A Tutorial of Factor Models and Their Implementation in R

Lucas Perin[1]

Abstract

This tutorial examines three factor models: the Fama-French factor model, a BARRA-type industry factor model and a PCA factor model. I discuss the mathematical aspects of each model and provide an R implementation, which I then use to construct minimum-variance weighted portfolios for each model. The resulting portfolios are then applied to new data.

1 Introduction and Overview

The modern portfolio theory introduced by Markowitz (1952) is widely used in practice. However, in order to obtain an efficient portfolio, an investor has to decide how quantify risks and returns. In order to determine the risks and returns, investors apply pricing or risk factor models. There are many factor models, the most famous being the CAPM, introduced independently by Sharpe (1964), Treynor (1962) and Litner (1965).

The CAPM is defined as:

$$E(R_i) = R_f + \beta_i(E(R_m) - R_f) \tag{1}$$

Where:

- $E(R_i)$: expected return on the capital asset
- R_f: risk-free rate
- β_i: sensitivity of the expected asset returns to the expected market returns
- $E(R_m)$: expected return of the market
- $E(R_m) - R_f$: market risk premium

Changing the returns-based factor model to a model based on **excess returns** we have:

$$E(R_i) - R_f = \beta_i(E(R_m) - R_f)$$
$$E(R) = \beta_i \lambda \tag{2}$$

[1] Foster School of Business, University of Washington - lmperin@uw.edu

Where:

- $E(R)$: expected excess returns
- λ: market risk premium

In the CAPM, all the risk is captured by λ: participating on the market is the single factor of risk. This makes the CAPM a single factor model. Assuming the model is correct, the challenge is how to find λ and β for the assets that will compose the portfolio.

To find λ, one common procedure is to observe the return of a broad index such as the S&P 500 or the Wilshire 3000 and subtract the risk-free rate observed using treasury bill data from the Federal Reserve.

To find β one usually uses Sharpe's regression:

$$R_{i,t} = \beta_i \lambda + \epsilon_{i,t} \tag{3}$$

Where:

- $R_{i,t}$: time-series of observed (excess) returns
- λ: market risk premium, as given above
- ϵ: random error, assumed to have mean zero and no autocorrelation
- β_i: resulting regression estimate

The CAPM is usually taught as the introductory model for asset pricing, and as shown above, assumes a single factor of risk. Although this is the feature that makes the CAPM simple to understand and broadly accepted, one frequent criticism of the CAPM is that it is *too* simple - other factors of risk must exist. But as one professor from Foster recently remarked, once you reject the CAPM, you are on your own: there are many other candidates, none of which is a clear winner against the CAPM.

In this work, I'm going to explore three alternative factor models and use them to derive a minimum variance portfolio. The first is a time-series factor model: the Fama-French three-factor model, described in Fama and French (1995). The second is a cross-sectional factor model, a BARRA-type industry factor model. The third is a statistical model, the principal components factor model.

After I explore the general form of each one of these three factor model types, I will present an implementation of each in R. I will then use five years of data from stocks that comprise the

Dow Jones Industrial Average to obtain a minimum variance portfolio. Finally, I will test the minimum variance portfolio obtained by each approach with two months of new data.

There are some empirical problems with this approach, one of which is that each portfolio was constructed with five years of past data but will be tested with only two months. There are many possible objections as to the validity of this test: maybe the results could be skewed by seasonality, for example. Another valid objection is that the DJIA represents a concentrated (in opposition to a diversified) portfolio, so diversifying using DJIA components still creates a concentrated portfolio. These are valid objections that will be addressed in future versions of this work.

2 Theoretical Background

2.1 Time series factor models

The general form of a time-series factor model for asset returns is:

$$R_{j,t} = \beta_{0,j} + \beta_{1,j} F_{1,t} + ... + \beta_{p,j} F_{p,t} + \epsilon_{j,t} \tag{4}$$

Where:

- $R_{j,t}$: excess return of the jth asset on time t.
- $F_{i,t}$: risk factors at time t.
- ϵ: uncorrelated, mean-zero risks.

Or, in matrix form:

$$\mathbf{R_t} = \beta_\mathbf{0} + \beta^\mathbf{T} \mathbf{F_t} + \epsilon_\mathbf{t} \tag{5}$$

Time series factor models use observable risk factors and estimate β through a regression of the risk factors on the returns.

One way to measure of risk of the assets above is by using the covariance matrix of the returns sample. That sample is unbiased, but has the undesirable property of having very large estimation errors. Instead, one frequently trades off these estimation errors by using Σ_F, the covariance matrix from the factors, to calculate a biased estimate of the returns through the formula below:

$$\mathbf{\Sigma_R} = \beta^{\mathbf{T}} \mathbf{\Sigma_F} \beta + \mathbf{\Sigma}_\epsilon \tag{6}$$

Although it can be shown that the above estimate is biased, its estimation error is usually much smaller than using the sample covariance matrix, and the bias is usually very small, although each one of the statements that are "usually" true in the previous sentence should be tested. Future versions of this work will show how to test for these conditions.

The minimum-variance portfolio is calculated by the following formula:

$$\mathbf{W_U} = \mathbf{\Sigma_R^{-1}} \times \mathbf{I_t} \tag{7}$$

$$\mathbf{W} = \frac{\mathbf{W_U}}{\sum_{\mathbf{i=1}}^{\mathbf{T}} \mathbf{W_i}} \tag{8}$$

$\mathbf{I_t}$ is a vector of length t comprised only of ones . Equation 7 calculates the relative weights of each asset and equation 8 normalizes the weights so they sum to 1.

2.1.1 The Fama-French three-factor model

To demonstrate the implementation of a time-series factor model, I will use the Fama-French factor model. It has three risk factors. The first is the same as the CAPM: the excess return of the market portfolio. The second risk factor is called small minus large (SML) and represents the difference in returns of a portfolio comprised only of stocks of companies with small market values and a portfolio comprised of stocks of companies of large market value. The third factor, called high minus low (HML) is the difference between the returns of a portfolio comprised of companies of high book-to-market value and a portfolio comprised of low book-to-market value companies. The equation of their model is:

$$R_{j,t} = \beta_{0,j} + \beta_{1,j}\lambda + \beta_{2,j} SML_t + \beta_{3,j} HML_t + \epsilon_{j,t} \tag{9}$$

It is important to notice that SML and HML in their model depend on t, and not on the assets being used - this is a key difference between time-series models and cross-sectional models. The HML and SML parameters are calculated by Professor French and made available in his website, at http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html.

