Code

Mean-variance Portfolio Optimization

Casey Tirshfield

Ledoit and Wolf (2003, 2004) propose to estimate μ by $\overline{\mathbf{X}}$ but to shrink the MLE of Σ toward structured covariance matrices that can have relatively small "estimation error" in comparison with the MLE of Σ. Let $\mathbf{S} = \sum_{t=1}^{n} (\mathbf{r}_t - \bar{\mathbf{r}})(\mathbf{r}_t - \bar{\mathbf{r}})^\top / n$. Ledoit and Wolf's rationale is that $\mathbf{S}$ has a large estimation error (or, more precisely, variances of the matrix entries) when $p(p+1)/2$ is comparable with n, whereas a structured covariance matrix $\mathbf{F}$) has much fewer parameters and can therefore be estimated with much smaller variances. In particular, they consider $\mathbf{F}$ that corresponds to the single-factor model in CAPM (see Sections 3.3 and 3.4 of of Tze Leung Lai and Haipeng Xing's book, "Statistical Models and Methods for Financial Markets") and point out that its disadvantage is that $\mathbf{\Sigma}$ may not equal $\mathbf{F}$, resulting in a bias of $\hat{\mathbf{F}}$ when the assumed structure (e.g., CAPM) does not hold. They therefore propose to estimate $\mathbf{\Sigma}$ by a convex combination of $\hat{\mathbf{F}}$ and $\mathbf{S}$:

$$\hat{\mathbf{\Sigma}} = \hat{\delta}\hat{\mathbf{F}} + (1 - \hat{\delta})\mathbf{S}, \quad (1)$$

where $\hat{\delta}$ is an estimator of the optimal shrinkage constant δ used to shrink the MLE toward the estimated structured covariance matrix $\hat{\mathbf{F}}$.

The file `m_ret_10stocks.txt` contains the monthly returns of ten stocks from January 1994 to December 2006. The ten stocks include Apple Computer, Adobe Systems, Automatic Data Processing, Advanced Micro Devices, Dell, Gateway, Hewlett-Packard Company, International Business Machines Corp., Microsoft Corp., and Oracle Corp. The file `m_sp500ret_3mtcm.txt` contains three columns. The second column gives the monthly returns of the S&P 500 index January 1994 to December 2006. The third column gives the monthly rates of the 3-month Treasury bill in the

secondary market, which are obtained from the Federal Reserve Bank of St. Louis and used as the risk-free rate here. Consider portfolios that consist of the ten stocks and allow short selling.

Hide

```
df <- read.table('m_sp500ret_3mtcm.txt', skip=1, header=TRUE)
logret <- read.table('m_logret_10stocks.txt', header=TRUE)
logret <- logret[,-1]
rfr <- df[,3] / (12 * 100)
```

(a) Using a single-index model for the structured covariance matrix $\mathbf{F}$, calculate the estimate $\hat{\mathbf{F}}$ of $\mathbf{F}$ in (1).

Hide

```
sp <- df[,2]
logret <- as.matrix(logret)
var_not <- var(sp)
model <- lm(logret ~ sp)
beta <- coef(model)[2,]
F1 <- var_not * (beta %*% t(beta)) + diag(diag(cov(resid(model))))
print('The estimated F hat is:')
```

```
[1] "The estimated F hat is:"
```

