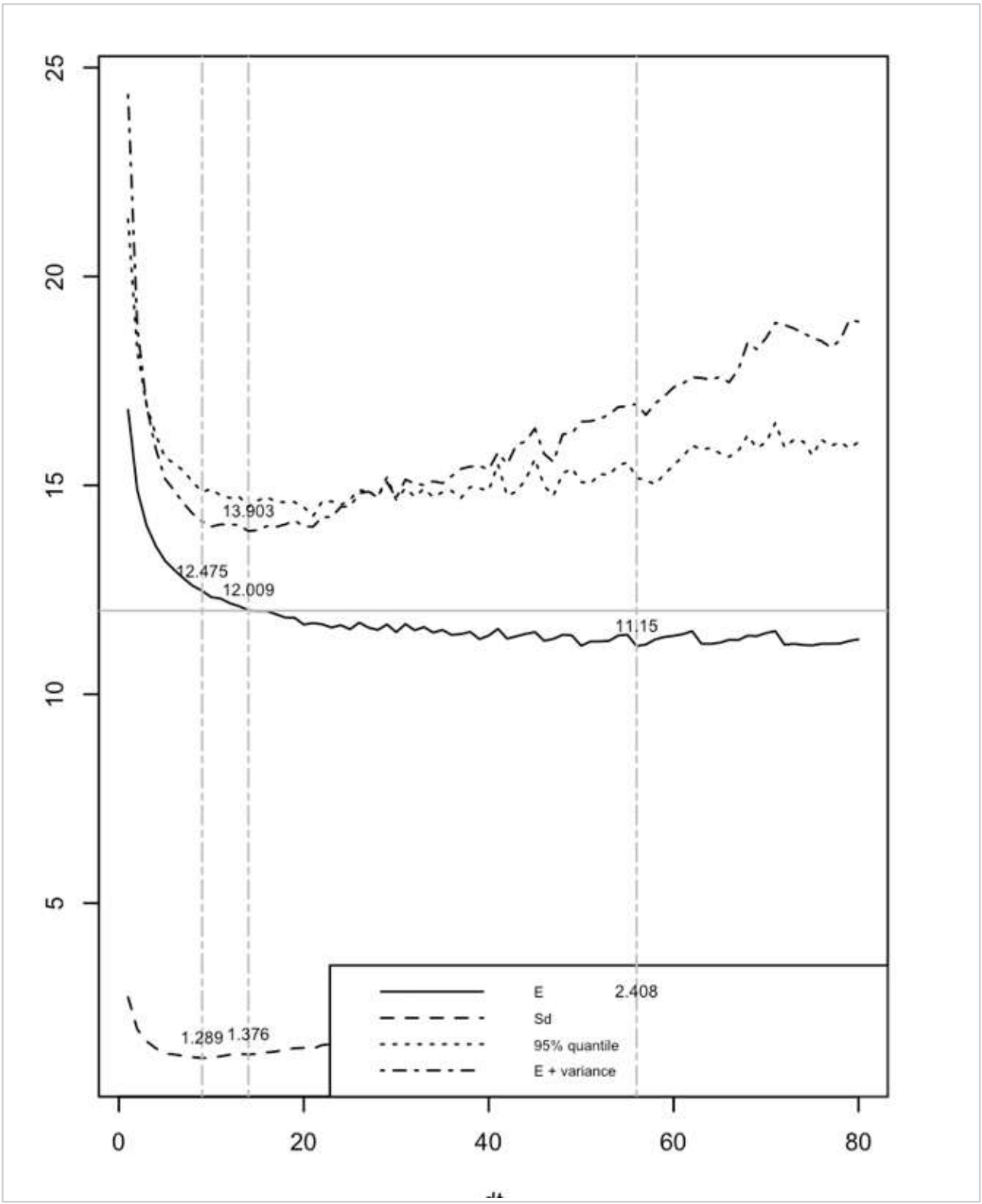
```
> (paste("E min =", z1, "cost of hedge = ",e[z1]," sd = ", s[z1]))
[1] "E min = 50 cost of hedge =  9.79184040508574  sd =  2.21227796458088"
> (paste("s min =", z2, "cost of hedge = ",e[z2]," sd = ", s[z2]))
[1] "s min = 1 cost of hedge =  14.5680033393436  sd =  0.302237879069943"
> (paste("U min =", z3, "u = ",u[z3],"cost of hedge = ",e[z3]," sd = ", s[z3]))
[1] "U min = 8 u =  11.0296321604941 cost of hedge =  10.2898541853535  sd =  0.860103467694772"
```

尽管最优化取决于参数，这张图阐明了存在交易成本时，期望成本与波动率之间的权衡。



6. 绝对交易成本情形下的最优对冲 现在的交易成本是成交金额的1%，其他参数不变。

```
> (paste("E min =", z1, "cost of hedge = ",e[z1]," sd = ", s[z1]))
[1] "E min = 56 cost of hedge =  11.1495374978655  sd =  2.4079570467643"
> (paste("s min =", z2, "cost of hedge = ",e[z2]," sd = ", s[z2]))
[1] "s min = 9 cost of hedge =  12.4747301348104  sd =  1.28919873150291"
> (paste("U min =", z3, "u = ",u[z3],"cost of hedge = ",e[z3]," sd = ", s[z3]))
[1] "U min = 14 u =  13.9033123535802 cost of hedge =  12.0090095949856  sd =  1.37633671701175"
```



通过以上最优化过程，可以看到： 存在交易成本时，仅仅考虑降低波动性可能对导致成本的大幅上升。因此，最优对冲策略需要同时考虑其影响。

三、进一步扩展

1. 金融资产的回报率通常不像BSM模型中假设的正态分布，它们的尾部比高斯曲线的预测更厚。使用GARCH模型可以表现出其波动的集聚性。

2. 另外一种方法，也可以通过在过程中建立随机跳跃，可以在极端回报率中捕捉到更高的概率。

3. 在模型中运用这些过程，会使衍生品的对冲更加昂贵，从而增大了成本分布的期望值和方差。

4. 除了delta中性组合，还有gamma中性组合：因为最新资产的gamma是零，不能通过仅仅持有衍生品和标的资产获得，我们需要为任意期限或执行价格的相同标的资产买入期权。