Basic Investment Model

Code

Casey Tirshfield

Let $\mathbf{y}_t$ be a $q \times 1$ vector of excess returns on q assets and let x_t be the excess return on the market portfolio (or, or more precisely, its proxy) at time t. THe capital asset pricing model can be associated with the null hypothesis $H_0 : \alpha = \mathbf{0}$ in the regression model $\mathbf{y}_t = \alpha + x_t\beta + \epsilon_t, \quad 1 \le t \le n$, where $\mathbb{E}[\epsilon_t] = \mathbf{0}$, $\mathrm{Cov}[\epsilon_t] = \mathbf{V}$, and $\mathbb{E}[x_t\epsilon_t] = 0$. Note that the regression model above is a multivariate representation of q regression models of the form $y_{tj} = \alpha_j + x_t\beta_j + \epsilon_{tj}, \quad j = 1, \dots, q$.

The file `m_sp500ret_3mtcm.txt` contains three columns. The second column gives the monthly returns of the S&P 500 index from January 1994 to December 2006. The third column gives the monthly rates of the 3-month U.S. Treasury bill in the secondary market, which is obtained from the Federal Reserve Bank of St. Louis and used as the risk-free asset here. Consider the ten monthly returns in the file `m_ret_10stocks.txt`.

Hide

```
# data frames
df <- read.table('m_sp500ret_3mtcm.txt', skip=1, header=TRUE)
logret <- read.table('m_logret_10stocks.txt', header=TRUE)
# removes date column for future matrix manipulation
logret <- logret[,-1]
# risk free rate column
rfr <- df[,3] / (12 * 100)
# excessive returns matrix
exc_logret <- apply(logret, 2, function(x){x - rfr})
# excessive return column for the S&P 500
exc_sp <- df[,2] - rfr
```

(a) Fit CAPM to the ten stocks. Give point estimates and 95% confidence intervals of α, β, the Sharpe index, and the Trynor index.

Hide

```
model <- lm(exc_logret ~ exc_sp)
alpha <- coef(model)[1,]
beta <- coef(model)[2,]
# formatting for printing to consol
alphabeta <- matrix(c(alpha, beta), nrow=10, ncol=2)
rownames(alphabeta) <- c(names(logret))
colnames(alphabeta) <- c('alpha', 'beta')
print('The alphas and betas are:')
```

```
[1] "The alphas and betas are:"
```

Hide

```
print(alphabeta)
```

```
             alpha      beta
AAPL  0.0038473784 1.3846398
ADBE  0.0046581721 1.5313505
ADP   0.0008070075 0.8476769
AMD  -0.0006186328 2.3238266
DELL  0.0088838584 1.6749919
GTW  -0.0054253352 2.2328015
HP    0.0019004141 0.8752343
IBM   0.0026256227 1.3479235
MSFT  0.0042555493 1.4585386
ORCL  0.0039104996 1.5676042
```

Hide

```
# n is the number of observations
n <- length(exc_sp)
# we use the methods in section 3.3.3 of Tze Leung Lai and Haipeng Xing's book, "Stat
istical Models and Methods for Financial Markets"
residual <- exc_logret - alpha - rep(1, n) %*% t(alpha) - exc_sp %*% t(beta)
vHat <- t(residual) %*% residual / n
# here we take the sqrt of the var from (3.27) of Tze Leung Lai and Haipeng Xing's bo
ok, "Statistical Models and Methods for Financial Markets" to find the standard devia
tion that we use to construct the confidence intervals
sd_beta <- sqrt(diag(vHat) / sum((exc_sp - mean(exc_sp)) ^ 2))
sd_alpha <- sqrt(diag(vHat) * (1 / n + mean(exc_sp) ^ 2 / sum((exc_sp - mean(exc_sp))
^ 2)))
# the 95% CI of alpha is
lb_alpha <- alpha - 1.96 * sd_alpha
ub_alpha <- alpha + 1.96 * sd_alpha
# formatting for printing to consol
CI_alpha <- matrix(c(lb_alpha, ub_alpha), nrow=10, ncol=2)
rownames(CI_alpha) <- c(names(logret))
colnames(CI_alpha) <- c('lower bound', 'upper bound')
print('The 95% confidence intervals of alpha are:')
```

```
[1] "The 95% confidence intervals of alpha are:"
```

