Financial Risk Models in R: Factor Models for Asset Returns and Interest Rate Models

Scottish Financial Risk Academy, March 15, 2011

Eric Zivot
Robert Richards Chaired Professor of Economics
Adjunct Professor, Departments of Applied Mathematics, Finance and Statistics
University of Washington
BlackRock Alternative Advisors, Seattle WA

Workshop Overview

- About Me
- Brief Introduction to R in Finance
- Factor Models for Asset Returns
- Estimation of Factor Models in R
- Factor Model Risk Analysis
- Factor Model Risk Analysis in R
- Modeling Interest Rates in R (brief discussion)

About Me

- Robert Richards Chaired Professor of Economics at the University of Washington
  - Adjunct Professor of Applied Mathematics, Finance, and Statistics
- Co-Director of MS Program in Computational Finance and Risk Management at UW
- BS in Economics and Statistics from UC Berkeley
- PhD in Economics from Yale University

About Me: R and Finance

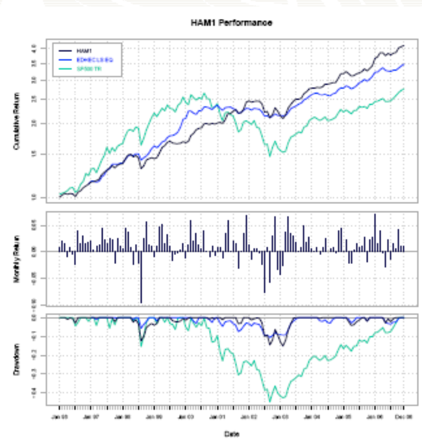
- 12 years programming in S language
- 8 years Research scientist and consultant for Mathsoft/Insightful (makers of SPLUS)
- Co-developed S+FinMetrics for Insightful
- Co-authored *Modeling Financial Time Series with SPLUS*, Springer
- 2 ½ years developing FoHF factor model based risk management system in R for BlackRock Alternative Advisors

Brief Introduction to R in Finance

- R is a language and environment for statistical computing and graphics
- R is based on the S language originally developed by John Chambers and colleagues at AT&T Bell Labs in the late 1970s and early 1980s
- R (sometimes called\GNU S") is free open source software licensed under the GNU general public license (GPL 2)
- R development was initiated by Robert Gentleman and Ross Ihaka at the University of Auckland, New Zealand
- R is formally known as The R Project for Statistical Computing
- www.r-project.org

What is R great at?

- Data analysis
- Data Manipulation
- Data Visualization
- Statistical Modeling and Programming



Plot from the PerformanceAnalytics package

S Language Implementations

- R is the most recent and full-featured implementation of the S language
- Original S - AT & T Bell Labs
- S-PLUS (S plus a GUI)
- Statistical Sciences, Inc.y
  - Mathsoft, Inc., Insightful, Inc., Tibco, Inc.
- R - The R Project for Statistical Computing

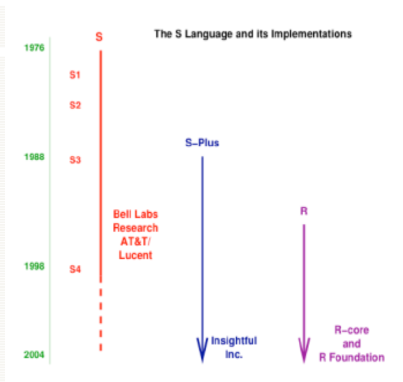
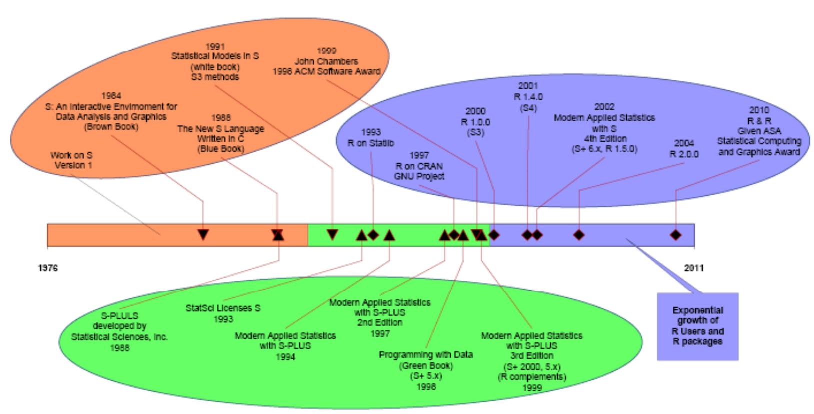


Figure from *The History of S and R, John Chambers, 2006*

R Timeline



Recognition for Software Excellence

Association for Computing Machinery

John Chambers received the 1998 ACM Software System Award

> *Dr. Chambers' work will forever alter the way people analyze, visualize, and manipulate data*

American Statistical Association

Robert Gentleman and Ross Ihaka received the 2009 ASA Statistical Computing and Graphics Award

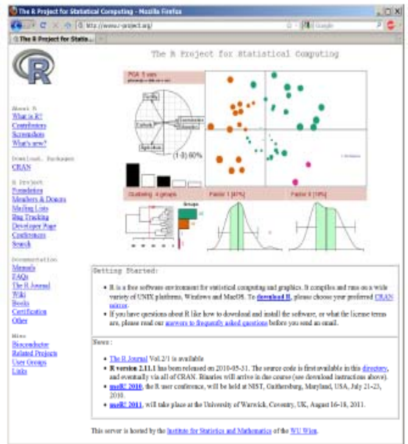
> *In recognition for their work in initiating the R Project for Statistical Computing*

The R Foundation

- The R Foundation is the non-profit organization located in Vienna, Austria which is responsible for developing and maintaining R
  - Hold and administer the copyright of R software and documentation
  - Support continued development of R
  - Organize meetings and conferences related to statistical computing

R Homepage

- http://www.r-project.org
- List of CRAN mirror sites
- Manuals
- FAQs
- Mailing Lists
- Links



CRAN – Comprehensive R Archive Network

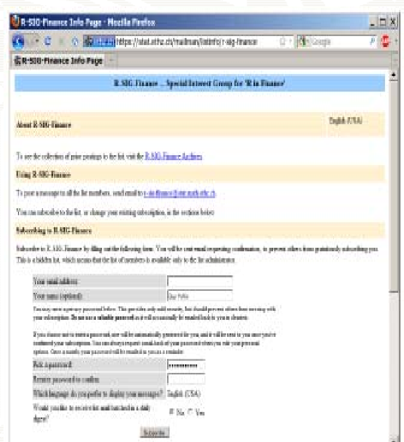
- http://cran.fhcrc.org
- CRAN Mirrors
  - About 75 sites worldwide
  - About 16 sites in US
- R Binaries
- R Packages
- R Sources
- Task Views