Hide

```
print(F1)
```

```
             AAPL         ADBE          ADP          AMD         DELL          GTW
HP           IBM         MSFT         ORCL
 [1,] 0.0045492771 0.0006594479 0.0003725823 0.0009955204 0.0007287262 0.0009657038 0
.0003780103 0.0005878055 0.0006343595 0.0006829952
 [2,] 0.0006594479 0.0044543592 0.0004222579 0.0011282508 0.0008258856 0.0010944589 0
.0004284095 0.0006661763 0.0007189372 0.0007740574
 [3,] 0.0003725823 0.0004222579 0.0007882798 0.0006374519 0.0004666182 0.0006183598 0
.0002420477 0.0003763838 0.0004061933 0.0004373357
 [4,] 0.0009955204 0.0011282508 0.0006374519 0.0071438193 0.0012467792 0.0016522248 0
.0006467386 0.0010056777 0.0010853271 0.0011685380
 [5,] 0.0007287262 0.0008258856 0.0004666182 0.0012467792 0.0035220840 0.0012094373 0
.0004734161 0.0007361614 0.0007944652 0.0008553760
 [6,] 0.0009657038 0.0010944589 0.0006183598 0.0016522248 0.0012094373 0.0065056598 0
.0006273683 0.0009755570 0.0010528208 0.0011335395
 [7,] 0.0003780103 0.0004284095 0.0002420477 0.0006467386 0.0004734161 0.0006273683 0
.0023049709 0.0003818672 0.0004121109 0.0004437070
 [8,] 0.0005878055 0.0006661763 0.0003763838 0.0010056777 0.0007361614 0.0009755570 0
.0003818672 0.0014830329 0.0006408320 0.0006899639
 [9,] 0.0006343595 0.0007189372 0.0004061933 0.0010853271 0.0007944652 0.0010528208 0
.0004121109 0.0006408320 0.0020186643 0.0007446088
[10,] 0.0006829952 0.0007740574 0.0004373357 0.0011685380 0.0008553760 0.0011335395 0
.0004437070 0.0006899639 0.0007446088 0.0038527785
```

(b) The $\hat{\delta}$ in (1) and suggested by Ledoit and Wolf (2003, 2004) is of the following form. Let $\hat{f}_{ij}$ and $\hat{\sigma}_{ij}$ denote the (i,j) the entry of $\hat{\mathbf{F}}$ and $\mathbf{S}$, respectively, and define:

$$\hat{\gamma} = \sum_{i=1}^{p}\sum_{i=1}^{p}(\hat{f}_{ij} - \hat{\sigma}_{ij})^2, \quad \overline{\sigma} = \frac{2}{p(p-1)}\sum_{i=1}^{p-1}\sum_{j=i+1}^{p}\frac{\hat{\sigma}_{ij}}{\sqrt{\hat{\sigma}_{ii}\hat{\sigma}_{jj}}},$$

$$\hat{\pi}_{ij} = n^{-1}\sum_{t=1}^{n}\{(r_{it} - \overline{r}_i)(r_{jt} - \overline{r}_j) - \hat{\sigma}_{ij}\}^2, \quad \hat{\pi} = \sum_{i=1}^{p}\sum_{i=1}^{p}\hat{\pi}_{ij},$$

$$\hat{\theta}_{k,ij} = n^{-1}\sum_{t=1}^{n}\{(r_{kt} - \overline{r}_k)^2 - \hat{\sigma}_{kk}\}\{(r_{it} - \overline{r}_i)(r_{jt} - \overline{r}_j) - \hat{\sigma}_{ij}\},$$

$$\hat{\rho} = \sum_{i=1}^{p}\hat{\pi}_{ii} + \sum_{i=1}^{p}\sum_{j\neq i,j=1}\frac{\overline{\sigma}}{2}\left\{\sqrt{\frac{\hat{\sigma}_{jj}}{\hat{\sigma}_{ii}}}\hat{\theta}_{i,ij} + \sqrt{\frac{\hat{\sigma}_{ii}}{\hat{\sigma}_{jj}}}\hat{\theta}_{j,ij}\right\}, \quad \hat{\kappa} = \frac{\hat{\pi} - \hat{\rho}}{\hat{\gamma}}$$

Then $\hat{\delta} = \min\left\{1, \left(\frac{\hat{\kappa}}{n}\right)_+\right\}$. Compute the covariance estimate (1) with $\hat{\mathbf{F}}$ in (a) and the $\hat{\delta}$ suggested by Ledoit and Wolf, and plot the estimated efficient frontier using this covariance estimate.

