Assignment2

Manish Kumar

Jan 25, 2023

Create efficient frontier, CAPM model and APT model for a group of stocks representing health care sector and industrial sector.

The names of the selected companies are in the file `Industrials_Health_Names.csv`.

The period of observation is `from="2014-7-1", to="2015-7-1"`.

For the sector indices use SPDR XLV (health care sector) and XLI (industrial sector).

For the broad market index use SPY.

For the risk-free rate use Fed Funds effective rate.

Note that it may not be possible to find interpretation of PCA factors in terms of real assets or indices.

In such cases it is possible to use PCA factors without interpretation.

```
suppressMessages(library(quantmod))
suppressMessages(library(dplyr))

# Load the csv files
SP500.Industrials.Health = read.csv(file="Industrials_Health_Names.csv", header=F)

# Remove UTX since the company has gone private and no history exists
SP500.Industrials.Health = SP500.Industrials.Health %>% filter(V1 != "UTX")

SP500.Industrials.Health.names = as.character(SP500.Industrials.Health[, 1])
FedFunds.Effective.Rate = read.csv(file="RIFSPFF_NB.csv")
FedFunds.Effective.Rate = FedFunds.Effective.Rate[15094:15345,]
```

```
# Download historical data for companies
getSymbols(SP500.Industrials.Health.names, from="2014-7-1", to="2015-7-1")
```

```
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
## pausing 1 second between requests for more than 5 symbols
```

```
##  [1] "CAT" "FDX" "GE"  "HON" "LMT" "NOC" "UNP" "UPS" "WM"  "ABT" "AET" "HUM"
## [13] "JNJ" "MDT" "PFE"
```

```
# Download historical data for sector indices
getSymbols("XLV", from="2014-7-1", to="2015-7-1")
```

```
## [1] "XLV"
```

```
getSymbols("XLI", from="2014-7-1", to="2015-7-1")
```

```
## [1] "XLI"
```

```
# Download historical data for broad market index
getSymbols("SPY", from="2014-7-1", to="2015-7-1")
```

```
## [1] "SPY"
```

1. Efficient Frontier

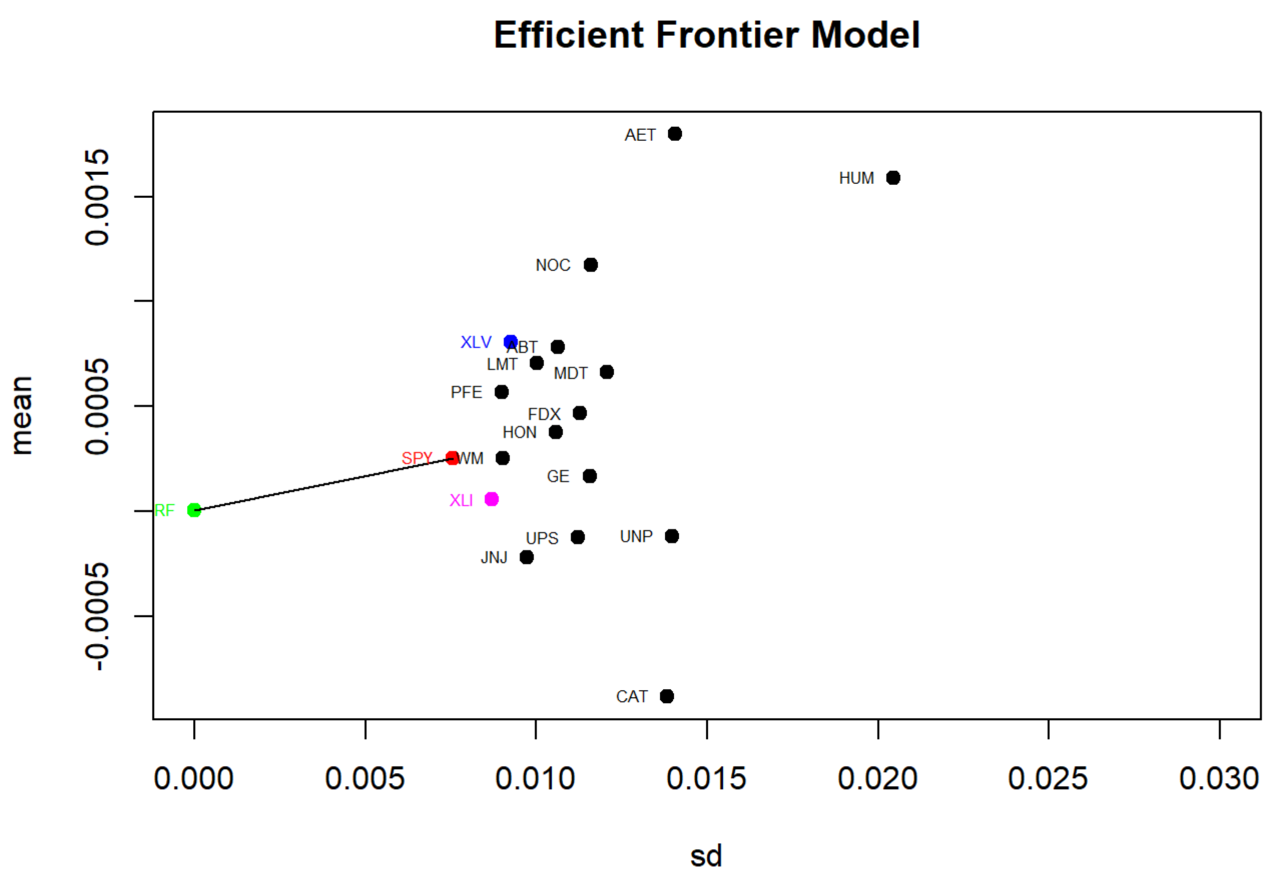
Calculating Mean and SD for each stock, sector and SPY

```
# Calculate mean and standard deviation for each stock
Mean.Sd.SP500.companies = cbind(
  sd=sapply(SP500.Industrials.Health.names, function(z) sd(ROC(Ad(get(z))), na.rm=T)),
  mean=sapply(SP500.Industrials.Health.names, function(z) mean(ROC(Ad(get(z))), na.rm=T))
)

head(Mean.Sd.SP500.companies)
```

```
##             sd          mean
## CAT 0.01383432 -0.0008847112
## FDX 0.01128369  0.0004650234
## GE  0.01156477  0.0001666116
## HON 0.01059194  0.0003788085
## LMT 0.01001458  0.0007044108
## NOC 0.01159261  0.0011759220
```

```r
# Calculate the mean and standard deviation for XLV, XLI, and SPY
Mean.Sd.XLV = c(sd(ROC(Ad(XLV)), na.rm=T), mean(ROC(Ad(XLV)), na.rm=T))
Mean.Sd.XLI = c(sd(ROC(Ad(XLI)), na.rm=T), mean(ROC(Ad(XLI)), na.rm=T))
Mean.Sd.SPY = c(sd(ROC(Ad(SPY)), na.rm=T), mean(ROC(Ad(SPY)), na.rm=T))
Mean.FedFunds = mean(FedFunds.Effective.Rate[,2])/100/360
```

```r
# Plotting the SPY companies on standard deviation-mean diagram
plot(Mean.Sd.SP500.companies,
     ylab="mean",
     xlab="sd",
     main="Efficient Frontier Model", pch=19, xlim=c(0,.03))

# Adding the points for SPY and risk-free rate
points(Mean.Sd.SPY[1], Mean.Sd.SPY[2], col="red", pch=19)
points(Mean.Sd.XLV[1],Mean.Sd.XLV[2],col="blue", pch=19)
points(Mean.Sd.XLI[1],Mean.Sd.XLI[2],col="magenta", pch=19)
points(0, Mean.FedFunds, col="green", pch=19)

# Adding the line connecting the points for risk-free rate and SPY
lines(c(0, Mean.Sd.SPY[1]), c(Mean.FedFunds, Mean.Sd.SPY[2]))

# Putting labels with the company names on the graph
text(Mean.Sd.SP500.companies, labels=rownames(Mean.Sd.SP500.companies), cex=.5, pos=2)
text(Mean.Sd.SPY[1], Mean.Sd.SPY[2], labels="SPY", cex=.5, col="red", pos=2)
text(Mean.Sd.XLV[1], Mean.Sd.XLV[2], labels="XLV", cex=.5, col="blue", pos=2)
text(Mean.Sd.XLI[1], Mean.Sd.XLI[2], labels="XLI", cex=.5, col="magenta", pos=2)
text(0, Mean.FedFunds, labels="FF.RF", cex=.5, col="green", pos=2)
```



