PAST

Portfolio Attribution and Simulation Toolkit

Vijay Vaidyanathan
vijay@ReturnMetrics.com

AAII Silicon Valley, 11March 2008

Agenda

- Screening vs. Backtesting
- Overview / PAST-SIPro Demo
- Setup PAST-SIPro
- The "R" programming language
- Sample backtests (Simple to Sophisticated)
- From Screens to Backtests
- ToDo List
- Questions?

About Me

- Silicon Valley “Techie” since 1990
  - CTO, Xoom.com
  - Chief Strategy Officer, NBC Internet
  - CEO, Yaga Inc.
- M. Sc degrees in Engineering (1986), Computer Science (1991) and Finance (2007)

Disclaimers

- I have nothing to sell and I don't give investment advice. Seriously.
- Nothing here should be considered advice
- I do have (many) opinions, which I may let slip
- My opinions are just that - opinions. They are worth even less than what you are paying me!
- PAST is a part-time project, I fix bugs daily!
- I use Mac, should work on Windows & Linux

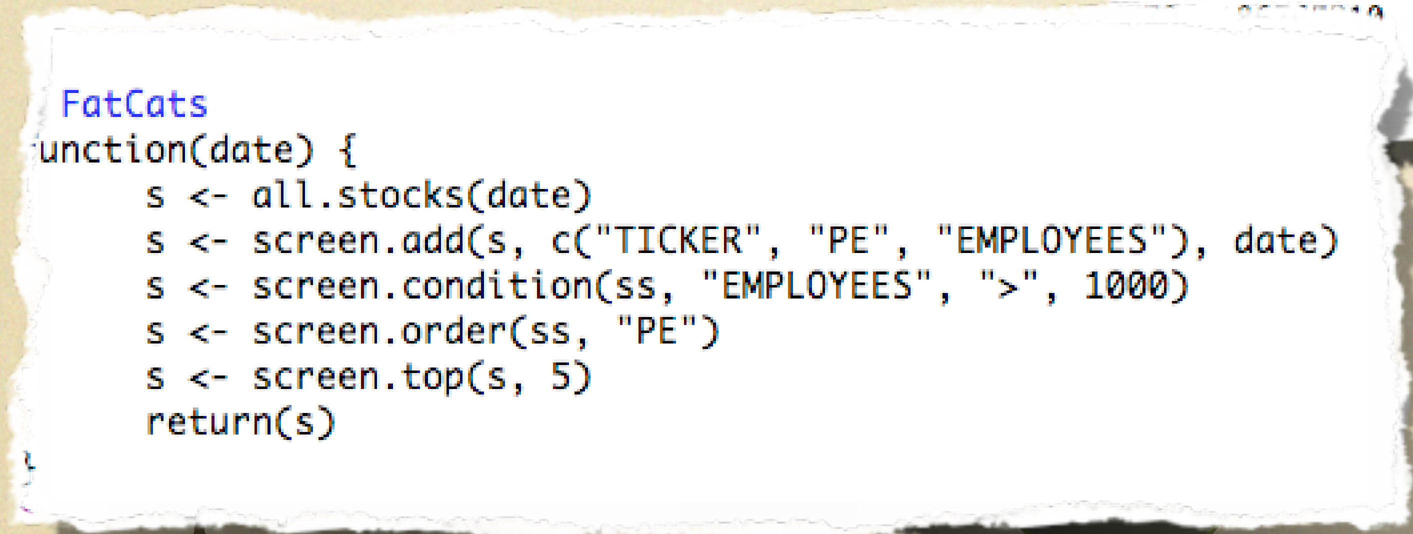
Agenda

- Screening vs. Backtesting
- Overview / PAST-SIPro Demo
- Setup PAST-SIPro
- The "R" programming language
- Sample backtests (Simple to Sophisticated)
- From Screens to Backtests
- ToDo List
- Questions?

What is a “screen”

- Filters a Universe of stocks, based on “criterion” as of a particular “date” to produce an “short list”

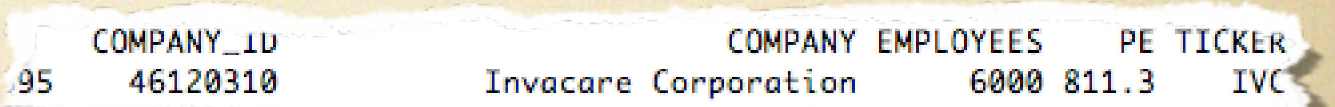
GOOG



```
 FatCats
unction(date) {
     s <- all.stocks(date)
     s <- screen.add(s, c("TICKER", "PE", "EMPLOYEES"), date)
     s <- screen.condition(ss, "EMPLOYEES", ">", 1000)
     s <- screen.order(ss, "PE")
     s <- screen.top(s, 5)
     return(s)
```

HD

:



```
    COMPANY_ID                   COMPANY EMPLOYEES    PE TICKER
95    46120310     Invacare Corporation       6000 811.3    IVC
                   SunPower Corporation       2219 690.9   SPWR
                         Public Storage       6000 651.3    PSA
                   salesforce.com, inc.       2070 576.8    CRM
```

SPWR

PSA

CRM

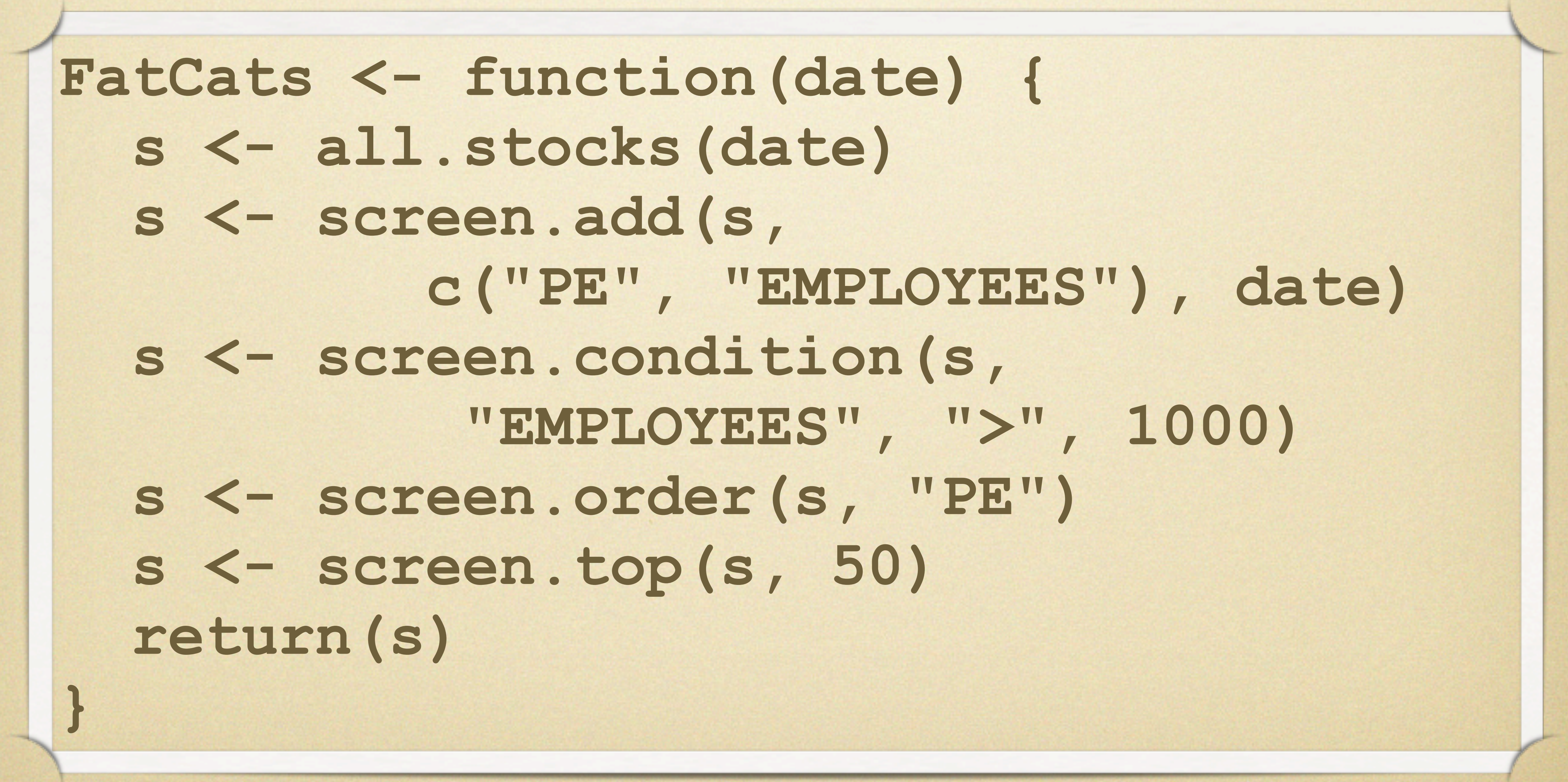
HTCH

A Screen

- Input
  - A Date
- Output
  - A list of Assets that “passes” the screen

e.g. FatCats

- Companies with over 1000 employees with the 50 highest Price-Earnings Ratios



```
FatCats <- function(date) {
   s <- all.stocks(date)
   s <- screen.add(s,
               c("PE", "EMPLOYEES"), date)
   s <- screen.condition(s,
                "EMPLOYEES", ">", 1000)
   s <- screen.order(s, "PE")
   s <- screen.top(s, 50)
   return(s)
}
```

e.g. S&P 500

- All the stocks in the S&P500 Index

```
sp500 <- function (date){

   s <- all.stocks(date)
   s <- screen.add(s, "SP", date)
   s <- screen.condition(s,
                 "SP", "=", 500)

   return(s)
}
```

Backtest

- A Backtest (in PAST) of a screen is a *simulation in time* that tells you how a portfolio that comprises of *all the stocks that pass that screen* would have performed under the simulated conditions

- Output: Portfolio and Trading metrics, including a *time series* of portfolio values.
  - + Portfolio Turnover, Transaction Costs etc.

- Output -> zoo() -> PerformanceAnalytics

e.g. Simulating FatCats

- Would you hold the FatCats portfolio?
- Start at Month #1, Buy all 50 FatCats (using some weighting scheme)
- Every "B" months, rebalance to the weighting
- Every "R" months, re-run the screen, selling the exits, buy the new screen entries
- e.g. Holding Equally Weighted, rebalancing Monthly, Re-run the screen Quarterly (B=1, R=3)
- Holding Market Cap-Weighted, rebalance Quarterly, Re-Screen Annually (B=3, R=12)

e.g. Simulating S&P

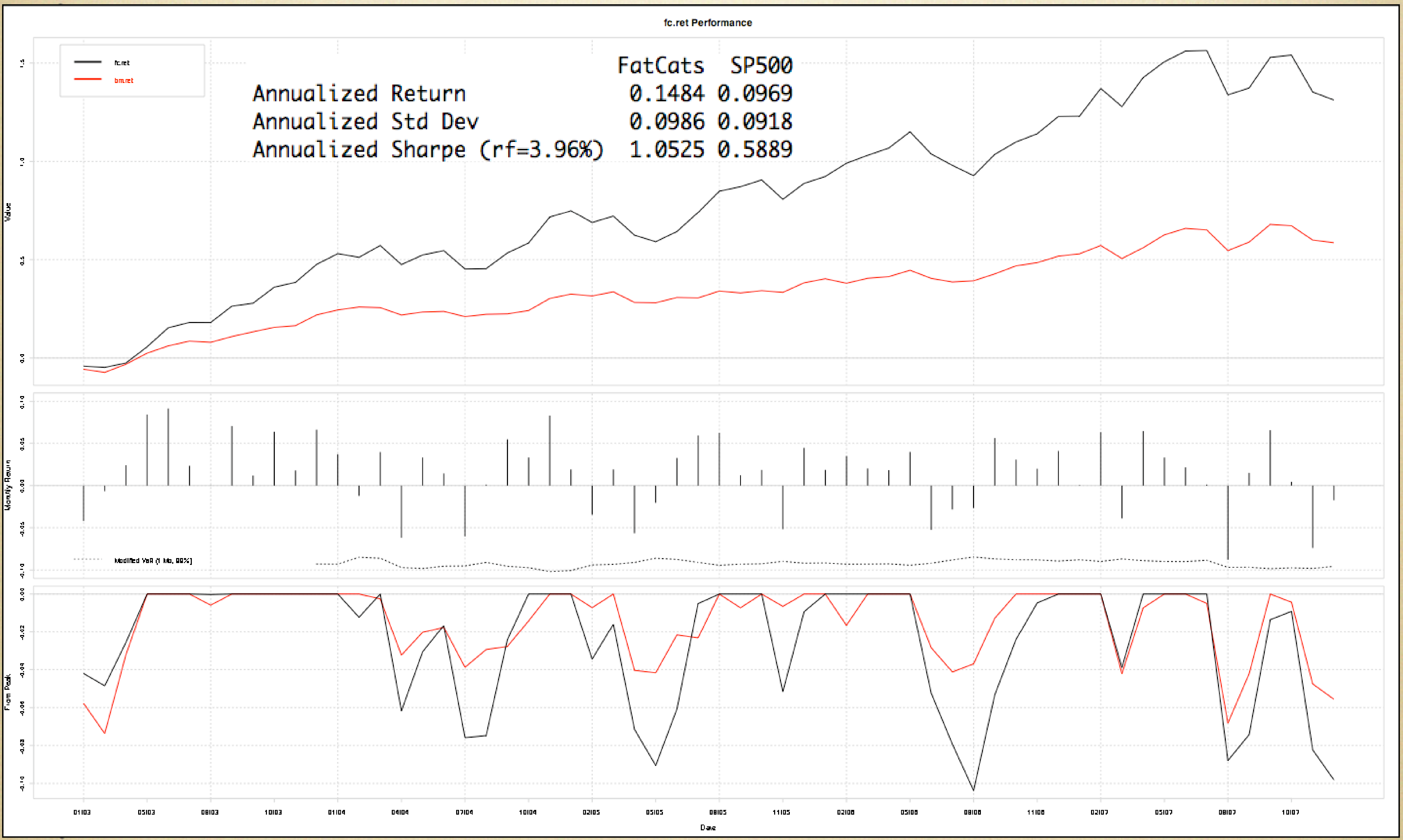
- Buy all sp500 stocks, weighted by Market Capitalization, rescreen monthly (rebalance monthly, but should have almost no effect)

FatCats vs. SP500

- Would you buy & follow the FatCats strategy?
- How would you have done compared to the S&P 500?
- Answer: Run the simulation/Backtest

(care to make a guess?)

Monthly Rebalancing, Quarterly Rescreening

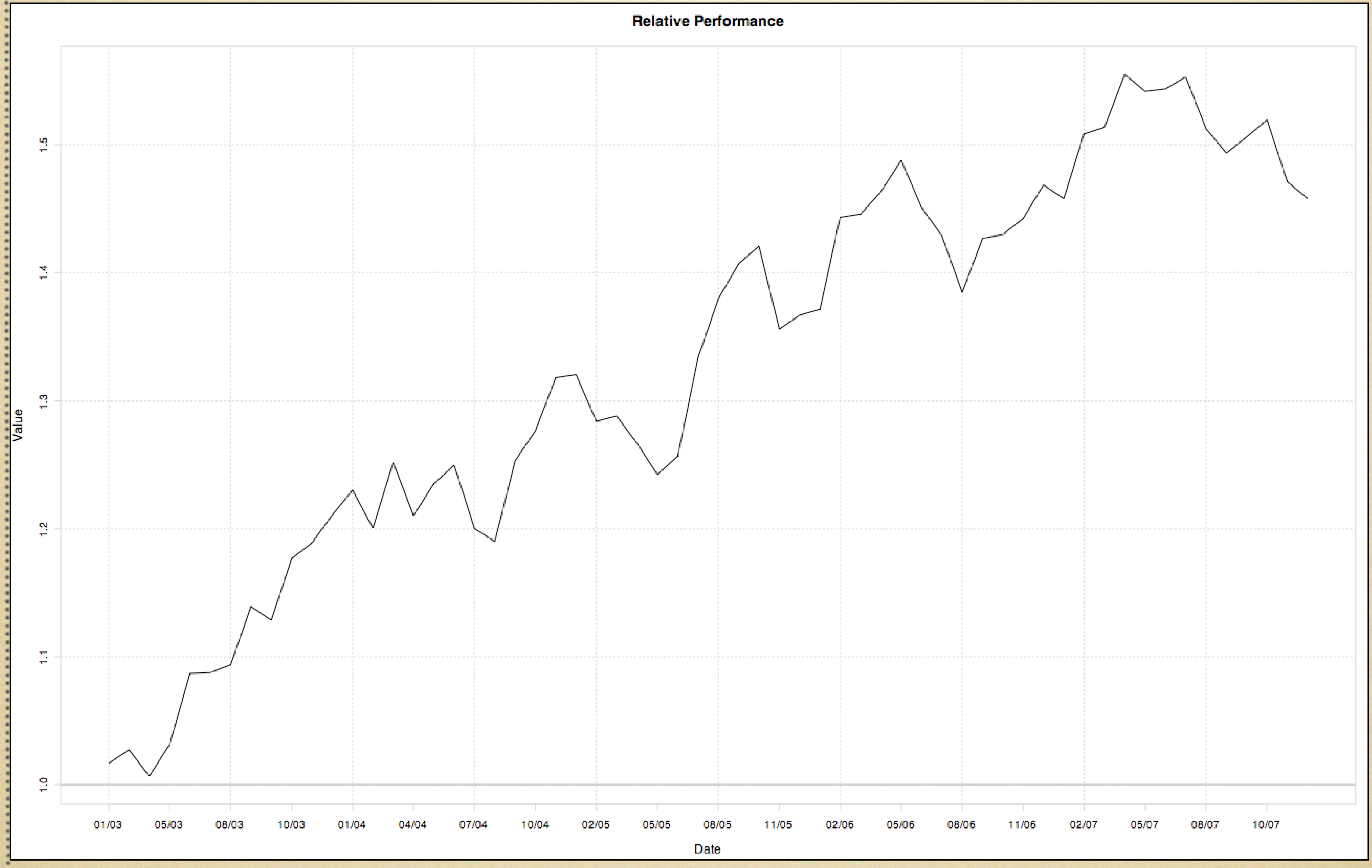


	FatCats	SP500
Annualized Return	0.1484	0.0969
Annualized Std Dev	0.0986	0.0918
Annualized Sharpe (rf=3.96%)	1.0525	0.5889

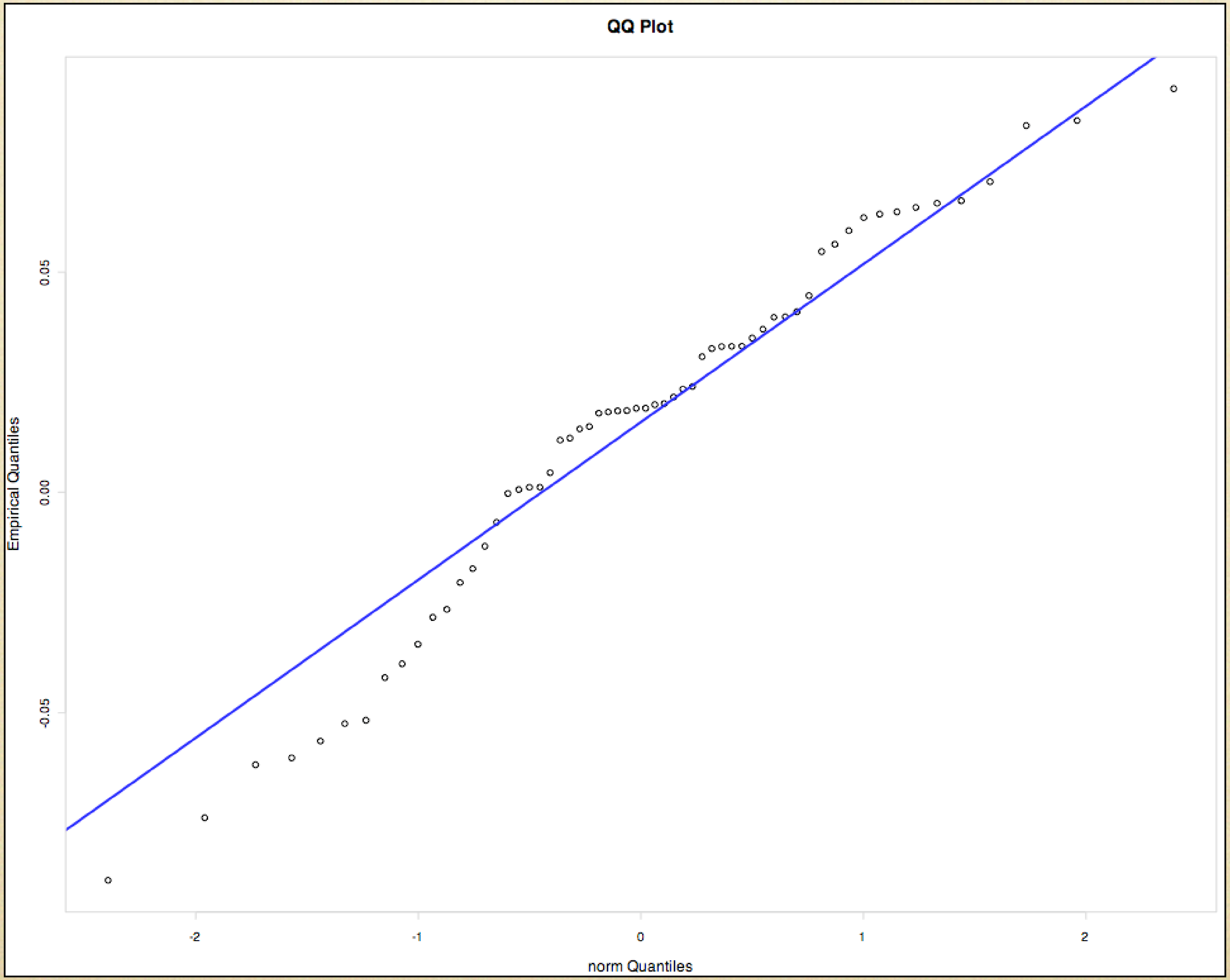
Why simulate?

- Thanks to “R” and PerformanceAnalytics, it is easy to perform in-depth analysis
- Screens are snapshots in time, simulations tell you what would have happened *over* time
- Time series lends itself to statistical analysis
- There may be a story hidden in the numbers ...
- For instance ...

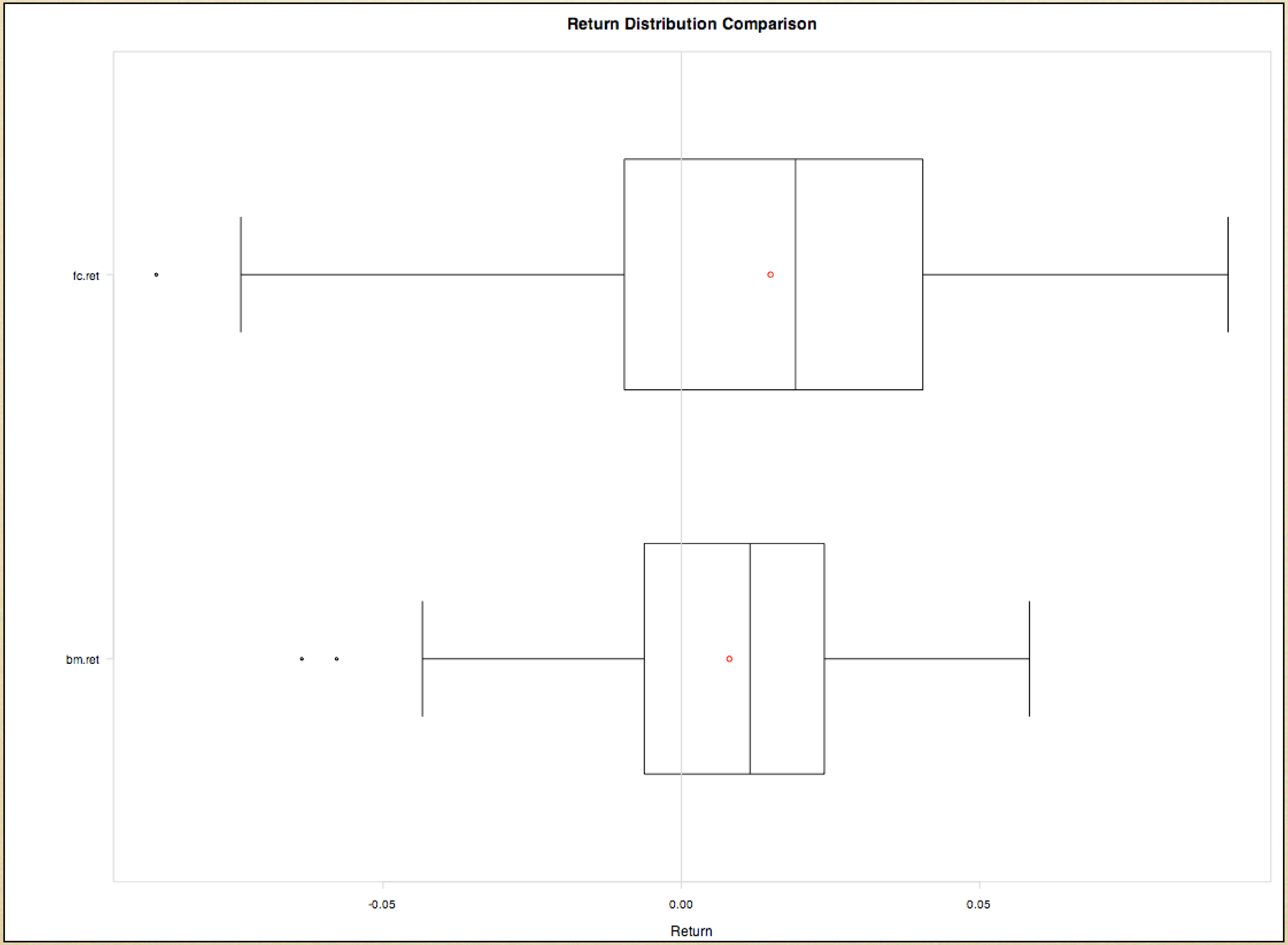
Relative Performance



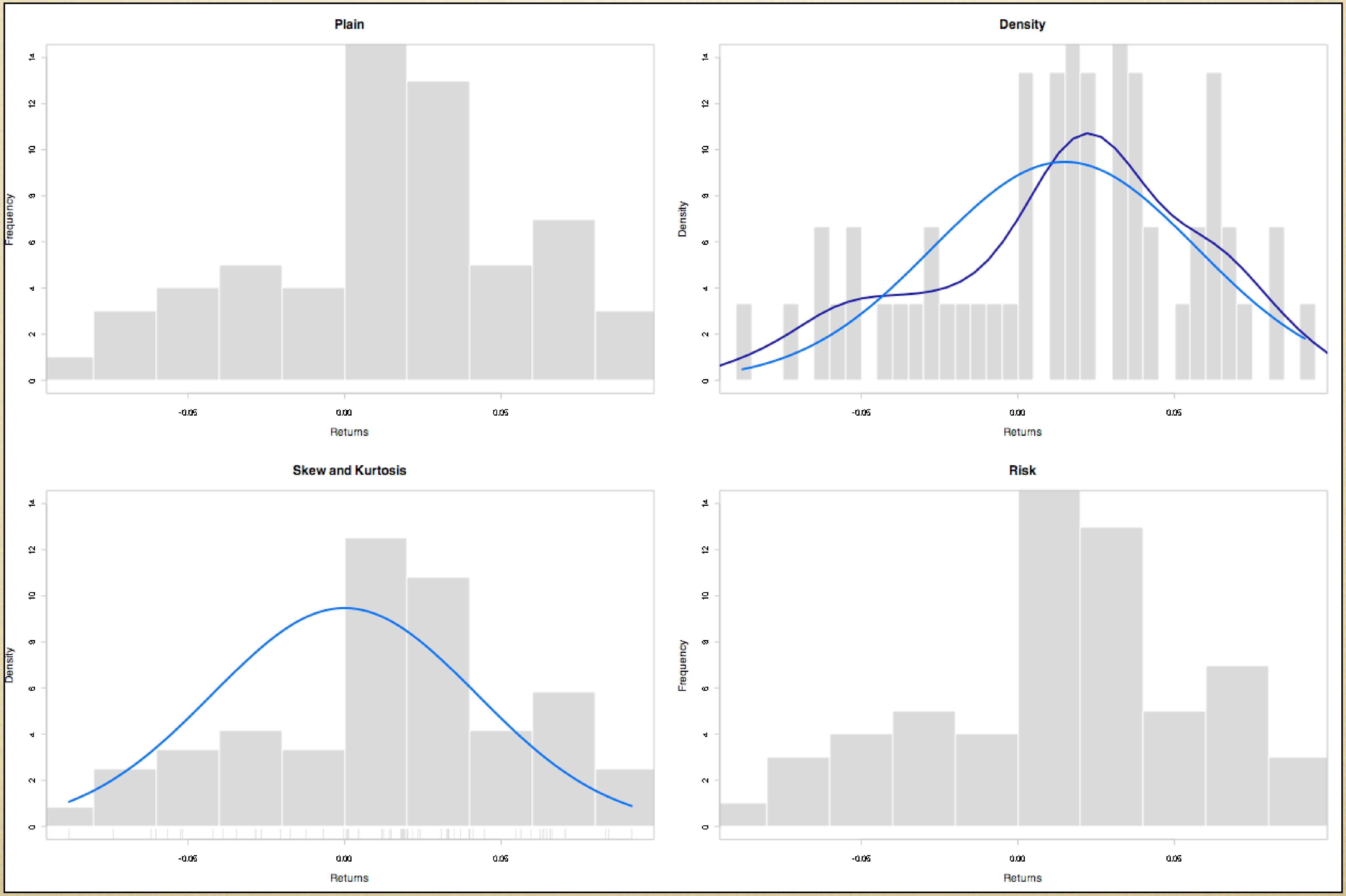
Gaussian Returns?