CRAN Task Views

- Organizes 2600+ R packages by application
- Relevant tasks for financial applications:
  - Finance
  - Time Series
  - Econometrics
  - Optimization
  - Machine Learning

R-Sig-Finance

- https://stat.ethz.ch/mailman/listinfo.r-sig-finance
- Nerve center of the R finance community
- Daily must read
- Exclusively for Finance-specific questions, not general R questions



Other Useful R Sites

- R Seek R specific search site:
  - http://www.rseek.org/
- R Bloggers Aggregation of about 100 R blogs:
  - http://www.r-bloggers.com
- Stack Overflow Excellent developer Q&A forum
  - http://stackoverflow.com
- R Graph Gallery Examples of many possible R graphs
  - http://addictedtor.free.fr/graphiques
- Blog from David Smith of Revolution
  - http://blog.revolutionanalytics.com
- Inside-R R community site by Revolution Analytics
  - http://www.inside-r.org

Estimation of Factor Models in R

- Data for examples
- Estimation of macroeconomic factor model
  Sharpe's single index model
- Estimation of fundamental factor model
  - BARRA-type industry model
- Estimation of statistical factor model
  - Principal components

Set Options and Load Packages

```
# set output options
> options(width = 70, digits=4)

# load required packages
> library(ellipse)                 # functions plotting
                                   # correlation matrices
> library(fEcofin)                 # various economic and
                                   # financial data sets
> library(PerformanceAnalytics)    # performance and risk
                                   # analysis functions
> library(zoo)                     # time series objects
                                   # and utility functions
```


Berndt Data

```
# load Berndt investment data from fEcofin package
> data(berndtInvest)
> class(berndtInvest)
[1] "data.frame"

>  colnames(berndtInvest)
 [1] "X.Y..m..d" "CITCRP"    "CONED"     "CONTIL"
 [5] "DATGEN"    "DEC"       "DELTA"     "GENMIL"
 [9] "GERBER"    "IBM"       "MARKET"    "MOBIL"
[13] "PANAM"     "PSNH"      "TANDY"     "TEXACO"
[17] "WEYER"     "RKFREE"

# create data frame with dates as rownames
> berndt.df = berndtInvest[, -1]
> rownames(berndt.df) = as.character(berndtInvest[, 1])
```

Berndt Data

```
> head(berndt.df, n=3)
           CITCRP  CONED CONTIL DATGEN    DEC
1978-01-01 -0.115 -0.079 -0.129 -0.084 -0.100
1978-02-01 -0.019 -0.003  0.037 -0.097 -0.063
1978-03-01  0.059  0.022  0.003  0.063  0.010

> tail(berndt.df, n=3)
           CITCRP  CONED CONTIL DATGEN    DEC
1987-10-01 -0.282 -0.017 -0.372 -0.342 -0.281
1987-11-01 -0.136 -0.012 -0.148 -0.075 -0.127
1987-12-01  0.064 -0.006  0.050  0.181  0.134
```


Sharpe's Single Index Model

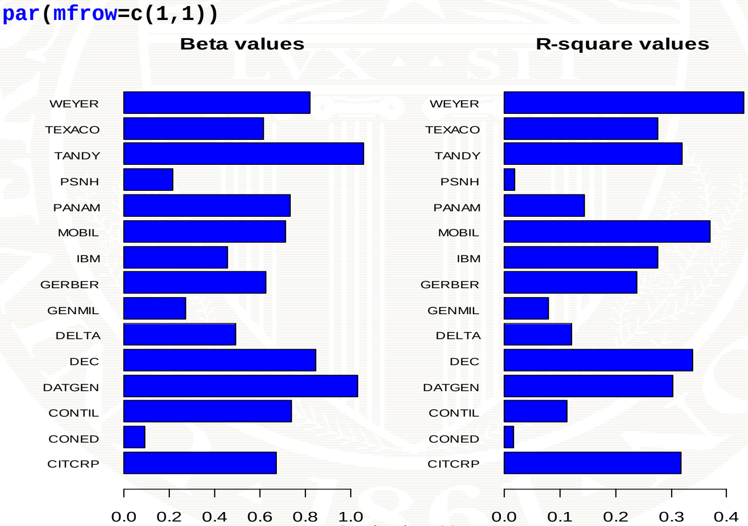
```
> returns.mat = as.matrix(berndt.df[, c(-10, -17)])
> market.mat = as.matrix(berndt.df[,10, drop=F])
> n.obs = nrow(returns.mat)
> X.mat = cbind(rep(1,n.obs),market.mat)
> colnames(X.mat)[1] = "intercept"
> XX.mat = crossprod(X.mat)

# multivariate least squares
> G.hat = solve(XX.mat)%*%crossprod(X.mat,returns.mat)
> beta.hat = G.hat[2,]
> E.hat = returns.mat - X.mat%*%G.hat
> diagD.hat = diag(crossprod(E.hat)/(n.obs-2))

# compute R2 values from multivariate regression
> sumSquares = apply(returns.mat, 2,
+              function(x) {sum( (x - mean(x))^2 )})
> R.square = 1 - (n.obs-2)*diagD.hat/sumSquares
```

Estimation Results

```
> cbind(beta.hat, diagD.hat, R.square)
        beta.hat diagD.hat R.square
CITCRP  0.66778  0.004511  0.31777
CONED   0.09102  0.002510  0.01532
CONTIL  0.73836  0.020334  0.11216
DATGEN  1.02816  0.011423  0.30363
DEC     0.84305  0.006564  0.33783
DELTA   0.48946  0.008152  0.12163
GENMIL  0.26776  0.003928  0.07919
GERBER  0.62481  0.005924  0.23694
IBM     0.45302  0.002546  0.27523
MOBIL   0.71352  0.004105  0.36882
PANAM   0.73014  0.015008  0.14337
PSNH    0.21263  0.011872  0.01763
TANDY   1.05549  0.011162  0.31986
TEXACO  0.61328  0.004634  0.27661
WEYER   0.81687  0.004154  0.43083
```



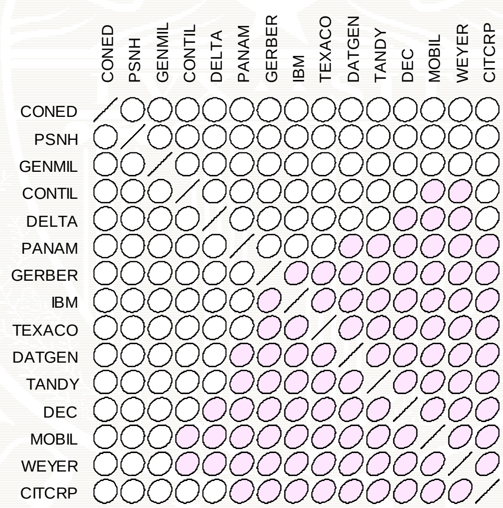
```
> par(mfrow=c(1,2))
> barplot(beta.hat, horiz=T, main="Beta values", col="blue",
+         cex.names = 0.75, las=1)
> barplot(R.square, horiz=T, main="R-square values", col="blue",
+         cex.names = 0.75, las=1)
> par(mfrow=c(1,1))
```