XLV did better than SPY while XLI did worse than the same. Stocks above the security line would be a better choice for a portfolio.

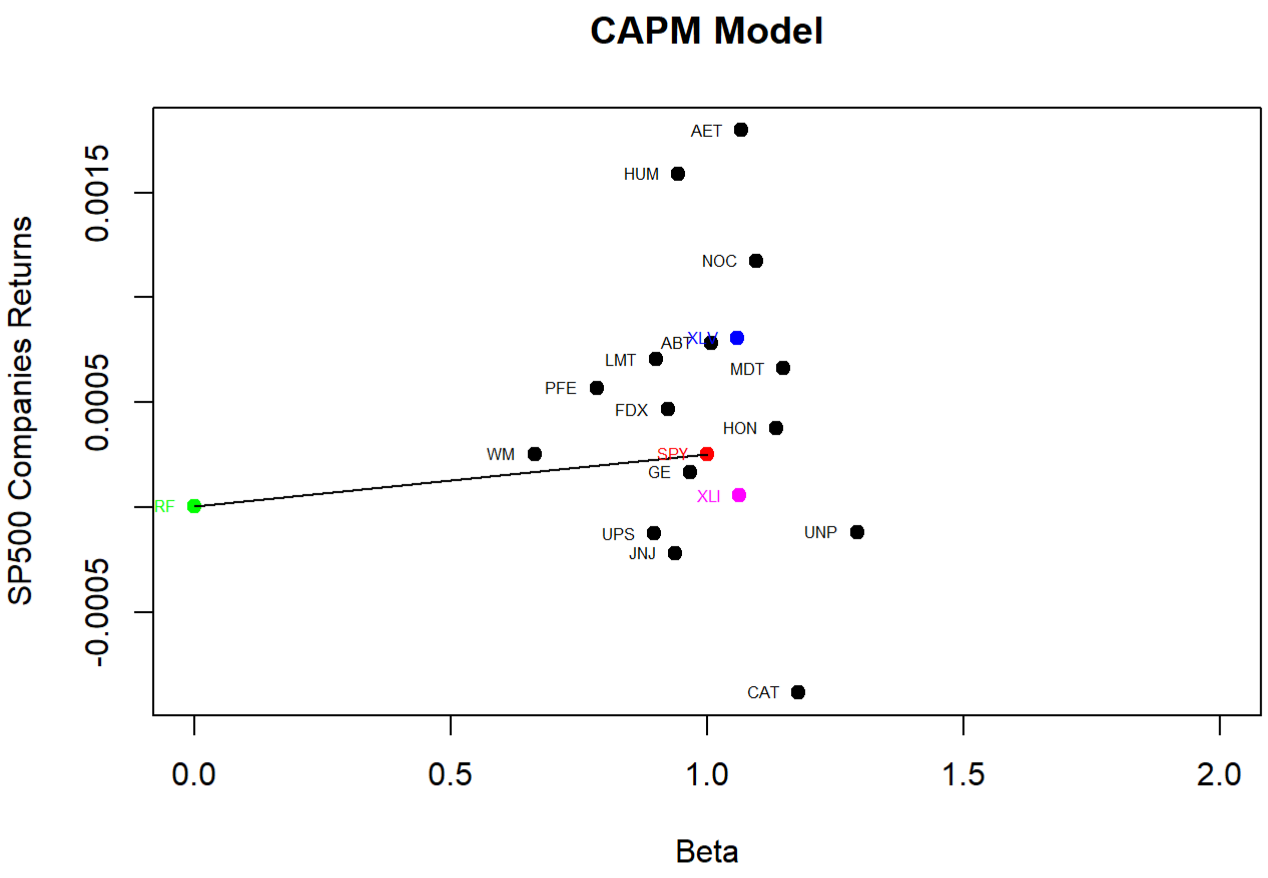
2. CAPM model

```
# Calculating the betas of all the companies on the list to SPY
SP500.companies.sector.betas = as.matrix(
  sapply(c(SP500.Industrials.Health.names, "XLV", "XLI"),
         function(z) lm(I(
           ROC(Ad(get(z))) - Mean.FedFunds
        ) ~-1 + I(ROC(Ad(SPY)) - Mean.FedFunds))$coefficients)
)

head(SP500.companies.sector.betas)
```

```
##                                          [,1]
## CAT.I(ROC(Ad(SPY)) - Mean.FedFunds) 1.1784133
## FDX.I(ROC(Ad(SPY)) - Mean.FedFunds) 0.9233727
## GE.I(ROC(Ad(SPY)) - Mean.FedFunds)  0.9666459
## HON.I(ROC(Ad(SPY)) - Mean.FedFunds) 1.1353500
## LMT.I(ROC(Ad(SPY)) - Mean.FedFunds) 0.8999203
## NOC.I(ROC(Ad(SPY)) - Mean.FedFunds) 1.0962287
```

```
# Given Row Names for betas without SPY
rownames(SP500.companies.sector.betas) = c(SP500.Industrials.Health.names, "XLV", "XLI")

# Combine Company and Sector Means
Mean.company.sector = c(Mean.Sd.SP500.companies[,2], "XLV"=Mean.Sd.XLV[2], "XLI"=Mean.Sd.XLI
[2])
```

```
# Plot the CAPM diagram
plot(SP500.companies.sector.betas[1:15], Mean.company.sector[1:15], pch=19,
     main="CAPM Model", xlim=c(0, 2), ylab= "SP500 Companies Returns", xlab="Beta")

# Adding the points for SPY and risk-free rate
points(1, Mean.Sd.SPY[2], col="red", pch=19)
points(0, Mean.FedFunds, col="green", pch=19)
points(SP500.companies.sector.betas[16], Mean.company.sector[16], col="blue", pch=19)
points(SP500.companies.sector.betas[17], Mean.company.sector[17], col="magenta", pch=19)

# Adding the line connecting the points for risk-free rate and SPY
lines(c(0, 1), c(Mean.FedFunds,Mean.Sd.SPY[2]))

# Putting labels with the company names on the graph
text(SP500.companies.sector.betas[1:15], Mean.company.sector[1:15],
     labels=rownames(SP500.companies.sector.betas)[1:15], cex=.5, pos=2)
text(1, Mean.Sd.SPY[2], labels="SPY", cex=.5, col="red", pos=2)
text(0, Mean.FedFunds, labels="FF.RF", cex=.5, col="green", pos=2)
text(SP500.companies.sector.betas[16], Mean.company.sector[16],
     labels="XLV", cex=.5, col="blue", pos=2)
text(SP500.companies.sector.betas[17], Mean.company.sector[17],
     labels="XLI", cex=.5, col="magenta", pos=2)
```