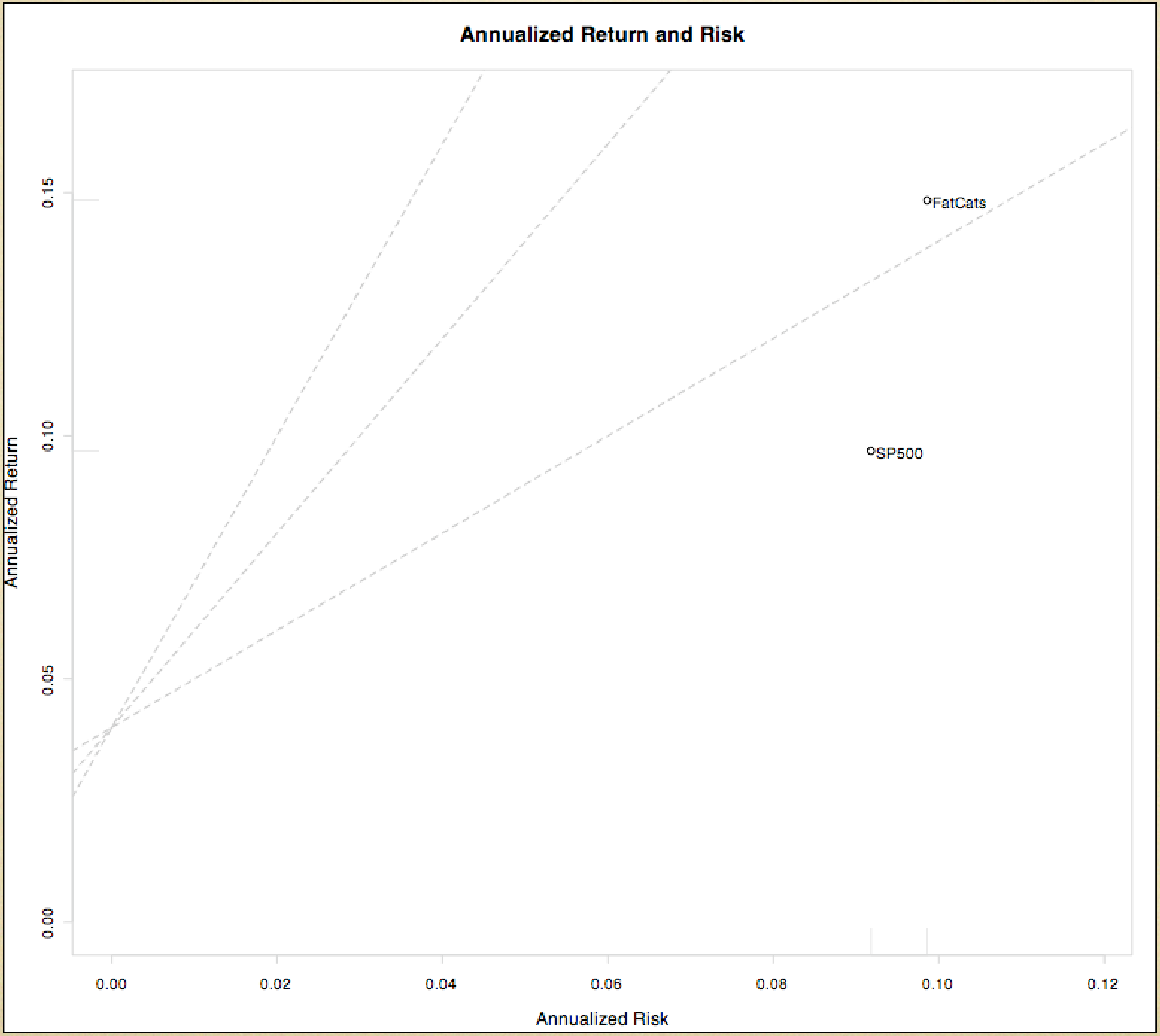


Easy Boxplots

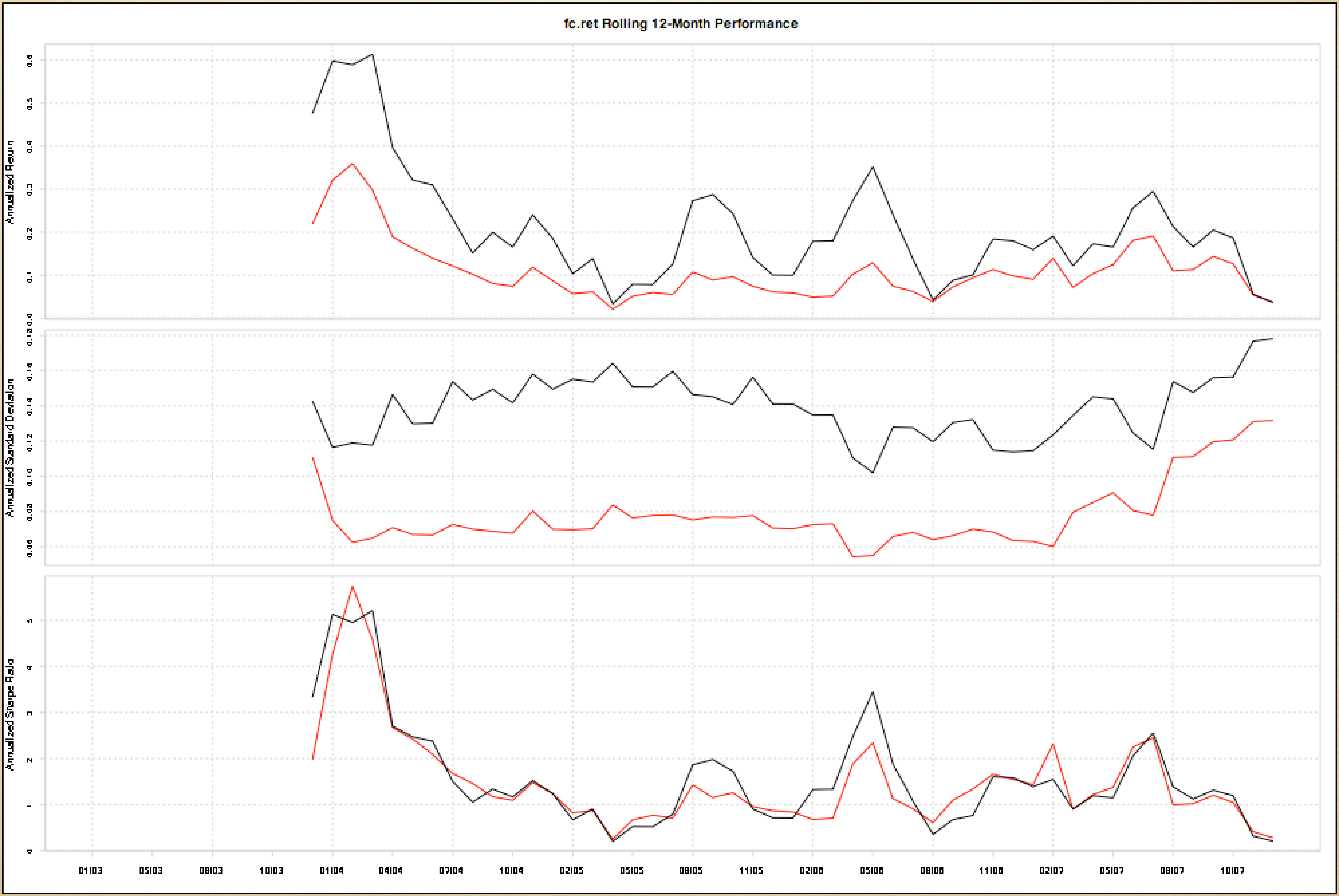


PerformanceAnalytics Charts (v. cool!)

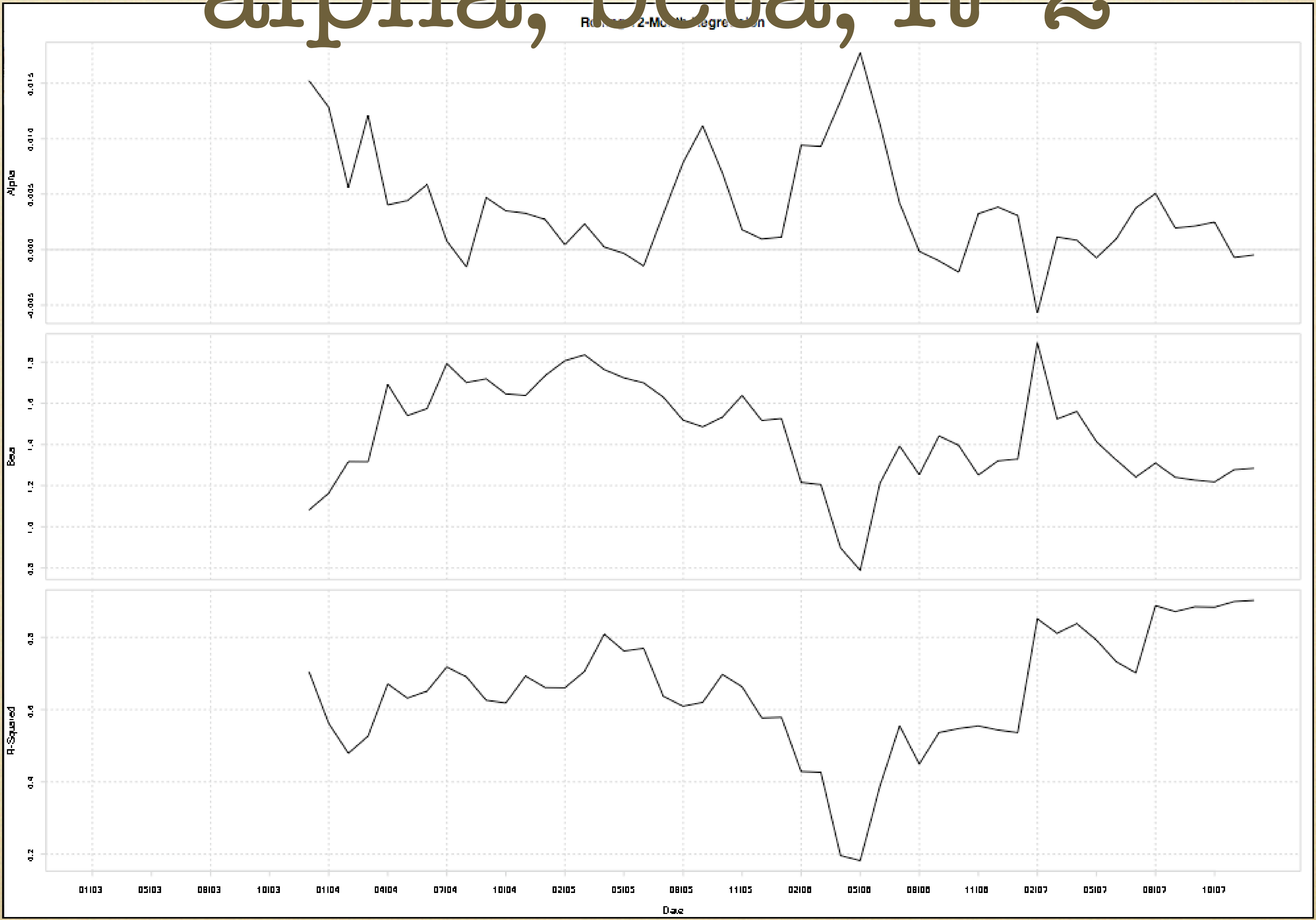



Annualized Return and Risk
Annualized Return
Annualized Risk
0.00
0.05
0.10
0.15
0.00
0.02
0.04
0.06
0.08
0.10
0.12
FatCats
SP500

Rolling Windows



Rolling CAPM alpha, beta, R^2



- and a lot more ...
- (see references at the end)

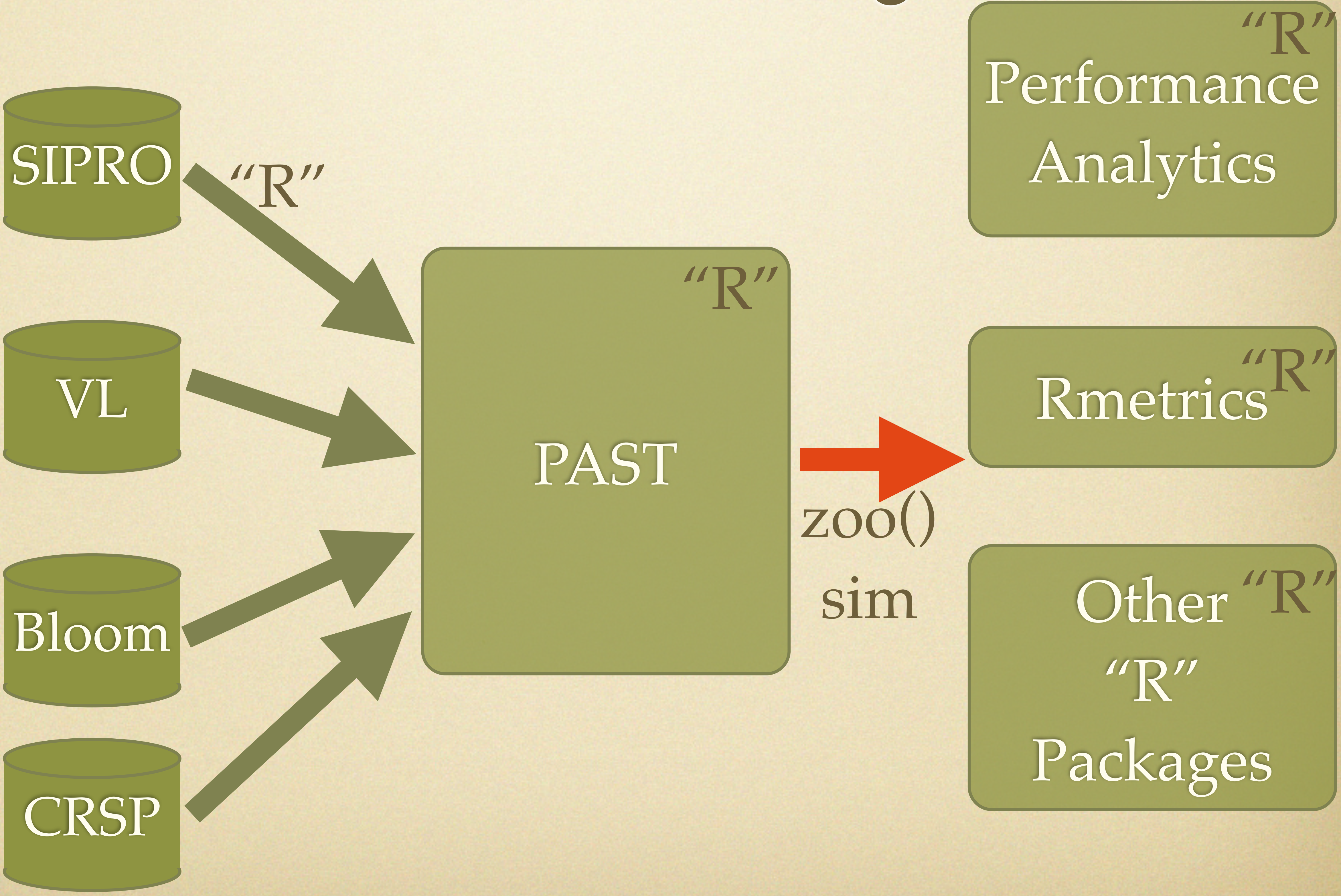
Agenda

- Screening vs. Backtesting
- Overview / PAST-SIPro Demo
- Setup PAST-SIPro
- The "R" programming language
- Sample backtests (Simple to Sophisticated)
- From Screens to Backtests
- ToDo List
- Questions?

Overview

- PAST is “Performance Attribution and Simulation Toolkit”
- A *Toolkit* - needs other pieces (DB, DBDriver ...)
- You use it to build your own backtester
- PAST needs a database, and a “driver” or “plugin” to connect to it
- AAII/SIPro exists, working on ValueLine next

The PAST Eco-System



Basic How-To

- Design strategy
- Write a “screen” in R
- Pick an allocator (e.g. EW, CW, Markowitz ...)
- Simulate over the time period
- Analyze Results

e.g. LeanMean

- Strategy: sort of the opposite of FatCats
- Between 10 and 1000 employees
- Lowest 50 by Price-Earnings Ratio
- How would you do this?

Screen Design

LeanMean <- function(date) {

s <- all.stocks(date)

- Start with all.stocks(date)

s <- ..., date)

s <-

s <- screen.condition(s, "EMPLOYEES", ">", 10)

- Add condition: > 10 employees

s <- screen.order(s, "PE", na.last=NA)

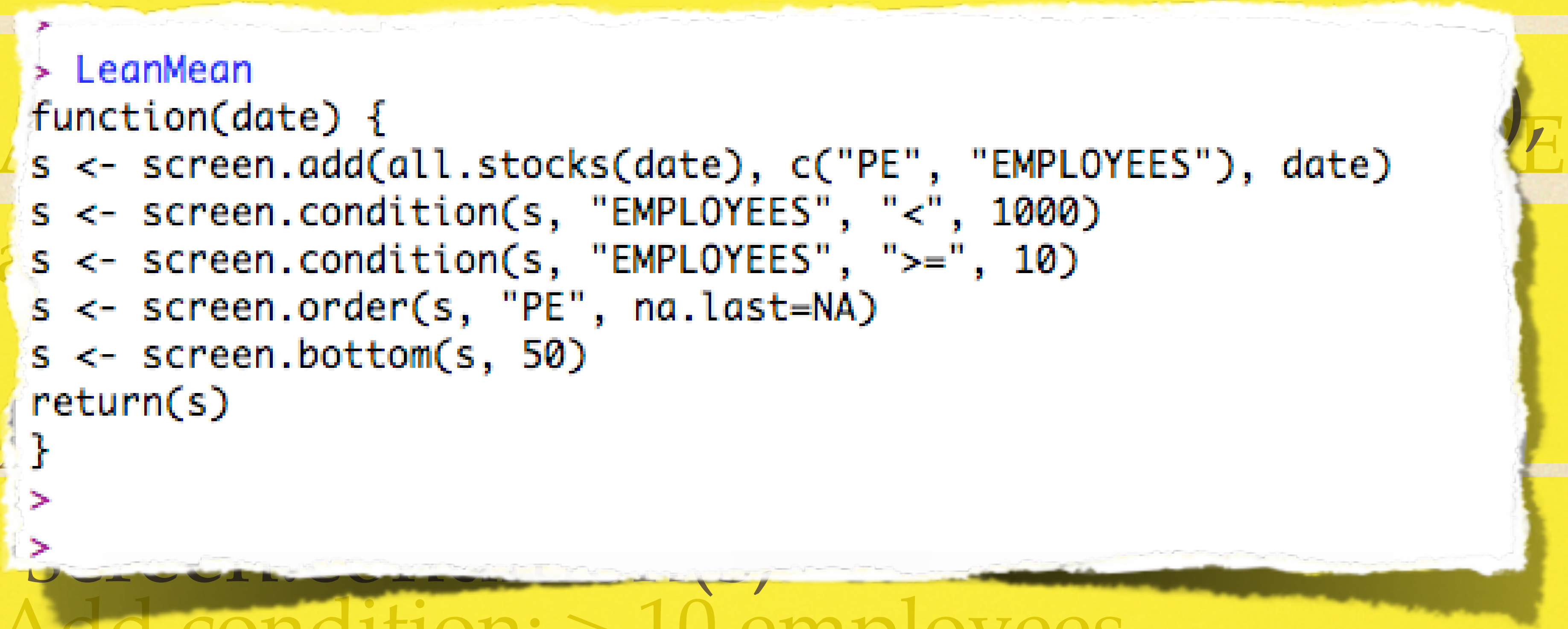
- Order by PE (remove NAs)

s <- screen.bottom(s, 50)

- Return the bottom 50

return (s)

}



```
> LeanMean
function(date) {
s <- screen.add(all.stocks(date), c("PE", "EMPLOYEES"), date)
s <- screen.condition(s, "EMPLOYEES", "<", 1000)
s <- screen.condition(s, "EMPLOYEES", ">=", 10)
s <- screen.order(s, "PE", na.last=NA)
s <- screen.bottom(s, 50)
return(s)
}
>
>
```

Agenda

- Screening vs. Backtesting
- Overview / PAST-SIPro Demo
- Setup PAST-SIPro
- The "R" programming language
- Sample backtests (Simple to Sophisticated)
- From Screens to Backtests
- ToDo List
- Questions?

Setting Up "R"

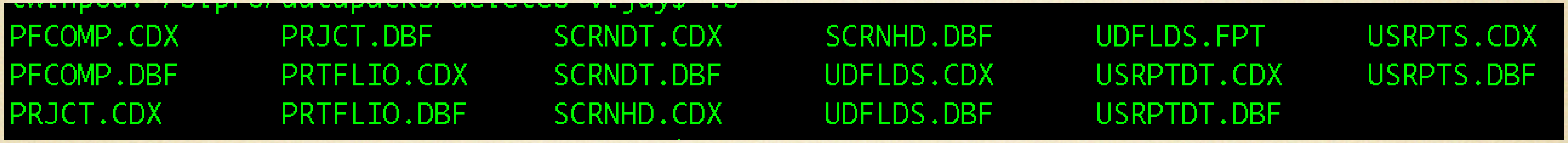
- Free download for Mac OS X, Windows and Linux from www.r-project.org
- Download and install is simple
- Install required "Packages"
- install.packages("PerformanceAnalytics", dependencies="Depends")
- PAST is not (yet) a package (ToDo)
- Install the Database ... (a mess)

Installing SIPRO data

- pick a directory/folder (e.g. ~/sipro/db)
- each month's data goes into a new folder
- e.g. FULLUPDATE20040227
- sub-folder FULLUPDATE20040227/DBFS
- put all *.DBF, *.CDX, *.FPT into this folder
- Things changed in January

Changes w.e.f. Jan ‘08

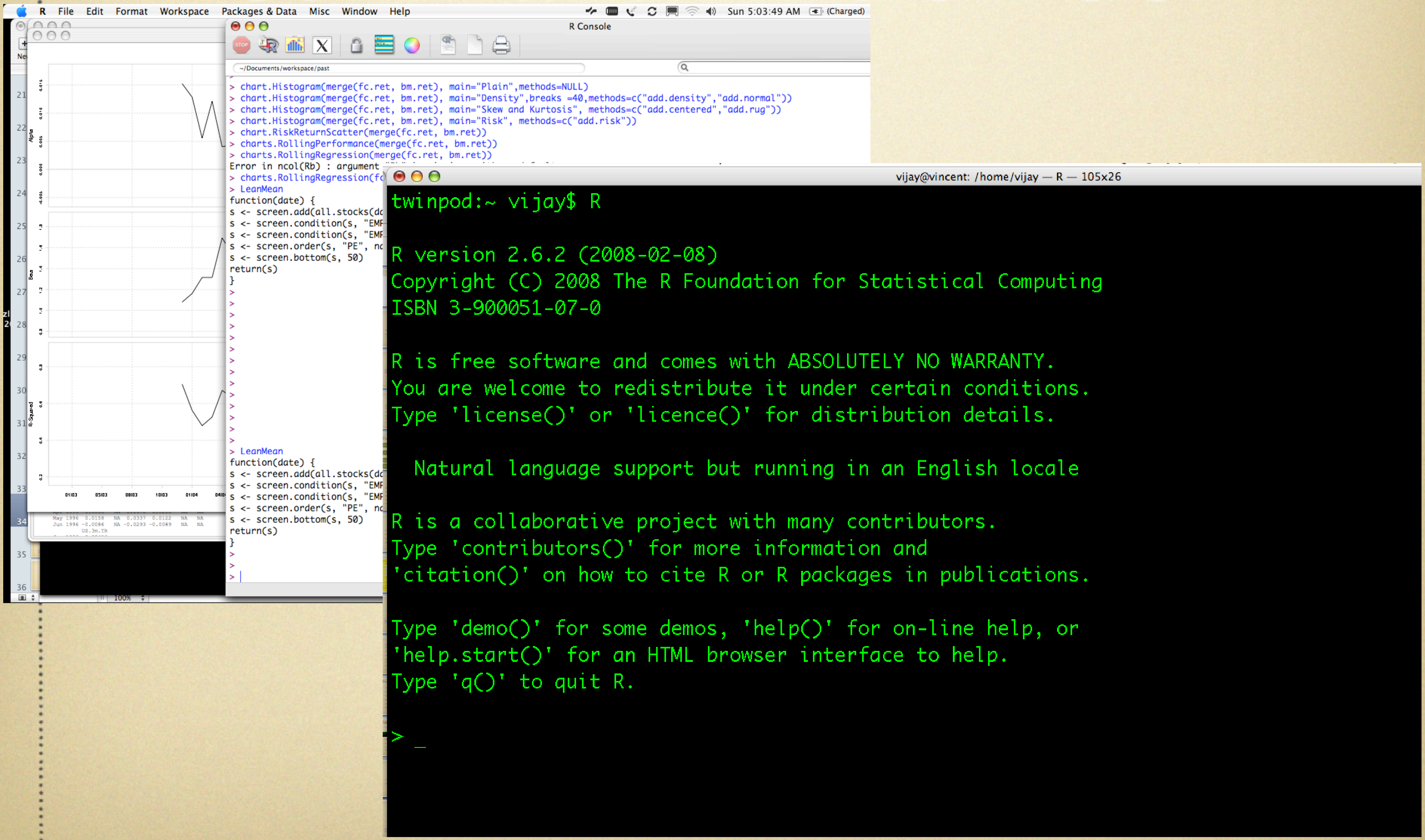
- New installer drops some new files
- Need to delete these new files:



```
PFCOMP.CDX      PRJCT.DBF       SCRNDT.CDX      SCRNHD.DBF      UDFLDS.FPT      USRPTS.CDX
PFCOMP.DBF      PRTFLIO.CDX     SCRNDT.DBF      UDFLDS.CDX      USRPTDT.CDX     USRPTS.DBF
PRJCT.CDX       PRTFLIO.DBF     SCRNHD.CDX      UDFLDS.DBF      USRPTDT.DBF
```

- DBF files are found in 4 dirs
- DataDict, Dynamic, Static, Weekly
- I use a script on Mac OS X, No windows required

Start "R"
Command Line OR GUI



```
> chart.Histogram(merge(fc.ret, bm.ret), main="Plain",methods=NULL)
> chart.Histogram(merge(fc.ret, bm.ret), main="Density",breaks =40,methods=c("add.density","add.normal"))
> chart.Histogram(merge(fc.ret, bm.ret), main="Skew and Kurtosis", methods=c("add.centered","add.rug"))
> chart.Histogram(merge(fc.ret, bm.ret), main="Risk", methods=c("add.risk"))
> chart.RiskReturnScatter(merge(fc.ret, bm.ret))
> charts.RollingPerformance(merge(fc.ret, bm.ret))
> charts.RollingRegression(merge(fc.ret, bm.ret))
Error in ncol(Rb) : argument
> charts.RollingRegression(fc
> LeanMean
function(date) {
s <- screen.add(all.stocks(da
s <- screen.condition(s, "EMF
s <- screen.condition(s, "EMF
s <- screen.order(s, "PE", nc
s <- screen.bottom(s, 50)
return(s)
}
>
>
>
>
>
>
>
>
>
>
>
>
>
>
> LeanMean
function(date) {
s <- screen.add(all.stocks(da
s <- screen.condition(s, "EMF
s <- screen.condition(s, "EMF
s <- screen.order(s, "PE", nc
s <- screen.bottom(s, 50)
return(s)
}
>
>
> |
```

```
twinpod:~ vijay$ R

R version 2.6.2 (2008-02-08)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> _
```

Initialize PAST, DB

- **THIS *WILL* CHANGE WHEN PACKAGED**
- Make sure you have the PAST source files
- (e.g. in ~/sipro/r)
- `setwd(“~/sipro/r/”)`
- `source(“aaii.R”)`
- `bt_setrepo("~/sipro/db/unpacked")`
- *[bt_setrepo() is to set/init the database repository]*
- *YOU ARE READY TO GO!*

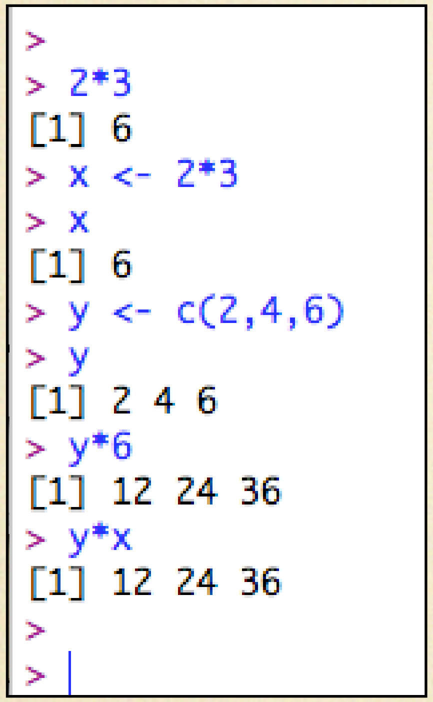
Agenda

- Screening vs. Backtesting
- Overview / PAST-SIPro Demo
- Setup PAST-SIPro
- The "R" programming language
- Sample backtests (Simple to Sophisticated)
- From Screens to Backtests
- ToDo List
- Questions?

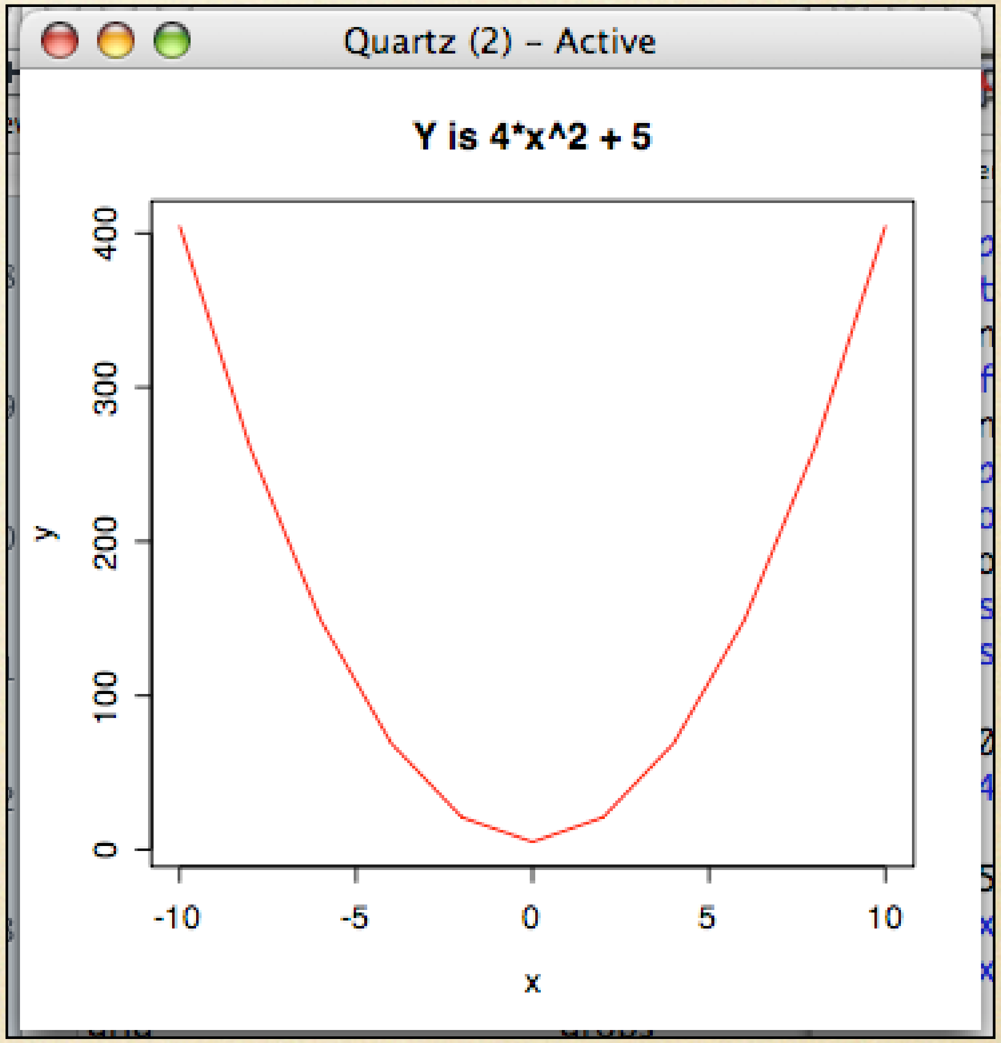
A super-short "R" 101

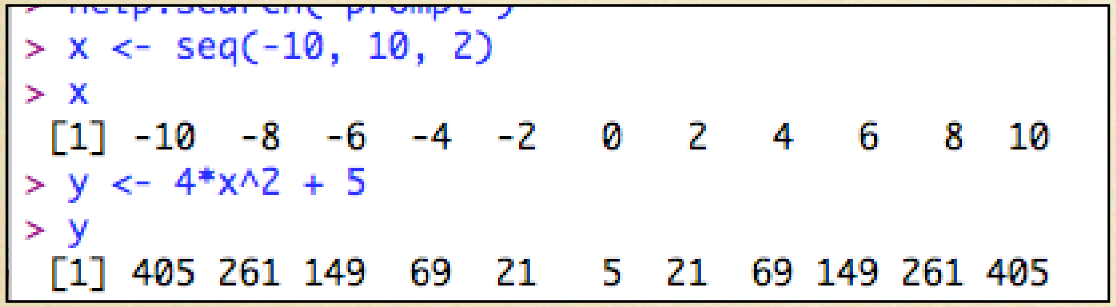
- R is a functional programming language
- Not really "object-oriented" ("object-based"?)
- Extremely elegant and powerful, but strange if you are unaccustomed to Functional Programs
- Functions inputs (arguments) and returns a single object as the result
- Lists, Matrices and Data Frames are "built in"
- So are operations (like + and *) on these

Some Examples

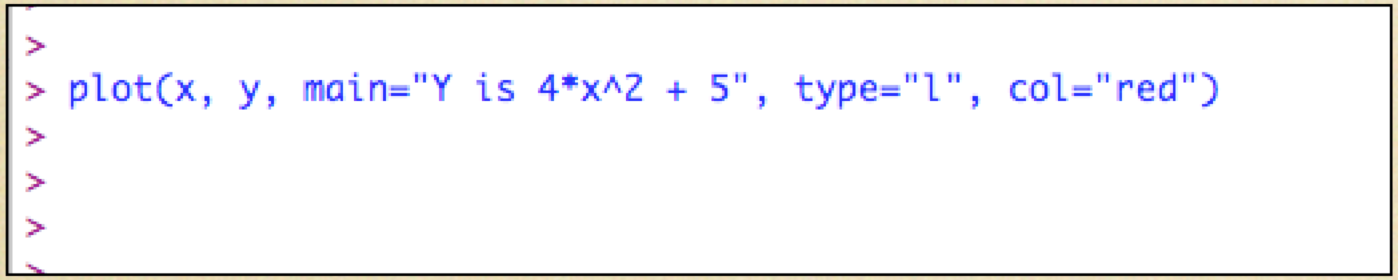


```
>
> 2*3
[1] 6
> x <- 2*3
> x
[1] 6
> y <- c(2,4,6)
> y
[1] 2 4 6
> y*6
[1] 12 24 36
> y*x
[1] 12 24 36
>
> |
```



