Lab 8. Testing CAPM

1

First, download the data into R and start preprocessing: combining the adjusted column data into a matrix

```
> close <- cbind(sp500[1:180,7], ford[1:180,7], ge[1:180,7], jnj[1:180,7])
> ## last 180 months for sp500 and 3 stocks
> dimnames(close) <- list(NULL, c("sp500", "ford", "ge", "jnj"))
> ## give the names to its column
> close <- apply(close, 2, rev)   ## reverse the time order for each column
> close <- log(close)      # log prices and reverse the order
>
> plot(as.ts(close))
> title("Monthly closing prices", "(a)")
>
> returns <- apply(close, 2, diff)   # take the difference for each column
> plot(as.ts(returns))
> title("Returns of the stocks", "(b)")
>
> returns <- data.frame(returns)
> ## change the object "returns" from a matrix to a data frame
>
> irx <- read.csv("IRX_raw.csv")
> tb3m <- rev( irx[1:179,7] )    ### last 179 months yields of 3m T-bill
>
> head(tb3m)
[1] 5.01 5.04 5.03 5.18 5.14 4.91
> head(returns)
         sp500        ford          ge          jnj
1  0.013342046  0.05210692 -0.00790518  0.002899393
2  0.022596160  0.01750592  0.06899287  0.055194188
3  0.002254153 -0.11968289  0.04752961  0.016304709
4 -0.046827521  0.01215082 -0.04858837 -0.035669076
5  0.018639176  0.03442476  0.01053751  0.034590327
6  0.052785302 -0.07007697  0.09588177  0.039677249
>
```

```
> tb3m <- tb3m/12   # convert to monthly return
> returns <- returns* 100   # convert to percentage
> apply(returns, 2, mean)
    sp500      ford        ge       jnj
0.4092980 0.3387415 0.4918052 0.7076000
>
> returns <- returns - tb3m ## excessive returns
> apply(returns, 2, mean)
     sp500       ford         ge        jnj
0.15745909 0.08690253 0.23996627 0.45576109
>
> rm(sp500, ford, ge, jnj)   ## remove unnessary objects from R
```

2

After computing the excess returns above, we are ready to run the least-squares regression. We first run separately and get its residuals.

```
> y <- returns[60:179, 2:4]   # last months e-return
> x <- returns[60:179, 1]
> ls.print( lsfit(x, y[,1]) )   # Ford
Residual Standard Error=13.3438
R-Square=0.3574
F-statistic (df=1, 118)=65.6298
p-value=0

          Estimate Std.Err t-value Pr(>|t|)
Intercept  -0.2602  1.2184 -0.2136   0.8312
X           2.0904  0.2580  8.1012   0.0000


> ls.print( lsfit(x, y[,2]) )   # GE
Residual Standard Error=5.9309
R-Square=0.5532
F-statistic (df=1, 118)=146.0749
p-value=0

          Estimate Std.Err t-value Pr(>|t|)
Intercept  -0.4157  0.5415 -0.7675   0.4443
X           1.3861  0.1147 12.0861   0.0000


> ls.print( lsfit(x, y[,3]) )   # JnJ
Residual Standard Error=3.8116
```

```
R-Square=0.2746
F-statistic (df=1, 118)=44.666
p-value=0

          Estimate Std.Err t-value Pr(>|t|)
Intercept   0.2591  0.3480  0.7444   0.4581
X           0.4926  0.0737  6.6833   0.0000


>
> residuals <- resid( lsfit(x, y))
> head(residuals)
             Y1         Y2         Y3
[1,]  15.648866 -0.3756076 -7.8289530
[2,]  -9.114765  5.0365355  5.7165445
[3,] -19.610913  0.7989647  0.1941740
[4,]   6.676167  4.0446293  3.8564168
[5,]   6.510850 -9.5513191  8.0932196
[6,]  -9.186307  3.5221844  0.5798894
> Sigma <- t(residuals) %*% residuals / 120
> alpha <- c( -0.2602, -0.4157, 0.2591 )
> mreturn <- mean(x)
> msigma <- var(x)
> T0 <- 120/( 1+mreturn^2/msigma ) * t(alpha) %*% solve(Sigma, alpha)
> 1 - pchisq(T0, 3)
          [,1]
[1,] 0.7232129
> T1 <- (120-3-1)/3/120*T0
> 1 - pf(T1, 3, 120-3-1)
          [,1]
[1,] 0.7340664
```