Hide

```
print(CI_alpha)
```

```
       lower bound upper bound
AAPL -0.0059879092 0.013682666
ADBE -0.0049195741 0.014235918
ADP  -0.0029430113 0.004557026
AMD  -0.0121766424 0.010939377
DELL  0.0008700769 0.016897640
GTW  -0.0163342912 0.005483621
HP   -0.0052759884 0.009076817
IBM  -0.0020682283 0.007319474
MSFT -0.0014872912 0.009998390
ORCL -0.0047824341 0.012603433
```

Hide

```
# the 95% CI of beta is
lb_beta <- beta - 1.96 * sd_beta
ub_beta <- beta + 1.96 * sd_beta
# formatting for printing to consol
CI_beta <- matrix(c(lb_beta, ub_beta), nrow=10, ncol=2)
rownames(CI_beta) <- c(names(logret))
colnames(CI_beta) <- c('lower bound', 'upper bound')
print('The 95% confidence intervals of beta are:')
```

```
[1] "The 95% confidence intervals of beta are:"
```

Hide

```
print(CI_beta)
```

```
     lower bound upper bound
AAPL   0.8339981    1.935281
ADBE   0.9951276    2.067573
ADP    0.6377271    1.057627
AMD    1.6767361    2.970917
DELL   1.2263296    2.123654
GTW    1.6220491    2.843554
HP     0.4734538    1.277015
IBM    1.0851320    1.610715
MSFT   1.1370181    1.780059
ORCL   1.0809187    2.054290
```

Hide

```
# the Sharpe ratio is
mu <- apply(exc_logret, 2, mean)
sig <- apply(exc_logret, 2, sd)
sharpe <- mu / sig
# formatting for printing to console
sharperatio <- matrix(c(sharpe), nrow=10, ncol=1)
rownames(sharperatio) <- c(names(logret))
colnames(sharperatio) <- c('Sharpe ratio')
print('The Sharpe ratios are')
```

```
[1] "The Sharpe ratios are"
```

Hide

```
print(sharperatio)
```

```
     Sharpe ratio
AAPL   0.05428406
ADBE   0.06685281
ADP    0.02503971
AMD   -0.01081325
DELL   0.14627308
GTW   -0.07080820
HP     0.03725599
IBM    0.06390360
MSFT   0.09073364
ORCL   0.05985554
```

Hide

```
# the 95% CI of the Sharpe ratio is
sd_sharpe <- sqrt(1 / n + mu ^ 2 / (2 * sig ^ 2 * n))
lb_sharpe <- sharpe - 1.96 * sd_sharpe
ub_sharpe <- sharpe + 1.96 * sd_sharpe
# formatting for printing to consol
CI_sharpe <- matrix(c(lb_sharpe, ub_sharpe), nrow=10, ncol=2)
rownames(CI_sharpe) <- c(names(logret))
colnames(CI_sharpe) <- c('lower bound', 'upper bound')
print('The 95% confidence intervals of the Sharpe ratio are:')
```

```
[1] "The 95% confidence intervals of the Sharpe ratio are:"
```

Hide

```
print(CI_sharpe)
```

```
     lower bound upper bound
AAPL -0.10275710  0.21132521
ADBE -0.09024802  0.22395364
ADP  -0.13191047  0.18198990
AMD  -0.16774343  0.14611693
DELL -0.01148966  0.30403583
GTW  -0.22793037  0.08631396
HP   -0.11972404  0.19423603
IBM  -0.09318212  0.22098932
MSFT -0.06651460  0.24798187
ORCL -0.09721054  0.21692162
```

Hide

```
# the Treynor ratio is
treynor <- mu / beta
# formatting for printing to console
trynorratio <- matrix(c(treynor), nrow=10, ncol=1)
rownames(trynorratio) <- c(names(logret))
colnames(trynorratio) <- c('Treynor ratio')
print('The Treynor ratios are')
```

```
[1] "The Treynor ratios are"
```