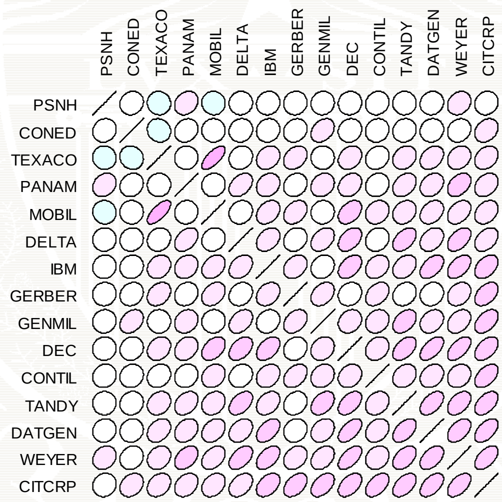
Compute Single Index Covariance

```
# compute single index model covariance/correlation
> cov.si =
as.numeric(var(market.mat))*beta.hat%*%t(beta.hat)
+         diag(diagD.hat)
> cor.si = cov2cor(cov.si)


# plot correlation matrix using plotcorr() from
# package ellipse
> ord <- order(cor.si[1,])
> ordered.cor.si <- cor.si[ord, ord]
> plotcorr(ordered.cor.si,
+          col=cm.colors(11)[5*ordered.cor.si + 6])
```


Single Index Correlation Matrix

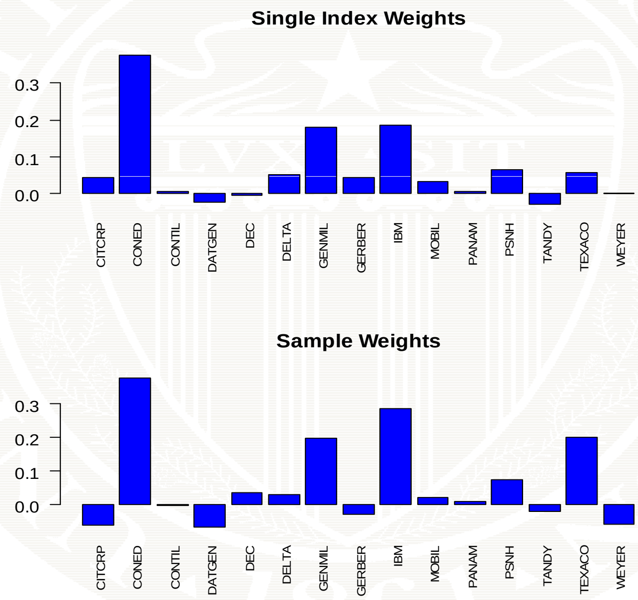


Sample Correlation Matrix



Minimum Variance Portfolio

```
# use single index covariance
> w.gmin.si = solve(cov.si)%*%rep(1,nrow(cov.si))
> w.gmin.si = w.gmin.si/sum(w.gmin.si)
> colnames(w.gmin.si) = "single.index"

# use sample covariance
> w.gmin.sample =
+         solve(var(returns.mat))%*%rep(1,nrow(cov.si))
> w.gmin.sample = w.gmin.sample/sum(w.gmin.sample)
> colnames(w.gmin.sample) = "sample"
```



Estimate Single Index Model in Loop

```
> asset.names = colnames(returns.mat)
> asset.names
 [1] "CITCRP" "CONED"  "CONTIL" "DATGEN" "DEC"
 [6] "DELTA"  "GENMIL" "GERBER" "IBM"    "MOBIL"
[11] "PANAM"  "PSNH"   "TANDY"  "TEXACO" "WEYER"

# initialize list object to hold regression objects
> reg.list = list()
# loop over all assets and estimate regression
> for (i in asset.names) {
+  reg.df = berndt.df[, c(i, "MARKET")]
+  si.formula = as.formula(paste(i,"~",
+                             "MARKET", sep=" "))
+  reg.list[[i]] = lm(si.formula, data=reg.df)
+ }
```

List Output

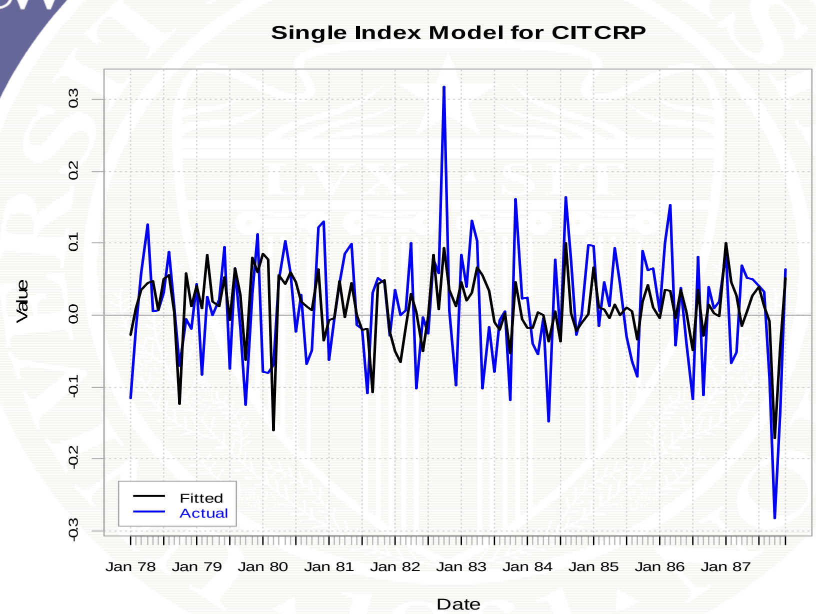
```
> names(reg.list)
 [1] "CITCRP" "CONED"  "CONTIL" "DATGEN" "DEC"
 [6] "DELTA"  "GENMIL" "GERBER" "IBM"    "MOBIL"
[11] "PANAM"  "PSNH"   "TANDY"  "TEXACO" "WEYER"
> class(reg.list$CITCRP)
[1] "lm"
> reg.list$CITCRP

Call:
lm(formula = si.formula, data = reg.df)

Coefficients:
(Intercept)       MARKET
    0.00252      0.66778
```