Same as the Efficient Frontier model, XLV did better than SPY while XLI did worse than the same.

3. Arbitrage Pricing Theory

```
# Calculate daily returns for each stock
Stock.Portfolio.Returns = as.data.frame(matrix(NA, nrow=dim(SPY)[1] - 1,
                                               ncol=length(SP500.Industrials.Health.names)))
colnames(Stock.Portfolio.Returns) = c(SP500.Industrials.Health.names)

for (i in colnames(Stock.Portfolio.Returns)){
  Stock.Portfolio.Returns[,i] = ROC(Ad(get(i)))[-1,]
}

head(Stock.Portfolio.Returns)
```

	CAT	FDX	GE	HON	LMT	NOC	
	<xts[,1]>	<xts[,1]>	<xts[,1]>	<xts[,1]>	<xts[,1]>	<xts[,1]>	
1	0.003566100	-0.0007223828	0.007922934	-0.003282323	-0.010387599	-0.005745778	-0.
2	0.013778178	0.0087617378	0.009351006	0.006343473	0.005770303	0.012037009	0.
3	-0.008316627	-0.0126439292	-0.004103706	-0.005283449	-0.010056694	-0.008951816	-0.
4	-0.006374463	-0.0033683589	-0.014307502	-0.001802853	-0.006909253	-0.006347472	-0.
5	0.006192811	0.0009922119	-0.001897826	0.003496299	0.004569474	0.003763302	0.

CAT	FDX	GE	HON	LMT	NOC	
<xts[,1]>	<xts[,1]>	<xts[,1]>	<xts[,1]>	<xts[,1]>	<xts[,1]>	
6-0.007107003	-0.0070294565	-0.004569926	-0.002753470	0.001581183	0.001000989	-0.

6 rows | 1-8 of 16 columns

```
# Calculate the returns for SPY, XLV, and XLI
SPY.returns = as.matrix(ROC(Ad(SPY))[-1])
XLV.returns = as.matrix(ROC(Ad(XLV))[-1])
XLI.returns = as.matrix(ROC(Ad(XLI))[-1])
```

3.1 Step 1: Selection of factors

```
# Start the process of factors selection by doing PCA on the stock portfolio
Stock.Portfolio.Returns.PCA = princomp(Stock.Portfolio.Returns)

# Calculate cumulative sum of all the standard deviations over the sum of standard deviations
cumsum(Stock.Portfolio.Returns.PCA$sdev/sum(Stock.Portfolio.Returns.PCA$sdev))
```

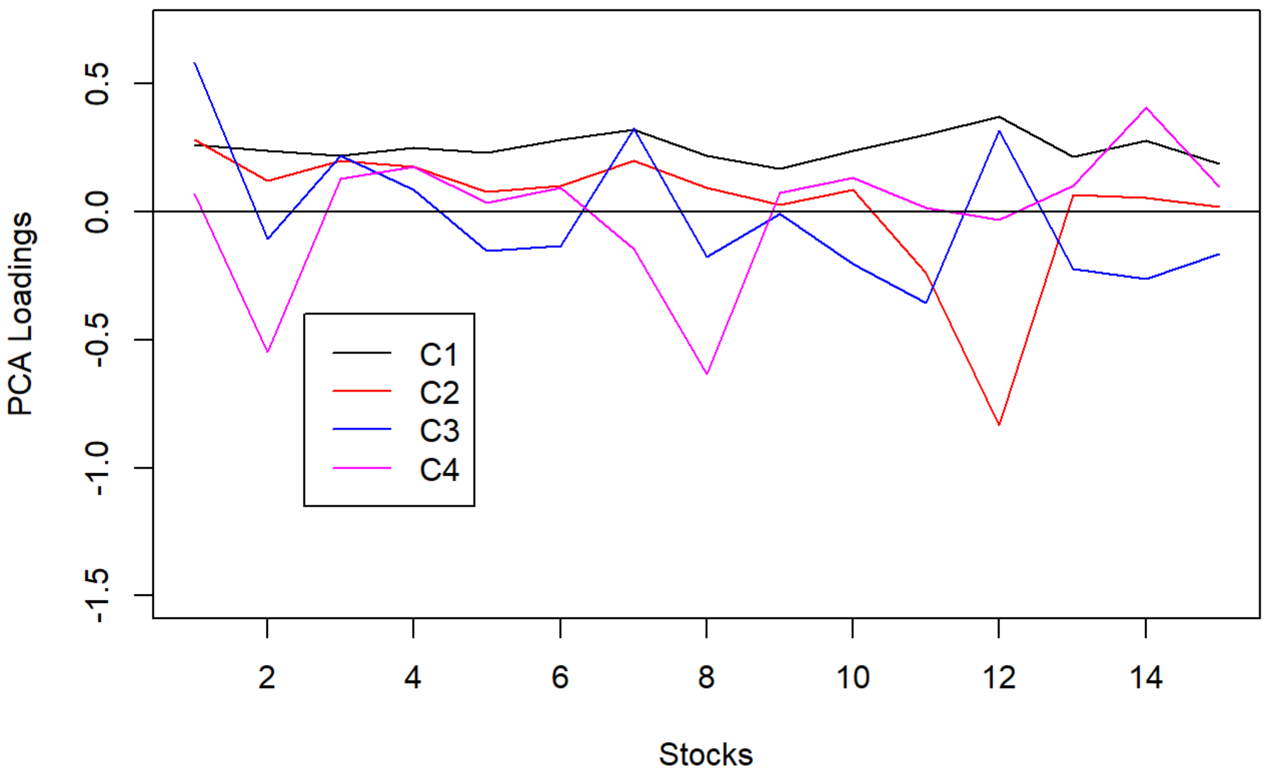
```
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8
## 0.2113686 0.3364545 0.4181374 0.4875410 0.5517611 0.6116146 0.6693570 0.7205440
##    Comp.9   Comp.10   Comp.11   Comp.12   Comp.13   Comp.14   Comp.15
## 0.7711462 0.8136195 0.8546923 0.8949620 0.9315689 0.9674921 1.0000000
```