Hide

```
n <- dim(logret)[1]
p <- dim(logret)[2]
# S
S <- t(logret - mean(logret, 2)) %*% (logret - mean(logret, 2)) / n
# gamma
gamma <- sum(sum((F1 - S) ^ 2))
# sig_overline
sig_overline <- 0
for (i in 1:(p - 1)){
  for (j in (i + 1):p){
    sig_overline <- sig_overline + S[i,j] / sqrt(S[i,i] * S[j,j])
  }
}
sig_overline <- sig_overline * 2 / (p * (p - 1))
# pi_ij
pi_ij <- function(i,j){
  sum(((logret[,i] - mean(logret[,i])) * (logret[,j] - mean(logret[,j])) - S[i,j]) ^
2) / n
}
# pi
pi <- 0
for (i in 1:p){
  for (j in 1:p){
    pi <- pi + pi_ij(i,j)
  }
}
# theta
theta <- function(k,i,j){
  sum(((logret[,k] - mean(logret[,k])) ^ 2 - S[k,k]) * ((logret[,i] - mean(logret[,i]
)) * (logret[,j] - mean(logret[,j])) - S[i,j])) / n
}
# rho
rho <- 0
for (i in 1:p){
  rho <- rho + pi_ij(i,i)
  for (j in 1:p){
    if (j!=i){
      rho <- rho + sig_overline * (sqrt(S[j,j] / S[i,i]) * theta(i,i,j) + sqrt(S[i,i]
/ S[j,j]) * theta(j,i,j)) / 2
    }
```

```
  }
}
# kappa
kappa <- (pi - rho) / gamma
# delta
delta <- min(1, max(kappa / n, 0))
# estimated efficient frontier
# this code was inspired by the following matlab code: https://web.stanford.edu/~xing
/statfinbook/_BookFun/ex3.2.4_plot_6assets_effifrontier.m
eef <- function(mean, Cov, mu_star){
  inverseCov <- solve(Cov)
  ind <- rep(1, length(mean))
  A <- as.numeric(mean %*% inverseCov %*% ind)
  B <- as.numeric(mean %*% inverseCov %*% mean)
  C <- as.numeric(ind %*% inverseCov %*% ind)
  D <- B * C - A ^ 2
  return((B * inverseCov %*% ind - A * inverseCov %*% mean + mu_star * (C * inverseCo
v %*% mean - A * inverseCov %*% ind)) / D)
}
mu <- apply(logret, 2, mean)
# Sigma
Sigma1 <- delta * F1 + (1 - delta) * S
print('The covariance estimate is:')
```

```
[1] "The covariance estimate is:"
```