It is worth noting the pervasiveness of the language of this model - the SML parameter

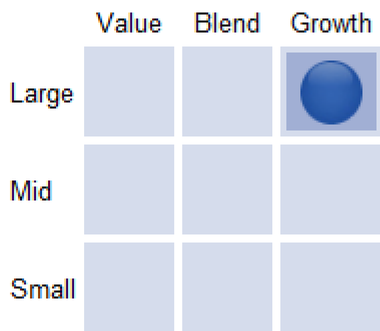

Figure 1: Example of Fama-French terminology in use

divides assets into "large-cap" and "small-cap" and the HML parameter divides assets into "value" (high book-to-market) and "growth" (low book-to-market). This language is widely used now. Figure 1 shows the composition of a mutual fund as of 12/31/2010.

The covariance matrix and minimum-weight portfolio will be calculated as described in section 2.1. An implementation of this model in R and the minimum-variance portfolio found by it using data from the 30 stocks that comprise the DJIA index are in section 3.1 and 4.1 respectively.

2.2 Cross-sectional factor models

Time-series models described above do not use asset-specific properties as risk-factors. If a theorist thinks that these are the important risk factors, an alternative is a **cross-sectional factor model** - it uses data from many assets but assumes the holding period is fixed. These are also called BARRA models after BARRA, Inc., a company that markets these models to financial managers. Since t is fixed, we can remove it from the general form equation, obtaining:

$$R_j = \beta_0 + \beta_1 F_{1,j} + ... + \beta_{p,j} F_{p,j} + \epsilon_j \tag{10}$$

Or, in matrix form:

$$\mathbf{R} = \beta_{\mathbf{0}} + \beta^{\mathbf{T}} \mathbf{F} + \epsilon \tag{11}$$

Different than time series factor models, the cross-sectional factor models make assumptions or observations about β and estimates F through a regression.

To estimate the F in the equation above, using the procedure described by Zivot and Wang (2006), we can use the following regression equation:

$$\hat{\mathbf{F}} = (\beta \mathbf{D^{-1}} \beta)^{-1} \mathbf{D^{-1}} \mathbf{R} \tag{12}$$

Where $\mathbf{D}$ is a matrix containing the variance (σ^2) of the errors. We don't know σ^2 *a priori*, but we can run a regular regression on equation 11 to estimate it.

The return covariances matrix is similar to the time-series model:

$$\mathbf{\Sigma_R} = \beta^{\mathbf{T}} \mathbf{\Sigma_F} \beta + \mathbf{\Sigma}_\epsilon \tag{13}$$

To find the portfolio weights, we use the same formulas described in equations 7 and 8.

2.2.1 A BARRA-type cross-sectional factor model

As an example, I will implement a cross-sectional factor model assuming that the industry is a risk factor for stocks comprising the DJIA. I will use six industries. In parentheses are the DJIA components that I assigned to each of the six industries: Materials (XOM, CVX, AA), Industrial (MMM, GE, BA, UTX, CAT, DD), Consumer (KO, HD, KFT, MCD, PG, WMT, DIS), Financial (TRV, AXP, BAC, JPM), Health (JNJ, MRK, PFE) and Technology (T, HPQ, INTC, IBM, MSFT, VZ, CSCO).

Applying these factors to the form described in the previous section, we obtain :

$$R_j = \beta_0 + \beta_1 \text{Mat}_j + \beta_2 \text{Ind}_j + \beta_3 \text{Cons}_j + \beta_4 \text{Fin}_j + \beta_5 \text{Health}_j + \beta_6 \text{Tech}_j + \epsilon_j \tag{14}$$

An implementation of this model in R and the minimum-variance portfolio found by it using data from the 30 stocks that comprise the DJIA index are in section 3.2 and 4.2 respectively.

2.3 PCA statistical factor models

Another type of model is an statistical factor model based on principal component analysis. Instead of theorizing on risk factors, this type of factor model tries to determine the risk factors by observing them in the data. It has the advantages of being easy to understand, compute and visualize, but on the other hand, it is more sensitive to the data used to compute the model.

The only parameter of this model is the number of factors to include. The number of factors is usually determined by running a principal component analysis on the available data and observing the result through a scree plot (named after the accumulation of the rock fragments at the base of mountains). To perform the principal component analysis, one calculates the *eigenvectors* V and *eigenvalues* D of the sample data matrix. A visual inspection of the scree plot can give an idea of how many factors capture the most variance.

Once the number of factors k is determined, one can obtain the covariance matrix by applying the formula:

$$\mathbf{\Sigma} = \mathbf{V}\mathbf{D}\mathbf{V}^{\mathbf{T}} \tag{15}$$

If not all the principal components are used, a small error ϵ is introduced. The covariance matrix then is:

$$\mathbf{\Sigma} = \beta^{\mathbf{T}}\mathbf{\Sigma_F}\beta + \Sigma_\epsilon \tag{16}$$

Where β are the first k eigenvectors, Σ_F is the variance of the sample rotated by the first k eigenvectors.

An implementation of this model in R and the minimum-variance portfolio found by it using data from the 30 stocks that comprise the DJIA index are in section 3.3 and 4.3 respectively.

3 Algorithm Implementation and Development

All references point to line numbers for the code that is in the appendix. I'm sticking to the format suggested for this assignment, but in general I would use the code in my explanation.

This section discusses the R implementation of the three factor models described above. It uses the returns of the stocks that comprise the DJIA between 2006/01/01 and 2010/12/31. Data with the prices of these securities is available on the file "djiaPrices.csv". The data was loaded in line 13 and converted from prices to returns in line 24, where it is also loaded in the variable `returns` . Note that the returns are **excess returns**, returns in excess of the risk-free rate. I have used the risk-free data from Prof. French's website, but one could use the Federal Reserve data instead, as it is the same data.

3.1 Implementation of the Fama-French three-factor model

As discussed in section 2.1.1, in order to calculate the betas for the Fama-French model, one needs to obtain the risk factors, available on Prof. French's website. The file I used is "FF-daily.csv". The data was loaded in line 12, and excess data points were removed in lines 20-21, which remove the rows for which we don't have returns for the securities and line 35, which leaves the data with only three columns: market returns, HML and SML.

The implementation of Fama-French is then very straightforward. We run a regression in line 38 using a simple regression: `lm returns ~ factors`. We calculate the covariance matrix using the factors in lines 50-53 using the formula described in equation 6. In lines 56-57 we find the weights for the minimum-variance portfolio, as described in equations 7 and 8.

The results are described in section 4.1.

3.2 Implementation of the BARRA-type factor model

Lines 63-85 assemble a matrix that has one row for each asset and one column for each industry. If the asset in row r belongs to the column c, the cell at position r, c of the matrix contains a 1, otherwise it contains a 0 - that is, it is a matrix of dummy variables. This is the matrix that contains the β in equation 14.