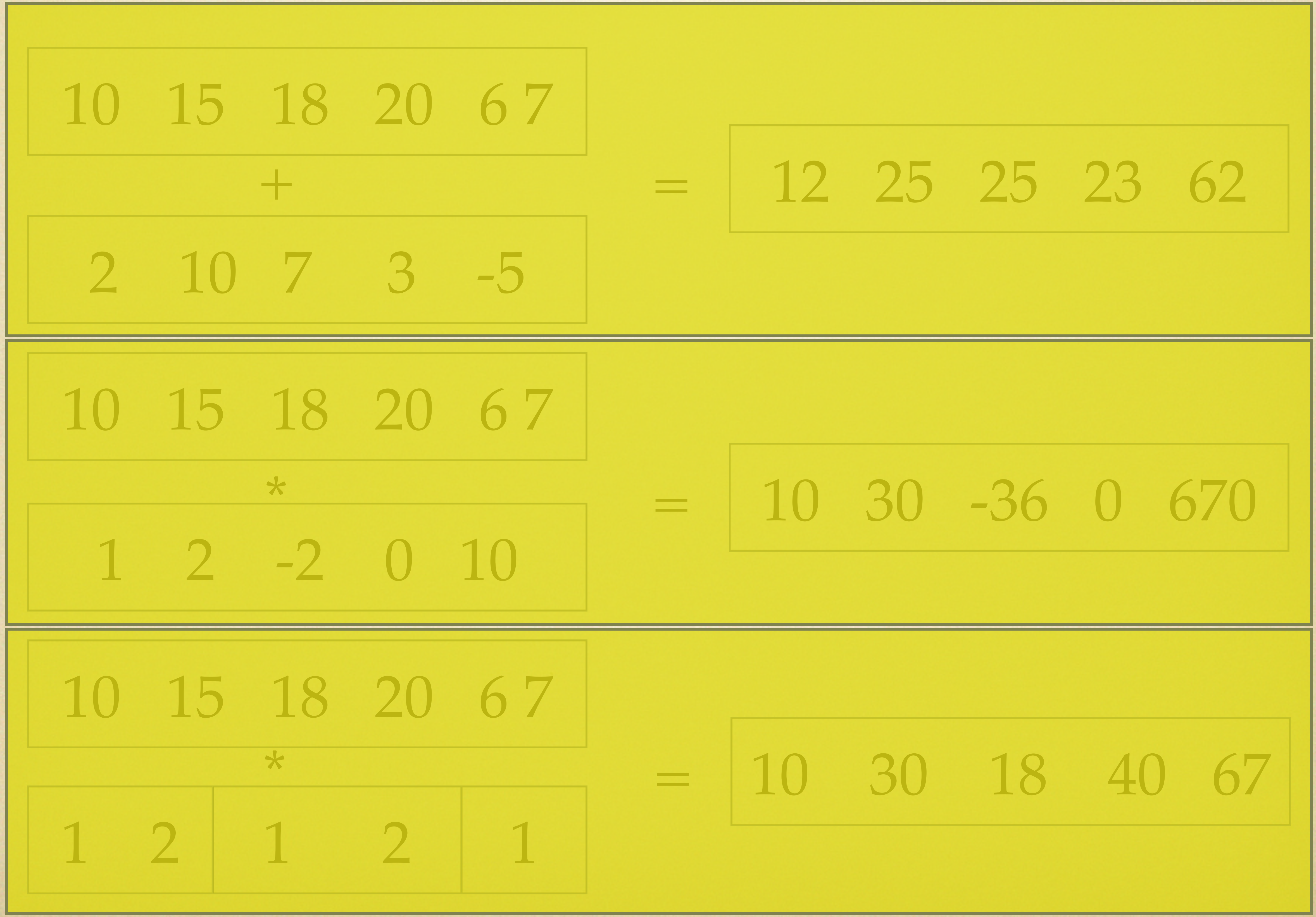


```
> x <- seq(-10, 10, 2)
> x
 [1] -10  -8  -6  -4  -2   0   2   4   6   8  10
> y <- 4*x^2 + 5
> y
 [1] 405 261 149  69  21   5  21  69 149 261 405
```



```
>
> plot(x, y, main="Y is 4*x^2 + 5", type="l", col="red")
>
>
>
```

Vector Arithmetic



10 15 18 20 67

+

2 10 7 3 -5

= 12 25 25 23 62

10 15 18 20 67

*

1 2 -2 0 10

= 10 30 -36 0 670

10 15 18 20 67

*

1 2 | 1 2 | 1

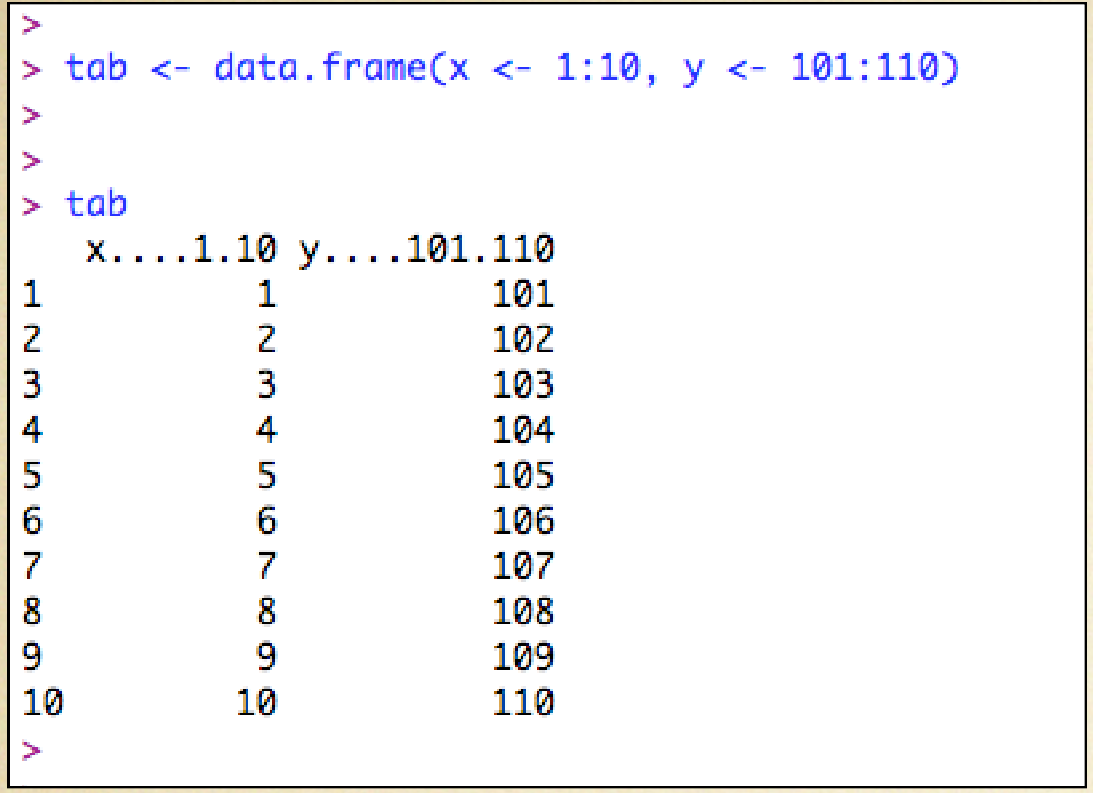
= 10 30 18 40 67

Lists, Matrices and Dataframes

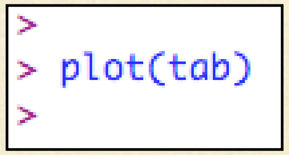
- Numbers, strings (scalars) are vectors of len=1
- A List is ... well ... a list of objects (of possibly different types)
- x <- list(“a”, 10, “b”, c(1, 3, 4))
- A Matrix is a 2 dimensional object of scalars
- A Dataframe is a list of columns, each of which has the same number of elements (rows)

Dataframes

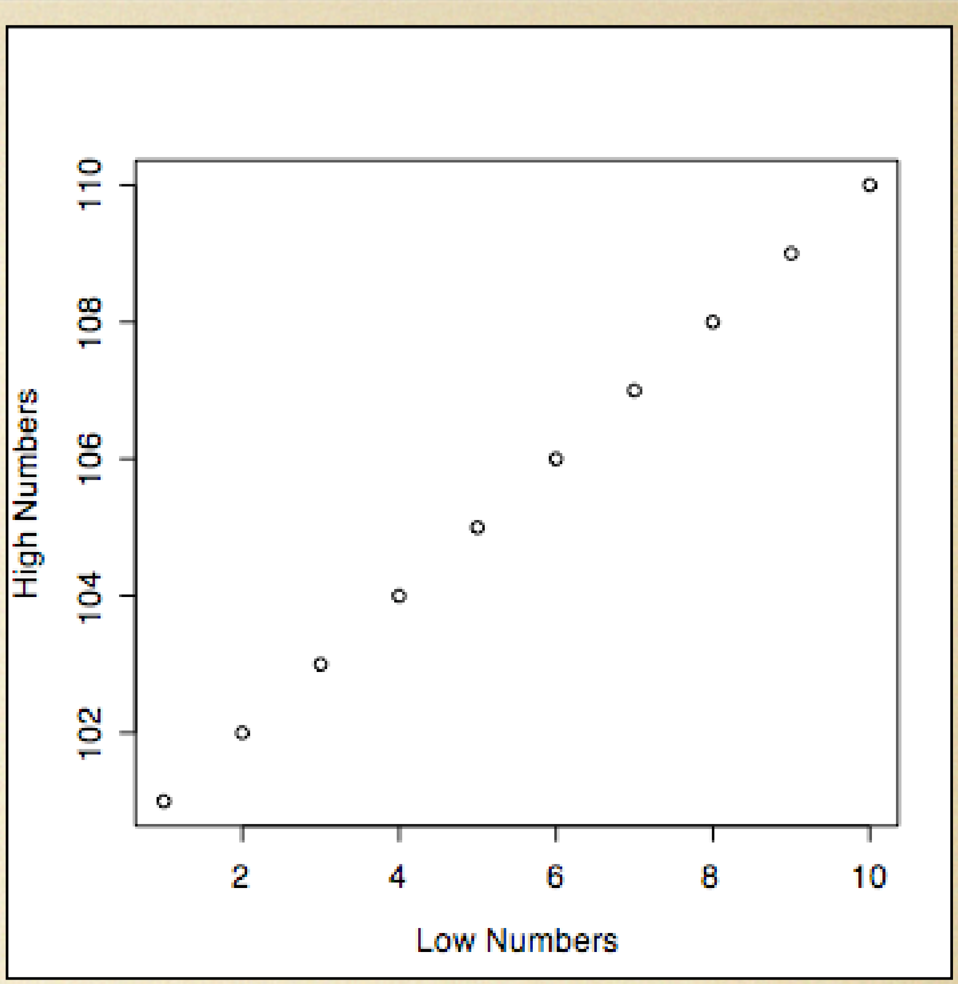
- Dataframe is sort of like a spreadsheet
- Each “row” is an “observation” of 1 or more “variables” (columns)
- Use str() to examine object structure
- rownames() and colnames() are cool

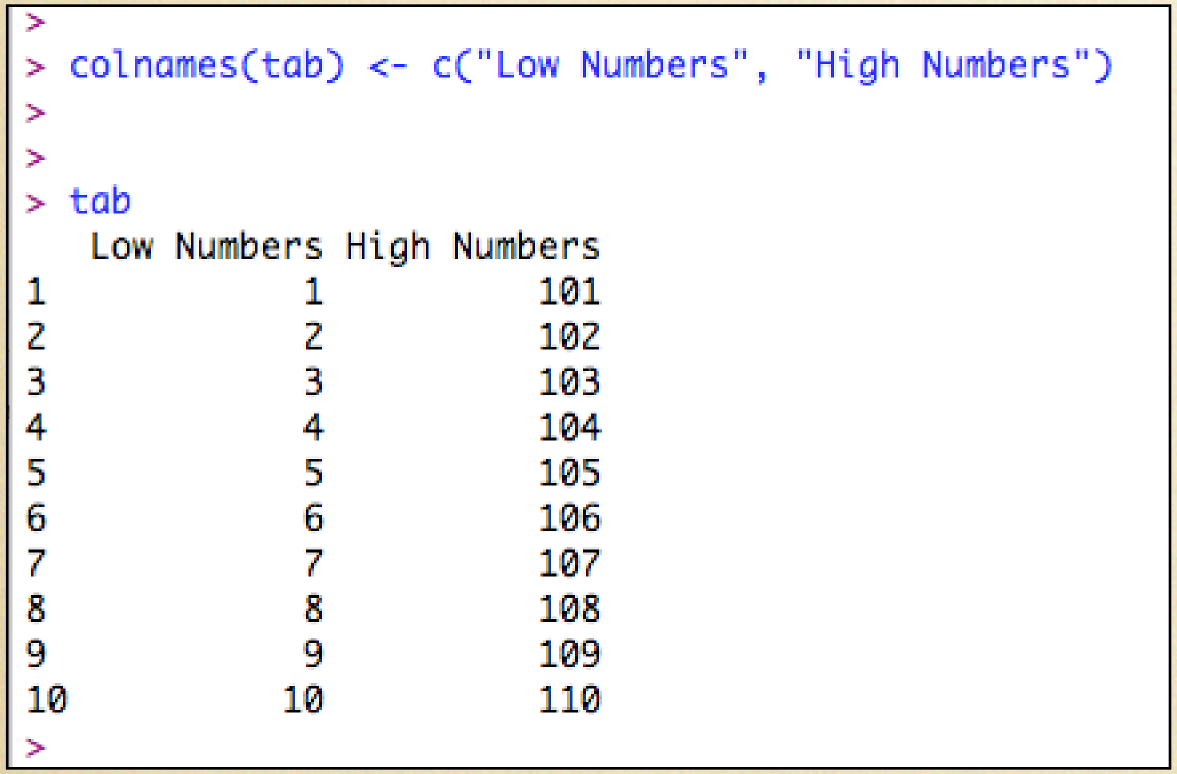


```
>
> tab <- data.frame(x <- 1:10, y <- 101:110)
>
>
> tab
   x....1.10 y....101.110
1          1          101
2          2          102
3          3          103
4          4          104
5          5          105
6          6          106
7          7          107
8          8          108
9          9          109
10        10          110
>
```

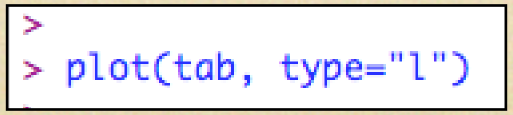


```
>
> plot(tab)
>
```

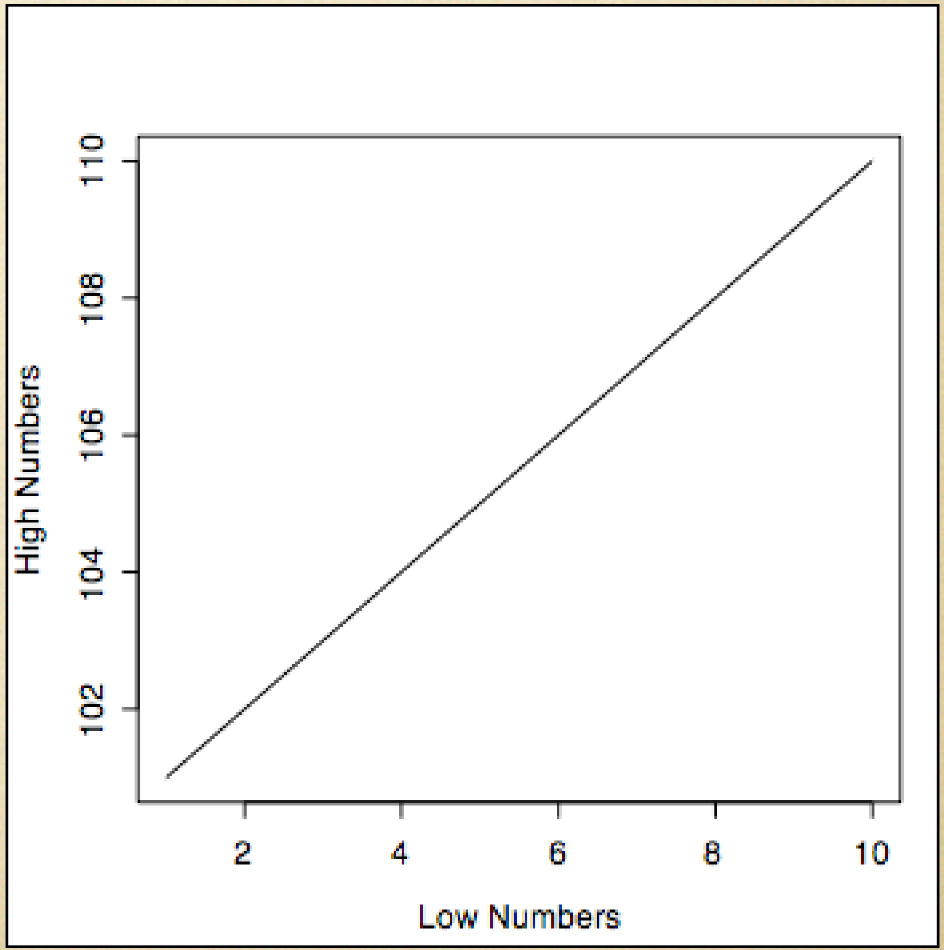


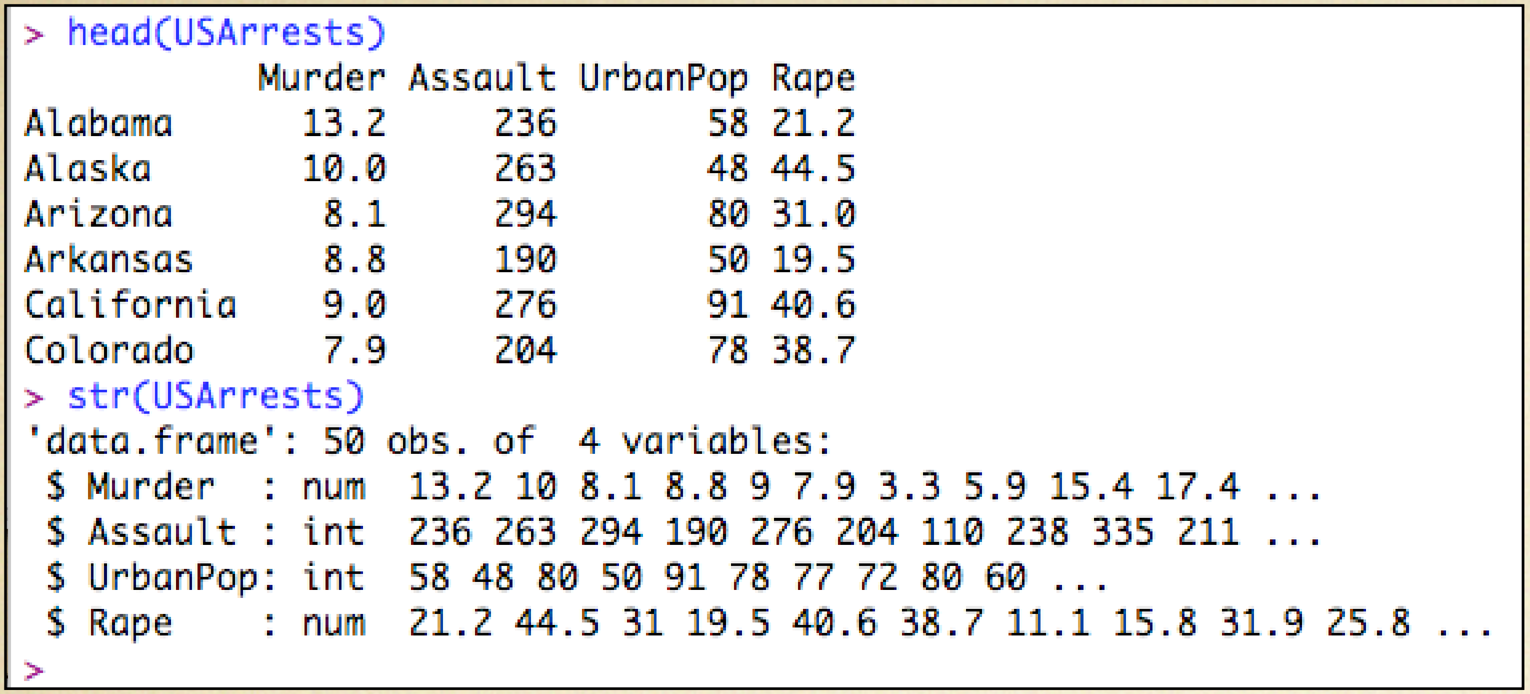


```
>
> colnames(tab) <- c("Low Numbers", "High Numbers")
>
>
> tab
   Low Numbers High Numbers
1            1          101
2            2          102
3            3          103
4            4          104
5            5          105
6            6          106
7            7          107
8            8          108
9            9          109
10          10          110
>
```

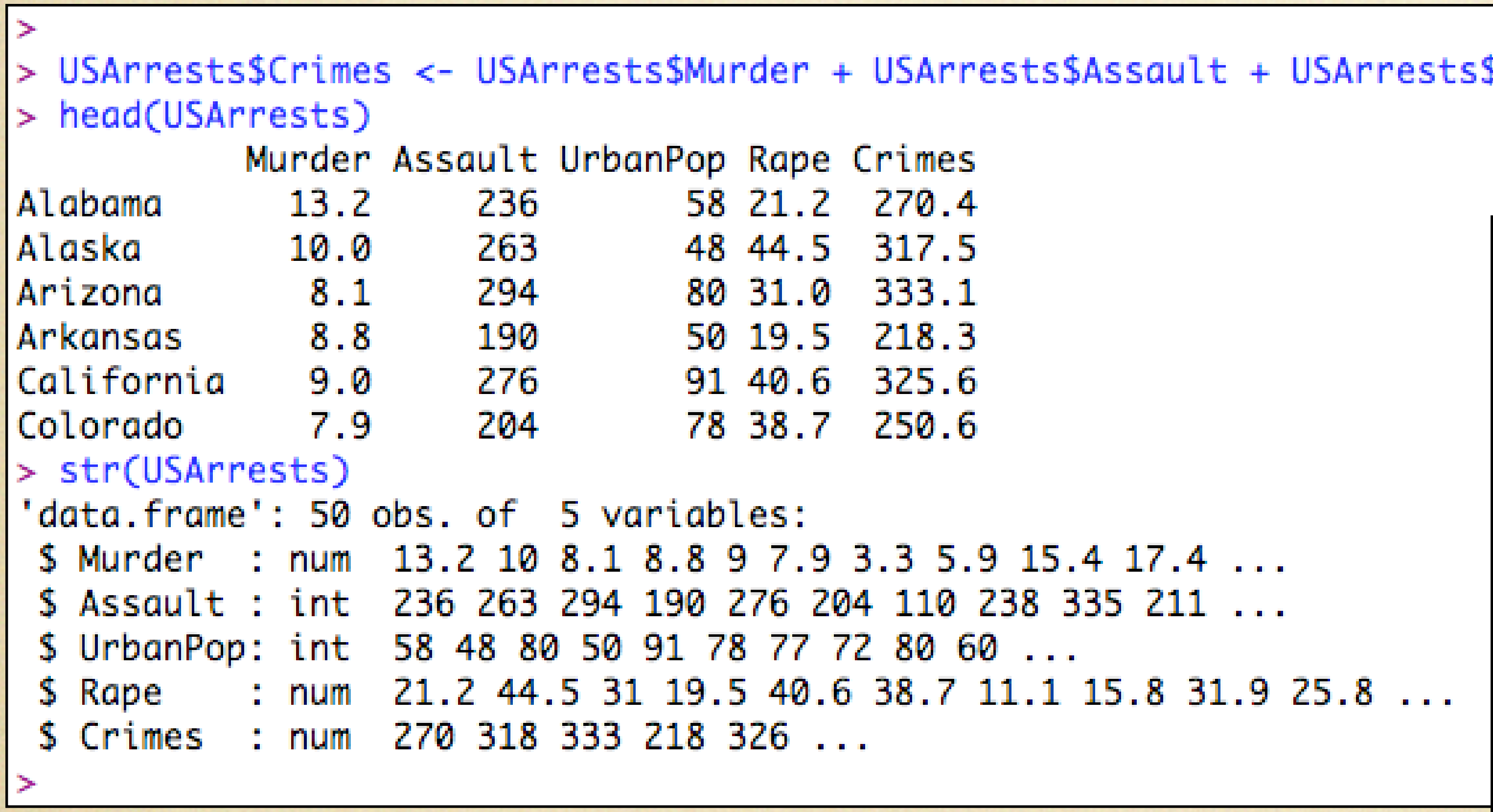


```
>
> plot(tab, type="l")
```

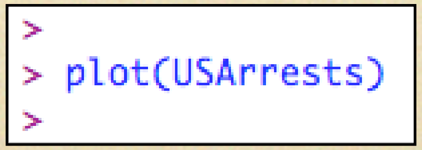




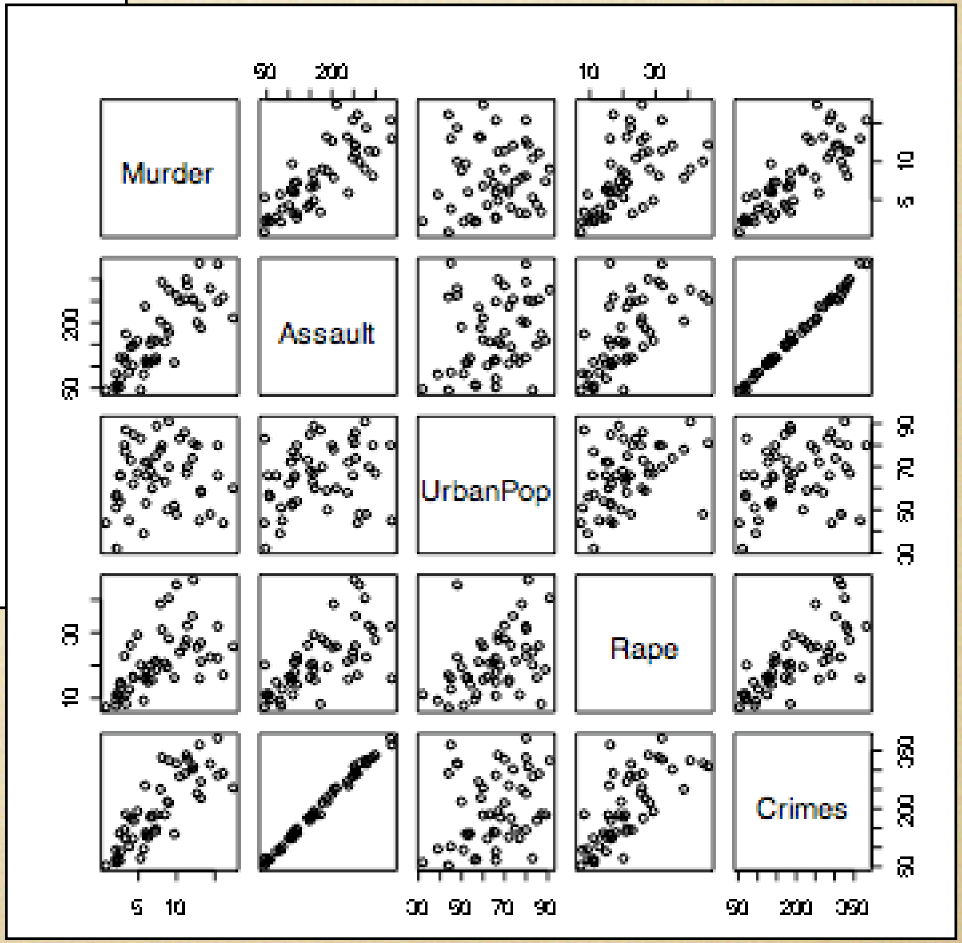
```
> head(USArrests)
           Murder Assault UrbanPop Rape
Alabama      13.2     236       58 21.2
Alaska       10.0     263       48 44.5
Arizona       8.1     294       80 31.0
Arkansas      8.8     190       50 19.5
California    9.0     276       91 40.6
Colorado      7.9     204       78 38.7
> str(USArrests)
'data.frame':	50 obs. of  4 variables:
 $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
 $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
 $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
 $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...
>
```



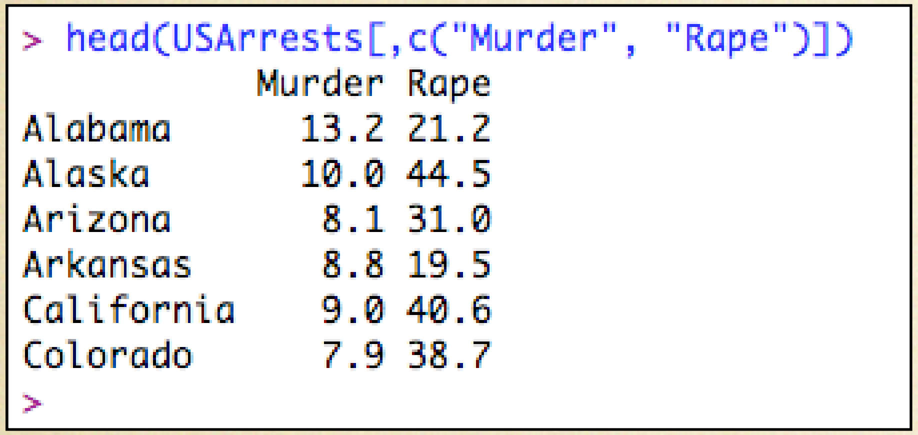
```
>
> USArrests$Crimes <- USArrests$Murder + USArrests$Assault + USArrests$Rape
> head(USArrests)
           Murder Assault UrbanPop Rape Crimes
Alabama      13.2     236       58 21.2  270.4
Alaska       10.0     263       48 44.5  317.5
Arizona       8.1     294       80 31.0  333.1
Arkansas      8.8     190       50 19.5  218.3
California    9.0     276       91 40.6  325.6
Colorado      7.9     204       78 38.7  250.6
> str(USArrests)
'data.frame':	50 obs. of  5 variables:
 $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
 $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
 $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
 $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...
 $ Crimes  : num  270 318 333 218 326 ...
>
```