Regression Summary Output

```
> summary(reg.list$CITCRP)

Call:
lm(formula = si.formula, data = reg.df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.16432 -0.05012  0.00226  0.04351  0.22467

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.00252    0.00626    0.40     0.69
MARKET       0.66778    0.09007    7.41  2.0e-11 ***

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

Residual standard error: 0.0672 on 118 degrees of freedom
Multiple R-squared: 0.318,      Adjusted R-squared: 0.312
F-statistic:   55 on 1 and 118 DF,  p-value: 2.03e-11
```

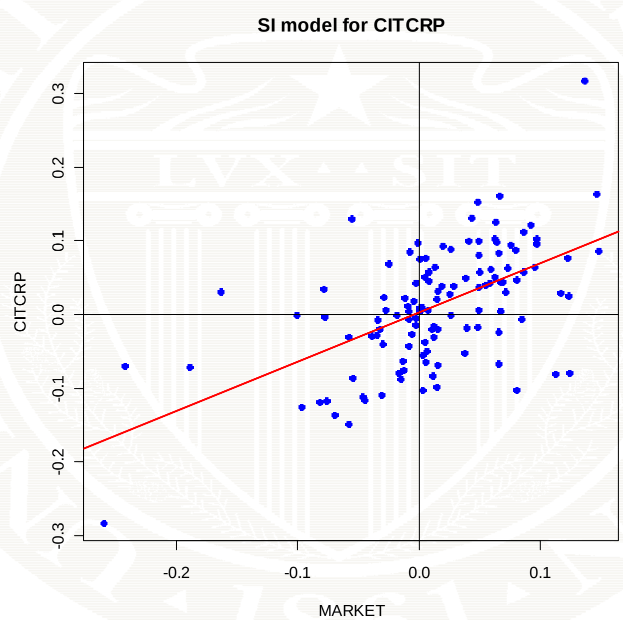
Plot Actual and Fitted Values: Time Series

```
# use chart.TimeSeries() function from
# PerformanceAnalytics package

> dataToPlot = cbind(fitted(reg.list$CITCRP),
+                    berndt.df$CITCRP)
> colnames(dataToPlot) = c("Fitted","Actual")
> chart.TimeSeries(dataToPlot,
+       main="Single Index Model for CITCRP",
+       colorset=c("black","blue"),
+       legend.loc="bottomleft")
```

Plot Actual and Fitted Values: Cross Section

```
> plot(berndt.df$MARKET, berndt.df$CITCRP,
       main="SI model for CITCRP",
+      type="p", pch=16, col="blue",
+      xlab="MARKET", ylab="CITCRP")
> abline(h=0, v=0)
> abline(reg.list$CITCRP, lwd=2, col="red")
```

Extract Regression Information 1

```
## extract beta values, residual sd's and R2's from list
## of regression objects by brute force loop
> reg.vals = matrix(0, length(asset.names), 3)
> rownames(reg.vals) = asset.names
> colnames(reg.vals) = c("beta", "residual.sd",
+                        "r.square")
> for (i in names(reg.list)) {
+     tmp.fit = reg.list[[i]]
+     tmp.summary = summary(tmp.fit)
+     reg.vals[i, "beta"] = coef(tmp.fit)[2]
+     reg.vals[i, "residual.sd"] = tmp.summary$sigma
+     reg.vals[i, "r.square"] = tmp.summary$r.squared
+}
```


Regression Results

```
> reg.vals
          beta residual.sd r.square
CITCRP 0.66778     0.06716  0.31777
CONED  0.09102     0.05010  0.01532
CONTIL 0.73836     0.14260  0.11216
DATGEN 1.02816     0.10688  0.30363
DEC    0.84305     0.08102  0.33783
DELTA  0.48946     0.09029  0.12163
GENMIL 0.26776     0.06268  0.07919
GERBER 0.62481     0.07697  0.23694
IBM    0.45302     0.05046  0.27523
MOBIL  0.71352     0.06407  0.36882
PANAM  0.73014     0.12251  0.14337
PSNH   0.21263     0.10896  0.01763
TANDY  1.05549     0.10565  0.31986
TEXACO 0.61328     0.06808  0.27661
WEYER  0.81687     0.06445  0.43083
```

Extract Regression Information 2

```
# alternatively use R apply function for list
# objects - lapply or sapply
extractRegVals = function(x) {
# x is an lm object
 beta.val = coef(x)[2]
 residual.sd.val = summary(x)$sigma
 r2.val = summary(x)$r.squared
 ret.vals = c(beta.val, residual.sd.val, r2.val)
 names(ret.vals) = c("beta", "residual.sd",
                     "r.square")
 return(ret.vals)
}
> reg.vals = sapply(reg.list, FUN=extractRegVals)
```


Regression Results

```
> t(reg.vals)
          beta residual.sd r.square
CITCRP 0.66778     0.06716  0.31777
CONED  0.09102     0.05010  0.01532
CONTIL 0.73836     0.14260  0.11216
DATGEN 1.02816     0.10688  0.30363
DEC    0.84305     0.08102  0.33783
DELTA  0.48946     0.09029  0.12163
GENMIL 0.26776     0.06268  0.07919
GERBER 0.62481     0.07697  0.23694
IBM    0.45302     0.05046  0.27523
MOBIL  0.71352     0.06407  0.36882
PANAM  0.73014     0.12251  0.14337
PSNH   0.21263     0.10896  0.01763
TANDY  1.05549     0.10565  0.31986
TEXACO 0.61328     0.06808  0.27661
WEYER  0.81687     0.06445  0.43083
```

Industry Factor Model

```
# create loading matrix B for industry factor model
> n.stocks = ncol(returns.mat)
> tech.dum = oil.dum = other.dum =
+            matrix(0,n.stocks,1)
> rownames(tech.dum) = rownames(oil.dum) =
+                     rownames(other.dum) = asset.names
> tech.dum[c(4,5,9,13),] = 1
> oil.dum[c(3,6,10,11,14),] = 1
> other.dum = 1 - tech.dum - oil.dum
> B.mat = cbind(tech.dum,oil.dum,other.dum)
> colnames(B.mat) = c("TECH","OIL","OTHER")
```