To obtain $\mathbf{D}$ for equation 12, we run a simple regression (lines 91-99). With it, we implement the formula in equation 12 in lines 101-104, and obtain the desired factors $\hat{\mathbf{F}}$. We use the $\hat{\mathbf{F}}$ to calculate a covariance matrix (line 115), and use the equations 7 and 8 to obtain the minimum variance portfolio (lines 117-119).

3.3 Implementation of the PCA factor model

For this section, we arbitrarily chose five factors. The usual way of choosing factors is to do a scree plot. The scree plot for the first ten principal components for the DJIA returns data is shown in Figure 2.

It is easy to notice that each additional component after the third or fourth adds little explanation to the variance.

To obtain $\mathbf{V}$, the *eigenvectors*, we use the first five *rotation* components of the object `pcaEq`, that was obtained using the command `prcomp` on the returns data, as shown in line 142. The eigenvalues are obtained in line 143, and the covariance matrix is calculated using equation 15,

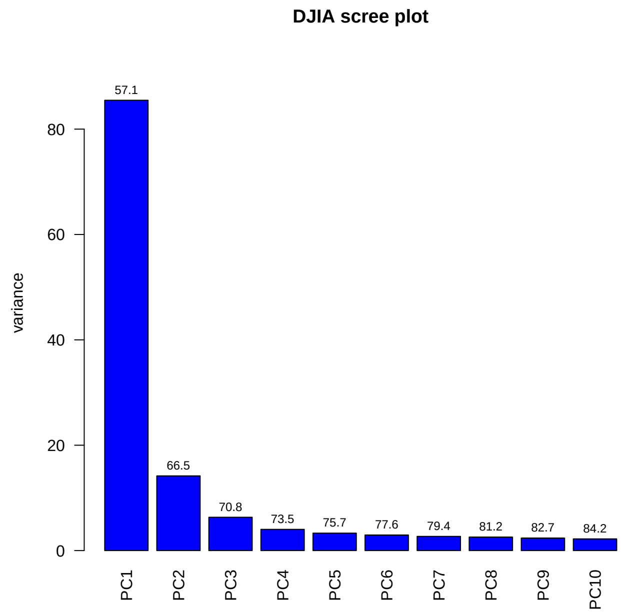

Figure 2: Scree plot for DJIA

in lines 152-153.

The minimum-weight portfolio is again calculated using equations 7 and 8 and the code for that calculation for the PCA model is in lines 155-157.

4 Computational Results

This section describes the results found using each one of the models presented earlier. The R-squared boxplots for each one of the methods employed are in Figure 3.

4.1 Results for the Fama-French three-factor model

The resulting betas for the Fama-French factor model are in Figure 4.

The resulting portfolio weights are listed in Table 1, and its comparison to the sample-based portfolio is on Figure 5.

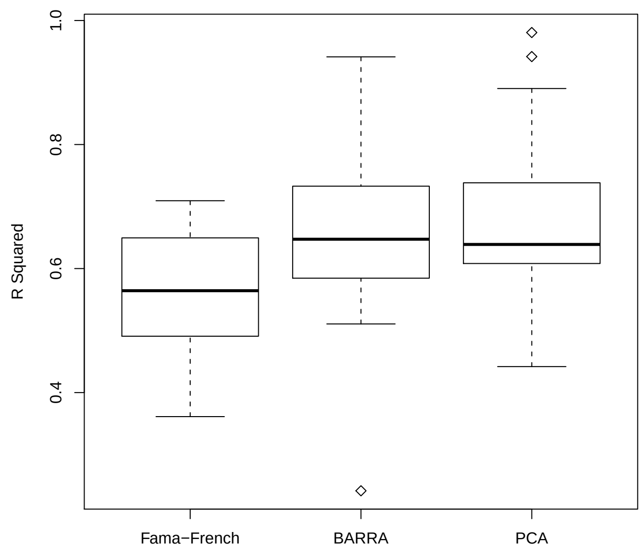

Figure 3: R-squared for each factor model

4.2 Results for the BARRA-type factor model

The resulting portfolio weights are listed in Table 2, and its comparison to the sample-based portfolio is on Figure 6.

4.3 Results for the PCA factor model

The resulting portfolio weights are listed in Table 3, and its comparison to the sample-based portfolio is on Figure 7.

5 Summary and Conclusions

In the code lines 263-301, I retrieve new pricing data for the assets that were used to construct the portfolios. I then apply each one of the weighted portfolios to the new returns. A graphical version of the returns is shown on Figure 8. Interestingly, the market portfolio offered much larger returns than the factor models provided.