Looking at the above values, we would need at least 13 factors to explain >90% variance. But we will proceed with 4 components, which accounts for nearly 50% of the variability.

```
# Create 4 factors and 4 loadings
Stock.Portfolio.Returns.PCA.factors = as.matrix(Stock.Portfolio.Returns.PCA$scores[,1:4])
Stock.Portfolio.Returns.PCA.loadings = Stock.Portfolio.Returns.PCA$loadings[,1:4]
Stock.Portfolio.Returns.PCA.zero.loading = Stock.Portfolio.Returns.PCA$center

head(Stock.Portfolio.Returns.PCA.loadings)
```

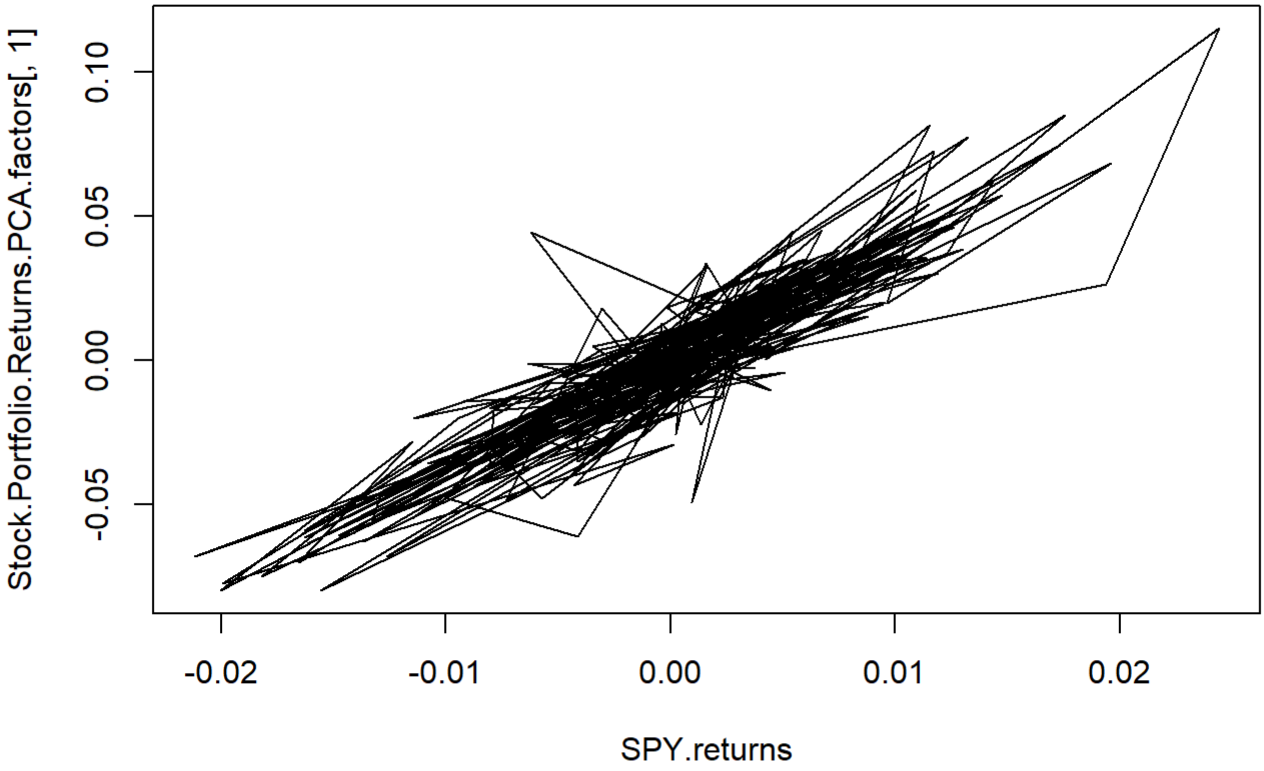
```
##        Comp.1     Comp.2      Comp.3      Comp.4
## CAT 0.2617008 0.28070973  0.58230589  0.07000261
## FDX 0.2413550 0.12119286 -0.10546125 -0.54947339
## GE  0.2210546 0.19872295  0.21931553  0.13092315
## HON 0.2529486 0.17638250  0.08817905  0.17744098
## LMT 0.2301699 0.07852472 -0.15347683  0.03614013
## NOC 0.2822664 0.10136532 -0.13444987  0.09404763
```

```
# Plot the Loadings
matplot(1:15, Stock.Portfolio.Returns.PCA.loadings, col=c('black', 'red', 'blue', 'magenta'),
        type="l", lty=1, ylab='PCA Loadings', xlab="Stocks", ylim=c(-1.5,0.7))
legend(2.5,-0.4,legend=c("C1", "C2", "C3", "C4"), col=c('black', 'red', 'blue', 'magenta'),
       lty=1)
abline(h=0)
```