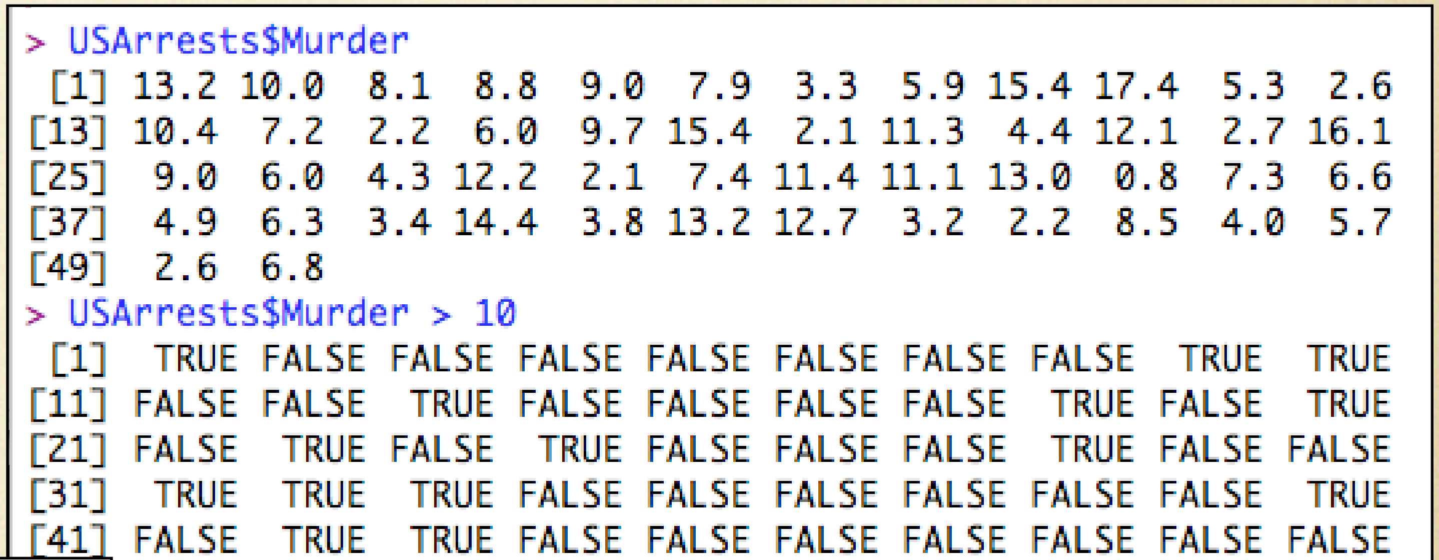
```
>
> plot(USArrests)
>
```



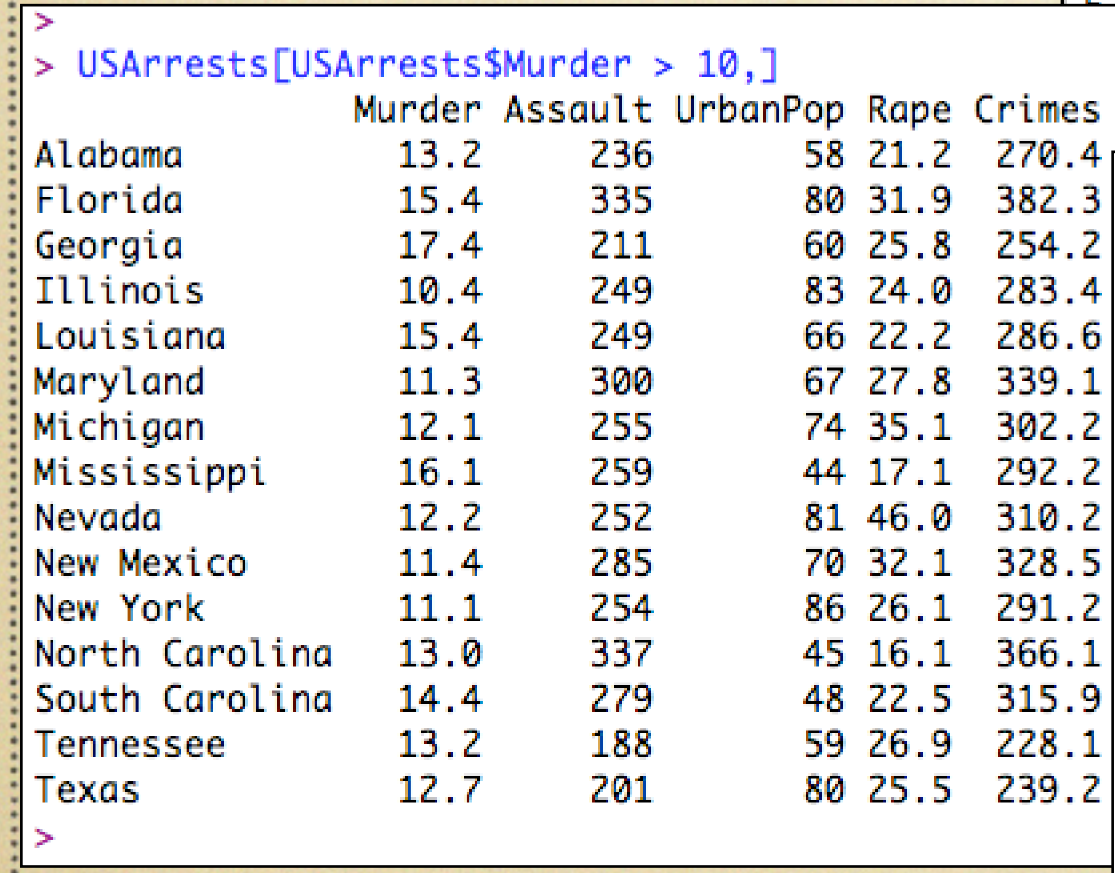
Rows and Columns



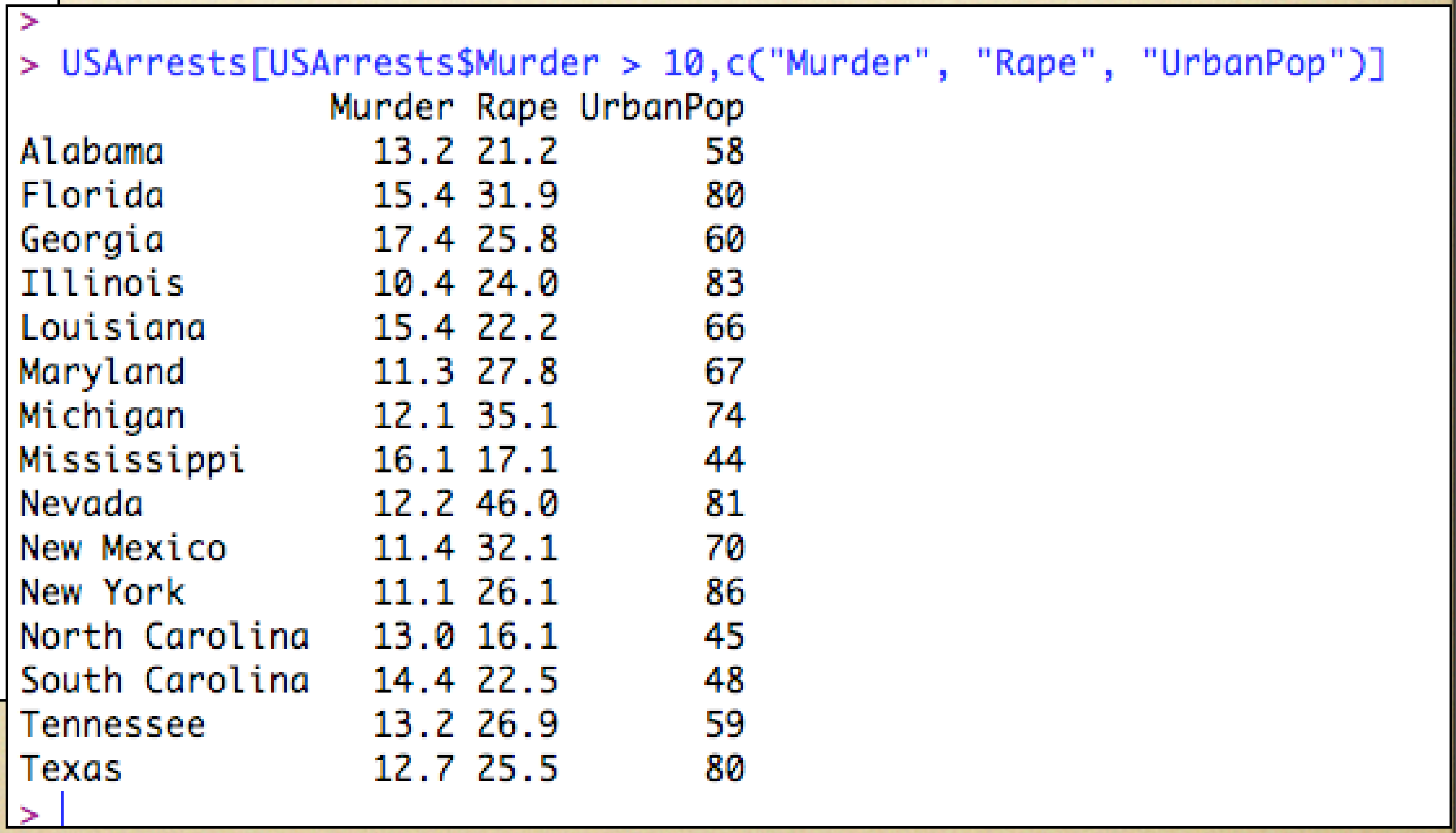
```
> head(USArrests[,c("Murder", "Rape")])
           Murder Rape
Alabama      13.2 21.2
Alaska       10.0 44.5
Arizona       8.1 31.0
Arkansas      8.8 19.5
California    9.0 40.6
Colorado      7.9 38.7
>
```



```
> USArrests$Murder
 [1] 13.2 10.0  8.1  8.8  9.0  7.9  3.3  5.9 15.4 17.4  5.3  2.6
[13] 10.4  7.2  2.2  6.0  9.7 15.4  2.1 11.3  4.4 12.1  2.7 16.1
[25]  9.0  6.0  4.3 12.2  2.1  7.4 11.4 11.1 13.0  0.8  7.3  6.6
[37]  4.9  6.3  3.4 14.4  3.8 13.2 12.7  3.2  2.2  8.5  4.0  5.7
[49]  2.6  6.8
> USArrests$Murder > 10
 [1]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
[11] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE
[21] FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE
[31]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
[41] FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

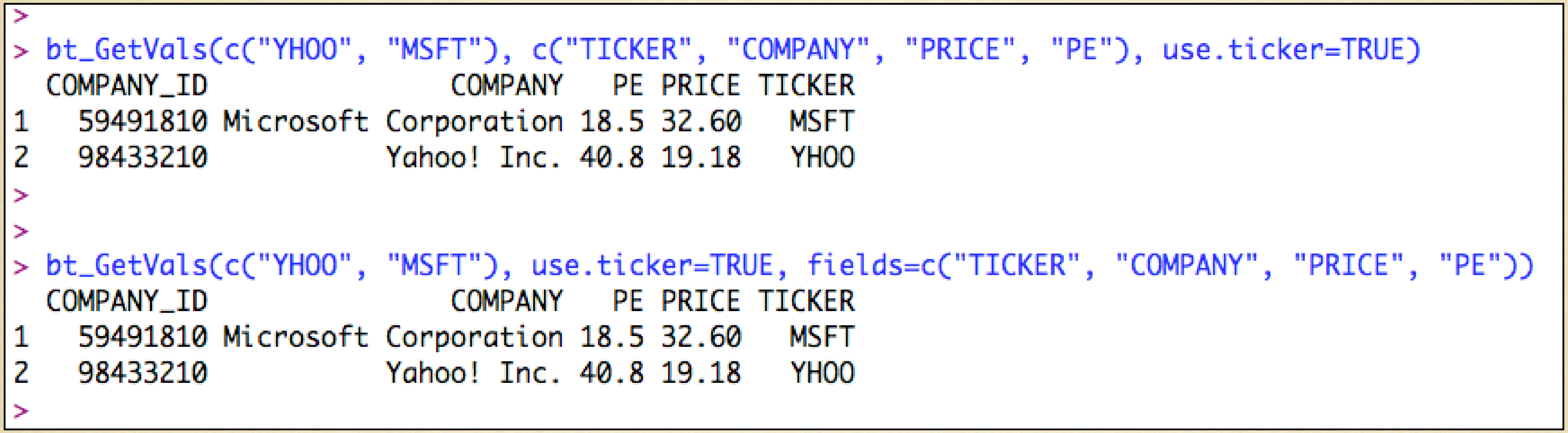


```
>
> USArrests[USArrests$Murder > 10,]
               Murder Assault UrbanPop Rape Crimes
Alabama          13.2     236       58 21.2  270.4
Florida          15.4     335       80 31.9  382.3
Georgia          17.4     211       60 25.8  254.2
Illinois         10.4     249       83 24.0  283.4
Louisiana        15.4     249       66 22.2  286.6
Maryland         11.3     300       67 27.8  339.1
Michigan         12.1     255       74 35.1  302.2
Mississippi      16.1     259       44 17.1  292.2
Nevada           12.2     252       81 46.0  310.2
New Mexico       11.4     285       70 32.1  328.5
New York         11.1     254       86 26.1  291.2
North Carolina   13.0     337       45 16.1  366.1
South Carolina   14.4     279       48 22.5  315.9
Tennessee        13.2     188       59 26.9  228.1
Texas            12.7     201       80 25.5  239.2
>
```

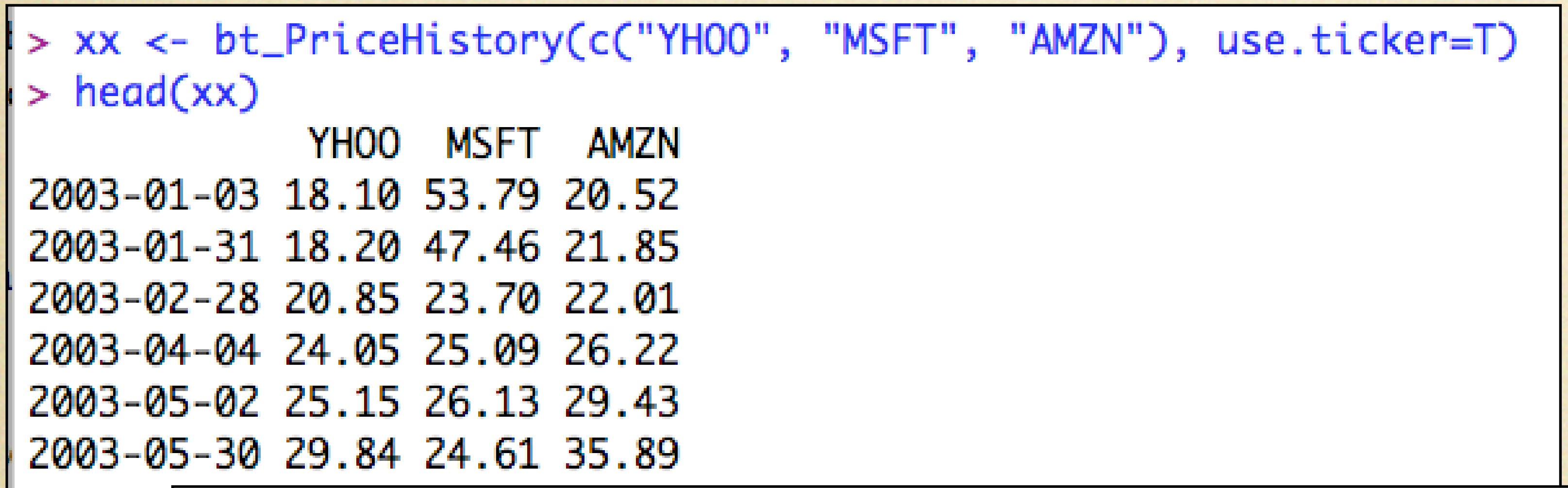


```
>
> USArrests[USArrests$Murder > 10,c("Murder", "Rape", "UrbanPop")]
               Murder Rape UrbanPop
Alabama          13.2 21.2       58
Florida          15.4 31.9       80
Georgia          17.4 25.8       60
Illinois         10.4 24.0       83
Louisiana        15.4 22.2       66
Maryland         11.3 27.8       67
Michigan         12.1 35.1       74
Mississippi      16.1 17.1       44
Nevada           12.2 46.0       81
New Mexico       11.4 32.1       70
New York         11.1 26.1       86
North Carolina   13.0 16.1       45
South Carolina   14.4 22.5       48
Tennessee        13.2 26.9       59
Texas            12.7 25.5       80
>
```

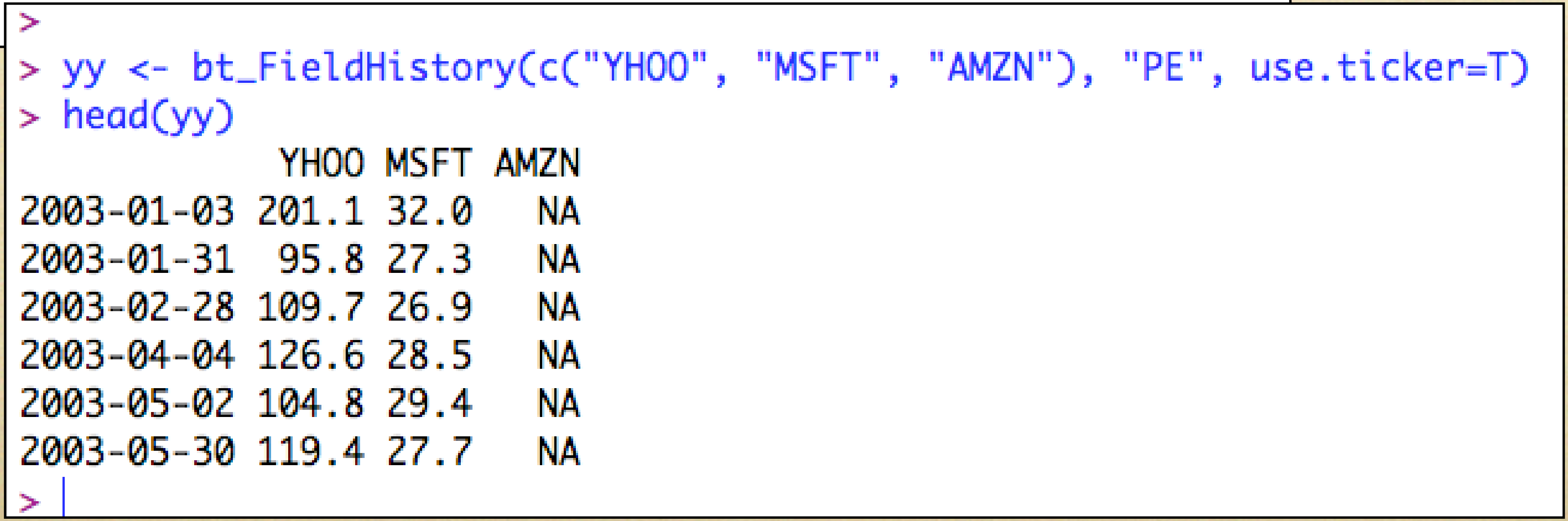
Calling Functions



```
>
> bt_GetVals(c("YHOO", "MSFT"), c("TICKER", "COMPANY", "PRICE", "PE"), use.ticker=TRUE)
  COMPANY_ID               COMPANY   PE PRICE TICKER
1   59491810 Microsoft Corporation 18.5 32.60   MSFT
2   98433210           Yahoo! Inc. 40.8 19.18   YHOO
>
>
> bt_GetVals(c("YHOO", "MSFT"), use.ticker=TRUE, fields=c("TICKER", "COMPANY", "PRICE", "PE"))
  COMPANY_ID               COMPANY   PE PRICE TICKER
1   59491810 Microsoft Corporation 18.5 32.60   MSFT
2   98433210           Yahoo! Inc. 40.8 19.18   YHOO
>
```

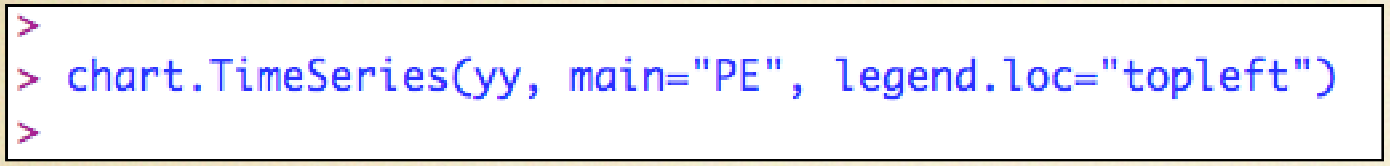


```
> xx <- bt_PriceHistory(c("YHOO", "MSFT", "AMZN"), use.ticker=T)
> head(xx)
            YHOO  MSFT  AMZN
2003-01-03 18.10 53.79 20.52
2003-01-31 18.20 47.46 21.85
2003-02-28 20.85 23.70 22.01
2003-04-04 24.05 25.09 26.22
2003-05-02 25.15 26.13 29.43
2003-05-30 29.84 24.61 35.89
>
```

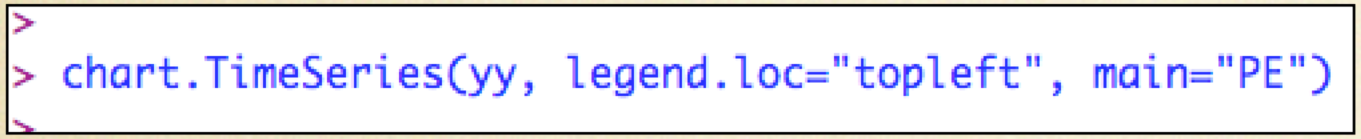


```
>
> yy <- bt_FieldHistory(c("YHOO", "MSFT", "AMZN"), "PE", use.ticker=T)
> head(yy)
            YHOO MSFT AMZN
2003-01-03 201.1 32.0   NA
2003-01-31  95.8 27.3   NA
2003-02-28 109.7 26.9   NA
2003-04-04 126.6 28.5   NA
2003-05-02 104.8 29.4   NA
2003-05-30 119.4 27.7   NA
>
```

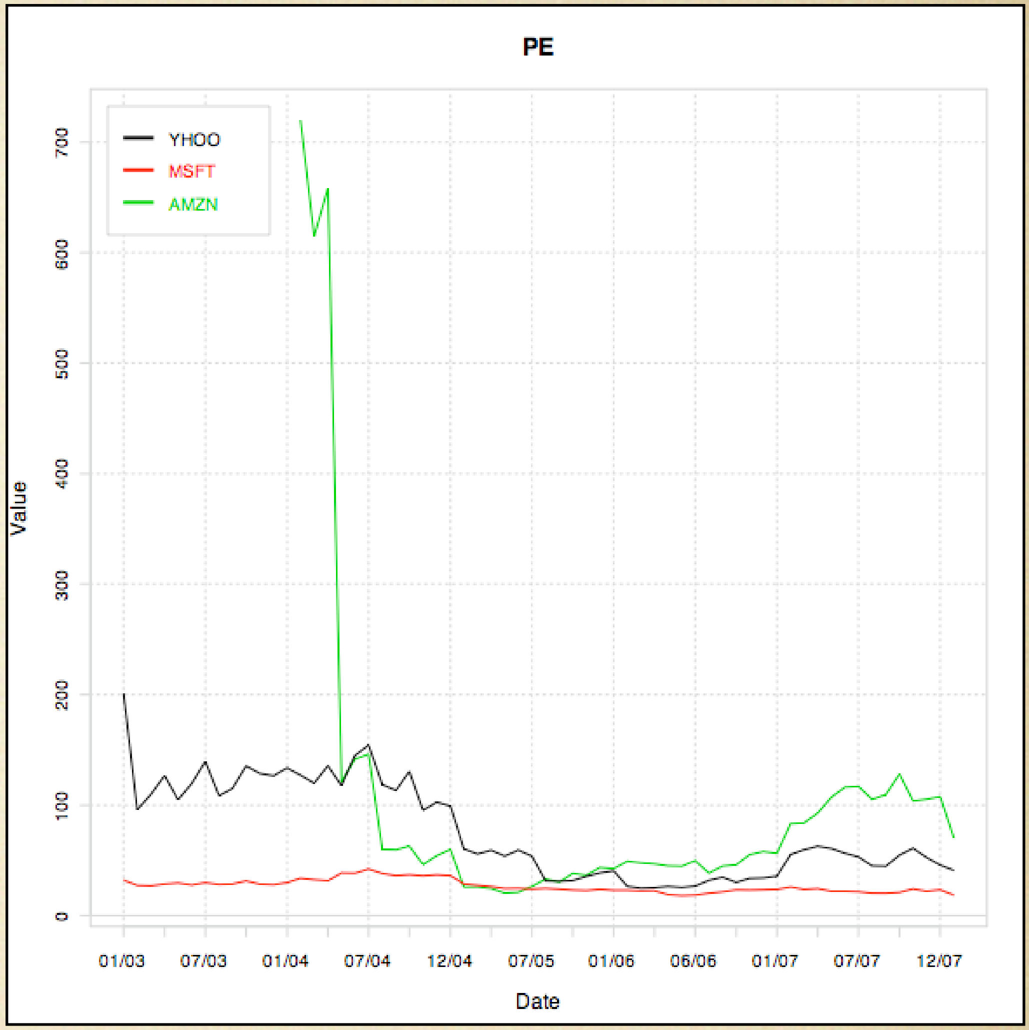
PerformanceAnalytics Charts

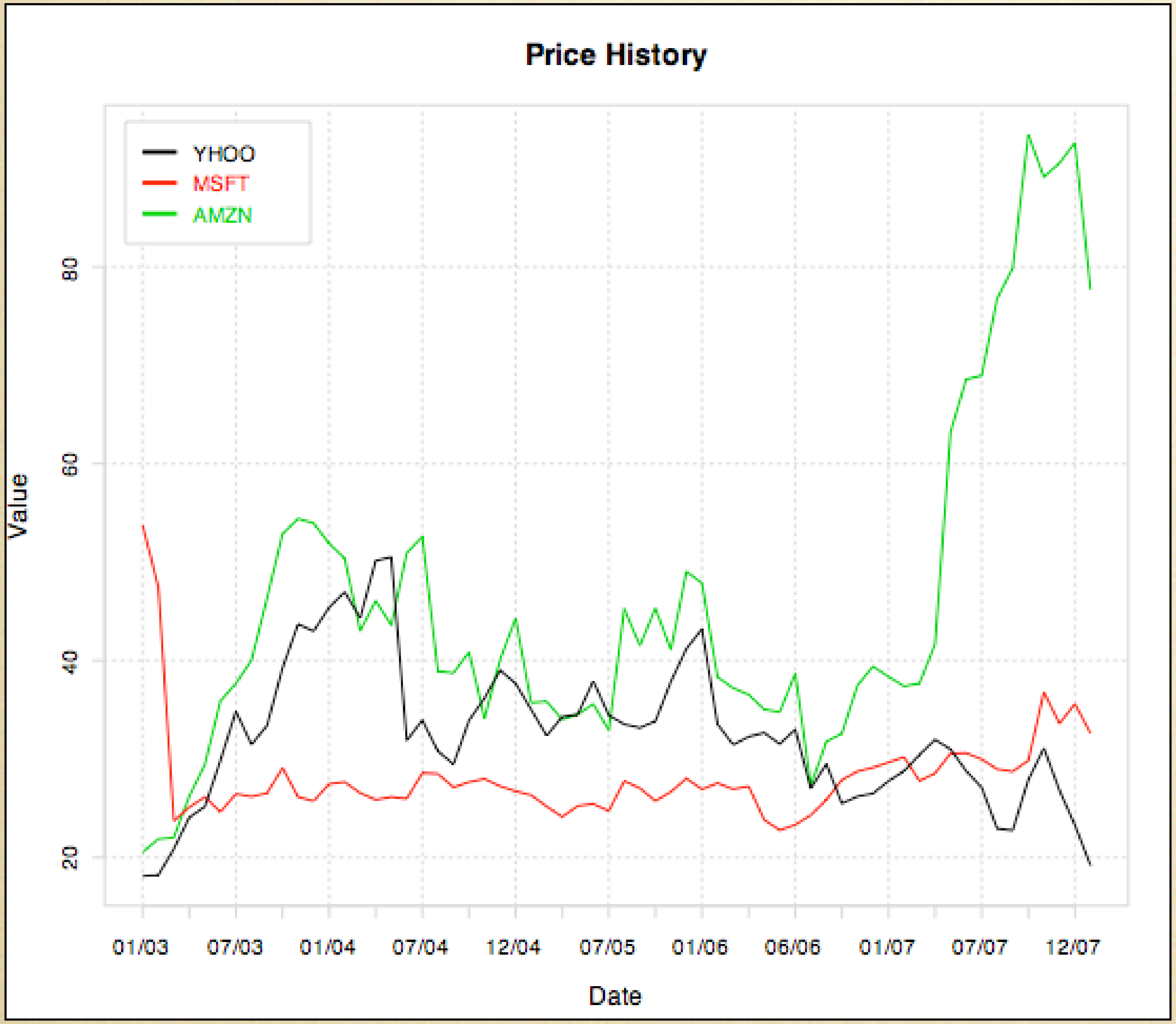


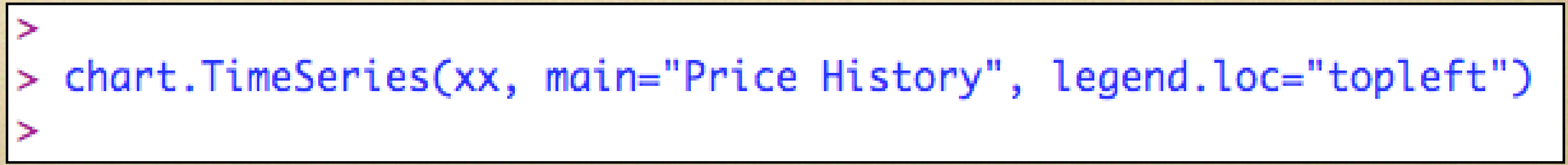
```
>
> chart.TimeSeries(yy, main="PE", legend.loc="topleft")
>
```



```
>
> chart.TimeSeries(yy, legend.loc="topleft", main="PE")
>
```

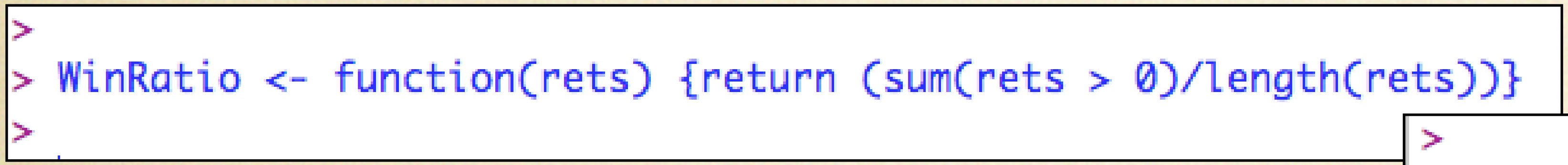




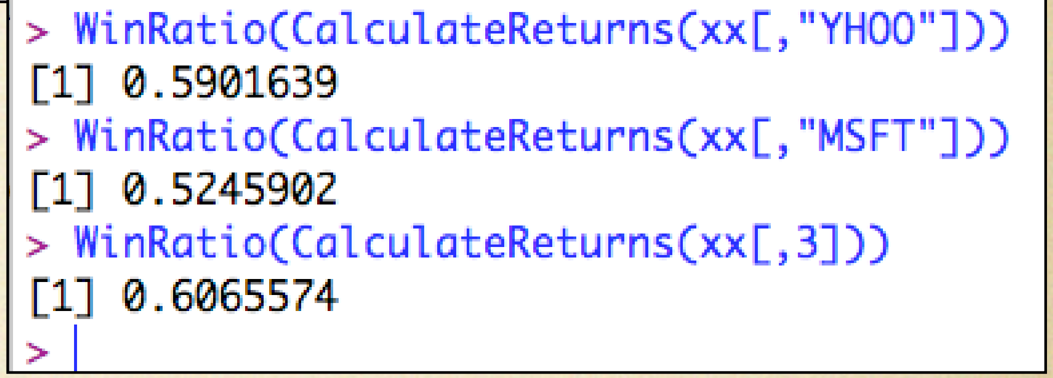


```
>
> chart.TimeSeries(xx, main="Price History", legend.loc="topleft")
>
```

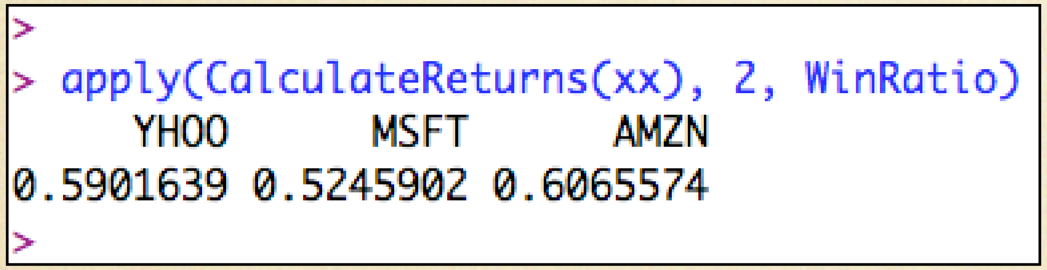
Defining Functions



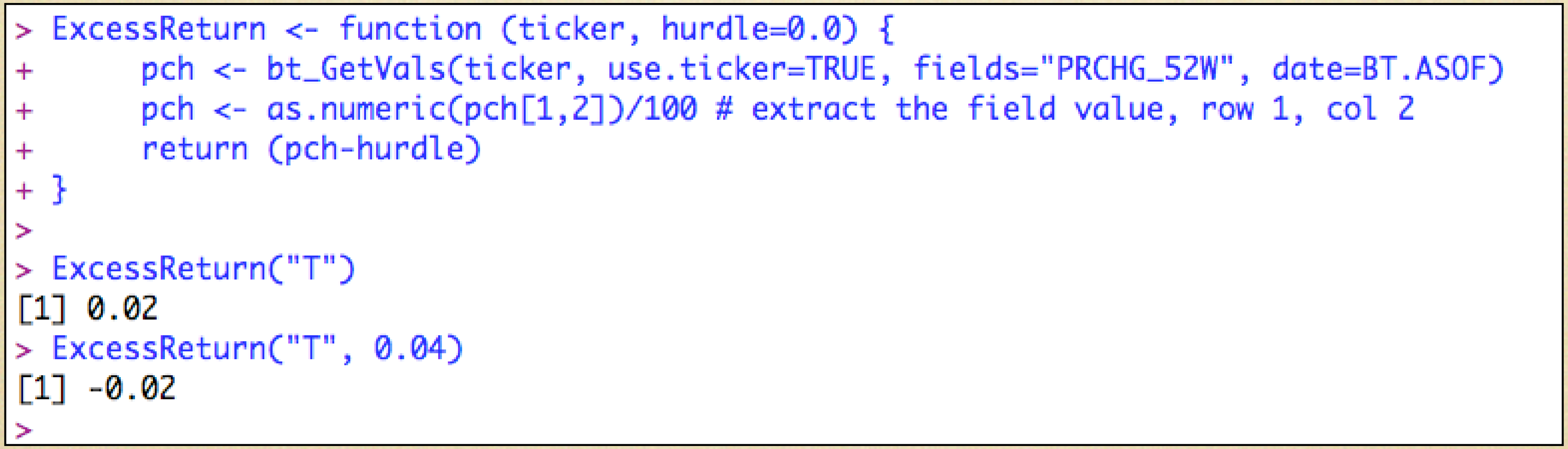
```
>
> WinRatio <- function(rets) {return (sum(rets > 0)/length(rets))}
>
```



```
>
> WinRatio(CalculateReturns(xx[,"YHOO"]))
[1] 0.5901639
> WinRatio(CalculateReturns(xx[,"MSFT"]))
[1] 0.5245902
> WinRatio(CalculateReturns(xx[,3]))
[1] 0.6065574
> |
```



```
>
> apply(CalculateReturns(xx), 2, WinRatio)
     YHOO      MSFT      AMZN 
0.5901639 0.5245902 0.6065574 
>
```