Hide

```
print(trynorratio)
```

```
     Treynor ratio
AAPL  0.0026512394
ADBE  0.0029144981
ADP   0.0008246488
AMD  -0.0003935868
DELL  0.0051764480
GTW  -0.0025572070
HP    0.0020439466
IBM   0.0018205281
MSFT  0.0027903063
ORCL  0.0023671970
```

Hide

```
# the 95% CI of the Treynor ratio is
sd_treynor <- sqrt((1 / beta ^ 2) * (sig ^ 2 / n) + (mu / beta ^ 2) ^ 2 * sd_beta ^ 2
)
lb_treynor <- treynor - 1.96 * sd_treynor
ub_treynor <- treynor + 1.96 * sd_treynor
# formatting for printing to consol
CI_treynor <- matrix(c(lb_treynor, ub_treynor), nrow=10, ncol=2)
rownames(CI_treynor) <- c(names(logret))
colnames(CI_treynor) <- c('lower bound', 'upper bound')
print('The 95% confidence intervals of the Treynor ratio are:')
```

```
[1] "The 95% confidence intervals of the Treynor ratio are:"
```

Hide

```
print(CI_treynor)
```

```
       lower bound upper bound
AAPL -0.0050852047 0.010387684
ADBE -0.0040024912 0.009831488
ADP  -0.0043475159 0.005996813
AMD  -0.0061065038 0.005319330
DELL -0.0005474598 0.010900356
GTW  -0.0082675098 0.003153096
HP   -0.0066163191 0.010704212
IBM  -0.0026641394 0.006305196
MSFT -0.0020746242 0.007655237
ORCL -0.0038823384 0.008616732
```

(b) Use the bootstrap procedure in Section 3.5 to estimate the standard error of the point estimates of α, β, and the Sharpe and Treynor indices.

Hide

```
# set number of bootstrap samples from page 87 section 3.5 of Tze Leung Lai and Haipe
ng Xing's book, "Statistical Models and Methods for Financial Markets"
B <- 500
alpha_boot <- matrix(0, B, 10)
beta_boot <- matrix(0, B, 10)
sharpe_boot <- matrix(0, B, 10)
treynor_boot <- matrix(0, B, 10)
for (i in 1:B){
  index <- sample(1:n, n, replace=TRUE)
  logret_boot <- exc_logret[index,]
  sp_boot <- exc_sp[index]
  model <- lm(logret_boot~sp_boot)
  alpha_boot[i,] <- coef(model)[1,]
  beta_boot[i,] <- coef(model)[2,]
  mu <- apply(logret_boot, 2, mean)
  sig <- apply(logret_boot, 2, sd)
  sharpe_boot[i,] <- mu / sig
  treynor_boot[i,] <- mu / beta_boot[i,]
}
sd_alpha_boot <- apply(alpha_boot, 2, sd)
sd_beta_boot <- apply(beta_boot, 2, sd)
sd_sharpe_boot <- apply(sharpe_boot, 2, sd)
sd_treynor_boot <- apply(treynor_boot, 2, sd)
mean_alpha_boot <- apply(alpha_boot, 2, mean)
mean_beta_boot <- apply(beta_boot, 2, mean)
mean_sharpe_boot <- apply(sharpe_boot, 2, mean)
mean_treynor_boot <- apply(treynor_boot, 2, mean)
# formatting for printing to consol
se <- matrix(c(sd_alpha_boot, sd_beta_boot, sd_sharpe_boot, sd_treynor_boot), nrow=10
, ncol=4)
rownames(se) <- c(names(logret))
colnames(se) <- c('alpha', 'beta', 'Sharpe', 'Treynor')
print('The standard errors are:')
```

```
[1] "The standard errors are:"
```

Hide

```
print(se)
```

```
           alpha      beta     Sharpe     Treynor
AAPL 0.005016156 0.3220267 0.08134045 0.004390568
ADBE 0.004754905 0.2611886 0.08062952 0.003823854
ADP  0.001890870 0.1379770 0.08091315 0.002737713
AMD  0.005831330 0.3484639 0.08016010 0.002959834
DELL 0.003958992 0.2690862 0.07692525 0.002876627
GTW  0.005529731 0.4445039 0.07523202 0.002821115
HP   0.003607360 0.1824784 0.07869802 0.004752505
IBM  0.002389007 0.1524500 0.07917591 0.002241406
MSFT 0.003006937 0.1663030 0.08247464 0.002471236
ORCL 0.004434156 0.2780629 0.07887088 0.003521192
```

(c) Test for each stock the null hypothesis $\alpha = 0$.