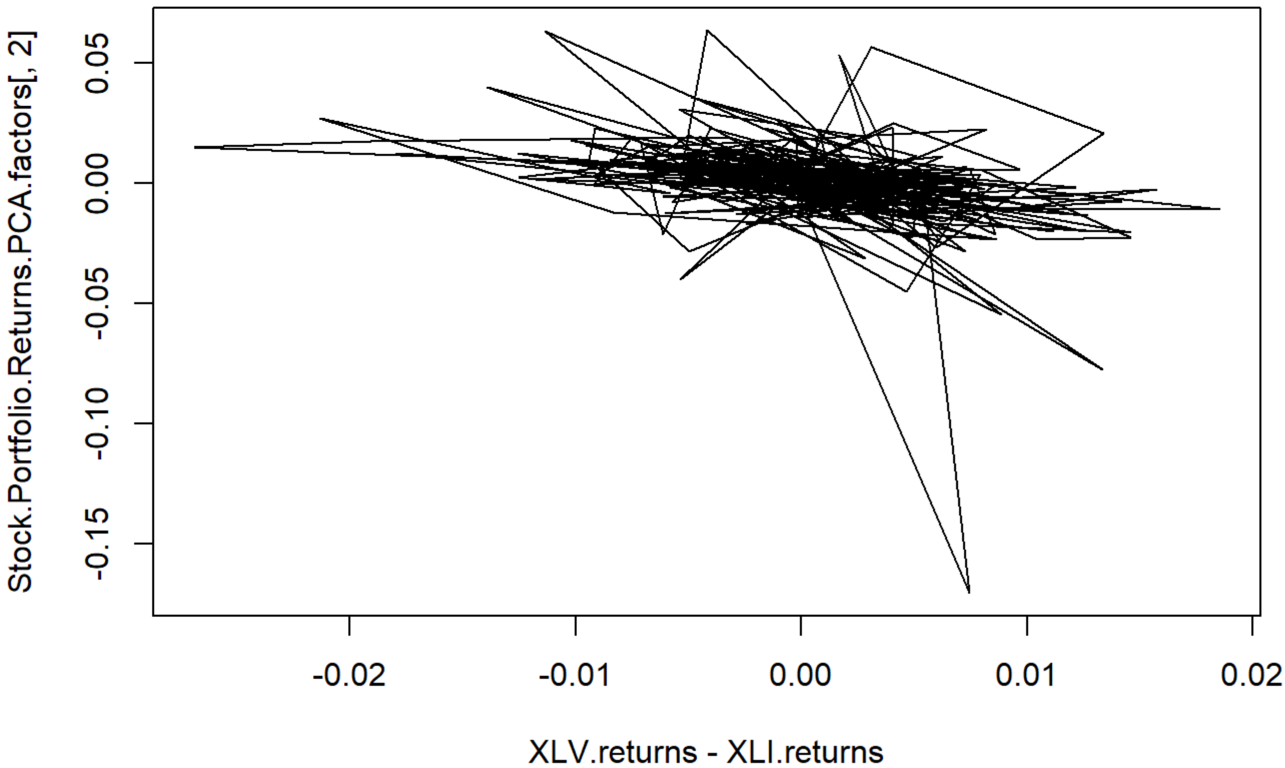


PCA loadings are positive for the first component for all 15 stocks. That means that the first factor is positively correlated with SPY. For the second component, all except 2, are positive. 3rd and 4th components aren't very expressive.

```
# Plot the first component with SPY
plot(SPY.returns,Stock.Portfolio.Returns.PCA.factors[,1], type="l")
```




```
# Plot the second component with the differences in the 2 sectors
plot(XLV.returns - XLI.returns, Stock.Portfolio.Returns.PCA.factors[,2], type = "l")
```



Fitting linear models explaining the interpretation of the components

```
lm.fit.factor1 = lm(Stock.Portfolio.Returns.PCA.factors[,1] ~ SPY.returns)
lm.fit.factor2 = lm(Stock.Portfolio.Returns.PCA.factors[,2] ~ I(XLV.returns - XLI.returns))
summary(lm.fit.factor1)
```

```
##
## Call:
## lm(formula = Stock.Portfolio.Returns.PCA.factors[, 1] ~ SPY.returns)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.051899 -0.007645  0.000223  0.008169  0.069564
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0009719  0.0008762  -1.109    0.268
## SPY.returns  3.8560362  0.1162021  33.184   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01387 on 249 degrees of freedom
## Multiple R-squared:  0.8156, Adjusted R-squared:  0.8148
## F-statistic:  1101 on 1 and 249 DF,  p-value: < 2.2e-16
```

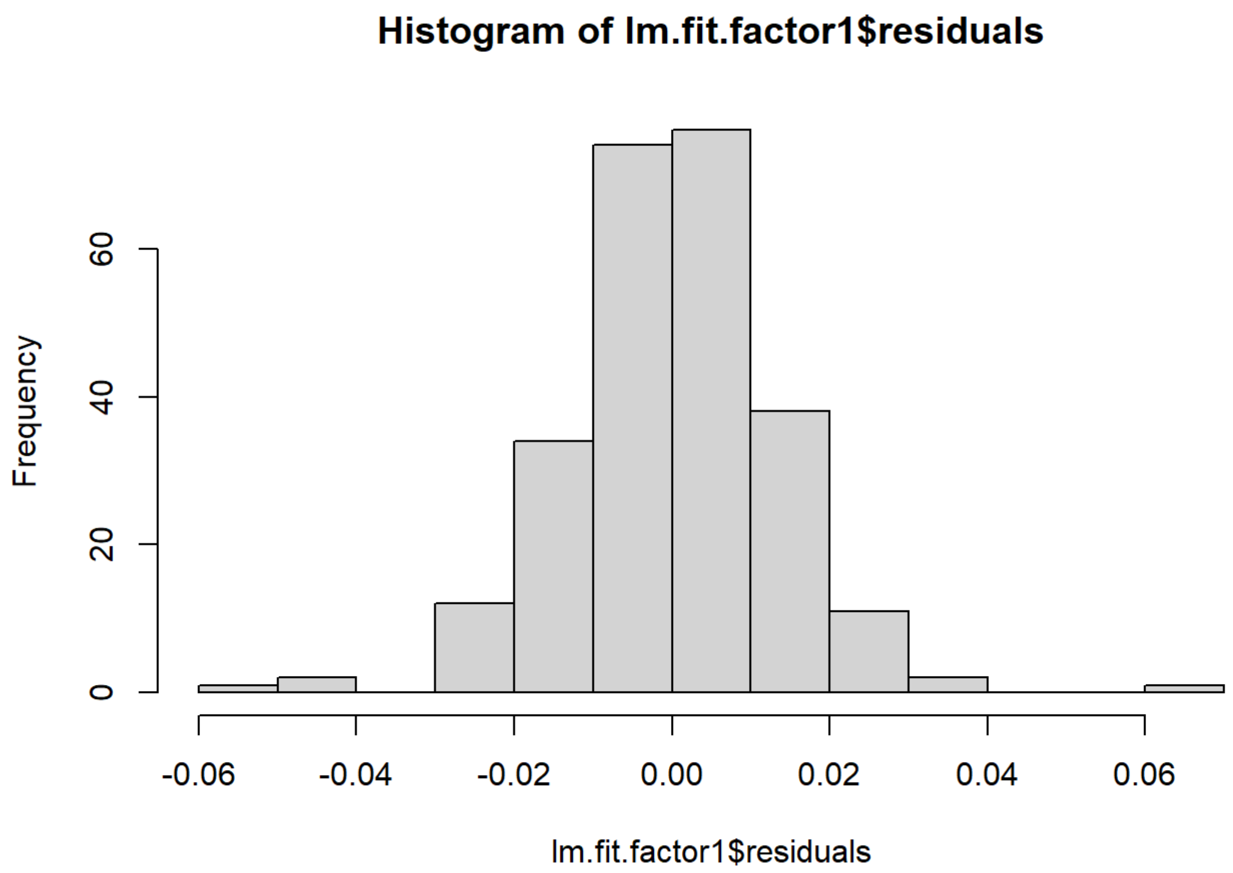
```
summary(lm.fit.factor2)
```

```
##
## Call:
## lm(formula = Stock.Portfolio.Returns.PCA.factors[, 2] ~ I(XLV.returns -
##     XLI.returns))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.162438 -0.006028 -0.000420  0.007744  0.059682
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   0.0008799  0.0011148   0.789    0.431
## I(XLV.returns - XLI.returns) -1.1722914  0.1715035  -6.835 6.27e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01754 on 249 degrees of freedom
## Multiple R-squared:  0.158,  Adjusted R-squared:  0.1546
## F-statistic: 46.72 on 1 and 249 DF,  p-value: 6.267e-11
```