```
> ExcessReturn <- function (ticker, hurdle=0.0) {
+      pch <- bt_GetVals(ticker, use.ticker=TRUE, fields="PRCHG_52W", date=BT.ASOF)
+      pch <- as.numeric(pch[1,2])/100 # extract the field value, row 1, col 2
+      return (pch-hurdle)
+ }
>
> ExcessReturn("T")
[1] 0.02
> ExcessReturn("T", 0.04)
[1] -0.02
>
```

Flow Control

- if (x > 0) {cat(“Yes, x > 0\n”)} else {cat(“no\n”)}
- for (i in 1:10) {...}
- sapply, apply, lapply
- return()

Useful “R” Functions

- apply, lapply, sapply, matrix ops (e.g. %*% ...)
- length, which, index, [], [[]], subset, colnames, rownames, names, dim, %in%
- str, class, type, summary, ls
- merge, rbind, cbind, zoo, plot, abline, points
- save, save.image, load
- print, cat, trace, head, tail
- ?<command> (e.g. ?sapply), help.search

Agenda

- Screening vs. Backtesting
- Overview / PAST-SIPro Demo
- Setup PAST-SIPro
- The "R" programming language
- Sample backtests (Simple to Sophisticated)
- From Screens to Backtests
- ToDo List
- Questions?

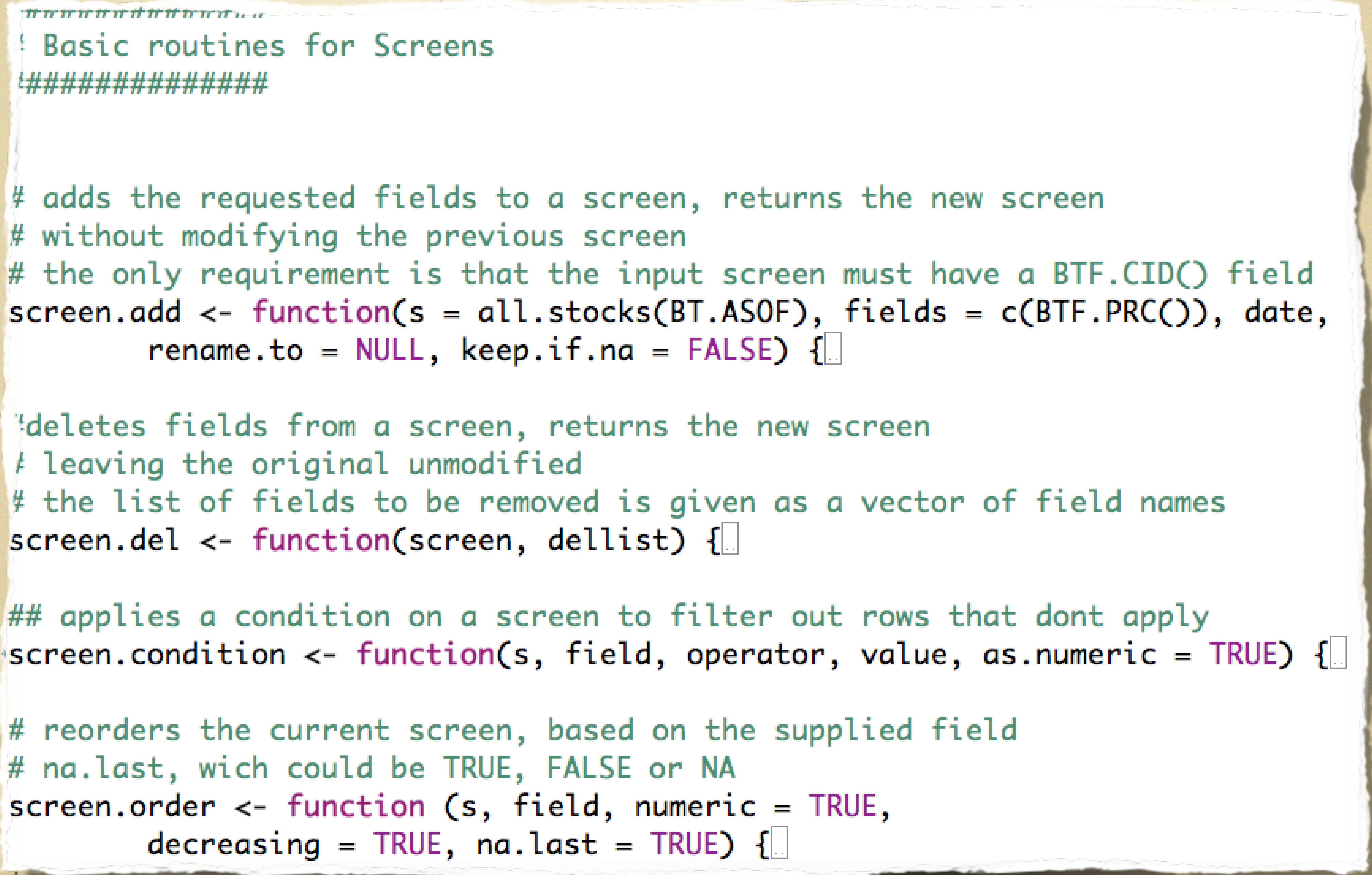
Simple Backtesting

- Recall - screen: Date -> Asset List
- simulation: Screen -> sim object
- bt_Returns: sim object -> zoo() returns
- zoo returns can be viewed/processed by PerformanceAnalytics

Building Screens

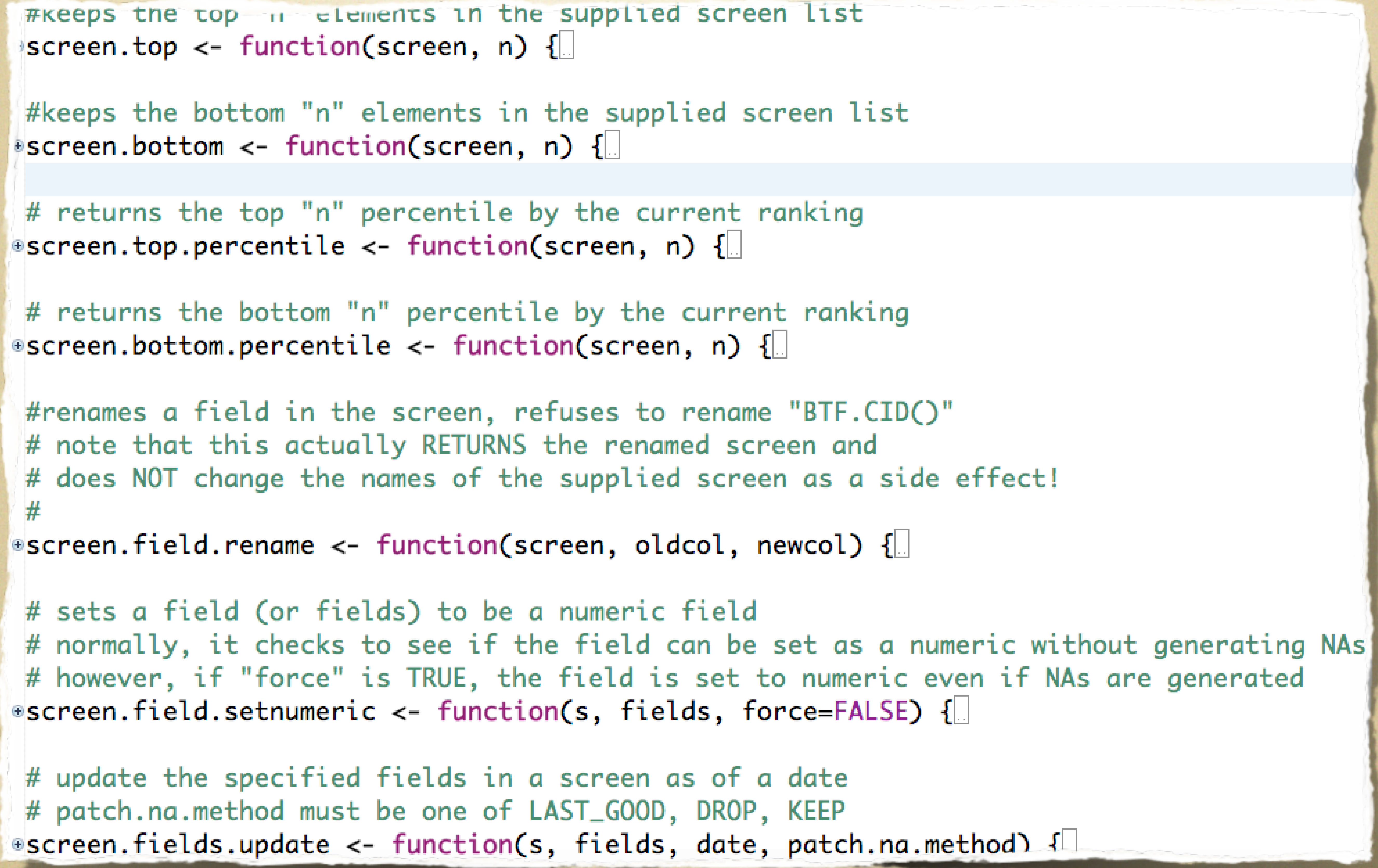
- start with all.stocks(date)
- add columns with screen.add()
- add conditions with screen.condition()
- delete fields (optional) with screen.del
- return the final screen

Screen Building Functions (1)



```
###############
# Basic routines for Screens
###############



# adds the requested fields to a screen, returns the new screen
# without modifying the previous screen
# the only requirement is that the input screen must have a BTF.CID() field
screen.add <- function(s = all.stocks(BT.ASOF), fields = c(BTF.PRC()), date,
        rename.to = NULL, keep.if.na = FALSE) {…

#deletes fields from a screen, returns the new screen
# leaving the original unmodified
# the list of fields to be removed is given as a vector of field names
screen.del <- function(screen, dellist) {…

## applies a condition on a screen to filter out rows that dont apply
screen.condition <- function(s, field, operator, value, as.numeric = TRUE) {…

# reorders the current screen, based on the supplied field
# na.last, wich could be TRUE, FALSE or NA
screen.order <- function (s, field, numeric = TRUE,
        decreasing = TRUE, na.last = TRUE) {…
```

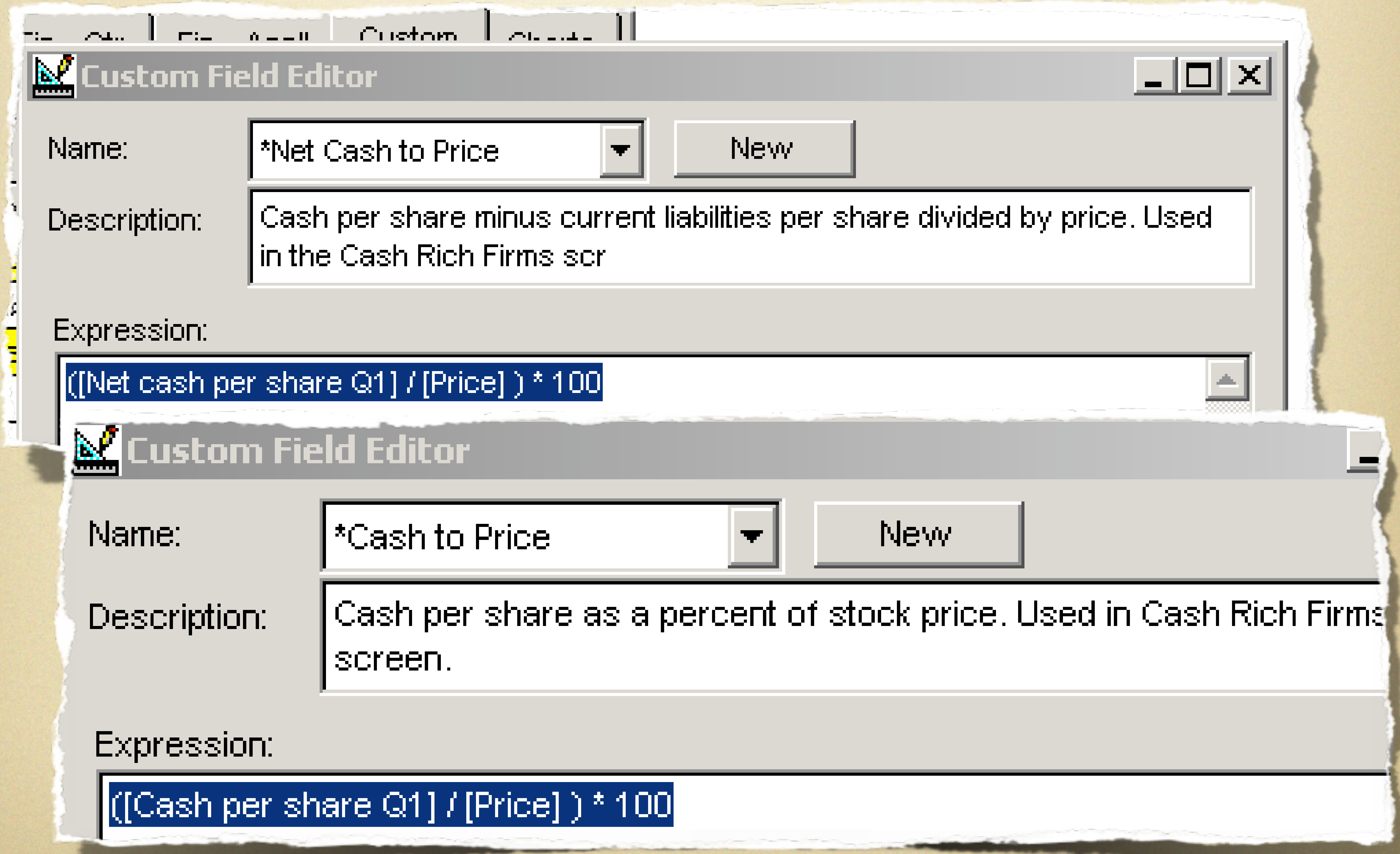
Screen Building Functions (2)



```
#keeps the top "n" elements in the supplied screen list
screen.top <- function(screen, n) {

#keeps the bottom "n" elements in the supplied screen list
screen.bottom <- function(screen, n) {

# returns the top "n" percentile by the current ranking
screen.top.percentile <- function(screen, n) {

# returns the bottom "n" percentile by the current ranking
screen.bottom.percentile <- function(screen, n) {

#renames a field in the screen, refuses to rename "BTF.CID()"
# note that this actually RETURNS the renamed screen and
# does NOT change the names of the supplied screen as a side effect!
#
screen.field.rename <- function(screen, oldcol, newcol) {

# sets a field (or fields) to be a numeric field
# normally, it checks to see if the field can be set as a numeric without generating NAs
# however, if "force" is TRUE, the field is set to numeric even if NAs are generated
screen.field.setnumeric <- function(s, fields, force=FALSE) {

# update the specified fields in a screen as of a date
# patch.na.method must be one of LAST_GOOD, DROP, KEEP
screen.fields.update <- function(s, fields, date, patch.na.method) {
```

Custom Fields

- Cash Rich Firms screen uses 2 custom fields



Name: *Net Cash to Price

Description: Cash per share minus current liabilities per share divided by price. Used in the Cash Rich Firms scr

Expression:

```
([Net cash per share Q1] / [Price] ) * 100
```

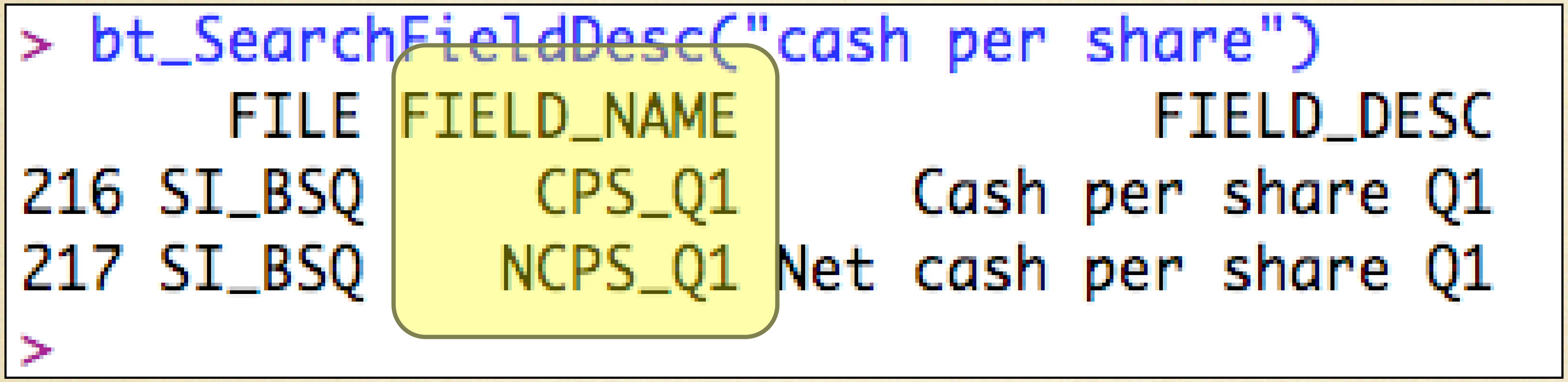
Name: *Cash to Price

Description: Cash per share as a percent of stock price. Used in Cash Rich Firms screen.

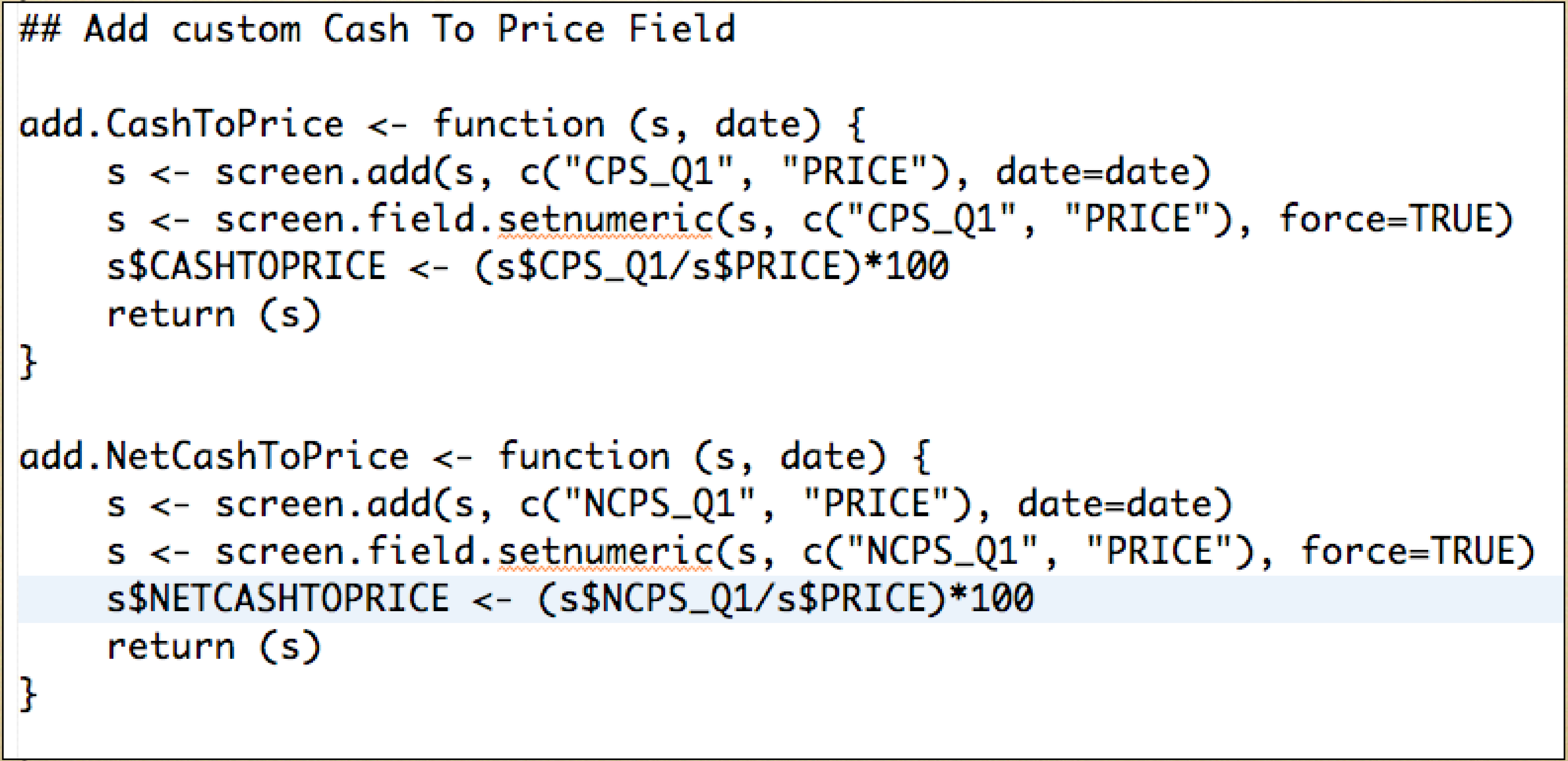
Expression:

```
([Cash per share Q1] / [Price] ) * 100
```

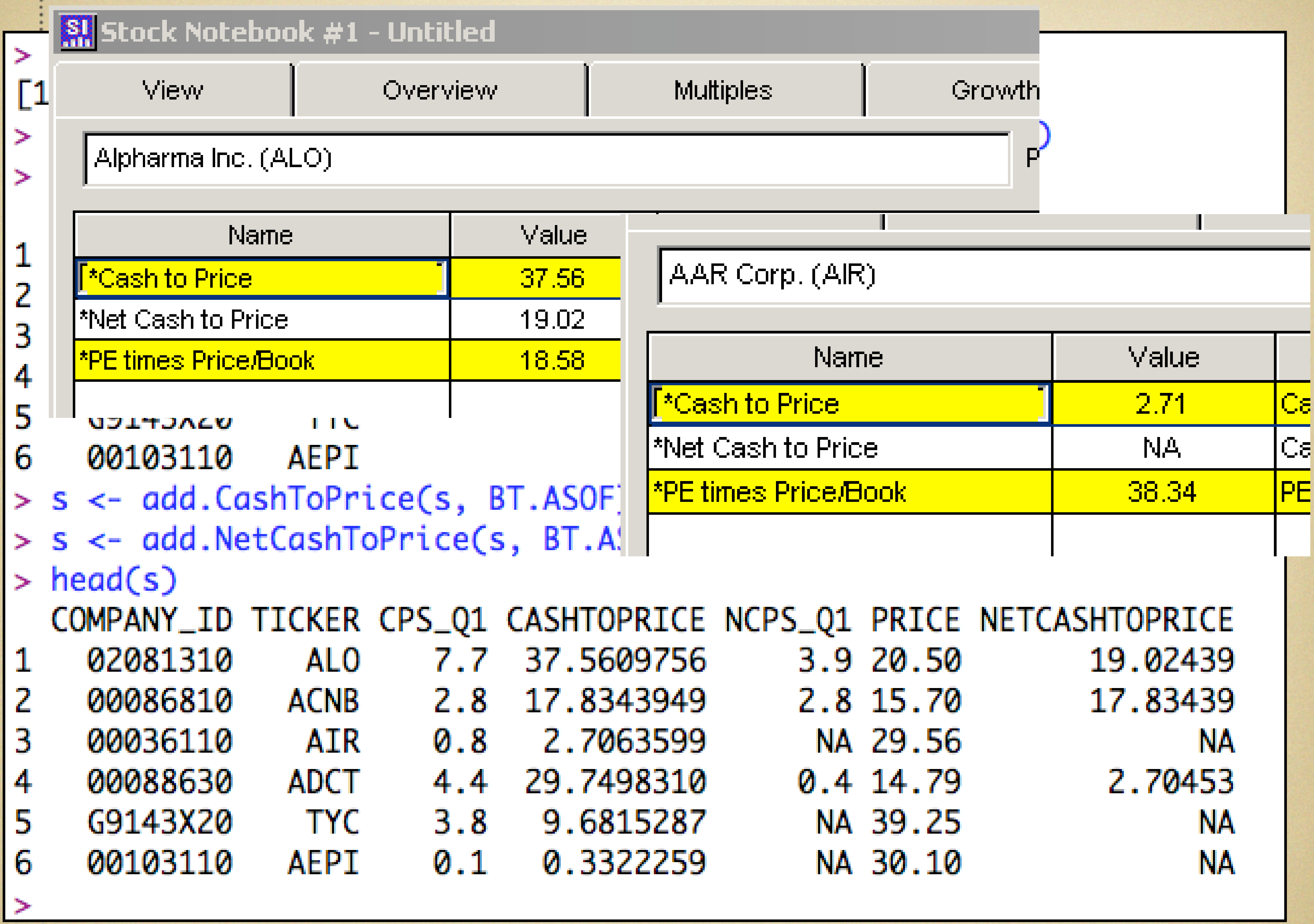
Add Custom Fields



```
> bt_SearchFieldDesc("cash per share")
      FILE FIELD_NAME             FIELD_DESC
216 SI_BSQ     CPS_Q1     Cash per share Q1
217 SI_BSQ    NCPS_Q1 Net cash per share Q1
>
```



```
## Add custom Cash To Price Field

add.CashToPrice <- function (s, date) {
    s <- screen.add(s, c("CPS_Q1", "PRICE"), date=date)
    s <- screen.field.setnumeric(s, c("CPS_Q1", "PRICE"), force=TRUE)
    s$CASHTOPRICE <- (s$CPS_Q1/s$PRICE)*100
    return (s)
}

add.NetCashToPrice <- function (s, date) {
    s <- screen.add(s, c("NCPS_Q1", "PRICE"), date=date)
    s <- screen.field.setnumeric(s, c("NCPS_Q1", "PRICE"), force=TRUE)
    s$NETCASHTOPRICE <- (s$NCPS_Q1/s$PRICE)*100
    return (s)
}
```



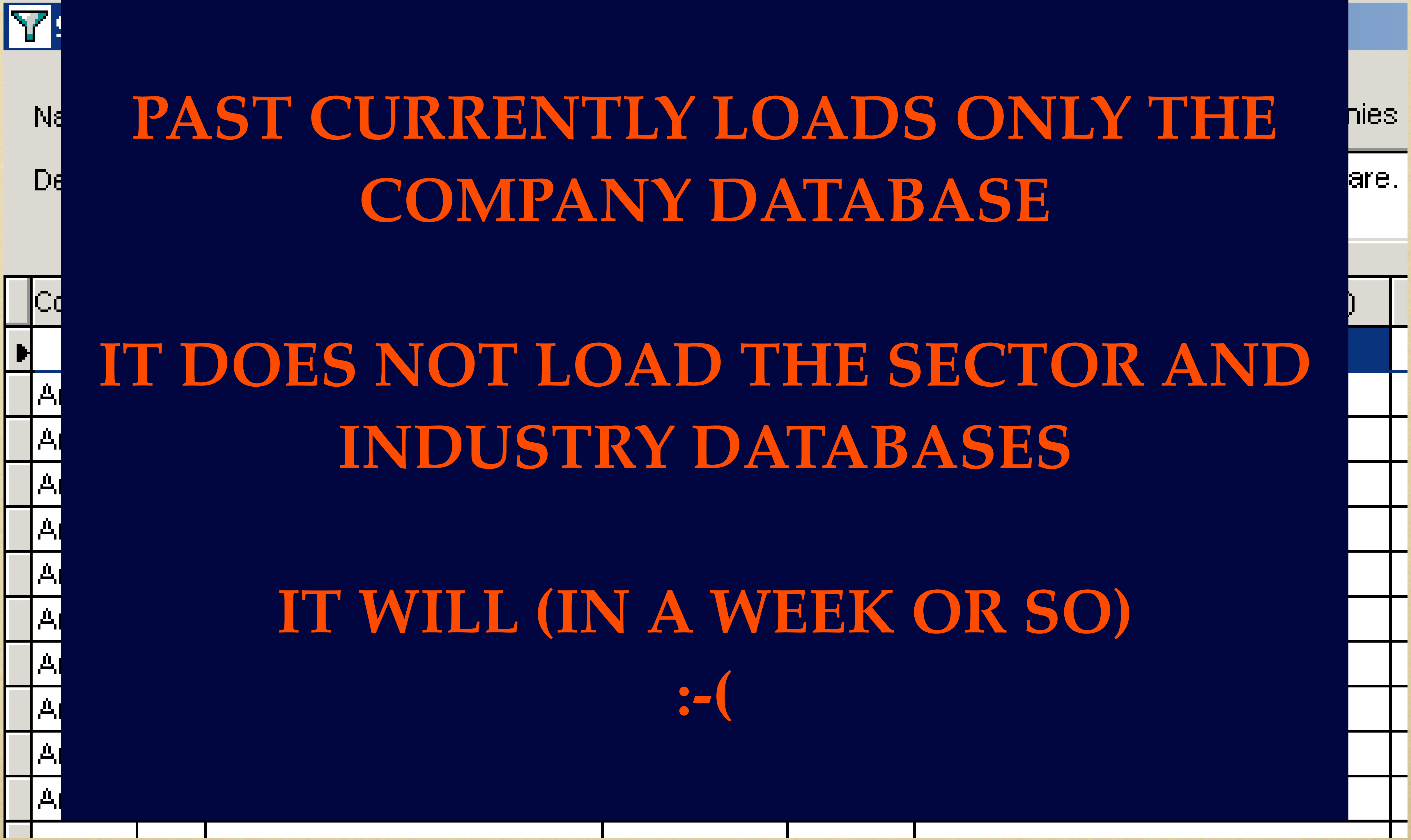
Alpharma Inc. (ALO)

Name	Value
*Cash to Price	37.56
*Net Cash to Price	19.02
*PE times Price/Book	18.58

AAR Corp. (AIR)

Name	Value	
*Cash to Price	2.71	Ca
*Net Cash to Price	NA	Ca
*PE times Price/Book	38.34	PE