Hide

```
# we employ the t-test
t_test <- alpha / sd_alpha
print(t_test)
```

```
      AAPL       ADBE        ADP        AMD       DELL 
 0.7667149  0.9532532  0.4217938 -0.1049074  2.1728022 
       GTW         HP        IBM       MSFT       ORCL 
-0.9747639  0.5190361  1.0963749  1.4523957  0.8817023 
```

Hide

```
# the t-value for a 95% confidence interval is 2.262
print('Since each stock has a t-value of less than 2.262, we fail to reject the null
hypothesis alpha=0 at the 5% significance level.')
```

```
[1] "Since each stock has a t-value of less than 2.262, we fail to reject the null hy
pothesis alpha=0 at the 5% significance level."
```

(d) Use the regression model (1) to test for the ten stocks the null hypothesis $\alpha = 0$.

Hide

```
model <- lm(exc_logret~exc_sp)
alpha <- coef(model)[1,]
beta <- coef(model)[2,]
n <- dim(exc_logret)[1]
m <- dim(exc_logret)[2]
residual <- exc_logret- alpha - rep(1, n) %*% t(alpha) - exc_sp %*% t(beta)
vHat <- t(residual) %*% residual / n
# we employ (3.28) from Tze Leung Lai and Haipeng Xing's book, "Statistical Models an
d Methods for Financial Markets"
fVal <- ((n - m - 1) / m) * alpha %*% solve(vHat) %*% alpha / (1 + mean(exc_sp) ^ 2 /
mean((exc_sp - mean(exc_sp)) ^ 2))
# we set the bounds
lb <- qf(0.025, m, n - m - 1)
ub <- qf(0.975, m, n - m - 1)
if (lb < fVal & fVal < ub){
  print('Since our F value is between our lower and upper bounds we cannot reject the
null hypothesis at the 95% significance level')
} else {
  print('Since our F value is outside of our lower and upper bounds we can reject the
null hypothesis at the 95% significance level')
}
```

```
[1] "Since our F value is between our lower and upper bounds we cannot reject the nul
l hypothesis at the 95% significance level"
```

(e) Perform a factor analysis on the excess returns of the ten stocks. Show the factor loadings and rotated factor loadings. Explain your choice of the number of fators.

Hide

```
corPCA <- princomp(exc_logret, cor=TRUE)
corPCA$loadings
```

```
Loadings:
     Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
AAPL -0.335         0.484  0.164  0.150        -0.362  0.641
ADBE -0.267  0.342  0.239 -0.647        -0.160  0.492  0.202
ADP  -0.235  0.417 -0.607 -0.222 -0.188  0.187 -0.386  0.263
AMD  -0.352         0.228  0.143         0.673  0.301 -0.221
DELL -0.391 -0.263               -0.178 -0.399              
GTW  -0.349 -0.130  0.193 -0.237 -0.461        -0.387 -0.461
HP   -0.181  0.677  0.140  0.473        -0.402        -0.280
IBM  -0.366        -0.329  0.378         0.221  0.246  0.160
MSFT -0.344 -0.392 -0.333               -0.326  0.339       
ORCL -0.279        -0.104 -0.234  0.827        -0.235 -0.337
     Comp.9 Comp.10
AAPL        -0.229 
ADBE  0.129  0.103 
ADP  -0.245        
AMD  -0.449        
DELL -0.439  0.617 
GTW   0.418        
HP          -0.161 
IBM   0.589  0.354 
MSFT        -0.618 
ORCL               

               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
SS loadings       1.0    1.0    1.0    1.0    1.0    1.0    1.0
Proportion Var    0.1    0.1    0.1    0.1    0.1    0.1    0.1
Cumulative Var    0.1    0.2    0.3    0.4    0.5    0.6    0.7
               Comp.8 Comp.9 Comp.10
SS loadings       1.0    1.0     1.0
Proportion Var    0.1    0.1     0.1
Cumulative Var    0.8    0.9     1.0
```