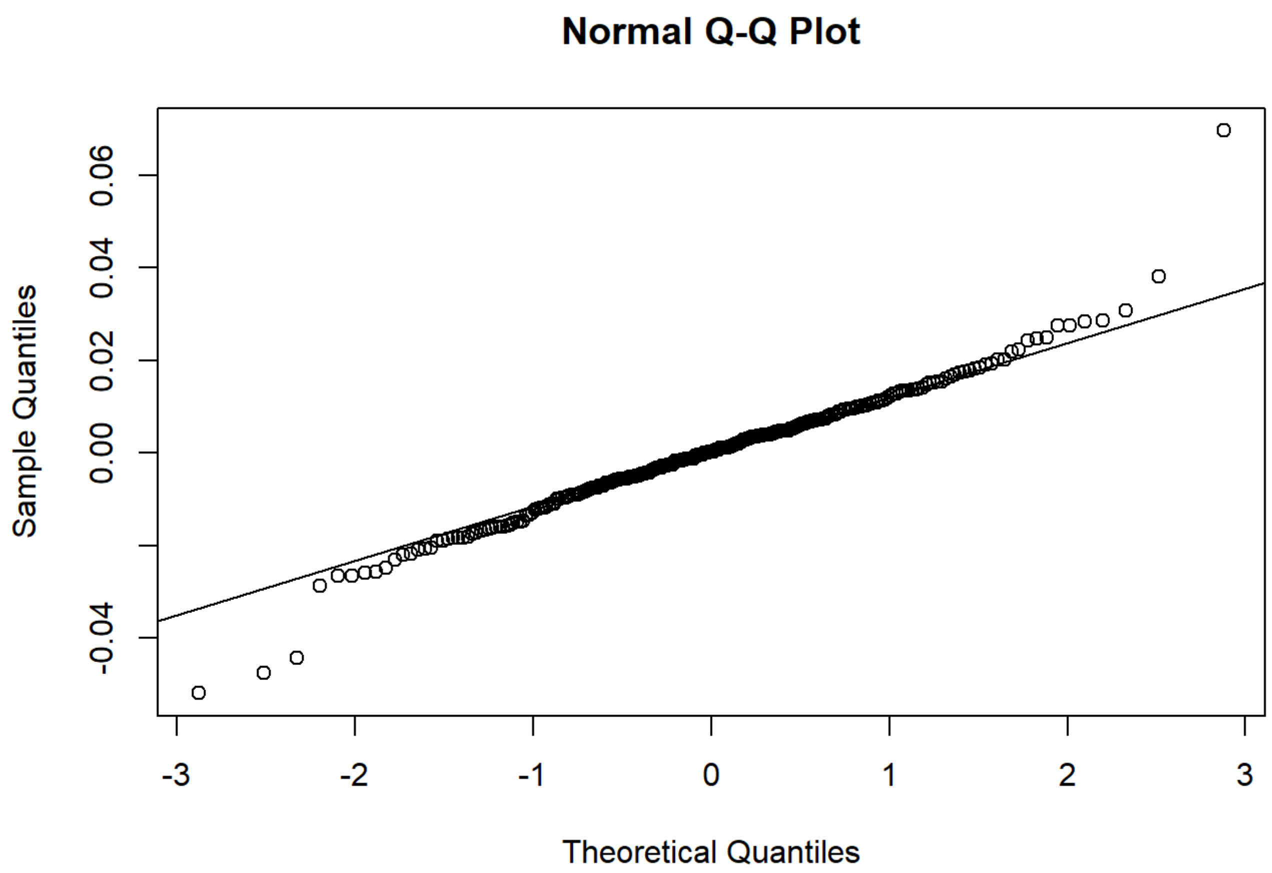
In both fits intercepts are practically insignificant, but both slopes are significant. The first factor fit has pretty good R^2, the second is not strong.

Check the residuals of both fits

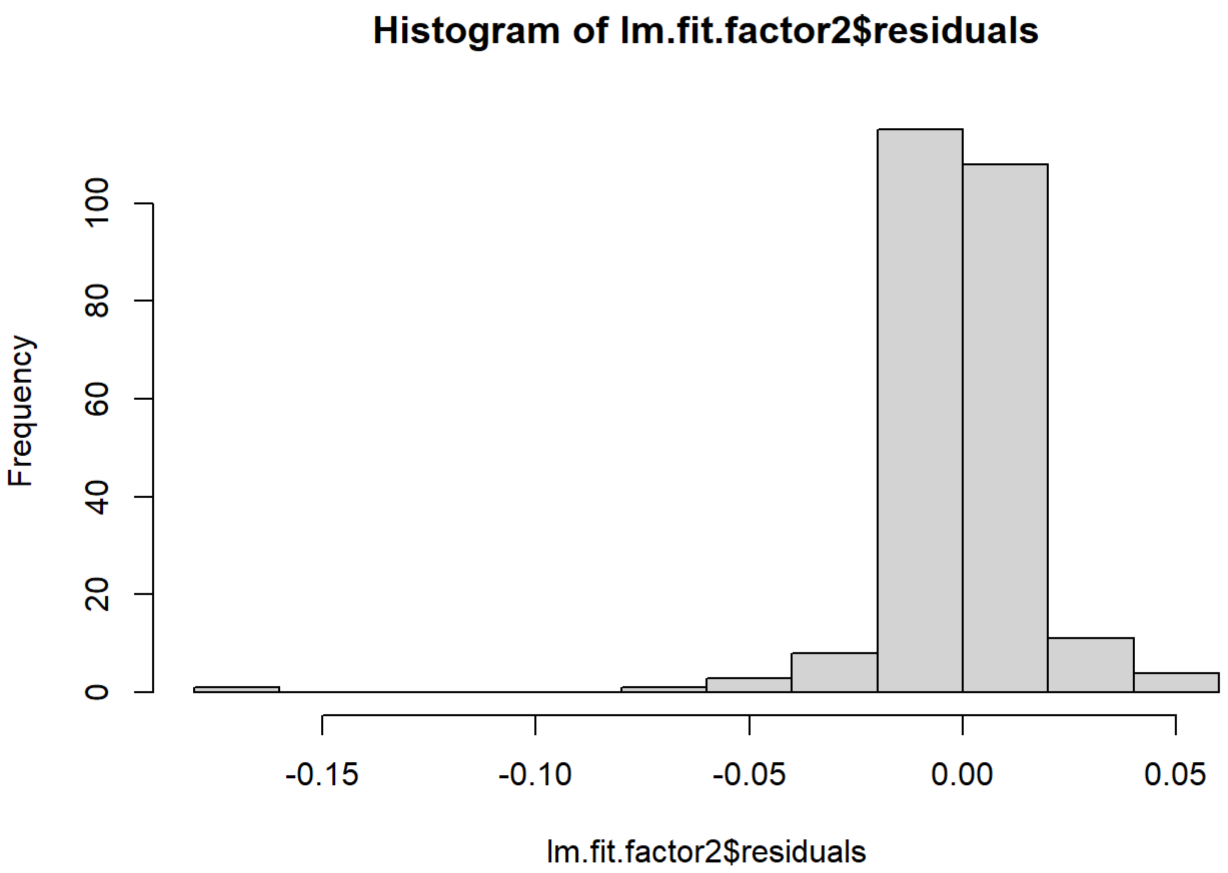
```
# Residuals of factor 1 fit
hist(lm.fit.factor1$residuals)
```