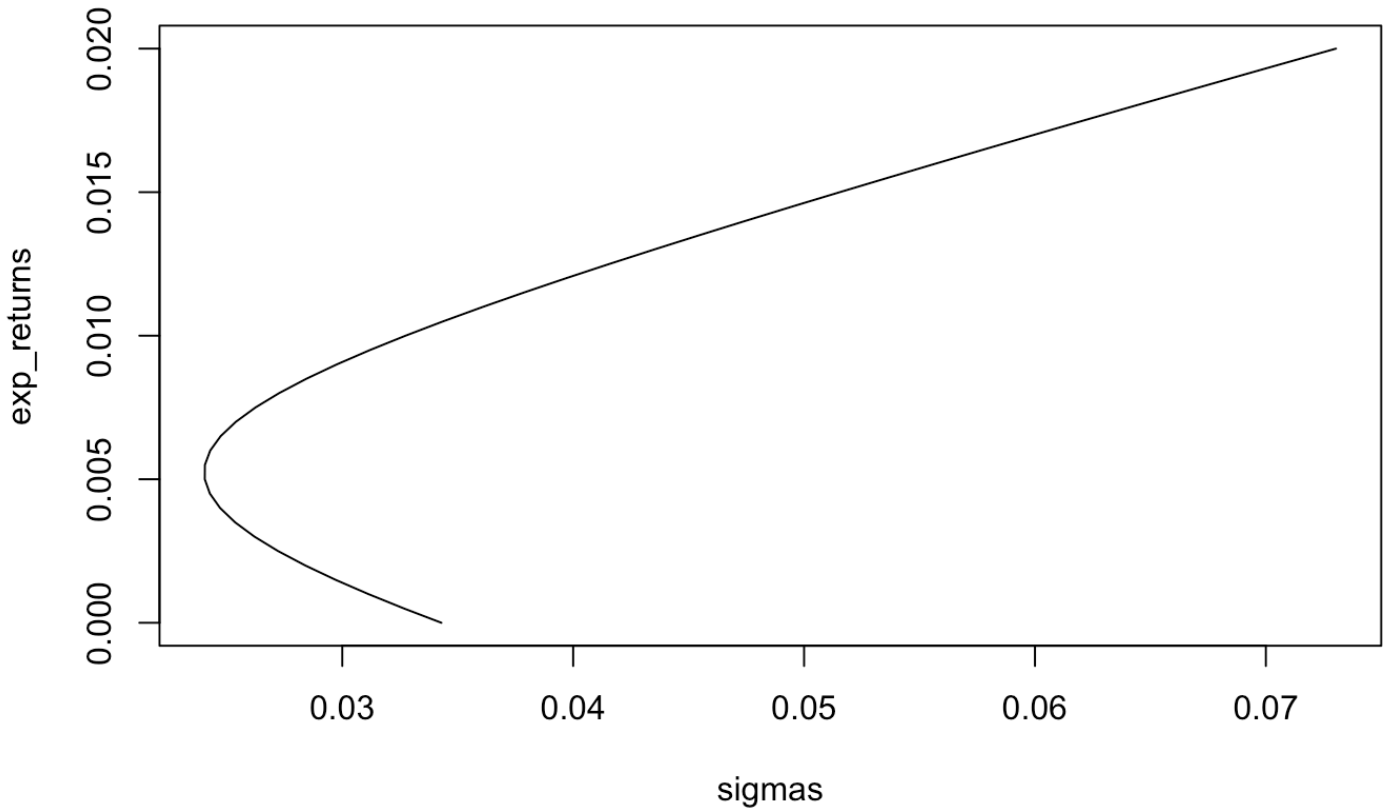
Hide

```
print(Sigma1)
```

```
              AAPL         ADBE          ADP          AMD         DELL          GTW
HP           IBM         MSFT         ORCL
 [1,] 0.0045289833 0.0011246155 0.0001994715 0.0021203392 0.0015665348 0.0018503128 0
.0006219641 0.0008382819 0.0008167951 0.0011404295
 [2,] 0.0011246155 0.0044355819 0.0004833082 0.0015576121 0.0010508003 0.0016714792 0
.0006106607 0.0005090313 0.0006407475 0.0010525509
 [3,] 0.0001994715 0.0004833082 0.0007887517 0.0005459344 0.0003920929 0.0006014173 0
.0002801375 0.0004028768 0.0003454025 0.0003948661
 [4,] 0.0021203392 0.0015576121 0.0005459344 0.0071228783 0.0017491509 0.0026693453 0
.0008257516 0.0014049672 0.0011482650 0.0014213212
 [5,] 0.0015665348 0.0010508003 0.0003920929 0.0017491509 0.0035281025 0.0021040546 0
.0005168702 0.0009674676 0.0013614036 0.0010908835
 [6,] 0.0018503128 0.0016714792 0.0006014173 0.0026693453 0.0021040546 0.0065308331 0
.0005206440 0.0010088434 0.0013133786 0.0011078543
 [7,] 0.0006219641 0.0006106607 0.0002801375 0.0008257516 0.0005168702 0.0005206440 0
.0022957632 0.0004512972 0.0001830560 0.0003462472
 [8,] 0.0008382819 0.0005090313 0.0004028768 0.0014049672 0.0009674676 0.0010088434 0
.0004512972 0.0014766263 0.0008194188 0.0007877517
 [9,] 0.0008167951 0.0006407475 0.0003454025 0.0011482650 0.0013614036 0.0013133786 0
.0001830560 0.0008194188 0.0020102282 0.0008630179
[10,] 0.0011404295 0.0010525509 0.0003948661 0.0014213212 0.0010908835 0.0011078543 0
.0003462472 0.0007877517 0.0008630179 0.0038356616
```

Hide

```
exp_returns <- seq(0, 0.02, 0.0005)
weights <- sapply(exp_returns, function(x){eef(mean=mu, Cov=Sigma1, x)})
risk <- function(x){sqrt(x %*% Sigma1 %*% x)}
sigmas <- apply(weights, 2, risk)
# plotting the estimated efficient frontier
plot(sigmas, exp_returns, type='l')
```



(c) Perform PCA on the ten stocks. Using the first two principal components as factors in a two-factor model for $\mathbf{F}$ (see Section 3.4.3 of Tze Leung Lai and Haipeng Xing's book, "Statistical Models and Methods for Financial Markets"), estimate $\mathbf{F}$.