```
>
[1
>
>

1
2
3
4
5   G9143X20    TYC
6   00103110   AEPI
> s <- add.CashToPrice(s, BT.ASOF
> s <- add.NetCashToPrice(s, BT.A
> head(s)
  COMPANY_ID TICKER CPS_Q1 CASHTOPRICE NCPS_Q1 PRICE NETCASHTOPRICE
1   02081310    ALO    7.7  37.5609756     3.9 20.50       19.02439
2   00086810   ACNB    2.8  17.8343949     2.8 15.70       17.83439
3   00036110    AIR    0.8   2.7063599      NA 29.56             NA
4   00088630   ADCT    4.4  29.7498310     0.4 14.79        2.70453
5   G9143X20    TYC    3.8   9.6815287      NA 39.25             NA
6   00103110   AEPI    0.1   0.3322259      NA 30.10             NA
>
```



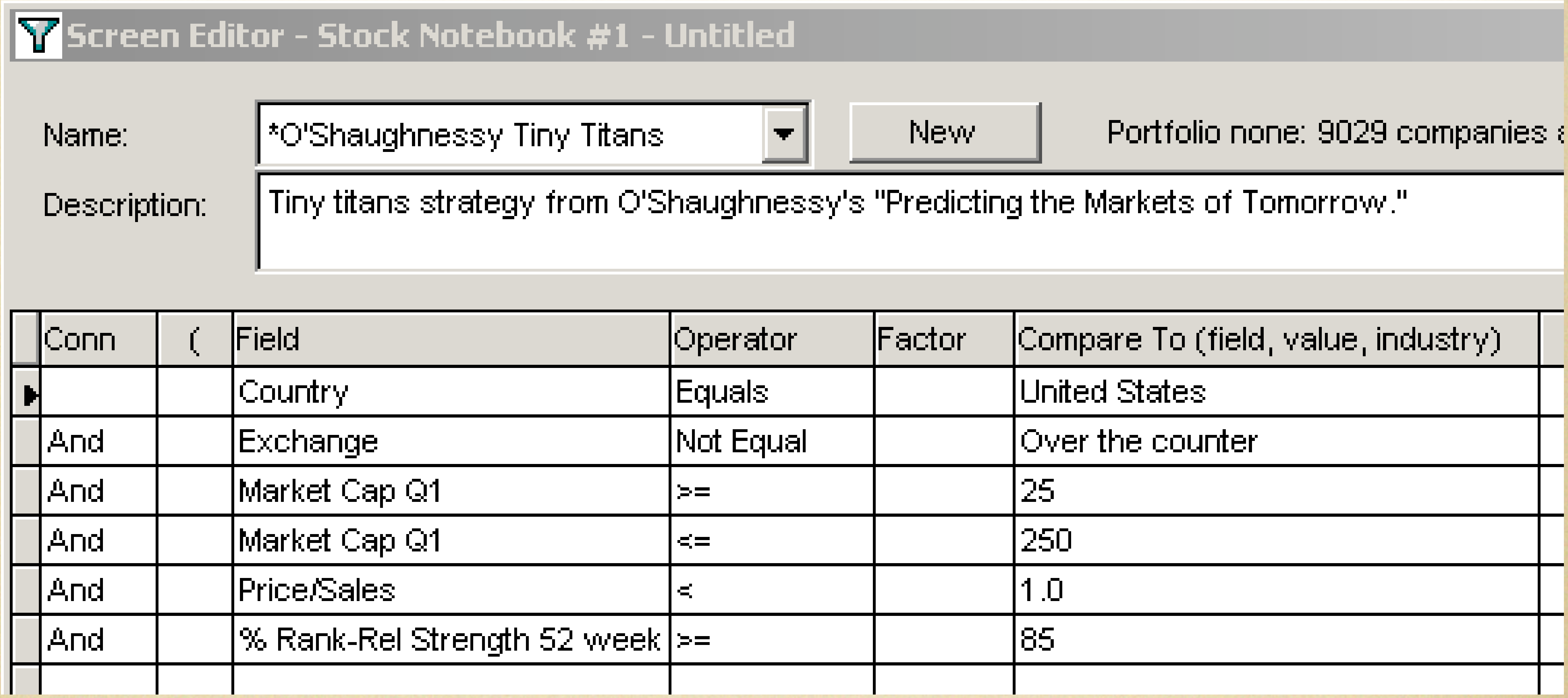
UNFORTUNATELY
WE CAN'T!!
(YET)

PAST CURRENTLY LOADS ONLY THE COMPANY DATABASE

IT DOES NOT LOAD THE SECTOR AND INDUSTRY DATABASES

IT WILL (IN A WEEK OR SO)
:-(

Let's build the "Tiny Titans" screen



Screen Editor - Stock Notebook #1 - Untitled

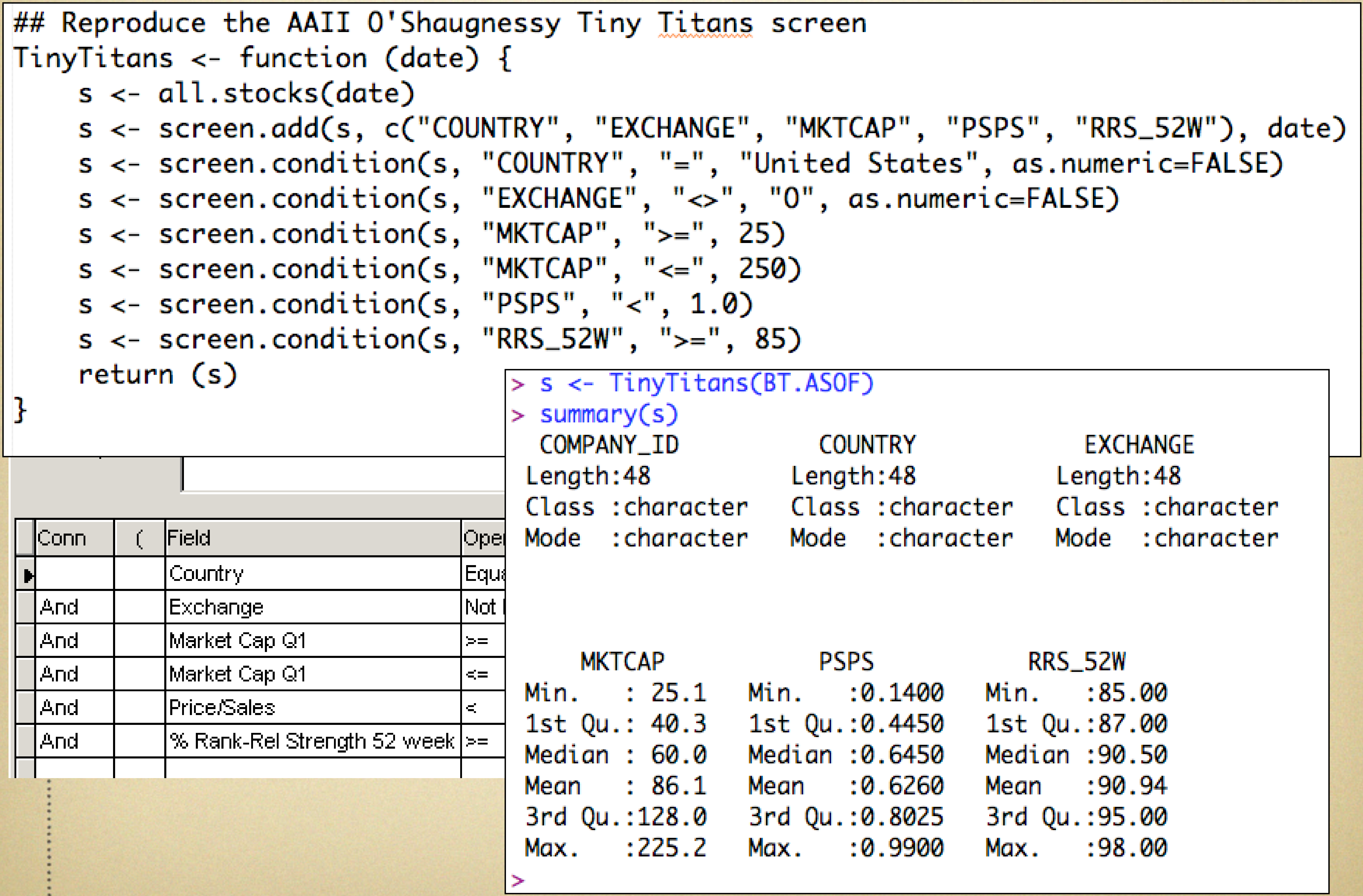
Name: *O'Shaughnessy Tiny Titans

Portfolio none: 9029 companies

Description: Tiny titans strategy from O'Shaughnessy's "Predicting the Markets of Tomorrow."

Conn	(	Field	Operator	Factor	Compare To (field, value, industry)
		Country	Equals		United States
And		Exchange	Not Equal		Over the counter
And		Market Cap Q1	>=		25
And		Market Cap Q1	<=		250
And		Price/Sales	<		1.0
And		% Rank-Rel Strength 52 week	>=		85

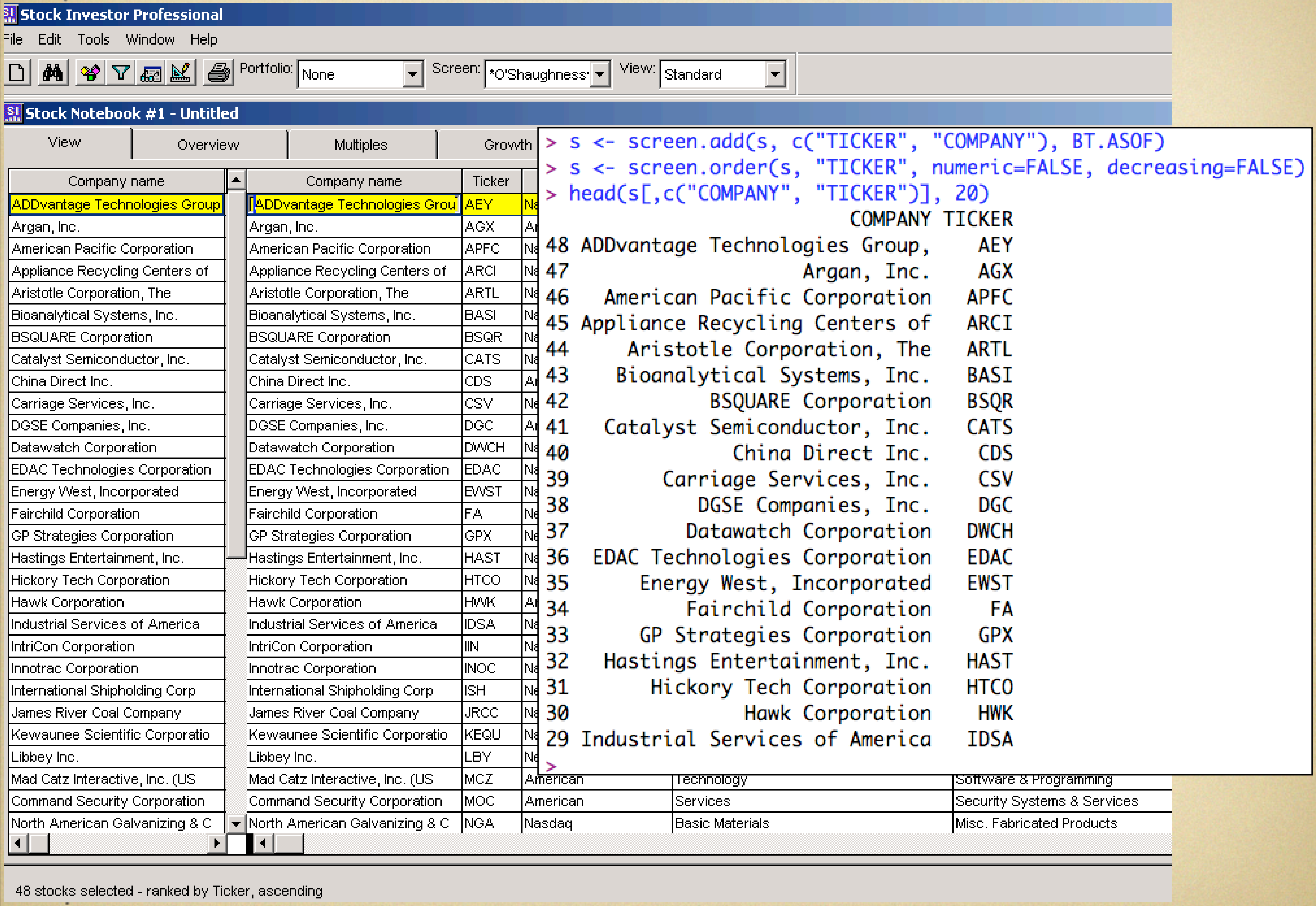
TinyTitans is easy!



```
## Reproduce the AAII O'Shaugnessy Tiny Titans screen
TinyTitans <- function (date) {
    s <- all.stocks(date)
    s <- screen.add(s, c("COUNTRY", "EXCHANGE", "MKTCAP", "PSPS", "RRS_52W"), date)
    s <- screen.condition(s, "COUNTRY", "=", "United States", as.numeric=FALSE)
    s <- screen.condition(s, "EXCHANGE", "<>", "O", as.numeric=FALSE)
    s <- screen.condition(s, "MKTCAP", ">=", 25)
    s <- screen.condition(s, "MKTCAP", "<=", 250)
    s <- screen.condition(s, "PSPS", "<", 1.0)
    s <- screen.condition(s, "RRS_52W", ">=", 85)
    return (s)
}
```

Conn	(	Field	Oper
		Country	Equa
And		Exchange	Not I
And		Market Cap Q1	>=
And		Market Cap Q1	<=
And		Price/Sales	<
And		% Rank-Rel Strength 52 week	>=

```
> s <- TinyTitans(BT.ASOF)
> summary(s)
  COMPANY_ID          COUNTRY             EXCHANGE
 Length:48          Length:48          Length:48
 Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character



     MKTCAP            PSPS              RRS_52W
 Min.   : 25.1    Min.   :0.1400    Min.   :85.00
 1st Qu.: 40.3    1st Qu.:0.4450    1st Qu.:87.00
 Median : 60.0    Median :0.6450    Median :90.50
 Mean   : 86.1    Mean   :0.6260    Mean   :90.94
 3rd Qu.:128.0    3rd Qu.:0.8025    3rd Qu.:95.00
 Max.   :225.2    Max.   :0.9900    Max.   :98.00
>
```

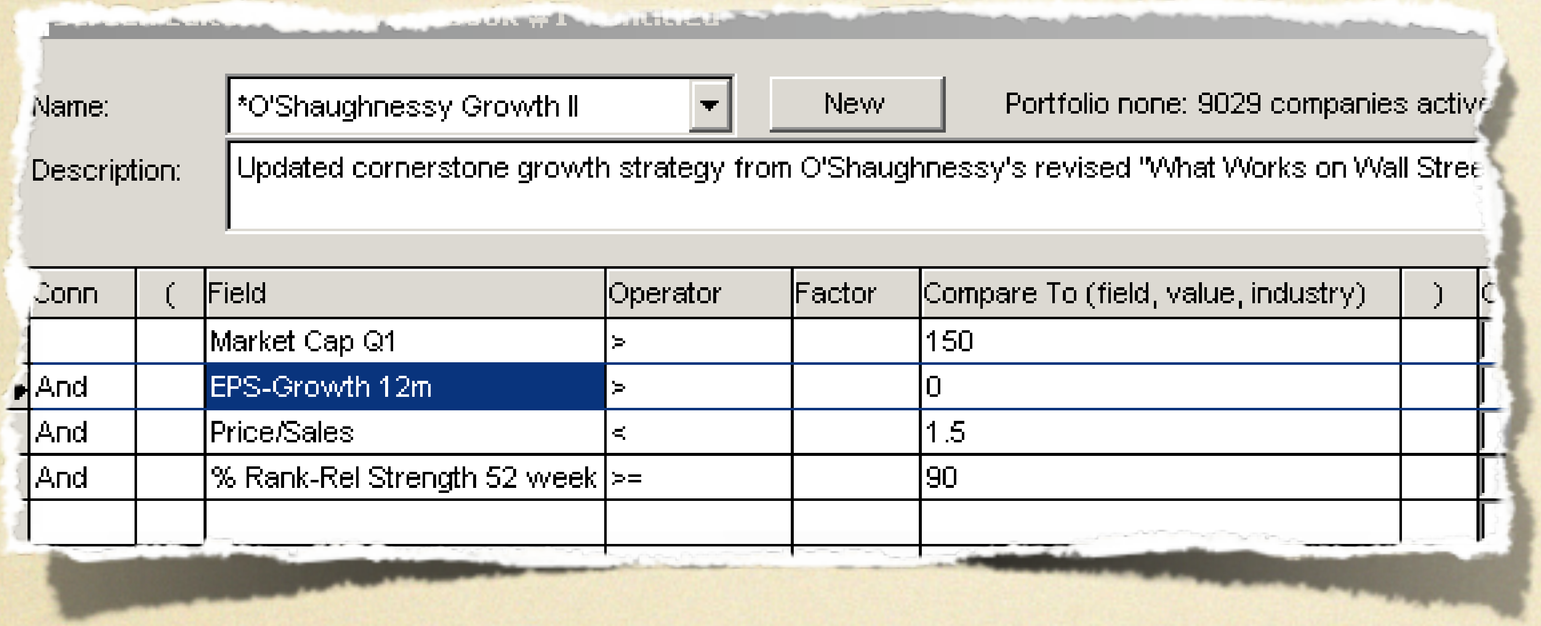

Stock Investor Professional
File Edit Tools Window Help
Portfolio: None
Screen: *O'Shaughness
View: Standard
Stock Notebook #1 - Untitled
View
Overview
Multiples
Growth

Company name	Company name	Ticker			
ADDvantage Technologies Group	ADDvantage Technologies Grou	AEY	Na		
Argan, Inc.	Argan, Inc.	AGX	Ar		
American Pacific Corporation	American Pacific Corporation	APFC	Na		
Appliance Recycling Centers of	Appliance Recycling Centers of	ARCI	Na		
Aristotle Corporation, The	Aristotle Corporation, The	ARTL	Na		
Bioanalytical Systems, Inc.	Bioanalytical Systems, Inc.	BASI	Na		
BSQUARE Corporation	BSQUARE Corporation	BSQR	Na		
Catalyst Semiconductor, Inc.	Catalyst Semiconductor, Inc.	CATS	Na		
China Direct Inc.	China Direct Inc.	CDS	Ar		
Carriage Services, Inc.	Carriage Services, Inc.	CSV	Ne		
DGSE Companies, Inc.	DGSE Companies, Inc.	DGC	Ar		
Datawatch Corporation	Datawatch Corporation	DWCH	Na		
EDAC Technologies Corporation	EDAC Technologies Corporation	EDAC	Na		
Energy West, Incorporated	Energy West, Incorporated	EWST	Na		
Fairchild Corporation	Fairchild Corporation	FA	Ne		
GP Strategies Corporation	GP Strategies Corporation	GPX	Ne		
Hastings Entertainment, Inc.	Hastings Entertainment, Inc.	HAST	Na		
Hickory Tech Corporation	Hickory Tech Corporation	HTCO	Na		
Hawk Corporation	Hawk Corporation	HWK	Ar		
Industrial Services of America	Industrial Services of America	IDSA	Na		
IntriCon Corporation	IntriCon Corporation	IIN	Na		
Innotrac Corporation	Innotrac Corporation	INOC	Na		
International Shipholding Corp	International Shipholding Corp	ISH	Ne		
James River Coal Company	James River Coal Company	JRCC	Na		
Kewaunee Scientific Corporatio	Kewaunee Scientific Corporatio	KEQU	Na		
Libbey Inc.	Libbey Inc.	LBY	Ne		
Mad Catz Interactive, Inc. (US	Mad Catz Interactive, Inc. (US	MCZ	American	Technology	Software & Programming
Command Security Corporation	Command Security Corporation	MOC	American	Services	Security Systems & Services
North American Galvanizing & C	North American Galvanizing & C	NGA	Nasdaq	Basic Materials	Misc. Fabricated Products

48 stocks selected - ranked by Ticker, ascending

```
> s <- screen.add(s, c("TICKER", "COMPANY"), BT.ASOF)
> s <- screen.order(s, "TICKER", numeric=FALSE, decreasing=FALSE)
> head(s[,c("COMPANY", "TICKER")], 20)
                           COMPANY TICKER
48 ADDvantage Technologies Group,    AEY
47                     Argan, Inc.    AGX
46    American Pacific Corporation   APFC
45  Appliance Recycling Centers of   ARCI
44      Aristotle Corporation, The   ARTL
43     Bioanalytical Systems, Inc.   BASI
42             BSQUARE Corporation   BSQR
41    Catalyst Semiconductor, Inc.   CATS
40               China Direct Inc.    CDS
39         Carriage Services, Inc.    CSV
38            DGSE Companies, Inc.    DGC
37           Datawatch Corporation   DWCH
36   EDAC Technologies Corporation   EDAC
35       Energy West, Incorporated   EWST
34           Fairchild Corporation     FA
33       GP Strategies Corporation    GPX
32    Hastings Entertainment, Inc.   HAST
31        Hickory Tech Corporation   HTCO
30                Hawk Corporation    HWK
29  Industrial Services of America   IDSA
>
```

O’Shaugnessy Growth II



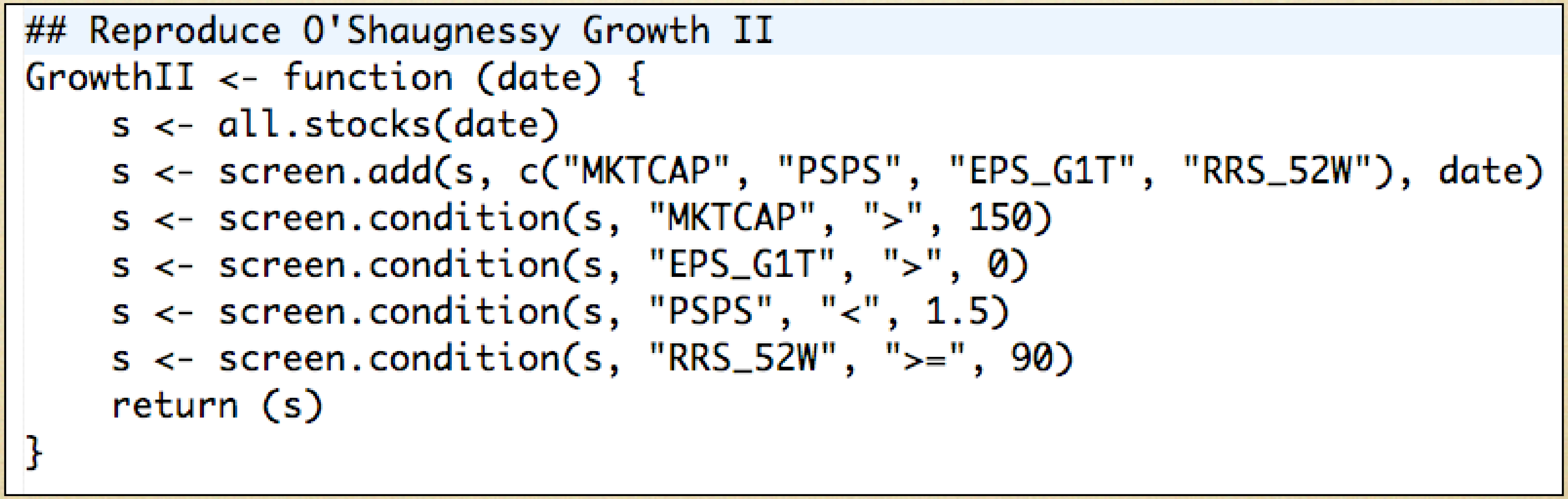
Name: *O'Shaughnessy Growth II

New

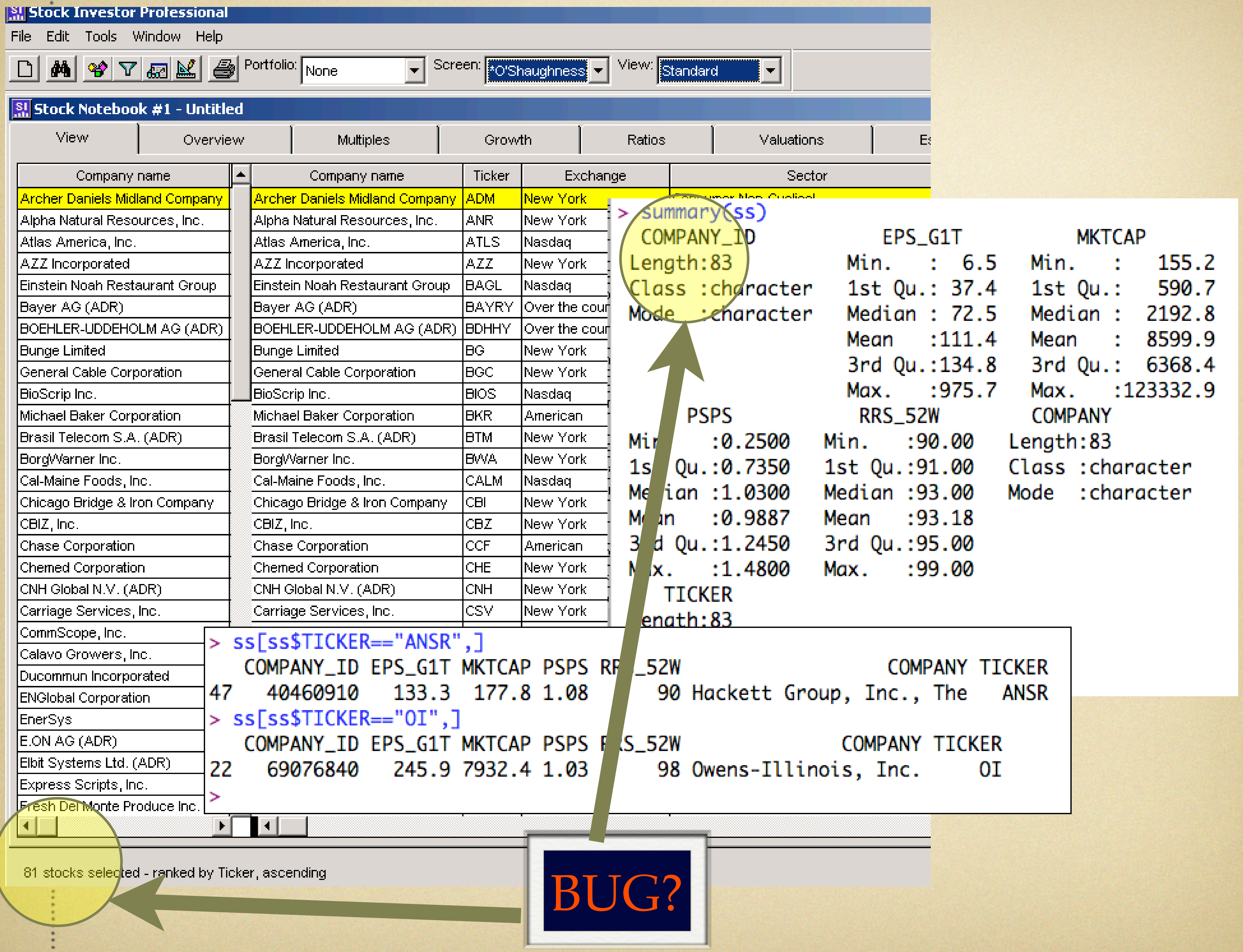
Portfolio none: 9029 companies activ

Description: Updated cornerstone growth strategy from O'Shaughnessy's revised "What Works on Wall Stree

Conn	(	Field	Operator	Factor	Compare To (field, value, industry)	)
		Market Cap Q1	>		150	
And		EPS-Growth 12m	>		0	
And		Price/Sales	<		1.5	
And		% Rank-Rel Strength 52 week	>=		90	



```
## Reproduce O'Shaugnessy Growth II
GrowthII <- function (date) {
    s <- all.stocks(date)
    s <- screen.add(s, c("MKTCAP", "PSPS", "EPS_G1T", "RRS_52W"), date)
    s <- screen.condition(s, "MKTCAP", ">", 150)
    s <- screen.condition(s, "EPS_G1T", ">", 0)
    s <- screen.condition(s, "PSPS", "<", 1.5)
    s <- screen.condition(s, "RRS_52W", ">=", 90)
    return (s)
}
```



Company name	Company name	Ticker	Exchange	Sector
Archer Daniels Midland Company	Archer Daniels Midland Company	ADM	New York	Consumer Non-Cyclical
Alpha Natural Resources, Inc.	Alpha Natural Resources, Inc.	ANR	New York	
Atlas America, Inc.	Atlas America, Inc.	ATLS	Nasdaq	
AZZ Incorporated	AZZ Incorporated	AZZ	New York	
Einstein Noah Restaurant Group	Einstein Noah Restaurant Group	BAGL	Nasdaq	
Bayer AG (ADR)	Bayer AG (ADR)	BAYRY	Over the cour	
BOEHLER-UDDEHOLM AG (ADR)	BOEHLER-UDDEHOLM AG (ADR)	BDHHY	Over the cour	
Bunge Limited	Bunge Limited	BG	New York	
General Cable Corporation	General Cable Corporation	BGC	New York	
BioScrip Inc.	BioScrip Inc.	BIOS	Nasdaq	
Michael Baker Corporation	Michael Baker Corporation	BKR	American	
Brasil Telecom S.A. (ADR)	Brasil Telecom S.A. (ADR)	BTM	New York	
BorgWarner Inc.	BorgWarner Inc.	BWA	New York	
Cal-Maine Foods, Inc.	Cal-Maine Foods, Inc.	CALM	Nasdaq	
Chicago Bridge & Iron Company	Chicago Bridge & Iron Company	CBI	New York	
CBIZ, Inc.	CBIZ, Inc.	CBZ	New York	
Chase Corporation	Chase Corporation	CCF	American	
Chemed Corporation	Chemed Corporation	CHE	New York	
CNH Global N.V. (ADR)	CNH Global N.V. (ADR)	CNH	New York	
Carriage Services, Inc.	Carriage Services, Inc.	CSV	New York	
CommScope, Inc.				
Calavo Growers, Inc.				
Ducommun Incorporated				
ENGlobal Corporation				
EnerSys				
E.ON AG (ADR)				
Elbit Systems Ltd. (ADR)				
Express Scripts, Inc.				
Fresh Del Monte Produce Inc.				

81 stocks selected - ranked by Ticker, ascending

```
> summary(ss)
  COMPANY_ID           EPS_G1T          MKTCAP
 Length:83          Min.   :  6.5   Min.   :   155.2
 Class :character   1st Qu.: 37.4   1st Qu.:   590.7
 Mode  :character   Median : 72.5   Median :  2192.8
                    Mean   :111.4   Mean   :  8599.9
                    3rd Qu.:134.8   3rd Qu.:  6368.4
                    Max.   :975.7   Max.   :123332.9
      PSPS            RRS_52W          COMPANY
 Min.   :0.2500   Min.   :90.00   Length:83
 1st Qu.:0.7350   1st Qu.:91.00   Class :character
 Median :1.0300   Median :93.00   Mode  :character
 Mean   :0.9887   Mean   :93.18
 3rd Qu.:1.2450   3rd Qu.:95.00
 Max.   :1.4800   Max.   :99.00
    TICKER
 Length:83
```