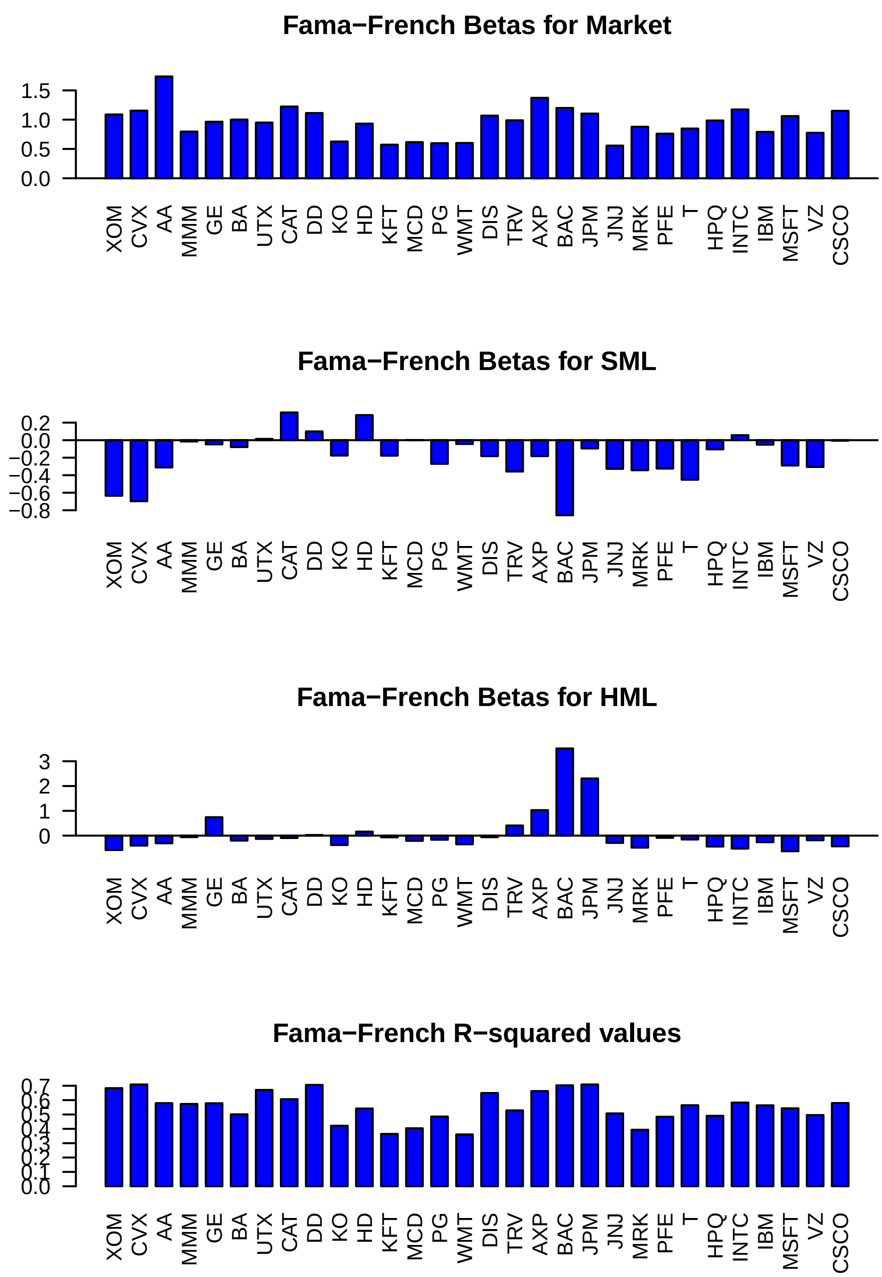

Figure 4: Betas and R-Squareds for Fama-French

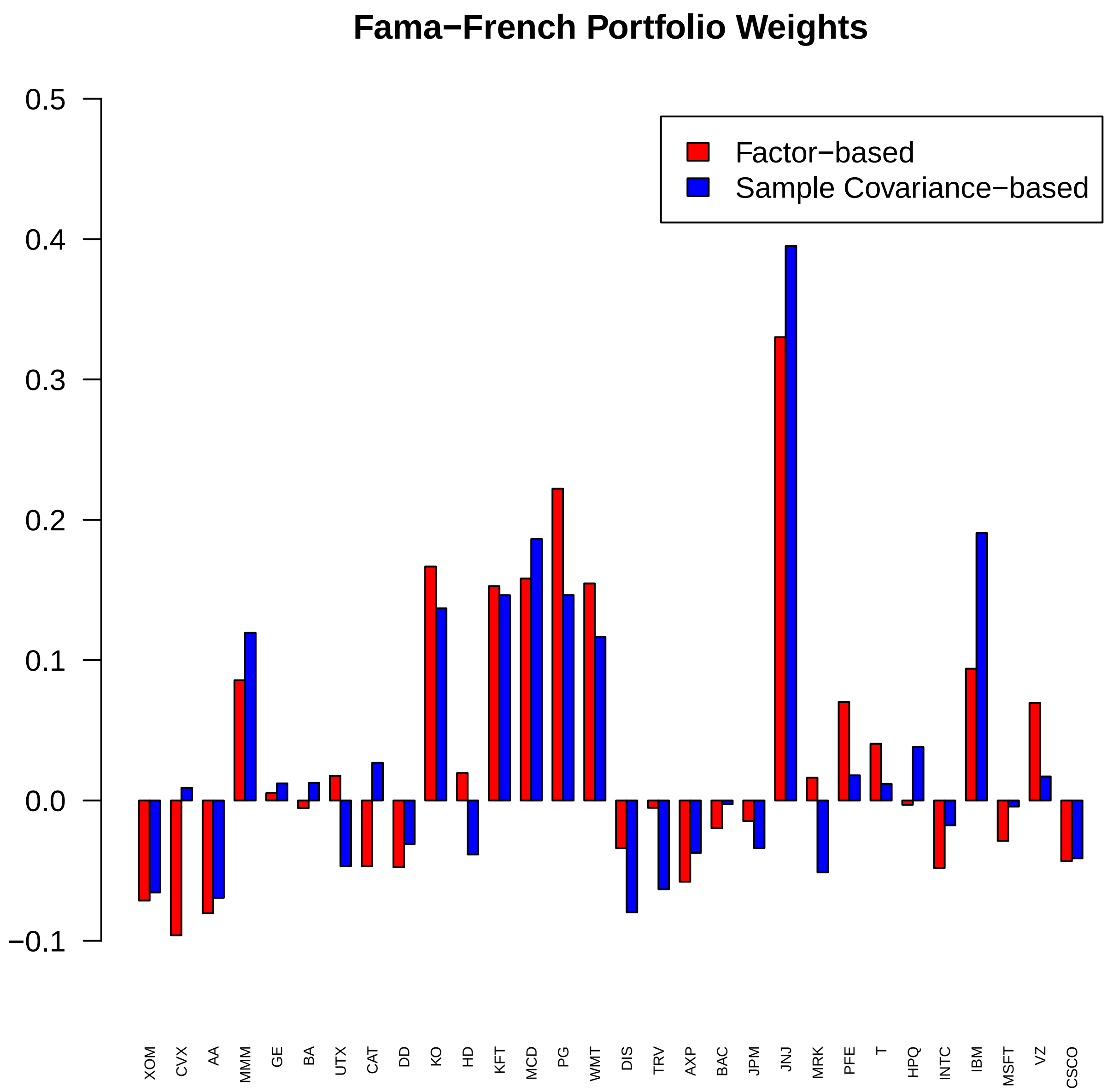

Figure 5: Portfolio Weights for Fama-French

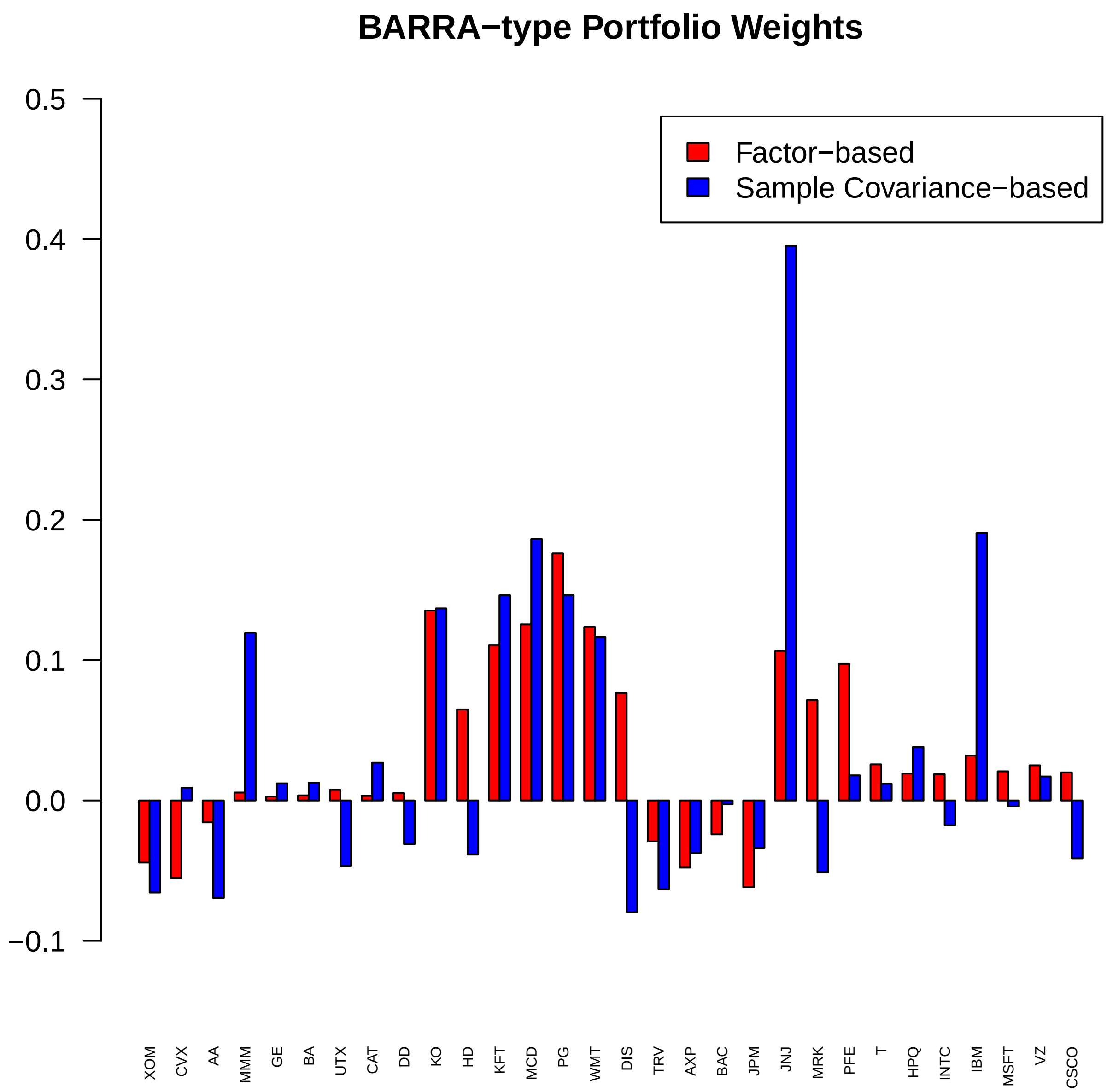

Figure 6: Portfolio Weights for BARRA-type

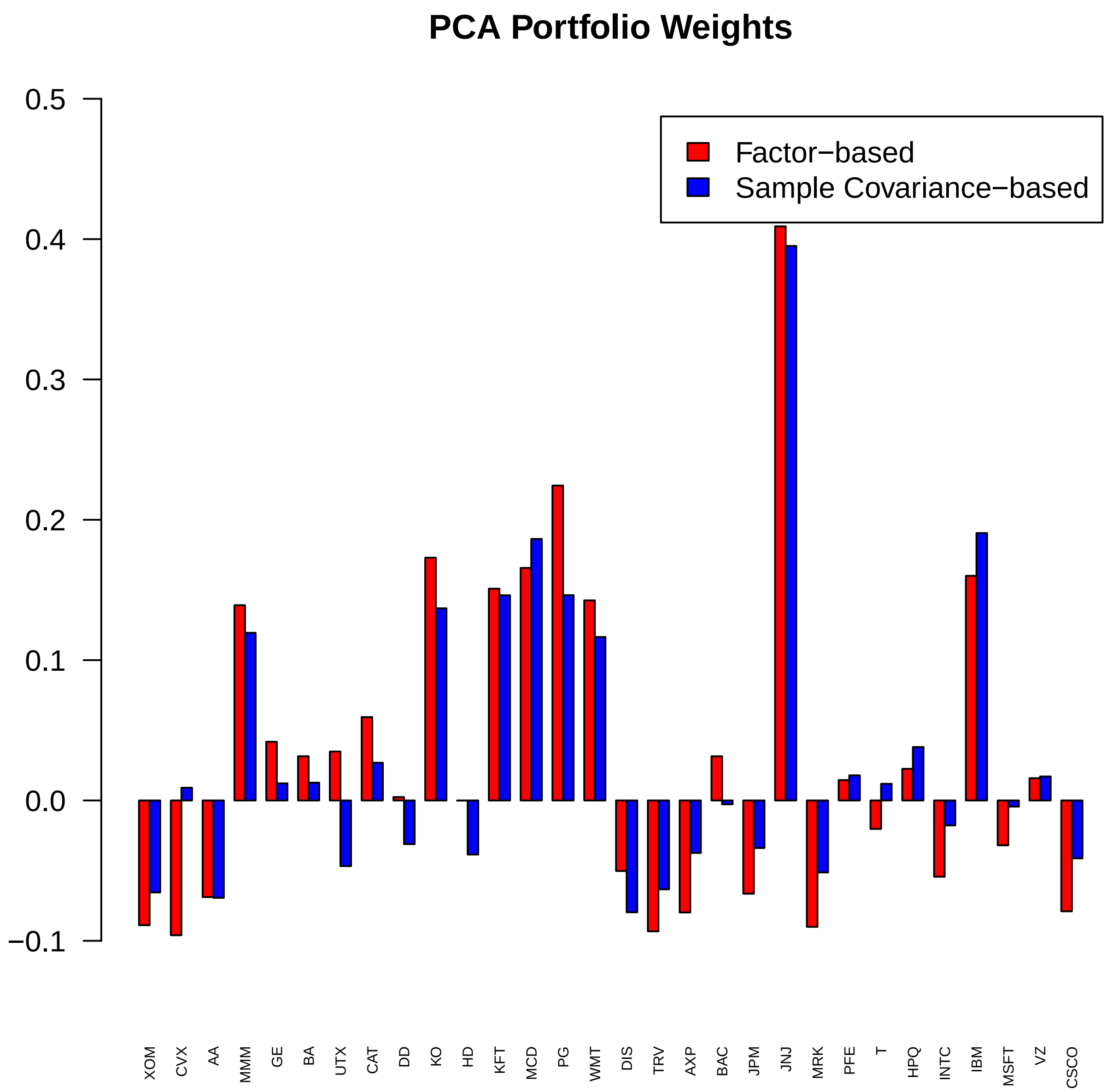

Figure 7: Portfolio Weights for PCA

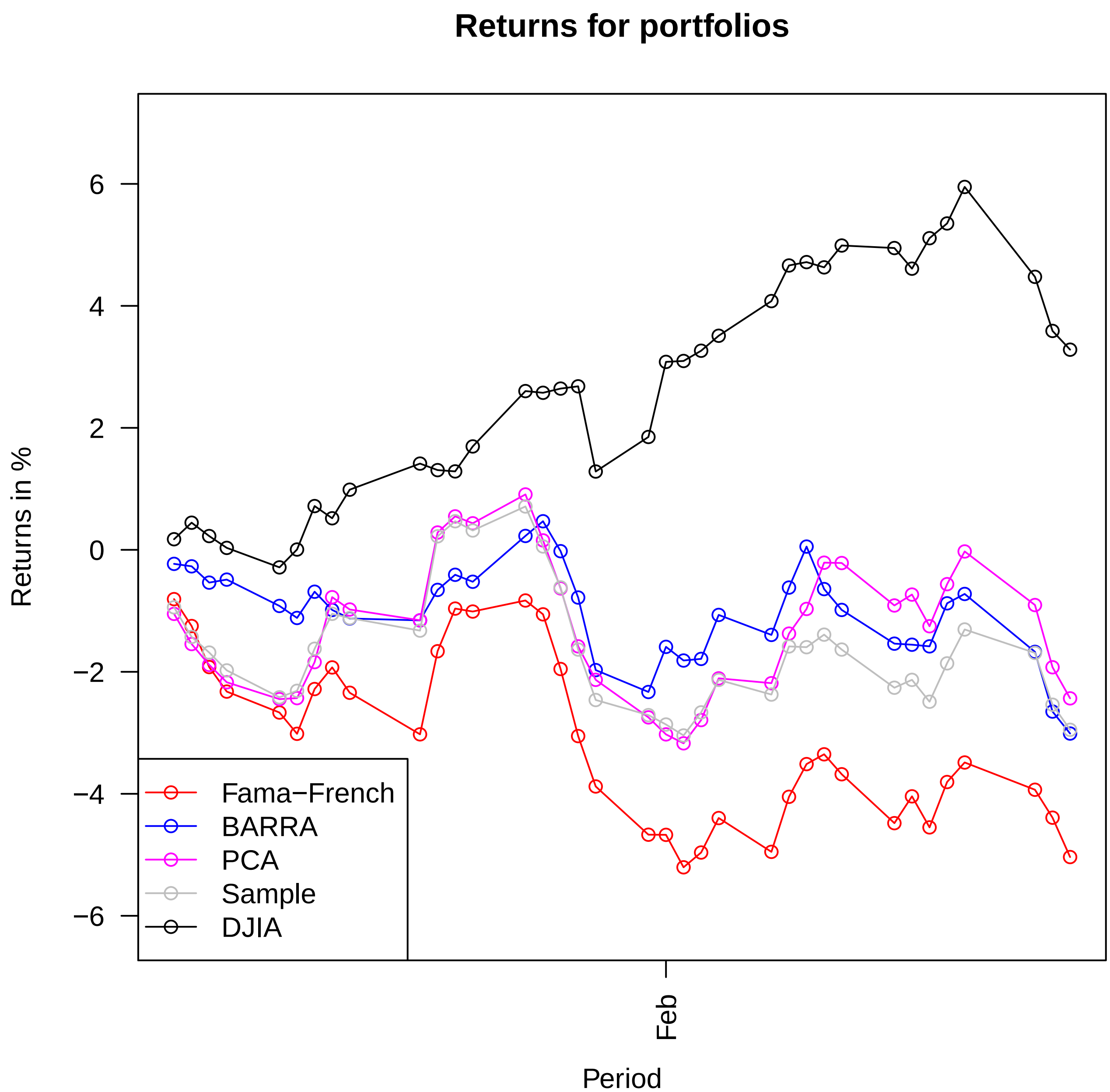

Figure 8: Returns of Portfolios

References

Fama, E. and K. French (1995): "Size and book-to-market factors in earnings and returns," *Journal of Finance*, 50, 131–155.

Litner, J. (1965): "Security prices, risk and maximal gains from diversification," *Journal of Finance*, 20, 587–615.

Markowitz, H. (1952): "Portfolio Selection," *The Journal of Finance*, 7, 77–91.

Sharpe, W. (1964): "Capital asset prices: A theory of market equilibrium under conditions of risk," *Journal of finance*, 19, 425–442.

Treynor, J. (1962): "Toward a theory of market value of risky assets," *Unpublished manuscript*, 15–22.

Zivot, E. and J. Wang (2006): *Modeling financial time series with S-PLUS*, Springer Verlag.