Hide

```
summary(corPCA)
```

```
Importance of components:
                          Comp.1    Comp.2     Comp.3
Standard deviation     1.9660670 1.0480926 0.99867855
Proportion of Variance 0.3865419 0.1098498 0.09973588
Cumulative Proportion  0.3865419 0.4963918 0.59612764
                           Comp.4     Comp.5     Comp.6
Standard deviation     0.93735990 0.90054737 0.81841133
Proportion of Variance 0.08786436 0.08109856 0.06697971
Cumulative Proportion  0.68399199 0.76509055 0.83207026
                           Comp.7     Comp.8     Comp.9
Standard deviation     0.71973625 0.69165215 0.59919459
Proportion of Variance 0.05180203 0.04783827 0.03590342
Cumulative Proportion  0.88387229 0.93171056 0.96761397
                          Comp.10
Standard deviation     0.56908723
Proportion of Variance 0.03238603
Cumulative Proportion  1.00000000
```

Hide

```
print('We choose 8 factors because the rule of thumb is that we want greater than 90%
of the cumulative proportion of variance to be explained.')
```

```
[1] "We choose 8 factors because the rule of thumb is that we want greater than 90% o
f the cumulative proportion of variance to be explained."
```

(f) Consider the model $r_t^e = \beta_1 \mathbf{1}_{\{t<t_0\}} r_M^e + \beta_2 \mathbf{1}_{\{t \geq t_0\}} r_M^e + \epsilon_t$, in which $r_t^e = r_t - r_f$ and $r_M^e = r_M - r_f$ are the excess returns of the stock and the S&P 500 index. The model suggests that the β in the CAPM might not be a constant (i.e., $\beta_1 \neq \beta_2$). Taking February 2001 as the month t_0, test for each stock the null hypothesis that $\beta_1 = \beta_2$.