Hide

```
# we compute the standardized log returns
stand_logret <- apply(logret, 2, scale)
# we perform PCA
logret_pca <- prcomp(stand_logret)
# the principal components are
logret_pc <- stand_logret %*% logret_pca$rotation
#the variance of the principal components are
logret_varpc <- logret_pca$sdev ^ 2
# we take the first two principal components as factors
f1 <- logret_pc[,1]
f2 <- logret_pc[,2]
model <- lm(logret ~ f1 + f2)
beta1 <- coef(model)[2,]
beta2 <- coef(model)[3,]
# we estimate F
F2 <- logret_varpc[1] * (beta1 %*% t(beta1)) + logret_varpc[2] * (beta2 %*% t(beta2))
+ diag(diag(cov(resid(model))))
print('The estimated F is:')
```

```
[1] "The estimated F is:"
```

Hide

```
print(F2)
```

```
              AAPL         ADBE          ADP          AMD         DELL          GTW
HP           IBM          MSFT         ORCL
 [1,] 0.0045492771 0.0013731827 0.0004846995 0.0025545453 0.0021329307 0.0025297257
5.069705e-04 0.0012272349  1.474016e-03 0.0015238713
 [2,] 0.0013731827 0.0044543592 0.0007521919 0.0021118726 0.0012138063 0.0016667121
1.395939e-03 0.0009721096  6.437053e-04 0.0011354828
 [3,] 0.0004846995 0.0007521919 0.0007882798 0.0007999888 0.0003906500 0.0005779259
6.517050e-04 0.0003628976  1.715005e-04 0.0004144836
 [4,] 0.0025545453 0.0021118726 0.0007999888 0.0071438193 0.0026138745 0.0031983470
1.100183e-03 0.0016223236  1.722004e-03 0.0019834961
 [5,] 0.0021329307 0.0012138063 0.0003906500 0.0026138745 0.0035220840 0.0027157031
2.260847e-04 0.0012684585  1.680487e-03 0.0015965098
 [6,] 0.0025297257 0.0016667121 0.0005779259 0.0031983470 0.0027157031 0.0065056598
5.543422e-04 0.0015400197  1.892815e-03 0.0019181523
 [7,] 0.0005069705 0.0013959393 0.0006517050 0.0011001829 0.0002260847 0.0005543422
2.304971e-03 0.0004750316 -8.966477e-05 0.0004996231
 [8,] 0.0012272349 0.0009721096 0.0003628976 0.0016223236 0.0012684585 0.0015400197
4.750316e-04 0.0014830329  8.460241e-04 0.0009482891
 [9,] 0.0014740158 0.0006437053 0.0001715005 0.0017220039 0.0016804868 0.0018928151 -
8.966477e-05 0.0008460241  2.018664e-03 0.0010821380
[10,] 0.0015238713 0.0011354828 0.0004144836 0.0019834961 0.0015965098 0.0019181523
4.996231e-04 0.0009482891  1.082138e-03 0.0038527785
```

(d) Using the estimated $\hat{\mathbf{F}}$ in (c) as the shrinkage target in (1), compute the new value of and the new shrinkage estimate (1) of $\boldsymbol{\Sigma}$. Plot the corresponding estimated efficient frontier and compare it with that in (b).