A R Script

```r
#######################################################
# Lucas Perin
# 2011-02-18
# Factor models and PCA
#######################################################

library(robust)
library(zoo)
library(fEcofin)

# Load the data
ff_data = read.zoo("FFdaily.csv", header=T, sep=",", format="%Y%m%d")
ret_data = read.zoo("djiaPrices.csv", header=T, sep=",")

###############
# Fama-French #
###############

# Drop the extra dates in the FF file
ff = window(ff_data, start=as.Date("2006-01-01"))
ff = ff[-1,]

# Calculate the returns
ret = 100*diff(ret_data)/lag(ret_data, -1) - ff$RF

# Is there an easier way to re-sort this big thing?
returns = cbind(ret$XOM, ret$CVX, ret$AA, ret$MMM, ret$GE, ret$BA,
    ret$UTX, ret$CAT, ret$DD, ret$KO, ret$HD, ret$KFT, ret$MCD, ret$PG,
    ret$WMT, ret$DIS, ret$TRV, ret$AXP, ret$BAC, ret$JPM, ret$JNJ, ret$MRK,
    ret$PFE, ret$T, ret$HPQ, ret$INTC, ret$IBM, ret$MSFT, ret$VZ, ret$CSCO)
```

```r

# Remove the risk-free, as we used it inside ret
factors = ff[,-4]

# find the betas
fit = lm(returns ~ factors)
slmfit = summary(fit)
cor.samp <- cor(t(returns))

# Calculate the r2
rsq <- as.numeric(sapply(X = slmfit, FUN = "[", i = 8))

# Calculate the errors
rerrv <- as.numeric(sapply(X = slmfit, FUN = "[", i = 6))^2

# with the betas and the variance matrices, calculate the covariance matrix
beta.hat <- coef(fit)[-1,]
Sigma.F <- var(factors)
Sigma.eps <- diag(rerrv)
cov.mat <- t(beta.hat) %*% Sigma.F %*% beta.hat + Sigma.eps

# Calculate the minimum-variance portfolio
w.ff = solve(cov.mat)%*%rep(1,nrow(cov.mat))
w.ff = w.ff/sum(w.ff)

write.table(cov.mat, file="cov-ff.dat", row.names=F, col.names=F)
write.table(t(w.ff), file="wmin-ff.dat", row.names=F, col.names=F)
write.table(t(rsq), file="rsq-ff.dat", row.names=F, col.names=F)

#############################
# Barra-type industry model #
#############################
```

```
# Prepare the industry data matrix
n.stocks = ncol(returns) # should be 30 :)

# Create several vectors with zeros for the six industry dummies
tech.dum = health.dum = fin.dum = cons.dum = ind.dum = mat.dum = matrix(0,n.stocks,1)

# Could probably be automated, but worth it?
mat.dum[1:3] = 1
ind.dum[4:9] = 1
cons.dum [10:16] = 1
fin.dum [17:20] = 1
health.dum [21:23] = 1
tech.dum [24:30] = 1

# Assemble the industry matrix
B.ind = cbind(mat.dum, ind.dum, cons.dum, fin.dum, health.dum, tech.dum)
dimnames(B.ind) = list(colnames(returns), c("Materials", "Industrial", "Consumer", "Fi
# If I was programming in C, ASSERT(sum(B) == n.stocks)

# Solve it Zivot-style
# (have to transpose the returns matrix for the matrix algebra)
returnsT = t(returns) # I kinda have some memory

# Multivariate OLS regression
F.hat = solve(crossprod(B.ind))%*%t(B.ind)%*%returnsT

# Residuals
E.hat = returnsT - B.ind%*%F.hat

# Residual variances
diagD.hat = apply(E.hat, 1, var)
```

```
Dinv.hat = diag(diagD.hat^(-1))

# FGLS
H = solve(t(B.ind)%*%Dinv.hat%*%B.ind)%*%t(B.ind)%*%Dinv.hat
F.hat = H%*%returnsT # would have saved me much aggravation if I just transposed retur
F.hat = t(F.hat)

E.hat2 = returnsT - B.ind %*% t(F.hat)

# Calculate the r-squared
ybar = rowMeans(returnsT)
SST = rowSums((returnsT - ybar)^2)
SSR = rowSums(E.hat2^2)
rsq.ind = 1-SSR/SST

# And the covariance matrix
cov.ind = B.ind%*%var(F.hat)%*%t(B.ind) + diag(diagD.hat)

# Calculate portfolio weights
w.ind = solve(cov.ind)%*%rep(1,nrow(cov.ind))
w.ind = w.ind/sum(w.ind)

w.gmin.sample = solve(var(returns))%*%rep(1,nrow(cov.ind))
w.gmin.sample = w.gmin.sample/sum(w.gmin.sample)

write.table(cov.ind, file="cov-ind.dat", row.names=F, col.names=F)
write.table(t(w.ind), file="wmin-ind.dat", row.names=F, col.names=F)
write.table(t(rsq.ind), file="rsq-ind.dat", row.names=F, col.names=F)


#####################
# 5-factor PCA model #
```

```
#####################

pcaEq = prcomp(returns)
tot.var = sum(pcaEq$sdev^2)

# Five factors
k = 5
x = as.matrix(returns)
n = ncol(x)
m = nrow(x)

# Center
xc = t(t(x) - colMeans(x))

# Betas
B = t(pcaEq$rotation[,1:k, drop=F])
f = x %*% pcaEq$rotation[, 1:k, drop=F]
tmp = x - f %*% B
alpha = colMeans(tmp)
tmp = t(t(tmp) - alpha)

# RSq
r2 = 1 - colSums(tmp^2)/colSums(xc^2)

# Covariance Matrix
Sigma = t(B) %*% var(f) %*% B
diag(Sigma) = diag(Sigma) + colSums(tmp^2)/(m-k-1)

# Weights
w.pca = solve(Sigma)%*%rep(1,nrow(Sigma))
w.pca = w.pca/sum(w.pca)