```
> ss[ss$TICKER=="ANSR",]
   COMPANY_ID EPS_G1T MKTCAP PSPS RRS_52W                  COMPANY TICKER
47   40460910   133.3  177.8 1.08      90 Hackett Group, Inc., The   ANSR
> ss[ss$TICKER=="OI",]
   COMPANY_ID EPS_G1T MKTCAP PSPS RRS_52W              COMPANY TICKER
22   69076840   245.9 7932.4 1.03      98 Owens-Illinois, Inc.     OI
>
```

BUG?

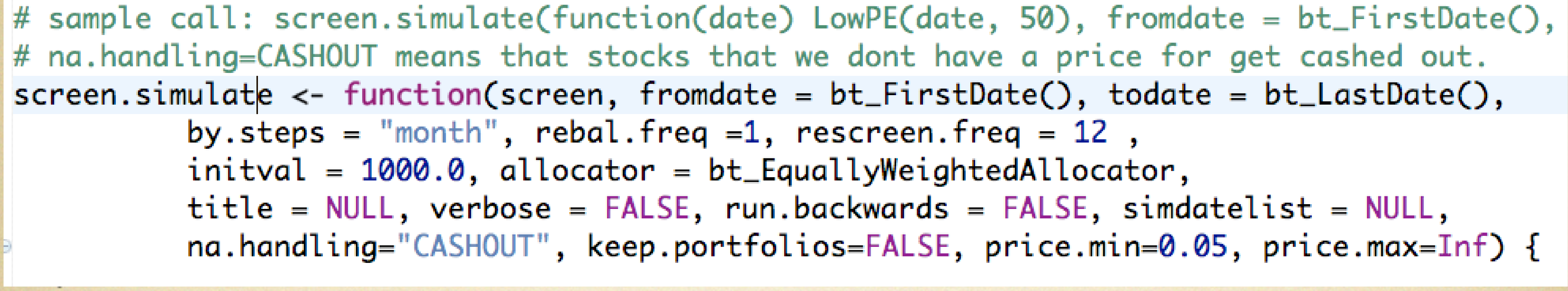
Agenda

- Screening vs. Backtesting
- Overview / PAST-SIPro Demo
- Setup PAST-SIPro
- The "R" programming language
- Sample backtests (Simple to Sophisticated)
- From Screens to Backtests
- ToDo List
- Questions?

Lets do something New!

- Until now, we haven't really done anything you couldn't have done with SI Pro
- e.g. TinyTitans
- So, let's look at how TinyTitans has done historically
- Decisions to be made!
- Allocation of funds, rescreen/rebal freq etc.

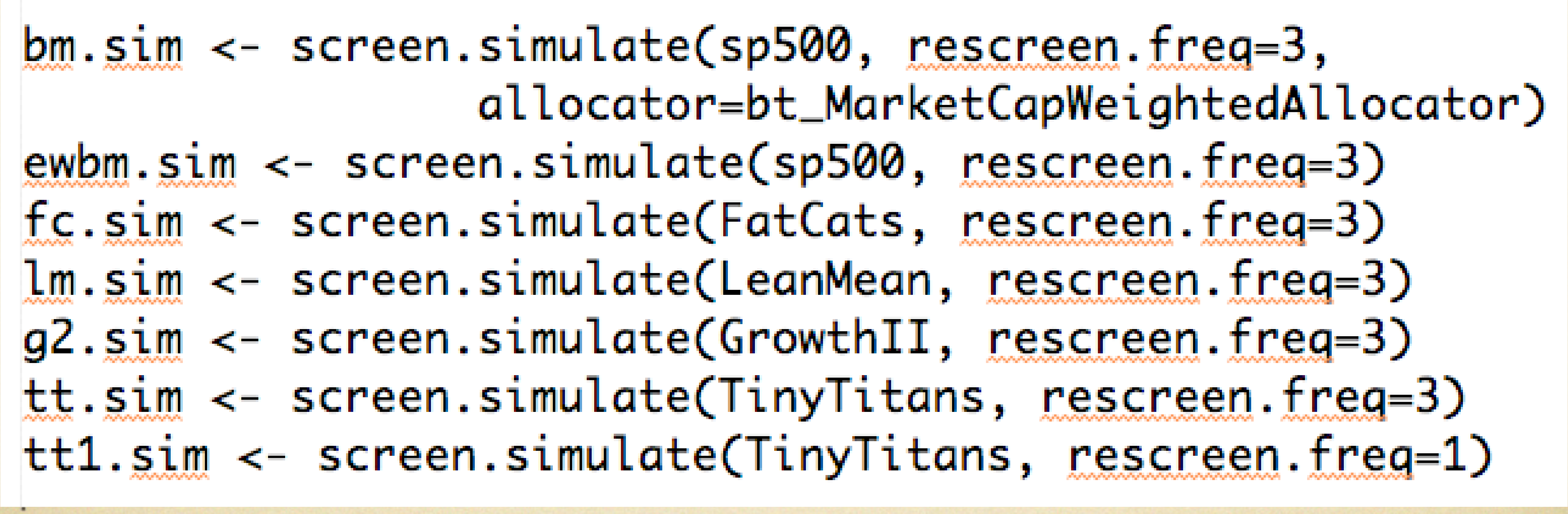
screen.simulate()



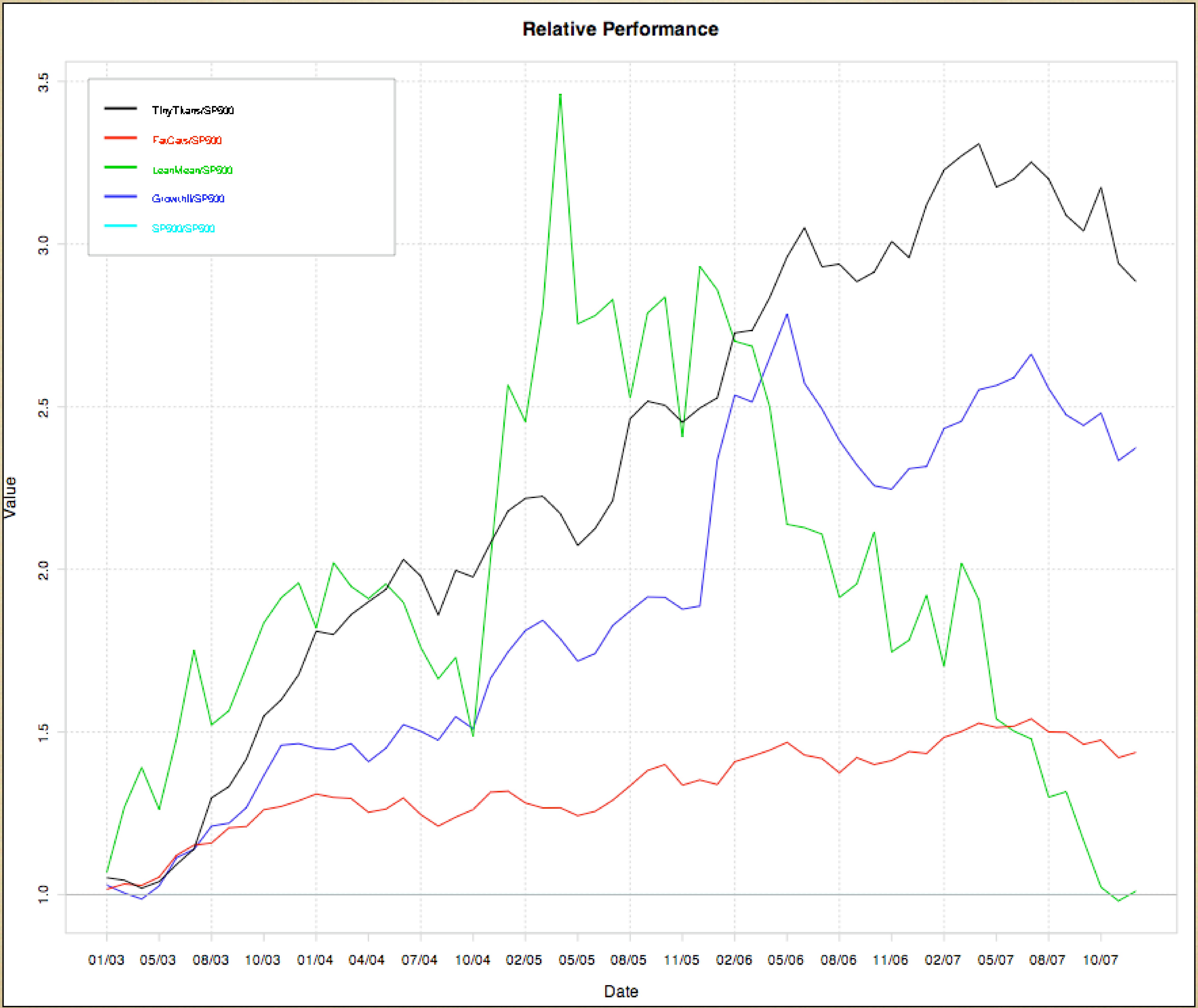
```
# sample call: screen.simulate(function(date) LowPE(date, 50), fromdate = bt_FirstDate(),
# na.handling=CASHOUT means that stocks that we dont have a price for get cashed out.
screen.simulate <- function(screen, fromdate = bt_FirstDate(), todate = bt_LastDate(),
          by.steps = "month", rebal.freq =1, rescreen.freq = 12 ,
          initval = 1000.0, allocator = bt_EquallyWeightedAllocator,
          title = NULL, verbose = FALSE, run.backwards = FALSE, simdatelist = NULL,
          na.handling="CASHOUT", keep.portfolios=FALSE, price.min=0.05, price.max=Inf) {
```

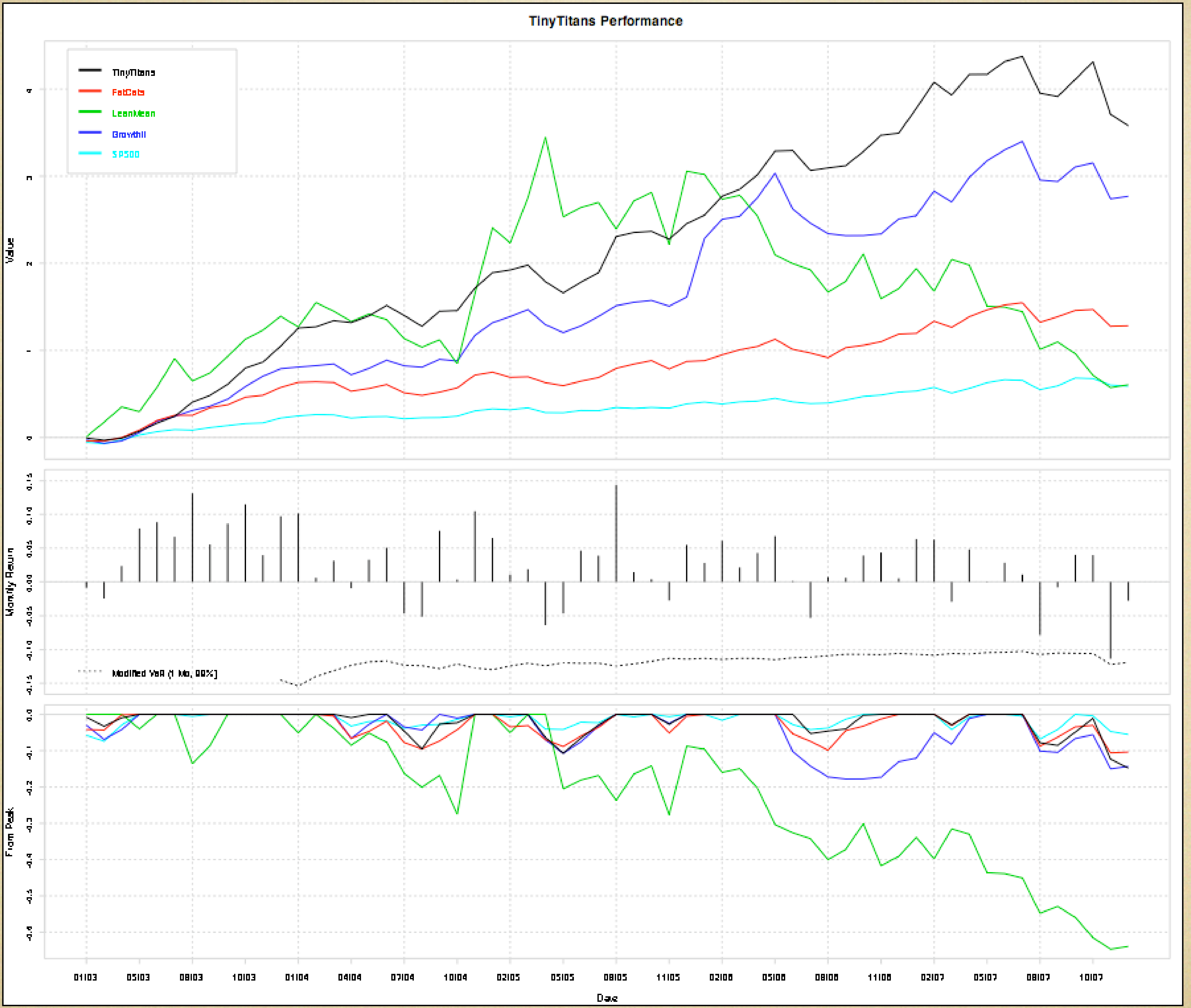
- Database Reading is slow the first time
- Defaults:
    - Annual Rescreening
    - Monthly Rebalancing
    - Equally Weighted

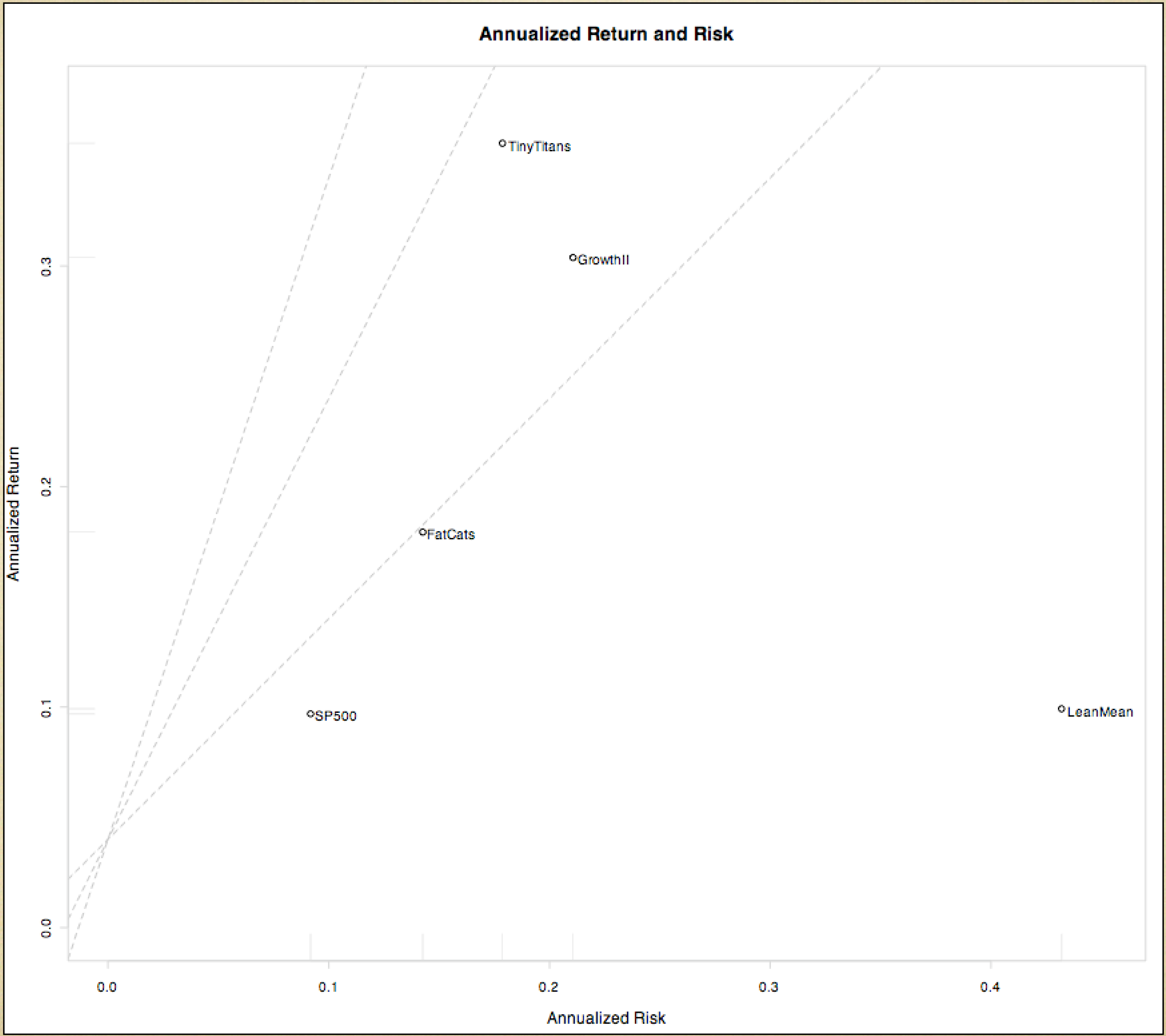
Let's Simulate!

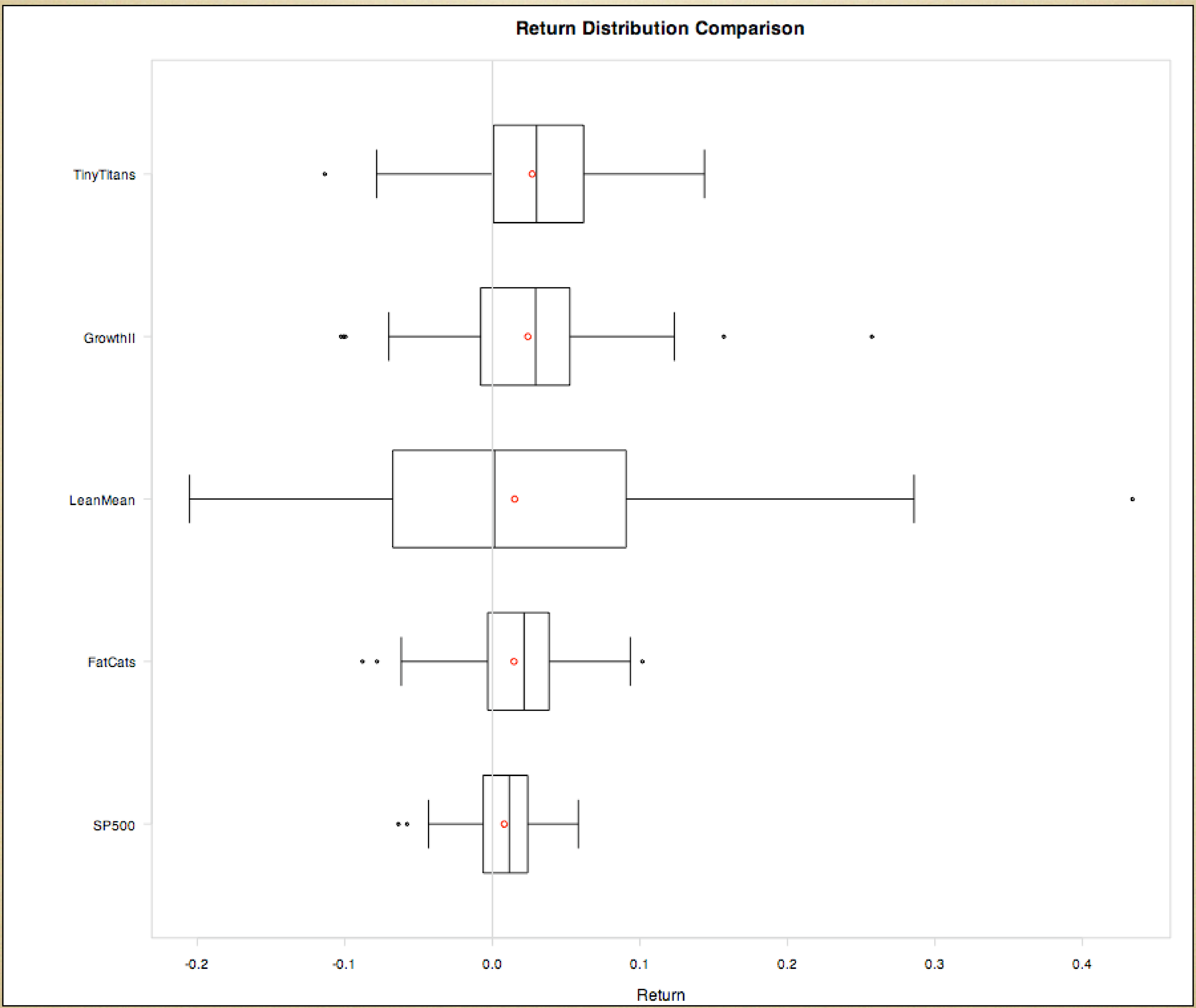


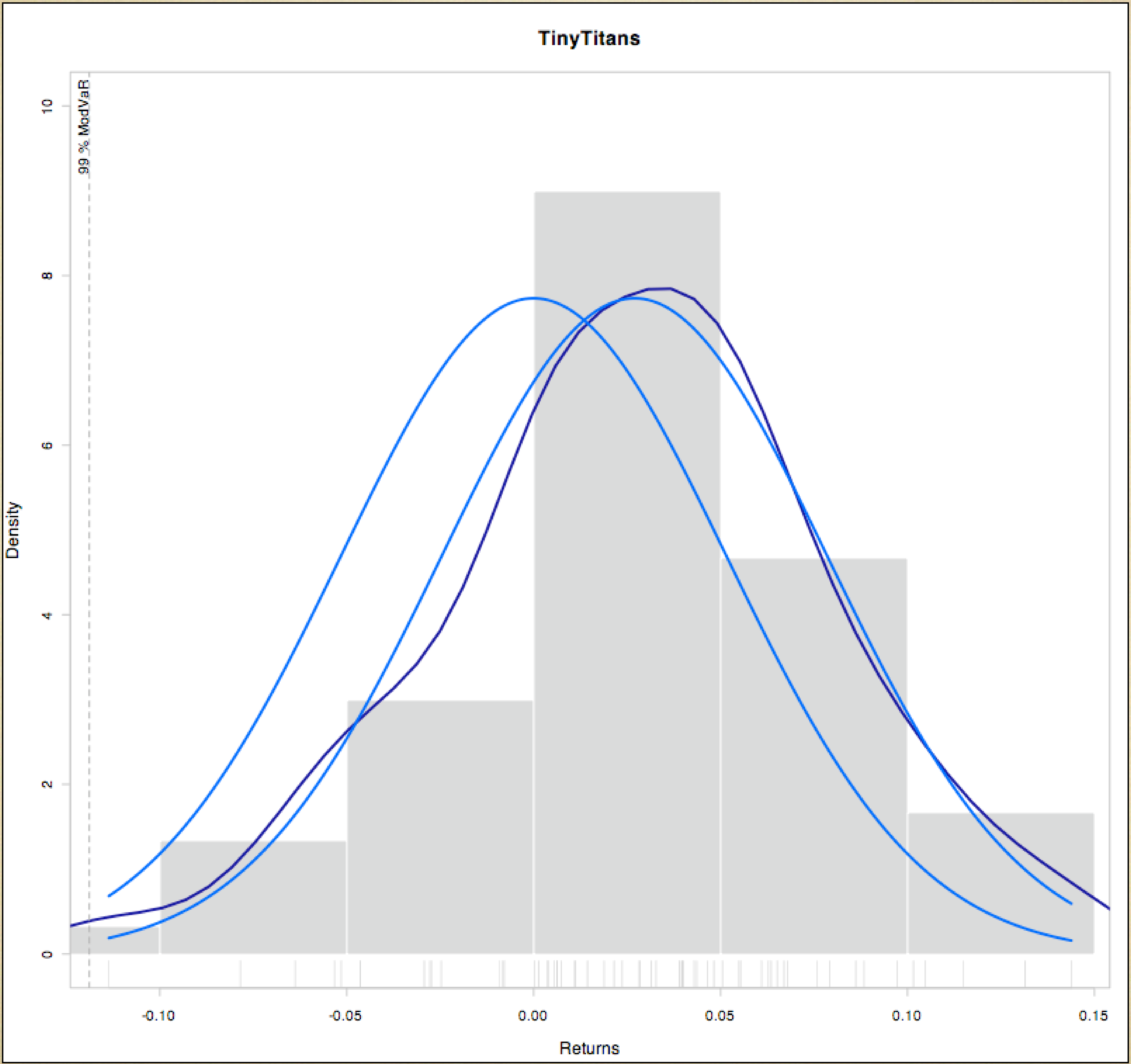
```
bm.sim <- screen.simulate(sp500, rescreen.freq=3,
                 allocator=bt_MarketCapWeightedAllocator)
ewbm.sim <- screen.simulate(sp500, rescreen.freq=3)
fc.sim <- screen.simulate(FatCats, rescreen.freq=3)
lm.sim <- screen.simulate(LeanMean, rescreen.freq=3)
g2.sim <- screen.simulate(GrowthII, rescreen.freq=3)
tt.sim <- screen.simulate(TinyTitans, rescreen.freq=3)
tt1.sim <- screen.simulate(TinyTitans, rescreen.freq=1)
```


Relative Performance
TinyTitans/SP900
FatCats/SP900
LeanMean/SP900
Growth/SP900
SP900/SP900
Value
3.5
3.0
2.5
2.0
1.5
1.0
01/03
05/03
08/03
10/03
01/04
04/04
07/04
10/04
02/05
05/05
08/05
11/05
02/06
05/06
08/06
11/06
02/07
05/07
08/07
10/07
Date

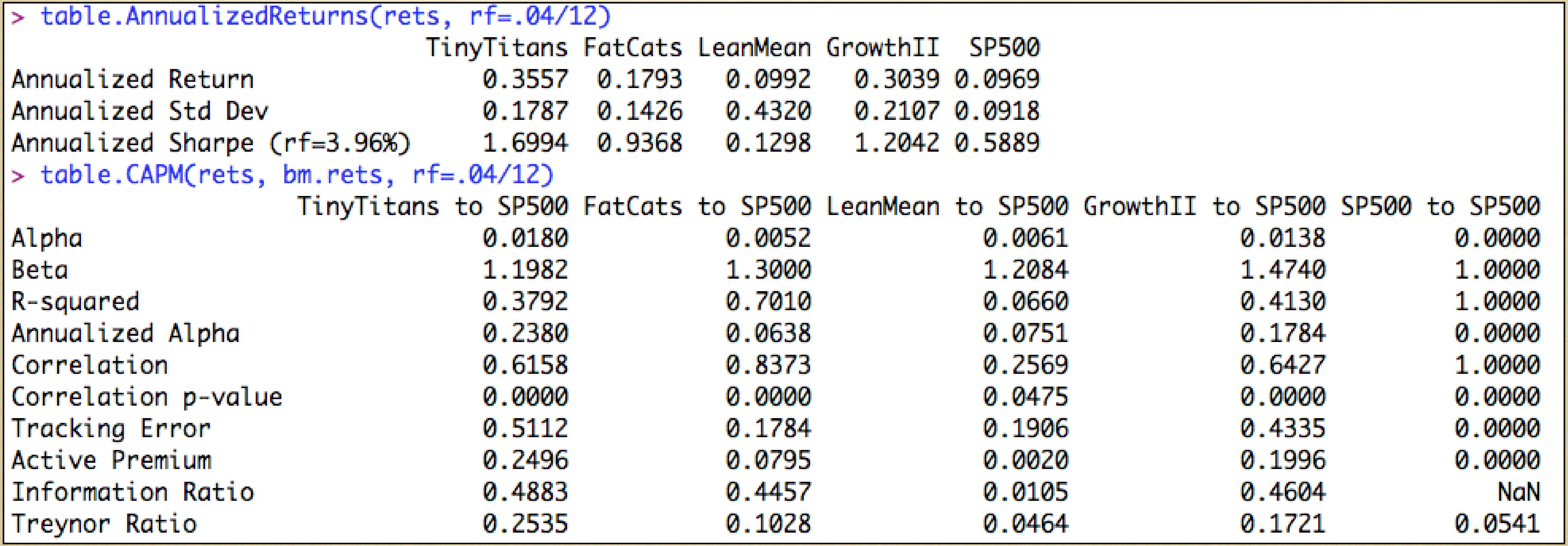

TinyTitans Performance
TinyTitans
FatCats
LeanMean
GrowthII
SP500
Value
0
1
2
3
4
Monthly Return
0.15
0.10
0.05
0.00
-0.05
-0.10
-0.15
Modified VaR (1 Mo, 99%)
From Peak
0.0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
01/03
05/03
08/03
10/03
01/04
04/04
07/04
10/04
02/05
05/05
08/05
11/05
02/06
05/06
08/06
11/06
02/07
05/07
08/07
10/07
Date


Annualized Return and Risk
Annualized Return
Annualized Risk
0.0
0.1
0.2
0.3
0.0
0.1
0.2
0.3
0.4
TinyTitans
GrowthII
FatCats
SP500
LeanMean


Return Distribution Comparison
TinyTitans
GrowthII
LeanMean
FatCats
SP500
-0.2
-0.1
0.0
0.1
0.2
0.3
0.4
Return

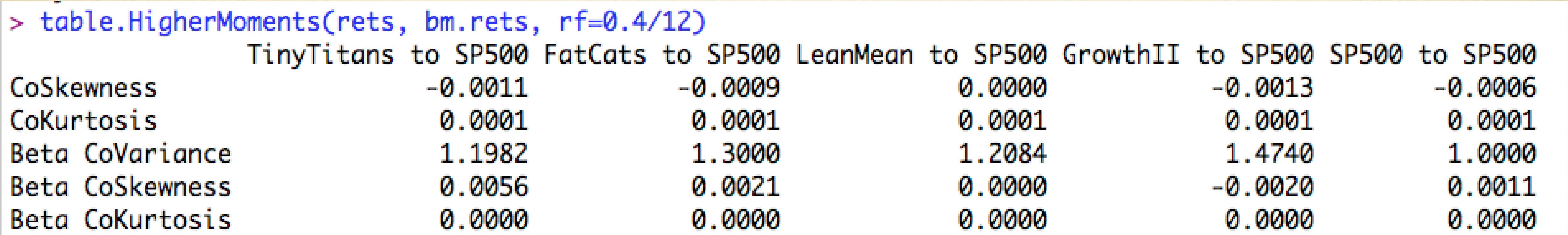
TinyTitans
99 % ModVaR
Density
0
2
4
6
8
10
-0.10
-0.05
0.00
0.05
0.10
0.15
Returns

Don’t underestimate Tables



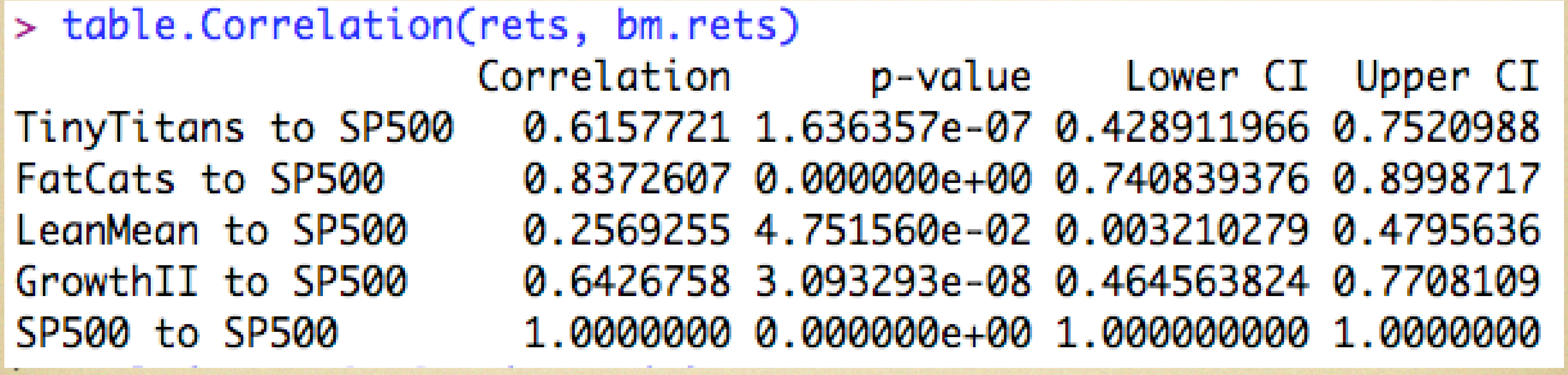
```
> table.AnnualizedReturns(rets, rf=.04/12)
                              TinyTitans FatCats LeanMean GrowthII  SP500
Annualized Return                 0.3557  0.1793   0.0992   0.3039 0.0969
Annualized Std Dev                0.1787  0.1426   0.4320   0.2107 0.0918
Annualized Sharpe (rf=3.96%)      1.6994  0.9368   0.1298   1.2042 0.5889
> table.CAPM(rets, bm.rets, rf=.04/12)
                     TinyTitans to SP500 FatCats to SP500 LeanMean to SP500 GrowthII to SP500 SP500 to SP500
Alpha                             0.0180           0.0052            0.0061            0.0138         0.0000
Beta                              1.1982           1.3000            1.2084            1.4740         1.0000
R-squared                         0.3792           0.7010            0.0660            0.4130         1.0000
Annualized Alpha                  0.2380           0.0638            0.0751            0.1784         0.0000
Correlation                       0.6158           0.8373            0.2569            0.6427         1.0000
Correlation p-value               0.0000           0.0000            0.0475            0.0000         0.0000
Tracking Error                    0.5112           0.1784            0.1906            0.4335         0.0000
Active Premium                    0.2496           0.0795            0.0020            0.1996         0.0000
Information Ratio                 0.4883           0.4457            0.0105            0.4604            NaN
Treynor Ratio                     0.2535           0.1028            0.0464            0.1721         0.0541
```

Higher Moments



```
> table.HigherMoments(rets, bm.rets, rf=0.4/12)
                 TinyTitans to SP500 FatCats to SP500 LeanMean to SP500 GrowthII to SP500 SP500 to SP500
CoSkewness                   -0.0011          -0.0009            0.0000           -0.0013        -0.0006
CoKurtosis                    0.0001           0.0001            0.0001            0.0001         0.0001
Beta CoVariance               1.1982           1.3000            1.2084            1.4740         1.0000
Beta CoSkewness               0.0056           0.0021            0.0000           -0.0020         0.0011
Beta CoKurtosis               0.0000           0.0000            0.0000            0.0000         0.0000
```

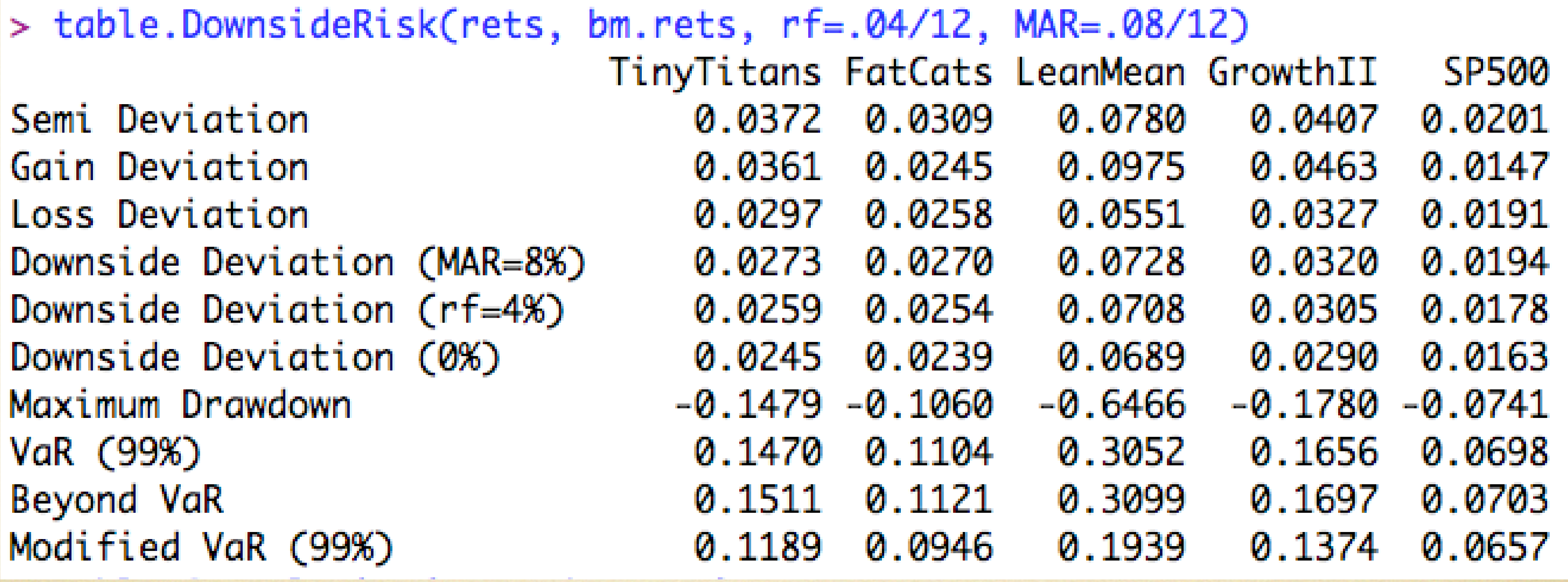
Correlations



```
> table.Correlation(rets, bm.rets)
                    Correlation      p-value    Lower CI  Upper CI
TinyTitans to SP500   0.6157721 1.636357e-07 0.428911966 0.7520988
FatCats to SP500      0.8372607 0.000000e+00 0.740839376 0.8998717
LeanMean to SP500     0.2569255 4.751560e-02 0.003210279 0.4795636
GrowthII to SP500     0.6426758 3.093293e-08 0.464563824 0.7708109
SP500 to SP500        1.0000000 0.000000e+00 1.000000000 1.0000000
```

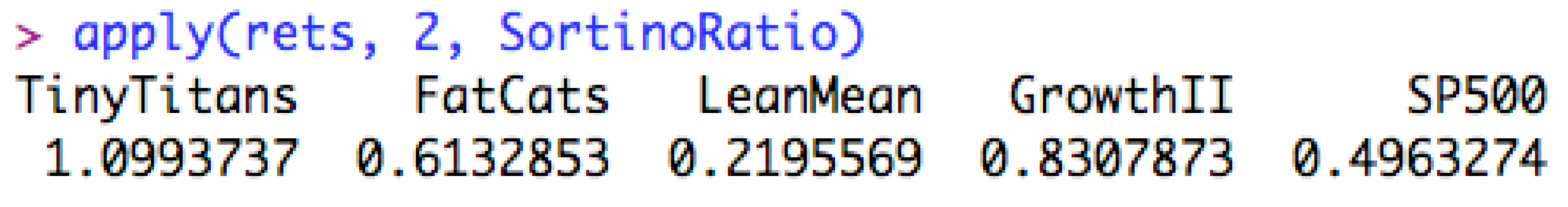
ci = 0.95

Downside Risk