Hide

```
# we implement the same steps as in section (b) using our new F
# S
S <- t(logret - mean(logret, 2)) %*% (logret - mean(logret, 2)) / n
# gamma
gamma <- sum(sum((F2 - S) ^ 2))
# sig_overline
sig_overline <- 0
for (i in 1:(p - 1)){
  for (j in (i + 1):p){
    sig_overline <- sig_overline + S[i,j] / sqrt(S[i,i] * S[j,j])
  }
}
sig_overline <- sig_overline * 2 / (p * (p - 1))
# pi_ij
pi_ij <- function(i,j){
  sum(((logret[,i] - mean(logret[,i])) * (logret[,j] - mean(logret[,j])) - S[i,j]) ^
2) / n
}
# pi
pi <- 0
for (i in 1:p){
  for (j in 1:p){
    pi <- pi + pi_ij(i,j)
  }
}
# theta
theta <- function(k,i,j){
  sum(((logret[,k] - mean(logret[,k])) ^ 2 - S[k,k]) * ((logret[,i] - mean(logret[,i]
)) * (logret[,j] - mean(logret[,j])) - S[i,j])) / n
}
# rho
rho <- 0
for (i in 1:p){
  rho <- rho + pi_ij(i,i)
  for (j in 1:p){
    if (j!=i){
      rho <- rho + sig_overline * (sqrt(S[j,j] / S[i,i]) * theta(i,i,j) + sqrt(S[i,i]
/ S[j,j]) * theta(j,i,j)) / 2
    }
  }
}
# kappa
kappa <- (pi - rho) / gamma
# delta
delta <- min(1, max(kappa / n, 0))
print('The new value of delta is:')
```

```
[1] "The new value of delta is:"
```

Hide

```
print(delta)
```

```
[1] 0.8997055
```

Hide

```
mu <- apply(logret, 2, mean)
# Sigma
Sigma2 <- delta * F2 + (1 - delta) * S
print('The new shrinkage estimate is:')
```

```
[1] "The new shrinkage estimate is:"
```