Factor Sensitivity Matrix

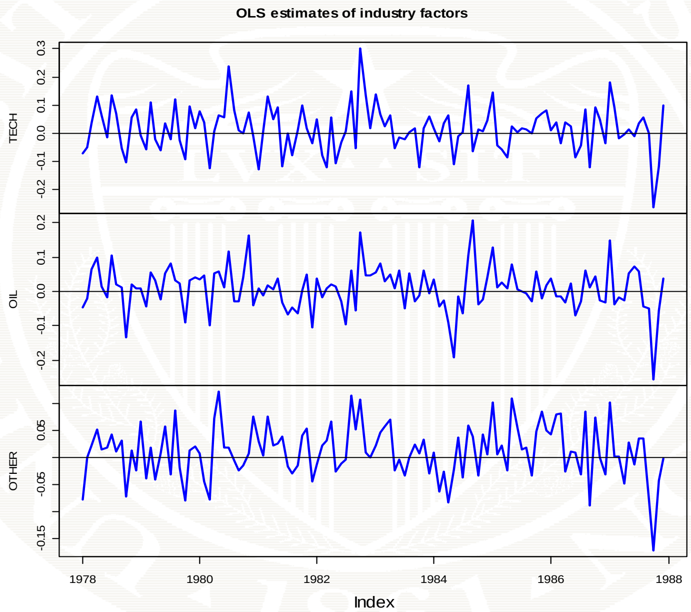
```
> B.mat
       TECH OIL OTHER
CITCRP    0   0     1
CONED     0   0     1
CONTIL    0   1     0
DATGEN    1   0     0
DEC       1   0     0
DELTA     0   1     0
GENMIL    0   0     1
GERBER    0   0     1
IBM       1   0     0
MOBIL     0   1     0
PANAM     0   1     0
PSNH      0   0     1
TANDY     1   0     0
TEXACO    0   1     0
WEYER     0   0     1
```

Multivariate Least Squares Estimation of Factor Returns

```
# returns.mat is T x N matrix, and fundamental factor
# model treats R as N x T.
> returns.mat = t(returns.mat)
# multivariate OLS regression to estimate K x T matrix
# of factor returns  (K=3)
> F.hat =
+ solve(crossprod(B.mat))%*%t(B.mat)%*%returns.mat

# rows of F.hat are time series of estimated industry
# factors
> F.hat
        1978-01-01 1978-02-01 1978-03-01 1978-04-01
TECH       -0.0720 -0.0517500     0.0335    0.13225
OIL        -0.0464 -0.0192000     0.0642    0.09920
OTHER      -0.0775 -0.0006667     0.0220    0.05133
```


Plot Industry Factors

```
# plot industry factors in separate panels - convert
# to zoo time series object for plotting with dates
> F.hat.zoo = zoo(t(F.hat), as.Date(colnames(F.hat)))
> head(F.hat.zoo, n=3)
TECH      OIL      OTHER
1978-01-01 -0.07200 -0.0464 -0.0775000
1978-02-01 -0.05175 -0.0192 -0.0006667
1978-03-01  0.03350  0.0642  0.0220000

# panel function to put horizontal lines at zero in each
panel
> my.panel <- function(...) {
+  lines(...)
+  abline(h=0)
+}
> plot(F.hat.zoo, main="OLS estimates of industry
+      factors“, panel=my.panel, lwd=2, col="blue")
```

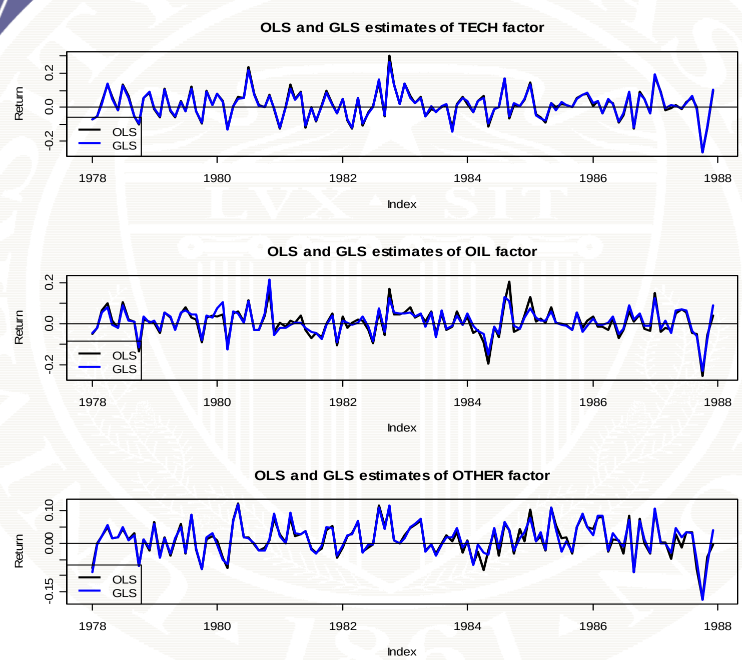


GLS Estimation of Factor Returns

```
# compute N x T matrix of industry factor model residuals
> E.hat = returns.mat - B.mat%*%F.hat
# compute residual variances from time series of errors
> diagD.hat = apply(E.hat, 1, var)
> Dinv.hat = diag(diagD.hat^(-1))

# multivariate FGLS regression to estimate K x T matrix
# of factor returns
> H.hat = solve(t(B.mat)%*%Dinv.hat%*%B.mat)
+         %*%t(B.mat)%*%Dinv.hat
> colnames(H.hat) = asset.names
# note: rows of H sum to one so are weights in factor
# mimicking portfolios
> F.hat.gls = H.hat%*%returns.mat
```

GLS Factor Weights

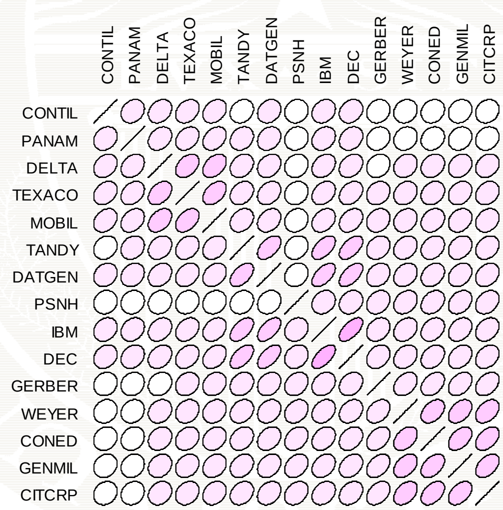
```
> t(H.hat)
          TECH    OIL    OTHER
CITCRP 0.0000 0.0000 0.19918
CONED  0.0000 0.0000 0.22024
CONTIL 0.0000 0.0961 0.00000
DATGEN 0.2197 0.0000 0.00000
DEC    0.3188 0.0000 0.00000
DELTA  0.0000 0.2233 0.00000
GENMIL 0.0000 0.0000 0.22967
GERBER 0.0000 0.0000 0.12697
IBM    0.2810 0.0000 0.00000
MOBIL  0.0000 0.2865 0.00000
PANAM  0.0000 0.1186 0.00000
PSNH   0.0000 0.0000 0.06683
TANDY  0.1806 0.0000 0.00000
TEXACO 0.0000 0.2756 0.00000
WEYER  0.0000 0.0000 0.15711
```