```
> table.DownsideRisk(rets, bm.rets, rf=.04/12, MAR=.08/12)
                              TinyTitans FatCats LeanMean GrowthII   SP500
Semi Deviation                    0.0372  0.0309   0.0780   0.0407  0.0201
Gain Deviation                    0.0361  0.0245   0.0975   0.0463  0.0147
Loss Deviation                    0.0297  0.0258   0.0551   0.0327  0.0191
Downside Deviation (MAR=8%)       0.0273  0.0270   0.0728   0.0320  0.0194
Downside Deviation (rf=4%)        0.0259  0.0254   0.0708   0.0305  0.0178
Downside Deviation (0%)           0.0245  0.0239   0.0689   0.0290  0.0163
Maximum Drawdown                 -0.1479 -0.1060  -0.6466  -0.1780 -0.0741
VaR (99%)                         0.1470  0.1104   0.3052   0.1656  0.0698
Beyond VaR                        0.1511  0.1121   0.3099   0.1697  0.0703
Modified VaR (99%)                0.1189  0.0946   0.1939   0.1374  0.0657
```

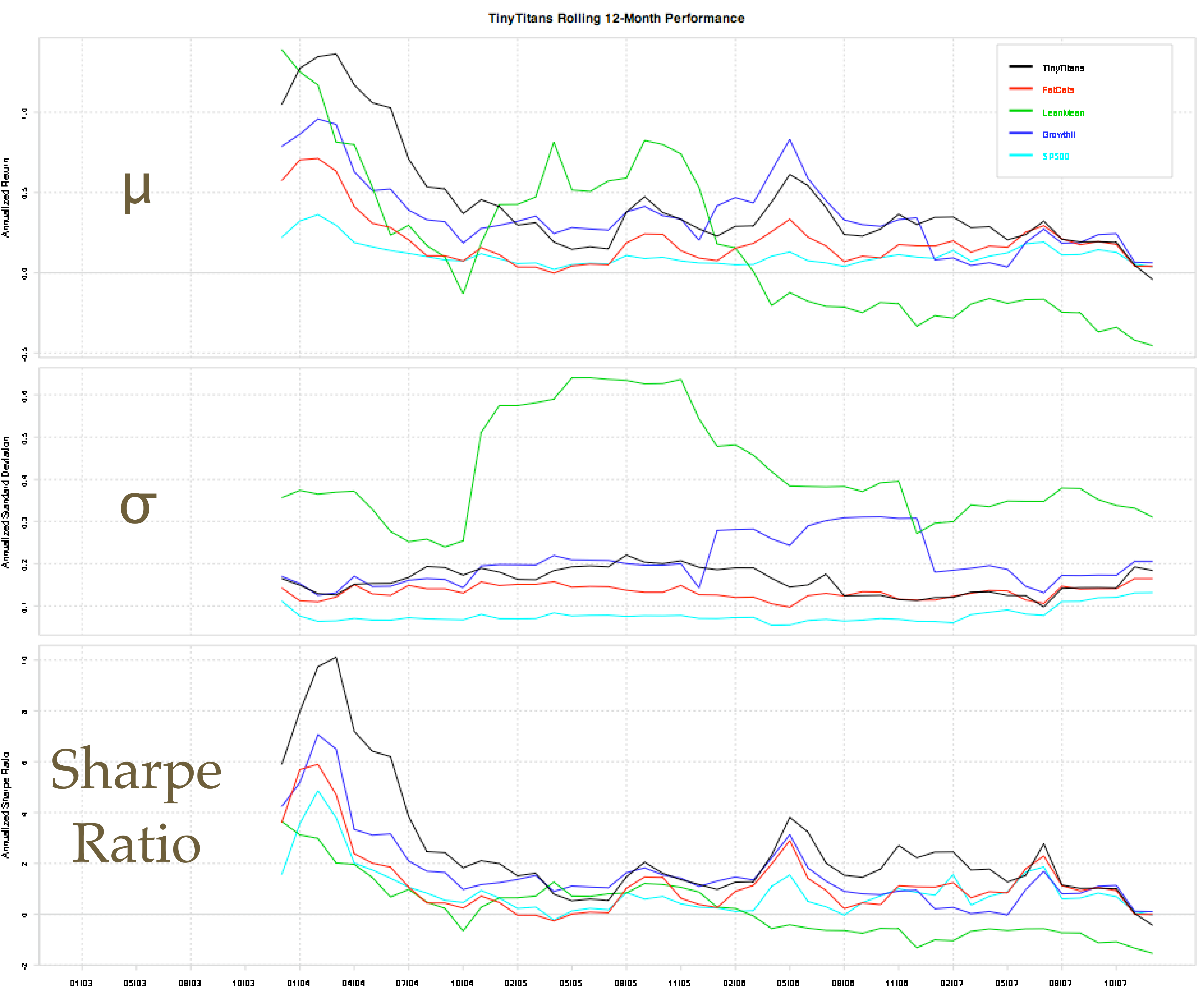
Sortino Ratio



```
> apply(rets, 2, SortinoRatio)
TinyTitans    FatCats   LeanMean   GrowthII      SP500
 1.0993737  0.6132853  0.2195569  0.8307873  0.4963274
```

Rolling Windows

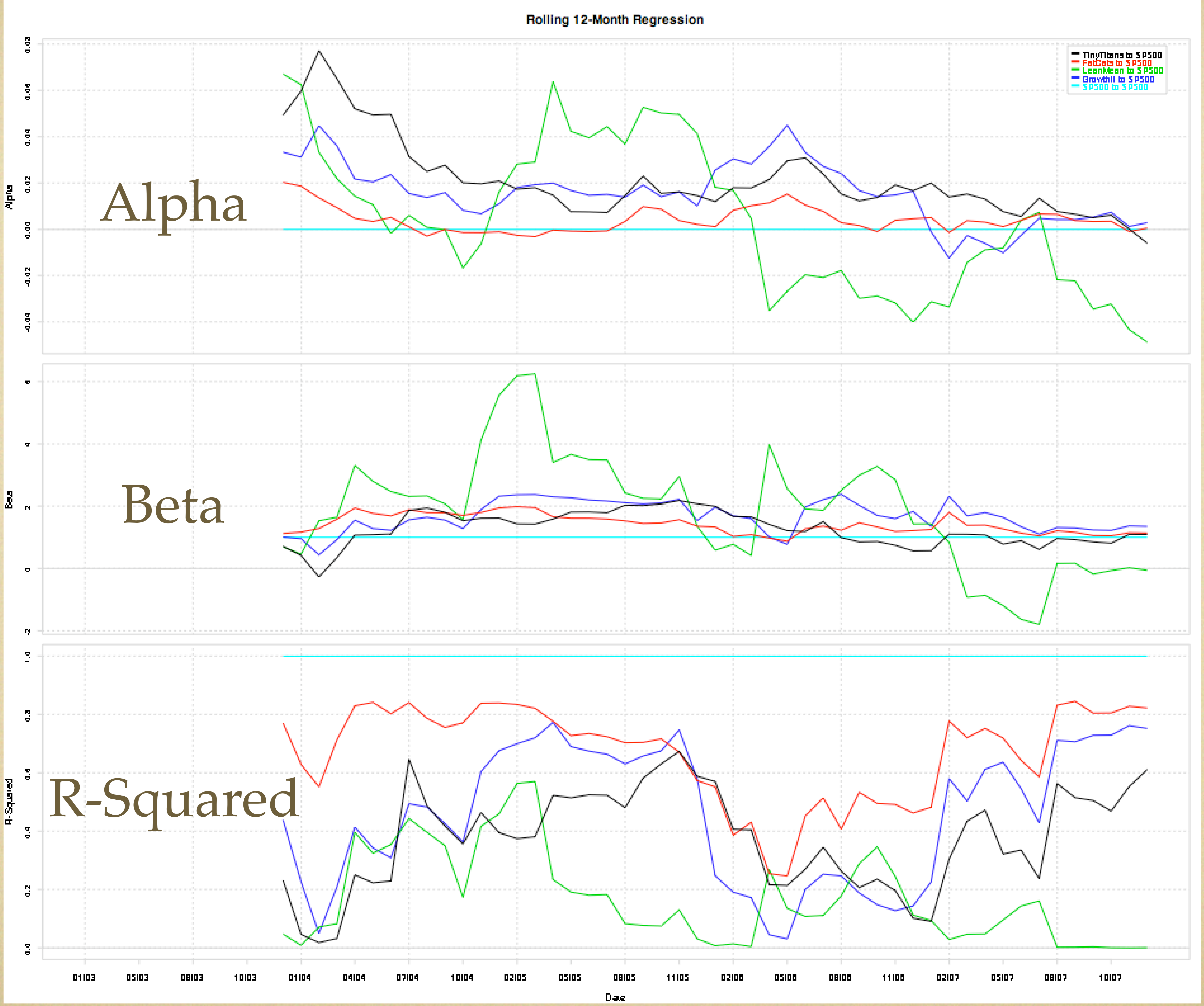
- Rolling Windows give a better sense of the persistence of results
- Window width is typically 12 months



μ

σ

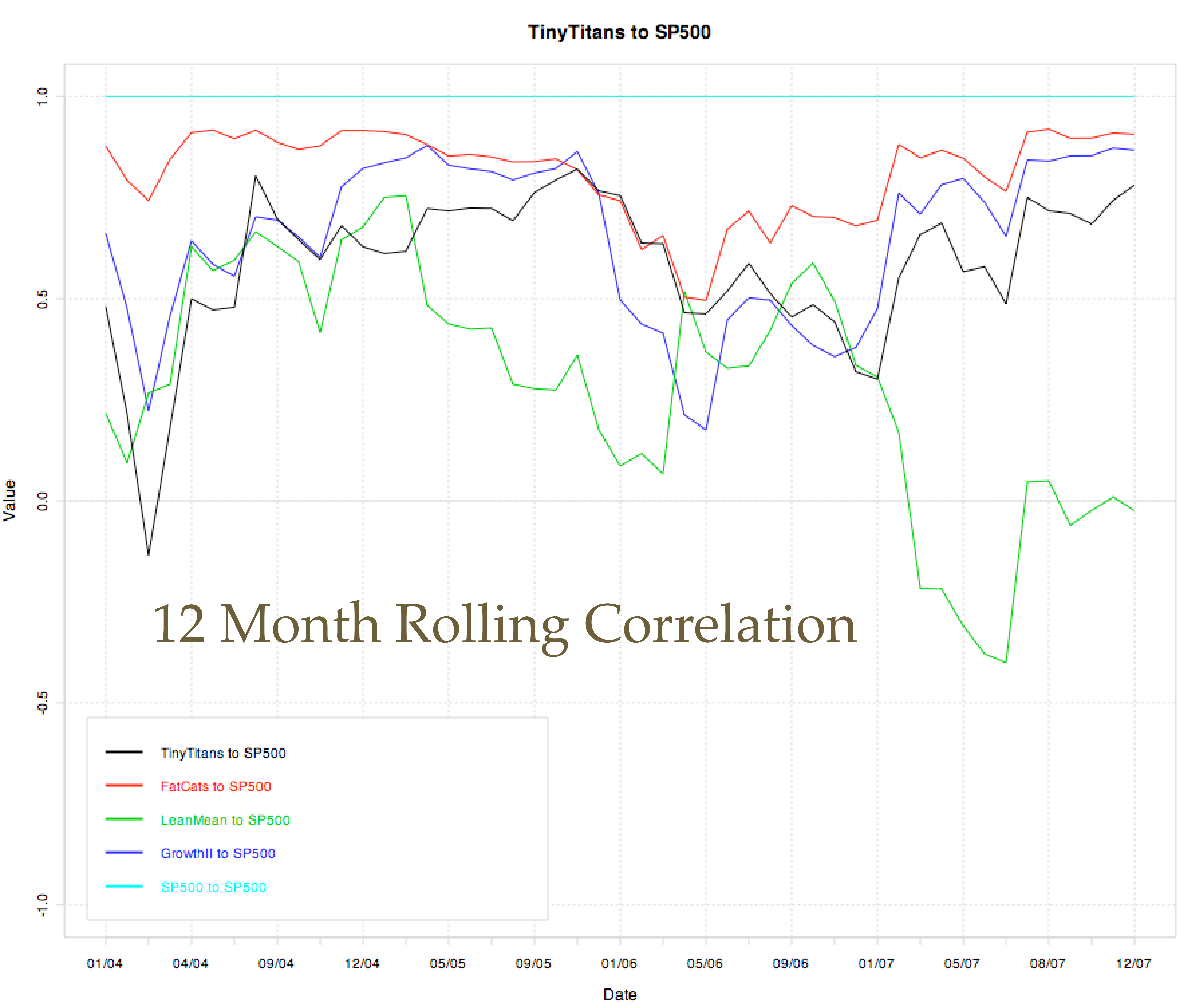
Sharpe Ratio



Alpha

Beta

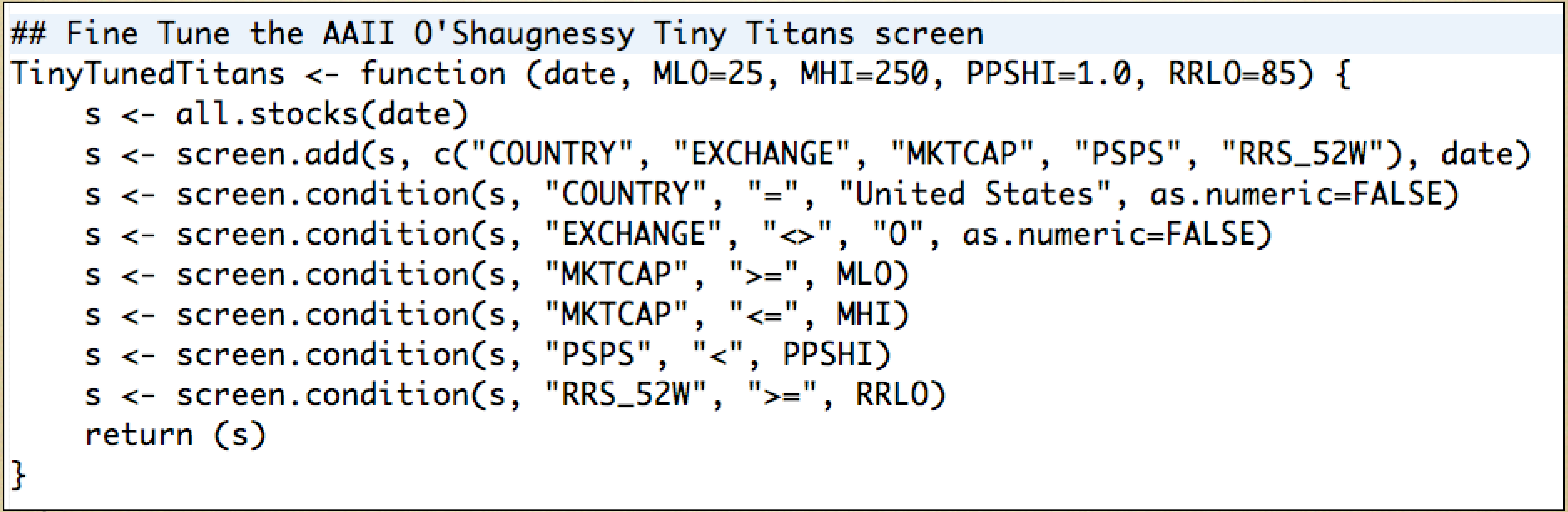
R-Squared



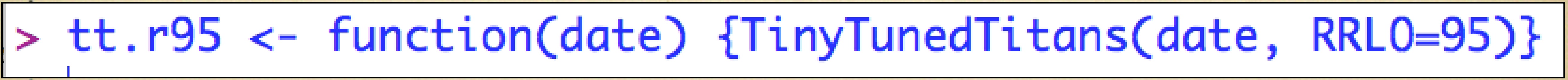
12 Month Rolling Correlation

Beyond TinyTitans

- Programmatic Screens allow more sophisticated screening



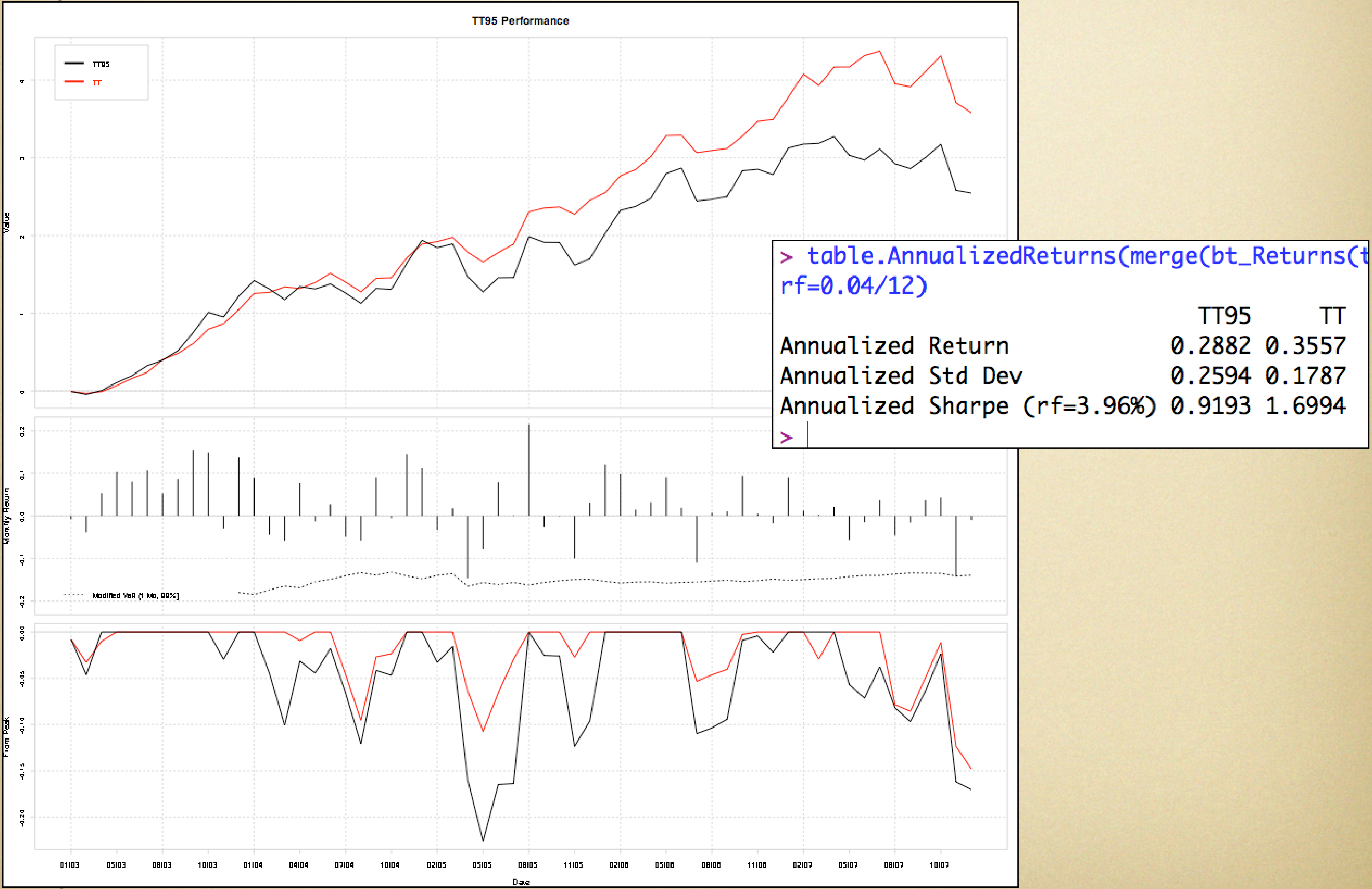
```
## Fine Tune the AAII O'Shaugnessy Tiny Titans screen
TinyTunedTitans <- function (date, MLO=25, MHI=250, PPSHI=1.0, RRLO=85) {
    s <- all.stocks(date)
    s <- screen.add(s, c("COUNTRY", "EXCHANGE", "MKTCAP", "PSPS", "RRS_52W"), date)
    s <- screen.condition(s, "COUNTRY", "=", "United States", as.numeric=FALSE)
    s <- screen.condition(s, "EXCHANGE", "<>", "O", as.numeric=FALSE)
    s <- screen.condition(s, "MKTCAP", ">=", MLO)
    s <- screen.condition(s, "MKTCAP", "<=", MHI)
    s <- screen.condition(s, "PSPS", "<", PPSHI)
    s <- screen.condition(s, "RRS_52W", ">=", RRLO)
    return (s)
}
```



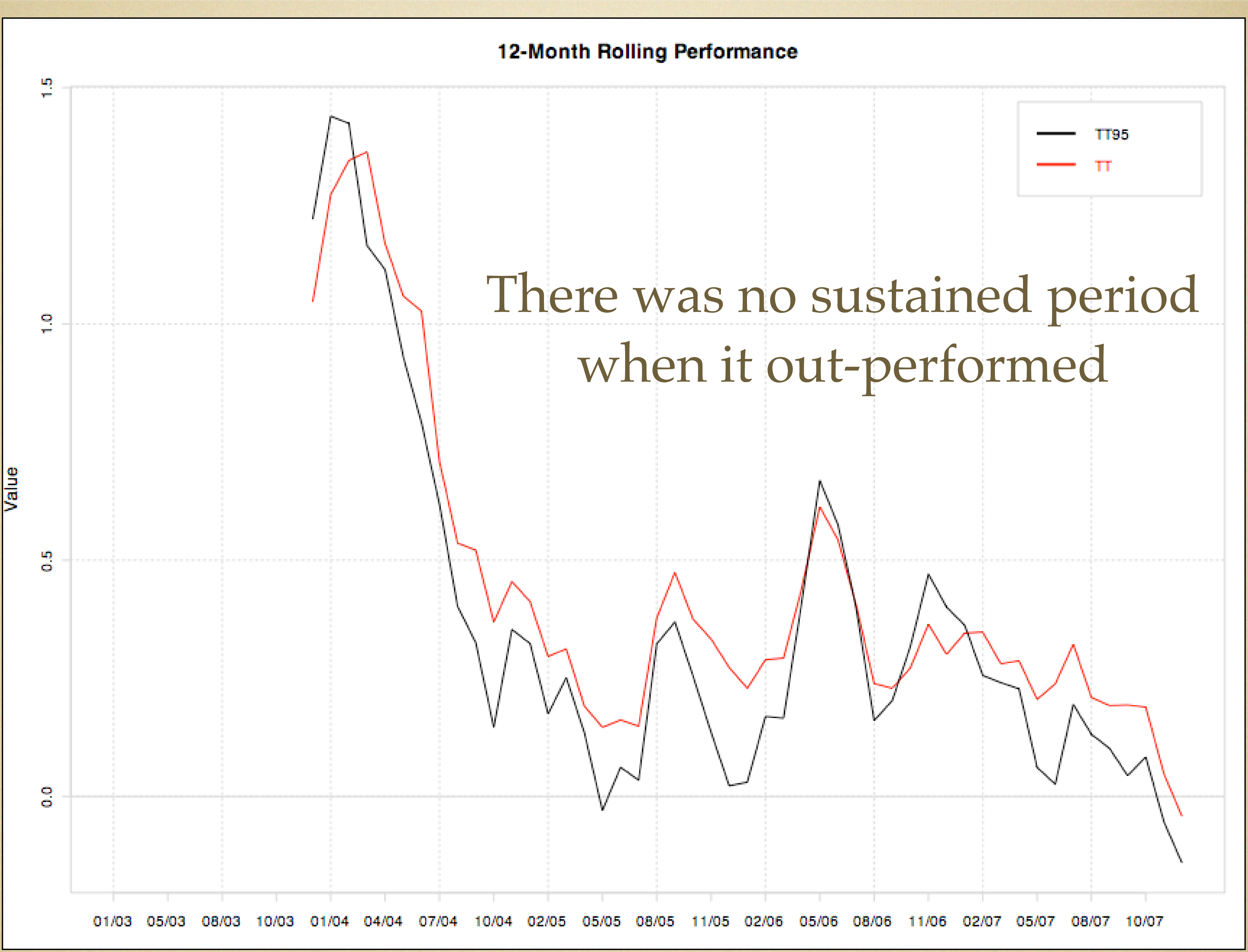
```
> tt.r95 <- function(date) {TinyTunedTitans(date, RRLO=95)}
```

Might this help?

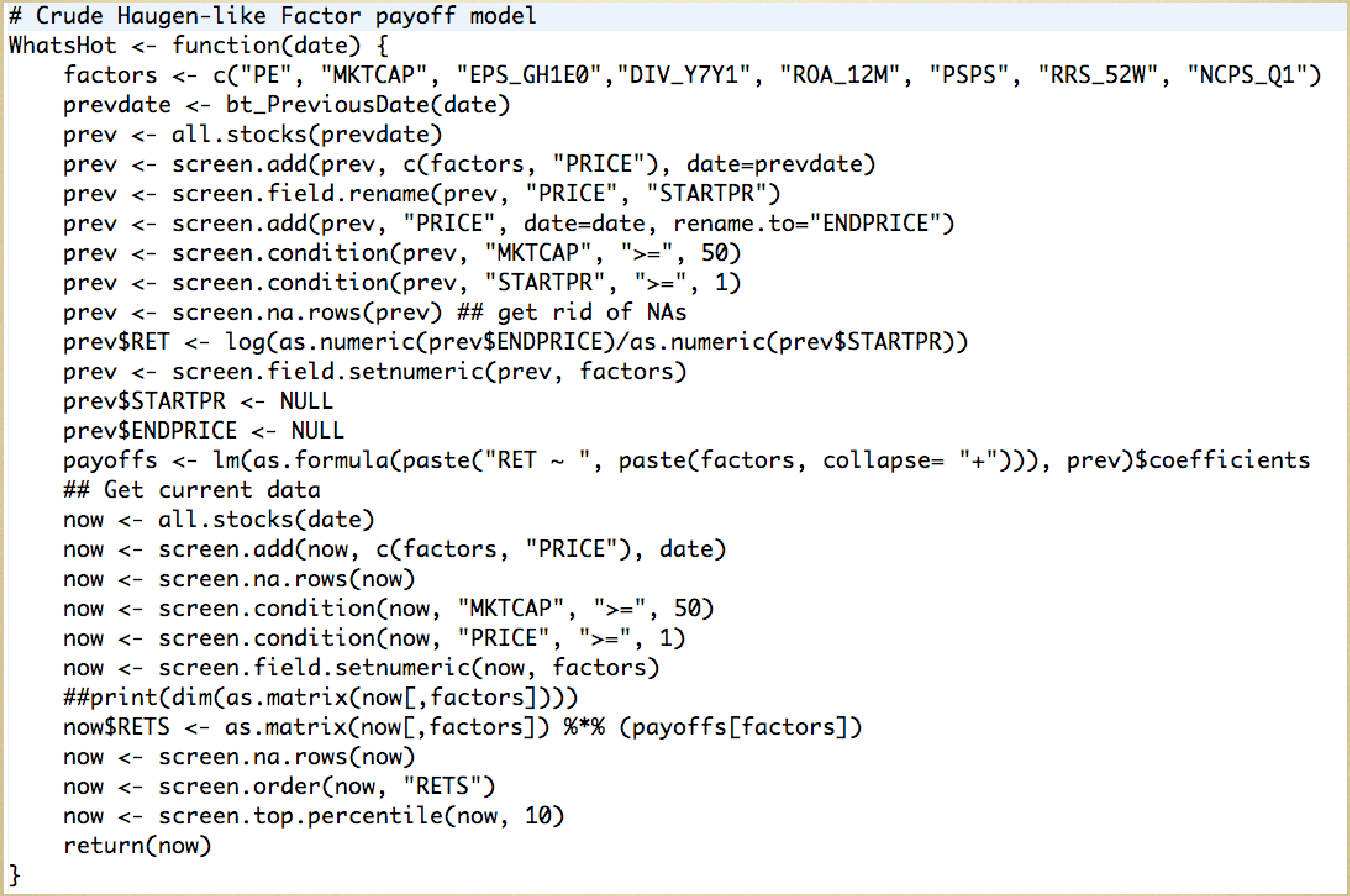
Nopes! It hurts!



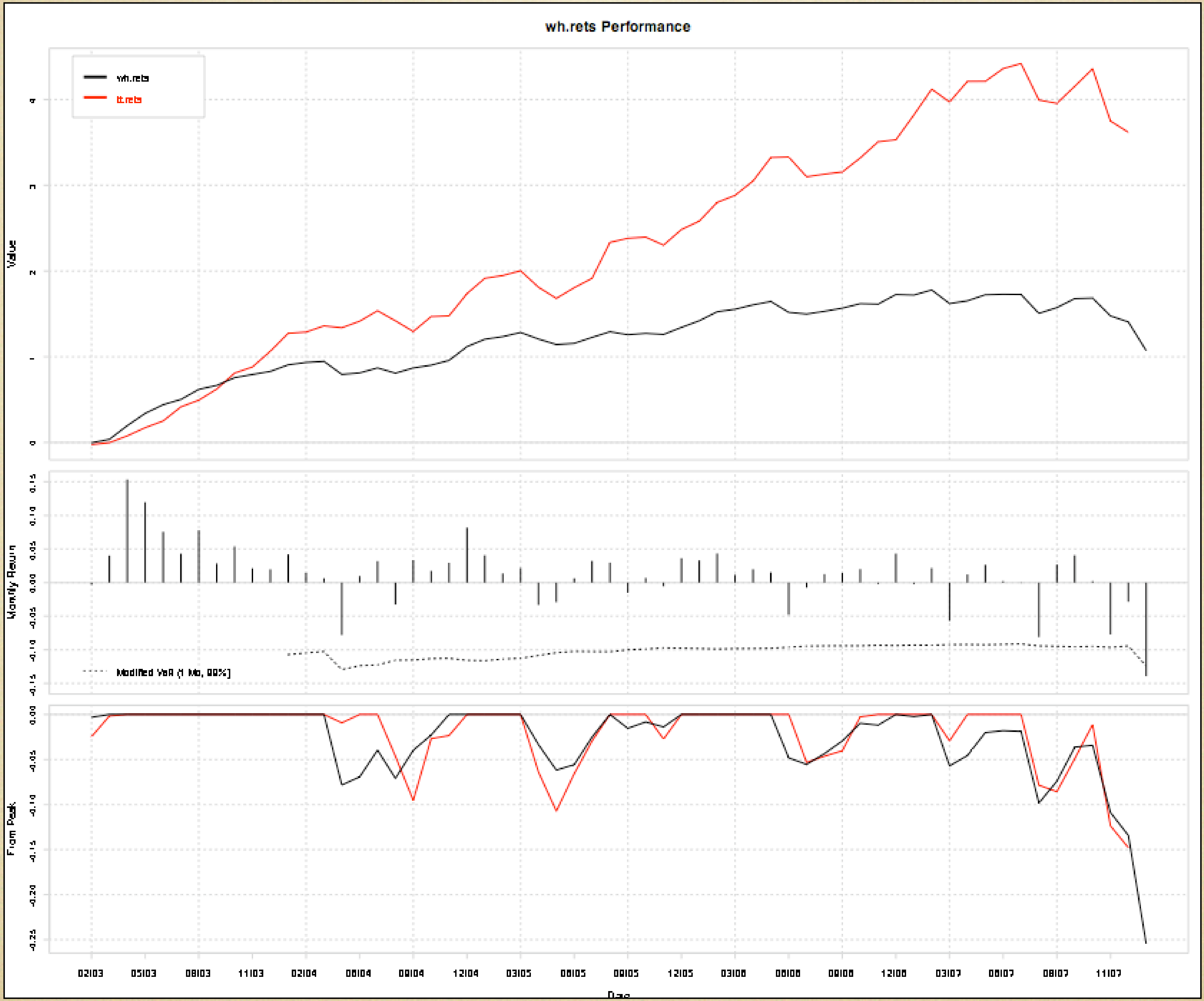
```
> table.AnnualizedReturns(merge(bt_Returns(t
rf=0.04/12)
                                 TT95     TT
Annualized Return              0.2882 0.3557
Annualized Std Dev             0.2594 0.1787
Annualized Sharpe (rf=3.96%) 0.9193 1.6994
> 
```



There was no sustained period when it out-performed



```
# Crude Haugen-like Factor payoff model
WhatsHot <- function(date) {
    factors <- c("PE", "MKTCAP", "EPS_GH1E0","DIV_Y7Y1", "ROA_12M", "PSPS", "RRS_52W", "NCPS_Q1")
    prevdate <- bt_PreviousDate(date)
    prev <- all.stocks(prevdate)
    prev <- screen.add(prev, c(factors, "PRICE"), date=prevdate)
    prev <- screen.field.rename(prev, "PRICE", "STARTPR")
    prev <- screen.add(prev, "PRICE", date=date, rename.to="ENDPRICE")
    prev <- screen.condition(prev, "MKTCAP", ">=", 50)
    prev <- screen.condition(prev, "STARTPR", ">=", 1)
    prev <- screen.na.rows(prev) ## get rid of NAs
    prev$RET <- log(as.numeric(prev$ENDPRICE)/as.numeric(prev$STARTPR))
    prev <- screen.field.setnumeric(prev, factors)
    prev$STARTPR <- NULL
    prev$ENDPRICE <- NULL
    payoffs <- lm(as.formula(paste("RET ~ ", paste(factors, collapse= "+"))), prev)$coefficients
    ## Get current data
    now <- all.stocks(date)
    now <- screen.add(now, c(factors, "PRICE"), date)
    now <- screen.na.rows(now)
    now <- screen.condition(now, "MKTCAP", ">=", 50)
    now <- screen.condition(now, "PRICE", ">=", 1)
    now <- screen.field.setnumeric(now, factors)
    ##print(dim(as.matrix(now[,factors])))
    now$RETS <- as.matrix(now[,factors]) %*% (payoffs[factors])
    now <- screen.na.rows(now)
    now <- screen.order(now, "RETS")
    now <- screen.top.percentile(now, 10)
    return(now)
}
```


wh.rets Performance
wh.rets
tt.rets
Value
0
1
2
3
4
Monthly Return
0.15
0.10
0.05
0.00
-0.05
-0.10
-0.15
Modified VaR (1 Mo, 99%)
From Peak
0.00
-0.05
-0.10
-0.15
-0.20
-0.25
02/03
05/03
08/03
11/03
02/04
06/04
09/04
12/04
03/05
06/05
09/05
12/05
03/06
06/06
09/06
12/06
03/07
06/07
09/07
11/07
Date

Agenda

- Screening vs. Backtesting
- Overview / PAST-SIPro Demo
- Setup PAST-SIPro
- The "R" programming language
- Sample backtests (Simple to Sophisticated)
- From Screens to Backtests
- ToDo List
- Questions?

(Partial) To Do List

- ValueLine Database Support
- Reading Sector/Industry database files
- Better DB File handling (save images to cache)
- Easier setup with conversion to Package
- GUI for screen building/simulation
- Testing on Windows
- Converting all SIPRO screens
- Keelix-like web interface?

Agenda

- Screening vs. Backtesting
- Overview / PAST-SIPro Demo
- Setup PAST-SIPro
- The "R" programming language
- Sample backtests (Simple to Sophisticated)
- From Screens to Backtests
- ToDo List
- Questions?

Questions?

Resources/Links

- www.r-project.org and cran.r-project.org
- CRAN Packages: PerformanceAnalytics, Rmetrics, Rggobi
- www.mayin.org/ajayshah/KB/R/index.html
- http://www.burns-stat.com/
- Econometrics in R: http://cran.r-project.org/doc/contrib/Farnsworth-EconometricsInR.pdf