Hide

```
tnot <- which(df[,1] == 'Feb-01')
exc_sp_post_tnot <- exc_sp
exc_sp_post_tnot[1:tnot - 1] <- 0
model <- lm(exc_logret ~ exc_sp + exc_sp_post_tnot - 1)
summary(model)
```

```
Response AAPL :

Call:
```

```
lm(formula = AAPL ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.33742 -0.03108  0.00414  0.04469  0.14286 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)    
exc_sp             1.4388     0.3802   3.784  0.00022 ***
exc_sp_post_tnot  -0.1254     0.5703  -0.220  0.82629    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.06322 on 154 degrees of freedom
Multiple R-squared:  0.1342,	Adjusted R-squared:  0.1229 
F-statistic: 11.93 on 2 and 154 DF,  p-value: 1.52e-05


Response ADBE :

Call:
lm(formula = ADBE ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.275594 -0.023959  0.002376  0.042462  0.279293 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)   
exc_sp             1.1956     0.3661   3.266  0.00135 **
exc_sp_post_tnot   0.7513     0.5491   1.368  0.17323   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.06087 on 154 degrees of freedom
Multiple R-squared:  0.1778,	Adjusted R-squared:  0.1671 
F-statistic: 16.65 on 2 and 154 DF,  p-value: 2.847e-07


Response ADP :

Call:
lm(formula = ADP ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.041287 -0.016113  0.000948  0.016292  0.064983 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
```

```
exc_sp              0.6818     0.1400   4.871 2.74e-06 ***
exc_sp_post_tnot   0.3724     0.2100   1.773   0.0781 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02328 on 154 degrees of freedom
Multiple R-squared:  0.3097,	Adjusted R-squared:  0.3008
F-statistic: 34.55 on 2 and 154 DF,  p-value: 4.025e-13


Response AMD :

Call:
lm(formula = AMD ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
      Min        1Q    Median        3Q       Max
-0.177953 -0.050162 -0.003217  0.043977  0.229415

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
exc_sp             1.6230     0.4361   3.722 0.000277 ***
exc_sp_post_tnot   1.5773     0.6541   2.411 0.017064 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.07251 on 154 degrees of freedom
Multiple R-squared:   0.27,	Adjusted R-squared:  0.2605
F-statistic: 28.47 on 2 and 154 DF,  p-value: 3.007e-11


Response DELL :

Call:
lm(formula = DELL ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
      Min        1Q    Median        3Q       Max
-0.123991 -0.025591  0.002821  0.041618  0.148689

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
exc_sp             2.1385     0.3079   6.945 9.99e-11 ***
exc_sp_post_tnot  -1.0508     0.4619  -2.275   0.0243 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.0512 on 154 degrees of freedom
Multiple R-squared:  0.2743,	Adjusted R-squared:  0.2649
F-statistic: 29.11 on 2 and 154 DF,  p-value: 1.896e-11
```

```
Response GTW :

Call:
lm(formula = GTW ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.36640 -0.04146 -0.00537  0.04138  0.15081 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)    
exc_sp             1.7572     0.4197   4.186 4.75e-05 ***
exc_sp_post_tnot   1.0749     0.6296   1.707   0.0898 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.06979 on 154 degrees of freedom
Multiple R-squared:  0.2594,	Adjusted R-squared:  0.2498 
F-statistic: 26.98 on 2 and 154 DF,  p-value: 9.037e-11


Response HP :

Call:
lm(formula = HP ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.159084 -0.029521  0.003743  0.031688  0.165366 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)   
exc_sp            0.84615    0.27405   3.088  0.00239 **
exc_sp_post_tnot  0.06372    0.41106   0.155  0.87702   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.04557 on 154 degrees of freedom
Multiple R-squared:  0.1065,	Adjusted R-squared:  0.09488 
F-statistic: 9.177 on 2 and 154 DF,  p-value: 0.0001717


Response IBM :

Call:
lm(formula = IBM ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
```

```
      Min        1Q    Median        3Q       Max 
-0.121136 -0.015113  0.001287  0.020130  0.102954 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)    
exc_sp             1.1953     0.1799   6.644 4.95e-10 ***
exc_sp_post_tnot   0.3410     0.2698   1.264    0.208    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02991 on 154 degrees of freedom
Multiple R-squared:  0.3996,	Adjusted R-squared:  0.3918 
F-statistic: 51.25 on 2 and 154 DF,  p-value: < 2.2e-16


Response MSFT :

Call:
lm(formula = MSFT ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.156145 -0.017064  0.005438  0.020023  0.126153 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)    
exc_sp             1.7096     0.2193   7.795 8.99e-13 ***
exc_sp_post_tnot  -0.5688     0.3290  -1.729   0.0858 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.03647 on 154 degrees of freedom
Multiple R-squared:  0.3486,	Adjusted R-squared:  0.3401 
F-statistic:  41.2 on 2 and 154 DF,  p-value: 4.645e-15


Response ORCL :

Call:
lm(formula = ORCL ~ exc_sp + exc_sp_post_tnot - 1)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.181931 -0.022507 -0.001783  0.031607  0.188443 

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)    
exc_sp           1.5656035  0.3343057   4.683 6.16e-06 ***
exc_sp_post_tnot 0.0009885  0.5014332   0.002    0.998    
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.05559 on 154 degrees of freedom
Multiple R-squared:  0.2041,	Adjusted R-squared:  0.1938 
F-statistic: 19.75 on 2 and 154 DF,  p-value: 2.313e-08
```

Hide

```
print('We observe that the null hypothesis that beta_1=beta_2 is rejected at the 95% significance level for AMD and DELL.')
```

```
[1] "We observe that the null hypothesis that beta_1=beta_2 is rejected at the 95% significance level for AMD and DELL."
```

(g) Estimate t_0 in (f) by the least squares criterion that minimizes the residual sum of squares over (β_1, β_2, t_0).

Hide

```
RSS <- rep(0, n)
for (tnot in 1:n){
  exc_sp_post_tnot <- exc_sp
  if (tnot > 1) {
    exc_sp_post_tnot[1:tnot - 1] <- 0
  }
  model <- lm(exc_logret ~ exc_sp + exc_sp_post_tnot - 1)
  RSS[tnot] <- sum(resid(model) ^ 2)
}
min_tnot <- which(RSS == min(RSS))
cat('The t0 that minimizes the RSS is: t0 =', min_tnot, 'which corresponds to', as.vector(df[min_tnot, 1]))
```

```
The t0 that minimizes the RSS is: t0 = 77 which corresponds to May-00
```