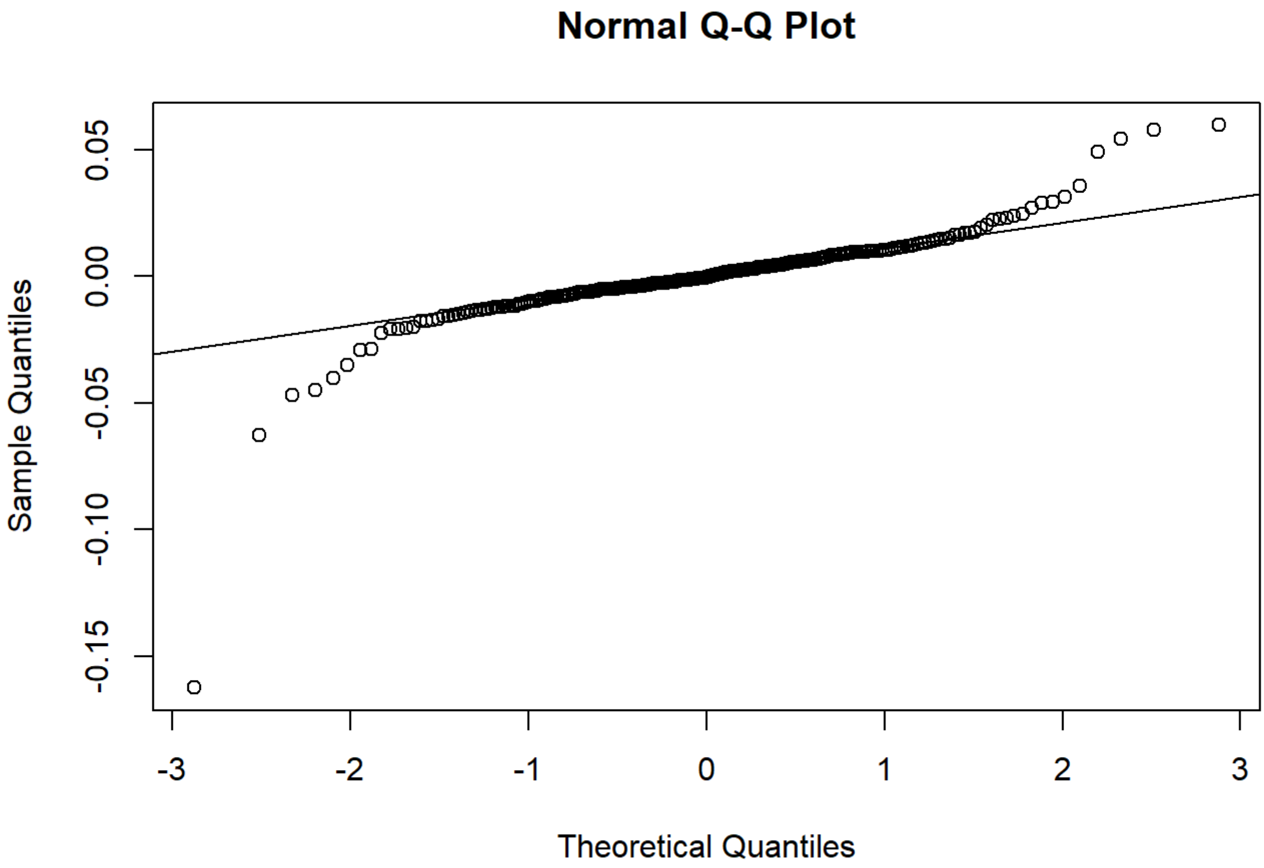

```
qqnorm(lm.fit.factor1$residuals)
qqline(lm.fit.factor1$residuals)
```



```r
# Residuals of factor 2 fit
hist(lm.fit.factor2$residuals)
```




```r
qqnorm(lm.fit.factor2$residuals)
qqline(lm.fit.factor2$residuals)
```



3.2 Step 2: Estimation of betas

```
# Check that betas are given by the PCA factor loadings
Stock.portfolio.betas = apply(Stock.Portfolio.Returns, 2,
                              function(z) lm(z~Stock.Portfolio.Returns.PCA.factors[,1] +
                                              Stock.Portfolio.Returns.PCA.factors[,2] +
                                              Stock.Portfolio.Returns.PCA.factors[,3] +
                                              Stock.Portfolio.Returns.PCA.factors[,4]
                                            )$coefficients)

rownames(Stock.portfolio.betas) = c("Alpha", "Factor.1", "Factor.2", "Factor.3", "Factor.4")
Stock.portfolio.betas = as.data.frame(t(Stock.portfolio.betas))

Stock.portfolio.betas
```

	Alpha <dbl>	Factor.1 <dbl>	Factor.2 <dbl>	Factor.3 <dbl>	Factor.4 <dbl>
CAT	-0.0008847112	0.2617008	0.28070973	0.582305894	0.07000261
FDX	0.0004650234	0.2413550	0.12119286	-0.105461254	-0.54947339
GE	0.0001666116	0.2210546	0.19872295	0.219315535	0.13092315
HON	0.0003788085	0.2529486	0.17638250	0.088179048	0.17744098
LMT	0.0007044108	0.2301699	0.07852472	-0.153476827	0.03614013
NOC	0.0011759220	0.2822664	0.10136532	-0.134449873	0.09404763

	Alpha	Factor.1	Factor.2	Factor.3	Factor.4
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
UNP	-0.0001184946	0.3218095	0.19861411	0.327460945	-0.14518114
UPS	-0.0001268312	0.2214923	0.09494905	-0.176475957	-0.63484832
WM	0.0002543294	0.1702028	0.02637558	-0.006119151	0.07479870
ABT	0.0007829838	0.2393350	0.08631066	-0.203754216	0.13560782