```

```
write.table(Sigma, file="cov-pca.dat", row.names=F, col.names=F)
write.table(t(w.pca), file="wmin-pca.dat", row.names=F, col.names=F)
write.table(t(r2), file="rsq-pca.dat", row.names=F, col.names=F)

################################################################
# For this version, graphs were done at the end in order to
# keep the line numbers in the report intact. Sorry. I would
# normally have the graphs in the appropriate sections of the
# code.
#
# I think this was a great assignment, so I'm probably going to
# reorganize this file when I have time and keep my tutorial
# up-to-date.
################################################################

# Scree plot
pdf("scree.pdf")
barplot((pcaEq$sdev^2)[1:10],col=4,ylim=c(0,max(pcaEq$sdev^2)*1.1),
    las=2,names.arg=paste("PC",1:10,sep=""),
    ylab="variance", main="DJIA scree plot")

text(x=seq(0.7,len=10,by=1.2),y=(pcaEq$sdev^2)[1:10]+2,
    round(100*cumsum((pcaEq$sdev^2)[1:10]/tot.var),1),cex=0.75)
dev.off()

# Box plot
pdf("R2.pdf")
boxplot(x=list(rsq,rsq.ind,r2), ylab="R Squared",
    names=c("Fama-French","BARRA","PCA"),pch=5)
dev.off()

# Sadly, I know how to read groups.csv to get the below
```

```
# but it takes longer than I can
names(returns) = c("XOM", "CVX", "AA", "MMM", "GE",
    "BA", "UTX", "CAT", "DD", "KO", "HD", "KFT", "MCD",
    "PG", "WMT", "DIS", "TRV", "AXP", "BAC", "JPM","JNJ",
    "MRK", "PFE", "T", "HPQ", "INTC", "IBM", "MSFT",
    "VZ", "CSCO")


# Betas and R-Squared for Fama-French
beta1 = coef(fit)[2,]
beta2 = coef(fit)[3,]
beta3 = coef(fit)[4,]

pdf("FF.pdf",width=5)
par(mfrow=c(4,1))
barplot(beta1,main="Fama-French Betas for Market",col=4, las=2,
  names.arg=colnames(returns),space=0.5)
abline(h=0)
barplot(beta2,main="Fama-French Betas for SML",col=4, las=2,
  names.arg=colnames(returns),space=0.5)
abline(h=0)
barplot(beta3,main="Fama-French Betas for HML",col=4, las=2,
  names.arg=colnames(returns),space=0.5)
abline(h=0)
barplot(rsq,names=names(beta1),main="Fama-French R-squared values", col=4, las=2,
  names.arg=colnames(returns),space=0.5)
dev.off()

# Fix the names of the portfolios
library(xtable)
rownames(w.ind) = rownames(w.pca) = rownames(w.ff) = names(returns)
rownames(w.gmin.sample) = names(returns)
```

```r

# Printing the weights of the minimum portfolios for use in LaTeX
print(xtable(w.ff))
print(xtable(w.ind))
print(xtable(w.pca))

# Required graphs of minimum portfolios:
pdf("ffport.pdf")
barplot(rbind(t(w.ff),t(w.gmin.sample)),col=c(2,4),
    legend = c("Factor-based", "Sample Covariance-based"),
    beside=T,names.arg=rownames(w.gmin.sample),cex.names=0.5,
    las=2, ylim=c(-0.15,0.5), main="Fama-French Portfolio Weights")
#text(x=seq(1,len=30,by=3),y=w.ff + (0.01 * sign(w.ff)),
    # round(w.ff,2),cex=0.75)
#text(x=seq(3,len=30,by=3),y=w.gmin.sample + (0.01 * sign(w.gmin.sample)),
#   round(w.gmin.sample,2),cex=0.75)
dev.off()

pdf("indport.pdf")
barplot(rbind(t(w.ind),t(w.gmin.sample)),col=c(2,4),
    legend = c("Factor-based", "Sample Covariance-based"),
    beside=T,names.arg=rownames(w.gmin.sample),cex.names=0.5,
    las=2, ylim=c(-0.15,0.5), main="BARRA-type Portfolio Weights")
# text(x=seq(1,len=30,by=3),y=w.ind + (0.01 * sign(w.ind)),
#   round(w.ind,2),cex=0.75)
# text(x=seq(3,len=30,by=3),y=w.gmin.sample + (0.01 * sign(w.gmin.sample)),
#   round(w.gmin.sample,2),cex=0.75)
dev.off()

pdf("pcaport.pdf")
barplot(rbind(t(w.pca),t(w.gmin.sample)),col=c(2,4),
    legend = c("Factor-based", "Sample Covariance-based"),
```

```
    beside=T,names.arg=rownames(w.gmin.sample),cex.names=0.5,
    las=2, ylim=c(-0.15,0.5), main="PCA Portfolio Weights")
 #text(x=seq(1,len=30,by=3),y=w.pca + (0.01 * sign(w.pca)),
#   round(w.pca,2),cex=0.75)
# text(x=seq(3,len=30,by=3),y=w.gmin.sample + (0.01 * sign(w.gmin.sample)),
#   round(w.gmin.sample,2),cex=0.75)
dev.off()

library(fImport)

newPrices = yahooSeries(names(returns),from="2011-01-01",to="2011-02-25")
djia = yahooSeries("^DJI",from="2011-01-01",to="2011-02-25")
djia = djia[,4]
newClosePrices = newPrices[,seq(4,len=30,by=6)]
djia = sort(djia)
newClosePrices = sort(newClosePrices)
newReturns = 100*diff(newClosePrices)/lag(newClosePrices,-1)
djiaReturns = 100*diff(djia)/lag(djia,-1)
newReturns = window(newReturns, start=as.Date("2011-01-04"), end=as.Date("2011-02-24")
djiaReturns = window(djiaReturns, start=as.Date("2011-01-04"), end=as.Date("2011-02-24
newReturns = sort(newReturns)
djiaReturns = sort(djiaReturns)

nr = apply(newReturns, 2, cumsum)
dd = apply(djiaReturns, 2, cumsum)
varrn = apply(newReturns, 2, var)
vardd = apply(djiaReturns, 2, var)

fun1 = nr %*% w.ff
fun2 = nr %*% w.ind
fun3 = nr %*% w.pca
fun4 = nr %*% w.gmin.sample
```

```r
fun5 = dd
fun = cbind(fun1, fun2, fun3, fun4, fun5)
names(fun) = c("Fama-French", "BARRA", "PCA", "Sample", "DJIA")

pdf("FinalReturns.pdf")
plot(time(newReturns), fun[,1] , type="o", col=2,
    main="Returns for portfolios", ylim=c(min(fun)-1,max(fun)+1),
    xlab="Period", las=2, ylab="Returns in %")
lines(time(newReturns), fun[,2], type="o", col=4, las=2)
lines(time(newReturns), fun[,3], type="o", col=6, las=2)
lines(time(newReturns), fun[,4], type="o", col=8, las=2)
lines(time(newReturns), fun[,5], type="o", col=9, las=2)
legend("bottomleft", names(fun),
    col=c(2,4,6,8,9), lty=1, pch=21)
dev.off()
```

Security	Weight
XOM	-0.07
CVX	-0.10
AA	-0.08
MMM	0.09
GE	0.01
BA	-0.01
UTX	0.02
CAT	-0.05
DD	-0.05
KO	0.17
HD	0.02
KFT	0.15
MCD	0.16
PG	0.22
WMT	0.15
DIS	-0.03
TRV	-0.01
AXP	-0.06
BAC	-0.02
JPM	-0.01
JNJ	0.33
MRK	0.02
PFE	0.07
T	0.04
HPQ	-0.00
INTC	-0.05
IBM	0.09
MSFT	-0.03
VZ	0.07
CSCO	-0.04

Table 1: Portfolio weights for Fama-French factor model

Security	Weight
XOM	-0.04
CVX	-0.06
AA	-0.02
MMM	0.01
GE	0.00
BA	0.00
UTX	0.01
CAT	0.00
DD	0.01
KO	0.14
HD	0.06
KFT	0.11
MCD	0.13
PG	0.18
WMT	0.12
DIS	0.08
TRV	-0.03
AXP	-0.05
BAC	-0.02
JPM	-0.06
JNJ	0.11
MRK	0.07
PFE	0.10
T	0.03
HPQ	0.02
INTC	0.02
IBM	0.03
MSFT	0.02
VZ	0.03
CSCO	0.02

Table 2: Portfolio weights for BARRA-type factor model

Security	Weight
XOM	-0.09
CVX	-0.10
AA	-0.07
MMM	0.14
GE	0.04
BA	0.03
UTX	0.03
CAT	0.06
DD	0.00
KO	0.17
HD	-0.00
KFT	0.15
MCD	0.17
PG	0.22
WMT	0.14
DIS	-0.05
TRV	-0.09
AXP	-0.08
BAC	0.03
JPM	-0.07
JNJ	0.41
MRK	-0.09
PFE	0.01
T	-0.02
HPQ	0.02
INTC	-0.05
IBM	0.16
MSFT	-0.03
VZ	0.02
CSCO	-0.08

Table 3: Portfolio weights for PCA factor model