Industry Factor Model Covariance

```
# compute covariance and correlation matrices
> cov.ind = B.mat%*%var(t(F.hat.gls))%*%t(B.mat) +
+            diag(diagD.hat)
> cor.ind = cov2cor(cov.ind)
# plot correlations using plotcorr() from ellipse
# package
> rownames(cor.ind) = colnames(cor.ind)
> ord <- order(cor.ind[1,])
> ordered.cor.ind <- cor.ind[ord, ord]
> plotcorr(ordered.cor.ind,
+           col=cm.colors(11)[5*ordered.cor.ind + 6])
```

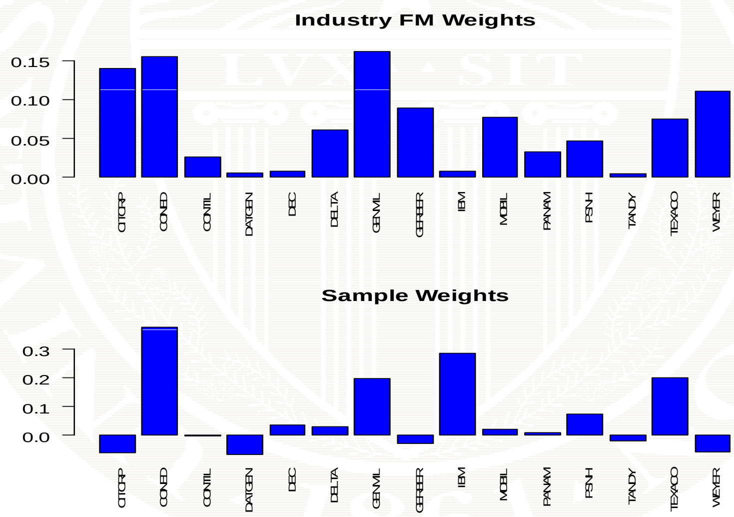

Industry Factor Model Correlations



Industry Factor Model Summary

```
> ind.fm.vals
       TECH OIL OTHER   fm.sd residual.sd r.square
CITCRP    0   0     1 0.07291     0.05468   0.4375
CONED     0   0     1 0.07092     0.05200   0.4624
CONTIL    0   1     0 0.13258     0.11807   0.2069
DATGEN    1   0     0 0.10646     0.07189   0.5439
DEC       1   0     0 0.09862     0.05968   0.6338
DELTA     0   1     0 0.09817     0.07747   0.3773
GENMIL    0   0     1 0.07013     0.05092   0.4728
GERBER    0   0     1 0.08376     0.06849   0.3315
IBM       1   0     0 0.10102     0.06356   0.6041
MOBIL     0   1     0 0.09118     0.06839   0.4374
PANAM     0   1     0 0.12222     0.10630   0.2435
PSNH      0   0     1 0.10601     0.09440   0.2069
TANDY     1   0     0 0.11159     0.07930   0.4950
TEXACO    0   1     0 0.09218     0.06972   0.4279
WEYER     0   0     1 0.07821     0.06157   0.3802
```


Global Minimum Variance Portfolios



Statistical Factor Model: Principal Components Method

```
# continue to use Berndt data
> returns.mat = as.matrix(berndt.df[, c(-10, -17)])
# use R princomp() function for principal component
# analysis
> pc.fit = princomp(returns.mat)

> class(pc.fit)
[1] "princomp"
> names(pc.fit)
[1] "sdev"     "loadings" "center"   "scale"    "n.obs"
[6] "scores"   "call"
```

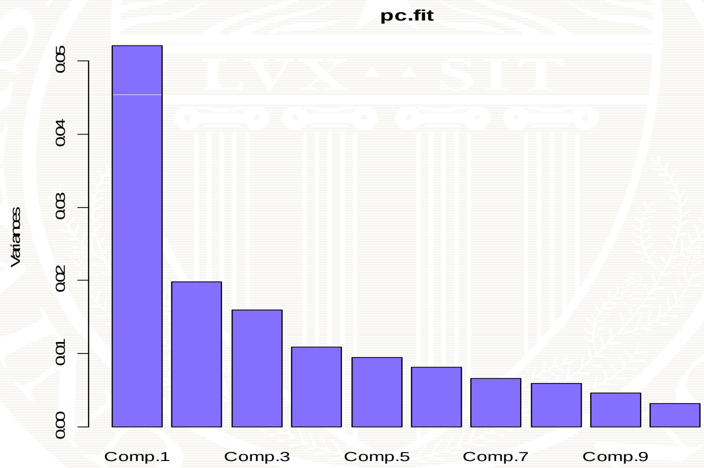
eigenvectors

principal components

Total Variance Contributions

```
> summary(pc.fit)
Importance of components:
                          Comp.1 Comp.2 Comp.3  Comp.4  Comp.5
Standard deviation     0.2282 0.1408 0.1264 0.10444 0.09741
Proportion of Variance 0.3543 0.1349 0.1087 0.07423 0.06458
Cumulative Proportion  0.3543 0.4892 0.5979 0.67218 0.73676
                        Comp.6  Comp.7  Comp.8  Comp.9
Standard deviation     0.09043 0.08123 0.07731 0.06791
Proportion of Variance 0.05565 0.04491 0.04068 0.03138
Cumulative Proportion  0.79241 0.83732 0.87800 0.90938
                       Comp.10 Comp.11 Comp.12 Comp.13
Standard deviation     0.05634 0.05353 0.04703 0.04529
Proportion of Variance 0.02160 0.01950 0.01505 0.01396
Cumulative Proportion  0.93098 0.95048 0.96553 0.97950
                       Comp.14  Comp.15
Standard deviation     0.04033 0.037227
Proportion of Variance 0.01107 0.009432
Cumulative Proportion  0.99057 1.000000
```

Eigenvalue Scree Plot




```
> plot(pc.fit)
```


Loadings (eigenvectors)

```
> loadings(pc.fit) # pc.fit$loadings
```

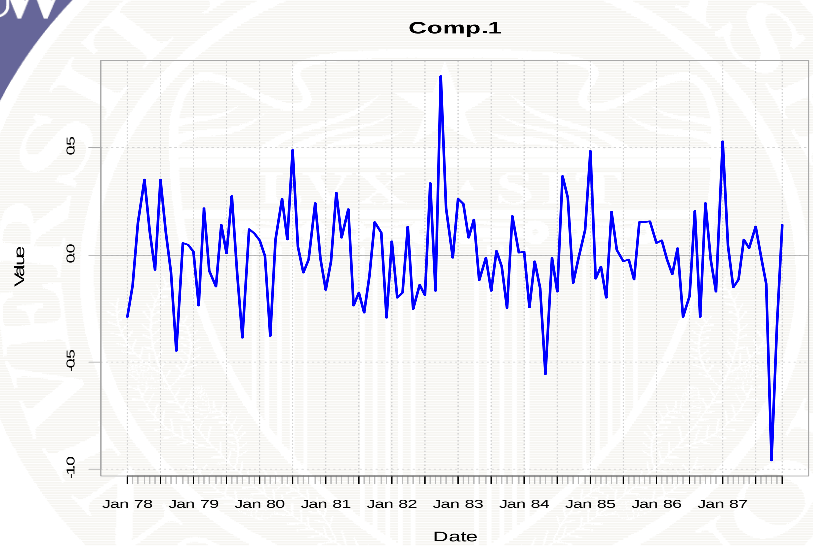
Loadings:

	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6	Comp.7
CITCRP	0.273						
CONED							
CONTIL	0.377	-0.824	-0.199	0.157	0.144	-0.191	
DATGEN	0.417	0.152	0.277	-0.329	0.287	-0.497	
DEC	0.305	0.129	0.202	-0.141			0.368
DELTA	0.250	0.179		0.258		0.242	0.481
GENMIL	0.133			0.128		0.249	0.117
GERBER	0.167	-0.199			-0.418	0.349	
IBM	0.146						0.142
MOBIL	0.155		0.248	-0.241	-0.459		-0.155
PANAM	0.311	0.365	-0.630	0.227	-0.343	-0.390	-0.197
PSNH			-0.527	-0.692	0.249	0.360	
TANDY	0.412	0.207	0.188	0.323	0.356	0.385	-0.564
TEXACO	0.132		0.245	-0.219	-0.430		-0.325
WEYER	0.265	0.131		-0.128	-0.111	0.152	0.291

Principal Component Factors

```
> head(pc.fit$scores[, 1:4])
              Comp.1    Comp.2   Comp.3     Comp.4
1978-01-01 -0.28998  0.069162 -0.07621  0.0217151
1978-02-01 -0.14236 -0.141967 -0.01794  0.0676476
1978-03-01  0.14927  0.113295 -0.09307  0.0326150
1978-04-01  0.35056 -0.032904  0.01128 -0.0168986
1978-05-01  0.10874  0.004943 -0.04640  0.0612666
1978-06-01 -0.06948  0.041330 -0.06757 -0.0009816
```

Note: Scores are based on centered (demeaned) returns




```
> chart.TimeSeries(pc.fit$scores[, 1, drop=FALSE],
+                  colorset="blue")
```

Direct Eigenvalue Computation

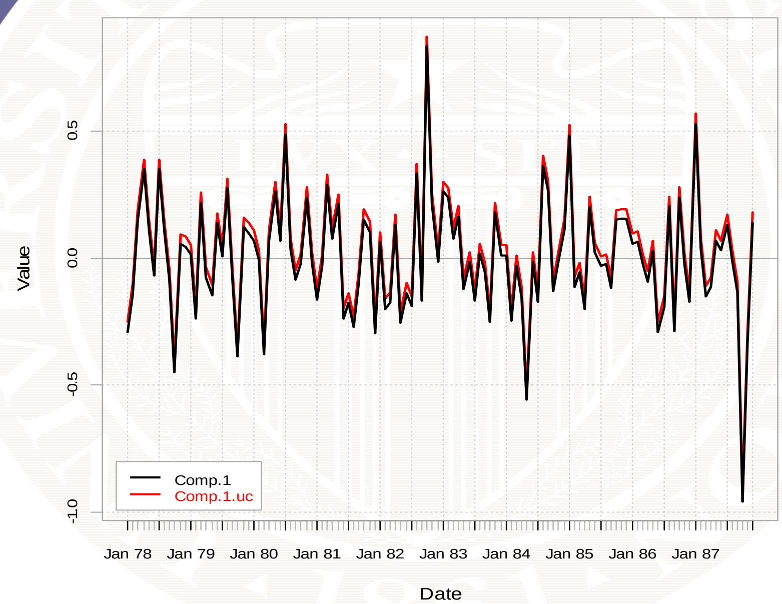
```
> eigen.fit = eigen(var(returns.mat))
> names(eigen.fit)
[1] "values"  "vectors"
> names(eigen.fit$values) =
+       rownames(eigen.fit$vectors) = asset.names

# compare princomp output with direct eigenvalue output
>  cbind(pc.fit$loadings[,1:2], eigen.fit$vectors[, 1:2])
         Comp.1    Comp.2
CITCRP  0.27271 -0.085495 -0.27271 -0.085495
CONED   0.04441  0.001193 -0.04441  0.001193
CONTIL  0.37694 -0.823575 -0.37694 -0.823575
DATGEN  0.41719  0.151818 -0.41719  0.151818
DEC     0.30493  0.129067 -0.30493  0.129067
…
```

Notice sign change!

Compare Centered and Uncentered Principal Component Factors

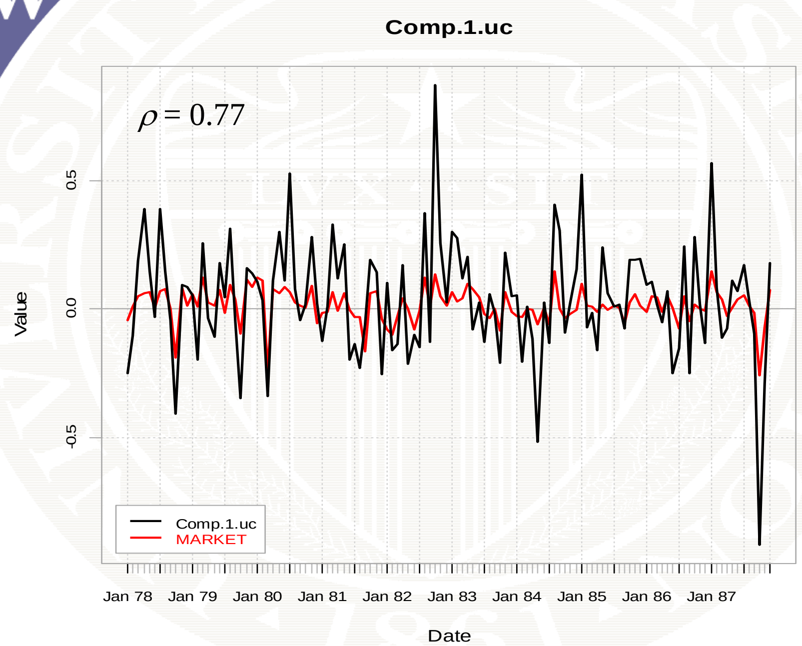
```
# compute uncentered pc factors from eigenvectors
# and return data
> pc.factors.uc = returns.mat %*% eigen.fit$vectors
> colnames(pc.factors.uc) =
+          paste(colnames(pc.fit$scores),".uc",sep="")

# compare centered and uncentered scores. Note sign
# change on first factor
> cbind(pc.fit$scores[,1,drop=F],
+       -pc.factors.uc[,1,drop=F])
              Comp.1   Comp.1.uc
1978-01-01 -0.289978 -0.250237
1978-02-01 -0.142355 -0.102614
1978-03-01  0.149273  0.189015
1978-04-01  0.350563  0.390304
1978-05-01  0.108743  0.148484
```