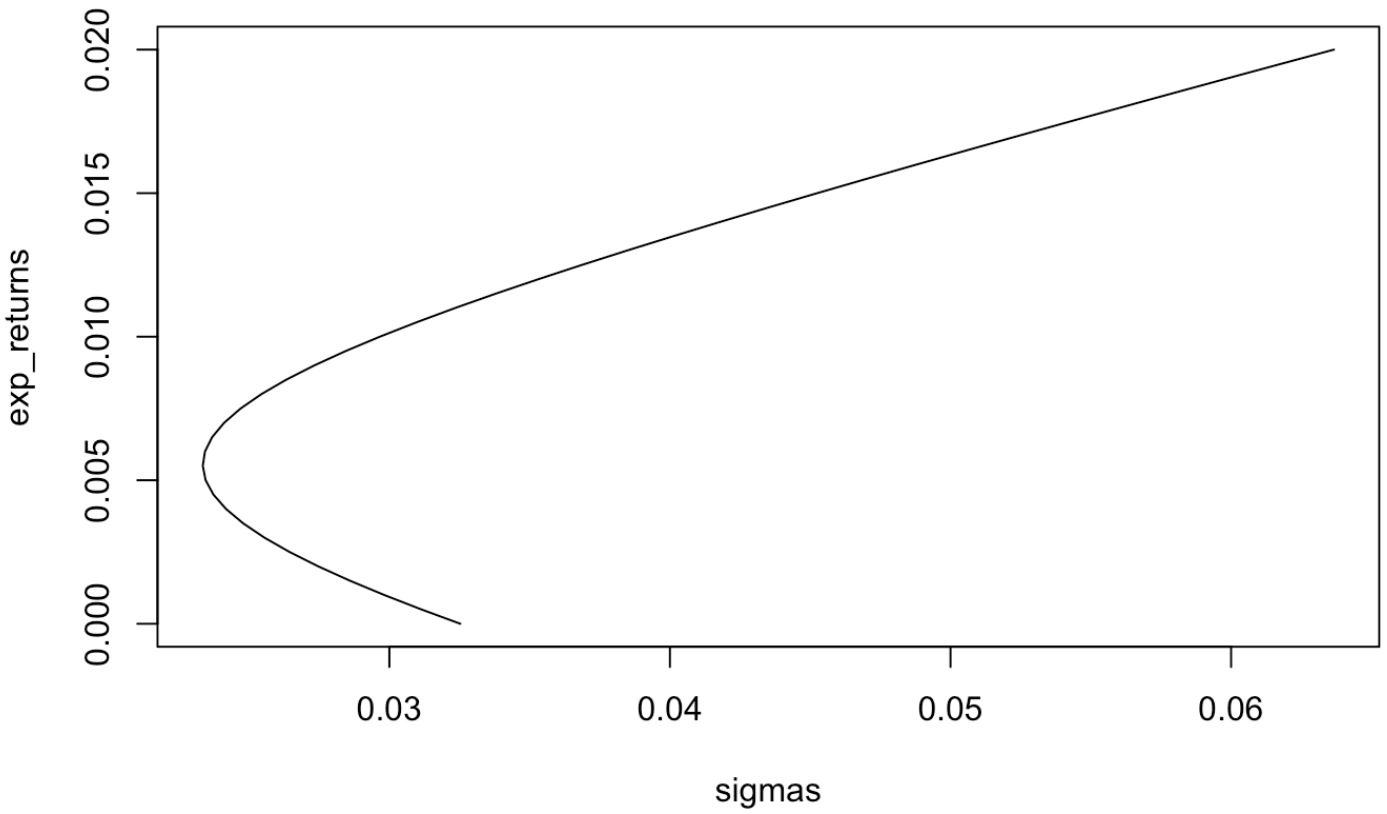
Hide

```
print(Sigma2)
```

```
              AAPL         ADBE          ADP          AMD         DELL          GTW
HP           IBM          MSFT         ORCL
 [1,] 0.0045463860 0.0013678680 0.0004487930 0.0025584283 0.0021114531 0.0024988866
5.287907e-04 0.0011987872  1.415793e-03 0.0015047033
 [2,] 0.0013678680 0.0044516841 0.0007277988 0.0020743887 0.0012069420 0.0016915220
1.324865e-03 0.0009190389  6.401116e-04 0.0011389087
 [3,] 0.0004487930 0.0007277988 0.0007883470 0.0007706495 0.0003876521 0.0005795675
6.160450e-04 0.0003680244  1.863785e-04 0.0004107252
 [4,] 0.0025584283 0.0020743887 0.0007706495 0.0071408360 0.0025483316 0.0031881811
1.080208e-03 0.0016173612  1.667115e-03 0.0019377724
 [5,] 0.0021114531 0.0012069420 0.0003876521 0.0025483316 0.0035229415 0.0026920824
2.570812e-04 0.0012480245  1.672391e-03 0.0015557292
 [6,] 0.0024988866 0.0016915220 0.0005795675 0.0031881811 0.0026920824 0.0065092461
5.464621e-04 0.0014881493  1.845688e-03 0.0018358009
 [7,] 0.0005287907 0.0013248654 0.0006160450 0.0010802076 0.0002570812 0.0005464621
2.303659e-03 0.0004755789 -7.197119e-05 0.0004801306
 [8,] 0.0011987872 0.0009190389 0.0003680244 0.0016173612 0.0012480245 0.0014881493
4.755789e-04 0.0014821202  8.508864e-04 0.0009363116
 [9,] 0.0014157932 0.0006401116 0.0001863785 0.0016671150 0.0016723912 0.0018456881 -
7.197119e-05 0.0008508864  2.017462e-03 0.0010651545
[10,] 0.0015047033 0.0011389087 0.0004107252 0.0019377724 0.0015557292 0.0018358009
4.801306e-04 0.0009363116  1.065155e-03 0.0038503400
```

Hide

```
exp_returns <- seq(0, 0.02, 0.0005)
weights <- sapply(exp_returns, function(x){eef(mean=mu, Cov=Sigma2, x)})
risk <- function(x){sqrt(x %*% Sigma2 %*% x)}
sigmas <- apply(weights, 2, risk)
# plotting the estimated efficient frontier to compare with that found in part (b)
plot(sigmas, exp_returns, type='l')
```



Hide

```
print('When comparing this estimated efficient frontier with that found in part (b) w
e see a reduction in variance.')
```

```
[1] "When comparing this estimated efficient frontier with that found in part (b) we
see a reduction in variance."
```