1-10 of 15 rows Previous 1 2 Next

```
cbind(zeroLoading=Stock.Portfolio.Returns.PCA.zero.loading,
      Stock.Portfolio.Returns.PCA.loadings[,1:4])
```

```
##       zeroLoading    Comp.1      Comp.2       Comp.3      Comp.4
## CAT -0.0008847112 0.2617008  0.28070973  0.582305894  0.07000261
## FDX  0.0004650234 0.2413550  0.12119286 -0.105461254 -0.54947339
## GE   0.0001666116 0.2210546  0.19872295  0.219315535  0.13092315
## HON  0.0003788085 0.2529486  0.17638250  0.088179048  0.17744098
## LMT  0.0007044108 0.2301699  0.07852472 -0.153476827  0.03614013
## NOC  0.0011759220 0.2822664  0.10136532 -0.134449873  0.09404763
## UNP -0.0001184946 0.3218095  0.19861411  0.327460945 -0.14518114
## UPS -0.0001268312 0.2214923  0.09494905 -0.176475957 -0.63484832
## WM   0.0002543294 0.1702028  0.02637558 -0.006119151  0.07479870
## ABT  0.0007829838 0.2393350  0.08631066 -0.203754216  0.13560782
## AET  0.0017974121 0.3016749 -0.23832453 -0.354283938  0.01723102
## HUM  0.0015888953 0.3712530 -0.83541566  0.318362316 -0.03020742
## JNJ -0.0002203025 0.2161350  0.06823679 -0.224152300  0.10323346
## MDT  0.0006614435 0.2780421  0.05553228 -0.262358378  0.40605256
## PFE  0.0005697645 0.1898185  0.02013955 -0.162528502  0.09778490
```

####3.3 Step 3. Estimation of market price of risk

```
Market.Prices.of.risk.fit = lm(I(Alpha-Mean.FedFunds)~.-1,data=Stock.portfolio.betas)
summary(Market.Prices.of.risk.fit)
```

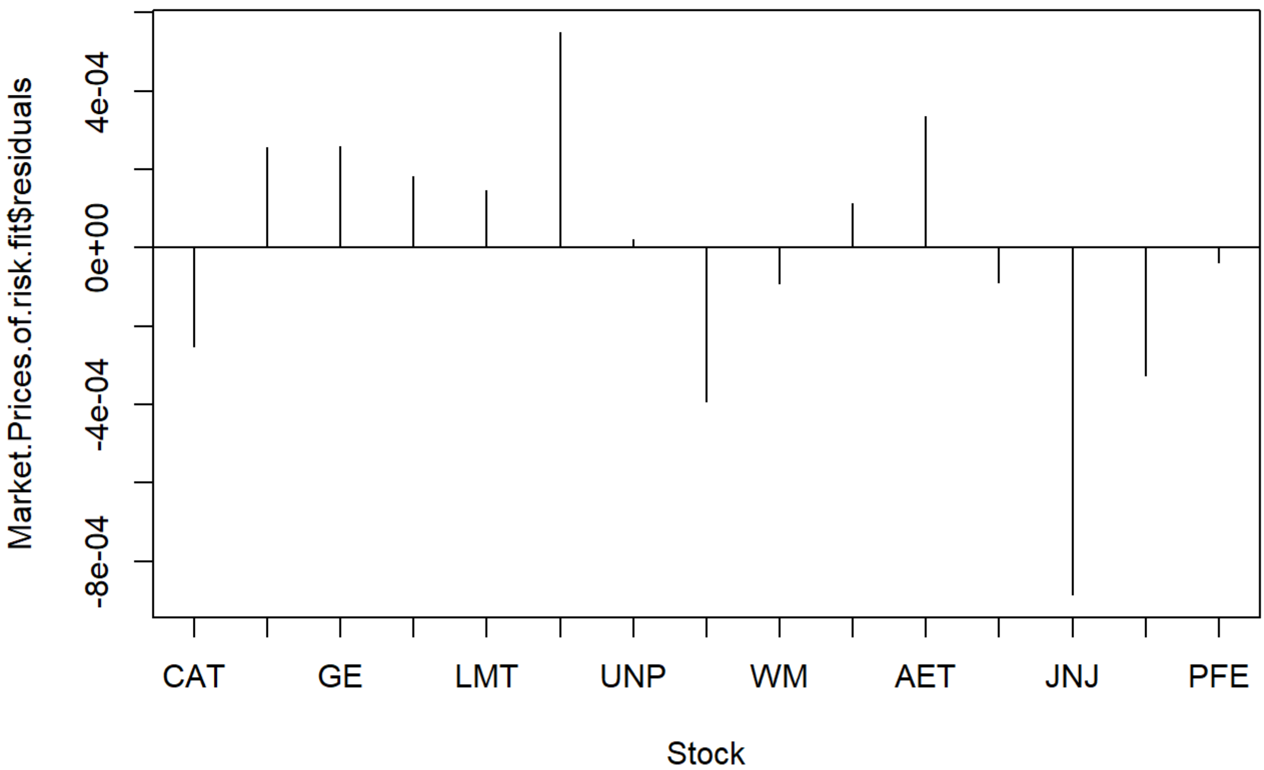
```
##
## Call:
## lm(formula = I(Alpha - Mean.FedFunds) ~ . - 1, data = Stock.portfolio.betas)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -8.851e-04 -1.719e-04  2.234e-05  2.200e-04  5.486e-04
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## Factor.1  0.0020370  0.0003975   5.124 0.000331 ***
## Factor.2 -0.0016027  0.0003975  -4.032 0.001976 **
## Factor.3 -0.0012850  0.0003975  -3.232 0.007981 **
## Factor.4  0.0004151  0.0003975   1.044 0.318760
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0003975 on 11 degrees of freedom
## Multiple R-squared:  0.8309, Adjusted R-squared:  0.7694
## F-statistic: 13.51 on 4 and 11 DF,  p-value: 0.0003167
```

Both R^2 and adjusted R^2 are high. Slope parameters are significant for Factor 1, 2 and 3 at 5% level. The F test can reject the utility hypothesis with 5% level. Note that slopes for factor 4 are not significant.

```
# Looking at coefficients of the factors from the linear model
Market.Prices.of.risk = c(Mean.FedFunds,Market.Prices.of.risk.fit$coefficients)
Market.Prices.of.risk
```

```
##                     Factor.1      Factor.2      Factor.3      Factor.4
##  3.010362e-06  2.036996e-03 -1.602729e-03 -1.284958e-03  4.151305e-04
```

```
# Plot the residuals
plot(Market.Prices.of.risk.fit$residuals,type="h",xaxt="n",xlab="Stock")
abline(h=0)
axis(1, at=1:15, labels=SP500.Industrials.Health.names)
```



When residual is positive it means that the mean excess return of the stock over the sample period is greater than predicted value, it means that the stock outperformed the predicted value of the model over the sample period and it is undervalued. When the residual is negative, it means that the stock gave less return than the model predicted value and it is overvalued by the model.