Centered and Uncentered Principle Component Factors



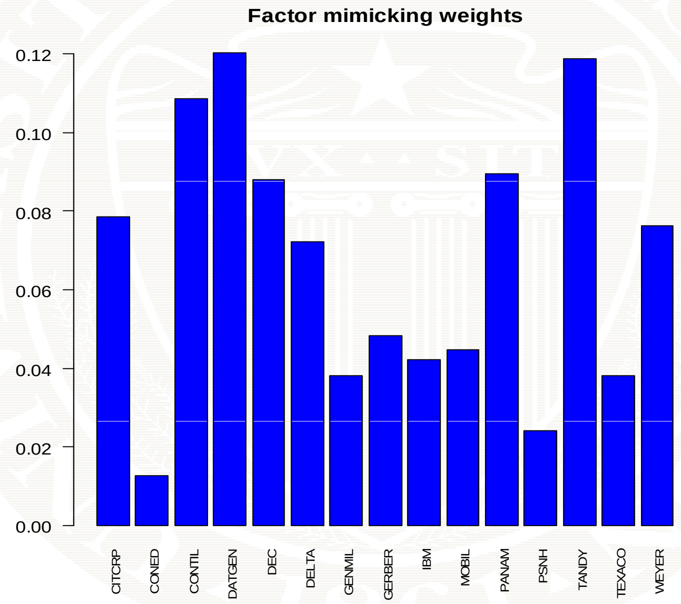
Interpreting Principal Component Factor

```
# Compute correlation with market return
> cor(cbind(pc.factors.uc[,1,drop=F],
+           berndt.df[, "MARKET",drop=F]))
          Comp.1.uc  MARKET
Comp.1.uc    1.0000 -0.7657
MARKET      -0.7657  1.0000

# Correlation with sign change
> cor(cbind(-pc.factors.uc[,1,drop=F],
+           berndt.df[, "MARKET",drop=F]))
          Comp.1.uc MARKET
Comp.1.uc    1.0000 0.7657
MARKET       0.7657 1.0000
```



Factor Mimicking Portfolio

```
> p1 = pc.fit$loadings[, 1]
> p1
 CITCRP   CONED  CONTIL  DATGEN     DEC   DELTA  GENMIL
0.27271 0.04441 0.37694 0.41719 0.30493 0.25017 0.13256
 GERBER     IBM   MOBIL   PANAM    PSNH   TANDY  TEXACO
0.16716 0.14644 0.15517 0.31067 0.08407 0.41193 0.13225
  WEYER
0.26488
> sum(p1)
[1] 3.471

# create factor mimicking portfolio by
normalizing
# weights to unity
> p1 = p1/sum(p1)
# normalized principle component factor
> f1 = returns.mat %*% p1
```



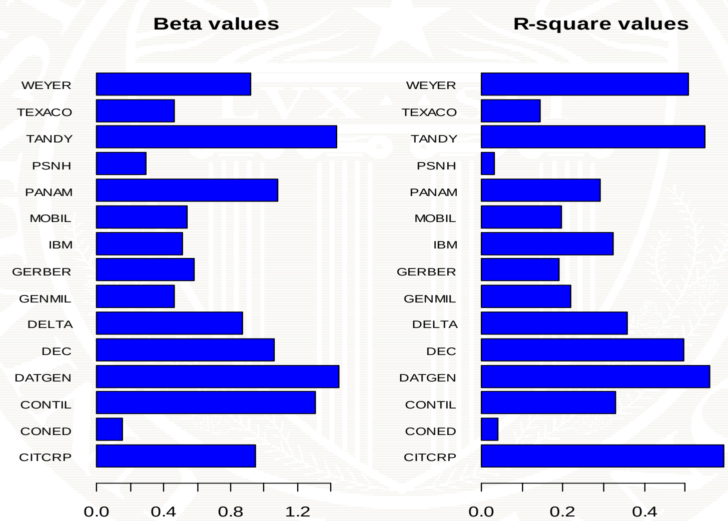
Estimate Factor Betas

```
# estimate factor betas by multivariate regression
> X.mat = cbind(rep(1,n.obs), f1)
> colnames(X.mat) = c("intercept", "Factor 1")
> XX.mat = crossprod(X.mat)
# multivariate least squares
> G.hat = solve(XX.mat)%*%crossprod(X.mat,returns.mat)
> beta.hat = G.hat[2,]
> E.hat = returns.mat - X.mat%*%G.hat
> diagD.hat = diag(crossprod(E.hat)/(n.obs-2))
# compute R2 values from multivariate regression
> sumSquares = apply(returns.mat, 2, function(x)
+                     {sum( (x - mean(x))^2 )})
> R.square = 1 - (n.obs-2)*diagD.hat/sumSquares
```

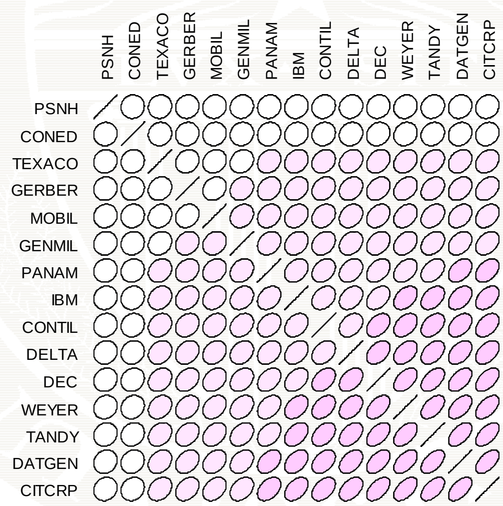
Regression Results

```
> cbind(beta.hat, diagD.hat, R.square)
       beta.hat diagD.hat R.square
CITCRP   0.9467  0.002674  0.59554
CONED    0.1542  0.002444  0.04097
CONTIL   1.3085  0.015380  0.32847
DATGEN   1.4483  0.007189  0.56176
DEC      1.0586  0.004990  0.49664
DELTA    0.8685  0.005967  0.35704
GENMIL   0.4602  0.003336  0.21808
GERBER   0.5803  0.006284  0.19058
IBM      0.5084  0.002378  0.32318
MOBIL    0.5387  0.005229  0.19600
PANAM    1.0785  0.012410  0.29168
PSNH     0.2918  0.011711  0.03096
TANDY    1.4300  0.007427  0.54746
TEXACO   0.4591  0.005480  0.14455
WEYER    0.9195  0.003583  0.50904
```


Regression Results



Principal Components Correlations



Global Minimum Variance Portfolios