Exploratory Data Analysis in Finance Using PerformanceAnalytics

Brian G. Peterson & Peter Carl

[1] Diamond Management & Technology Consultants
Chicago, IL
brian@braverock.com

[2] Guidance Capital
Chicago, IL
peter@braverock.com

UseR! International User and Developer Conference, Ames, Iowa, 8-10 Aug 2007

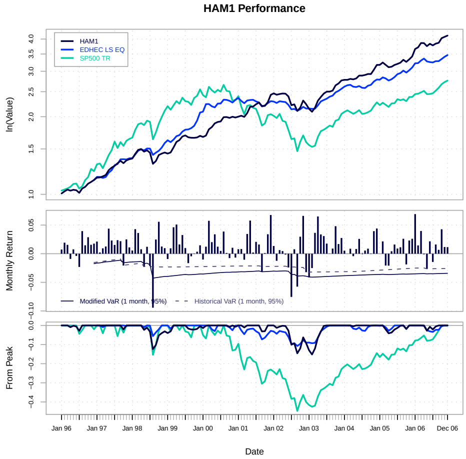
Outline

Overview

- Exploratory data analysis with finance data often starts with visual examination to:
  - examine properties of asset returns
  - compare an asset to other similar assets
  - compare an asset to one or more benchmarks
- Application of performance and risk measures can build a set of statistics for comparing possible investments
- Examples are developed using data for six (hypothetical) managers, a peer index, and an asset class index
- Hypothetical manager data was developed from real manager timeseries using *accuracy* and *perturb* packages to disguise the data while maintaining some of the statistical properties of the original data.

Draw a Performance Summary Chart.

```
> charts.PerformanceSummary(managers[, c(manager.col, indexes.cols)],
+     colorset = rich6equal, lwd = 2, ylog = TRUE)
```



Show Calendar Performance.

```
> t(table.CalendarReturns(managers[, c(manager.col, indexes.cols)]))

             1996 1997 1998 1999 2000  2001  2002 2003 2004 2005 2006
Jan           0.7  2.1  0.6 -0.9 -1.0   0.8   1.4 -4.1  0.5  0.0  6.9
Feb           1.9  0.2  4.3  0.9  1.2   0.8  -1.2 -2.5  0.0  2.1  1.5
Mar           1.6  0.9  3.6  4.6  5.8  -1.1   0.6  3.6  0.9 -2.1  4.0
Apr          -0.9  1.3  0.8  5.1  2.0   3.5   0.5  6.5 -0.4 -2.1 -0.1
May           0.8  4.4 -2.3  1.6  3.4   5.8  -0.2  3.4  0.8  0.4 -2.7
Jun          -0.4  2.3  1.2  3.3  1.2   0.2  -2.4  3.1  2.6  1.6  2.2
Jul          -2.3  1.5 -2.1  1.0  0.5   2.1  -7.5  1.8  0.0  0.9 -1.4
Aug           4.0  2.4 -9.4 -1.7  3.9   1.6   0.8  0.0  0.5  1.1  1.6
Sep           1.5  2.2  2.5 -0.4  0.1  -3.1  -5.8  0.9  0.9  2.6  0.7
Oct           2.9 -2.1  5.6 -0.1 -0.8   0.1   3.0  4.8 -0.1 -1.9  4.3
Nov           1.6  2.5  1.3  0.4  1.0   3.4   6.6  1.7  3.9  2.3  1.2
Dec           1.8  1.1  1.0  1.5 -0.7   6.8  -3.2  2.8  4.4  2.6  1.1
HAM1         13.6 20.4  6.1 16.1 17.7  22.4  -8.0 23.7 14.9  7.8 20.5
EDHEC LS EQ    NA 21.4 14.6 31.4 12.0  -1.2  -6.4 19.3  8.6 11.3 11.7
SP500 TR     23.0 33.4 28.6 21.0 -9.1 -11.9 -22.1 28.7 10.9  4.9 15.8
```

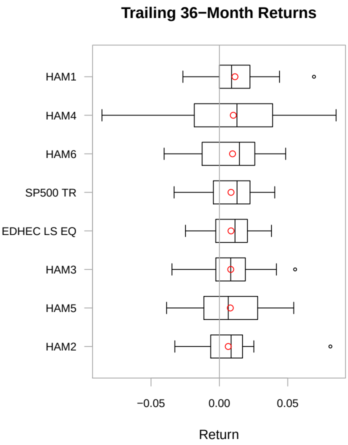
Calculate Statistics.

```
> table.Stats(managers[, c(manager.col, peers.cols)])

                    HAM1     HAM2     HAM3     HAM4    HAM5    HAM6
Observations    132.0000 125.0000 132.0000 132.0000 77.0000 64.0000
NAs               0.0000   7.0000   0.0000   0.0000 55.0000 68.0000
Minimum          -0.0944  -0.0371  -0.0718  -0.1759 -0.1320 -0.0404
Quartile 1        0.0000  -0.0098  -0.0054  -0.0198 -0.0164 -0.0016
Median            0.0112   0.0082   0.0102   0.0138  0.0038  0.0128
Arithmetic Mean   0.0111   0.0141   0.0124   0.0110  0.0041  0.0111
Geometric Mean    0.0108   0.0135   0.0118   0.0096  0.0031  0.0108
Quartile 3        0.0248   0.0252   0.0314   0.0460  0.0309  0.0255
Maximum           0.0692   0.1556   0.1796   0.1508  0.1747  0.0583
SE Mean           0.0022   0.0033   0.0032   0.0046  0.0052  0.0030
LCL Mean (0.95)   0.0067   0.0076   0.0062   0.0019 -0.0063  0.0051
UCL Mean (0.95)   0.0155   0.0206   0.0187   0.0202  0.0145  0.0170
Variance          0.0007   0.0013   0.0013   0.0028  0.0021  0.0006
Stdev             0.0256   0.0367   0.0365   0.0532  0.0457  0.0238
Skewness         -0.6514   1.4406   0.7819  -0.4262  0.0724 -0.2735
Kurtosis          2.2807   2.2937   2.5972   0.8049  2.1772 -0.4311
```

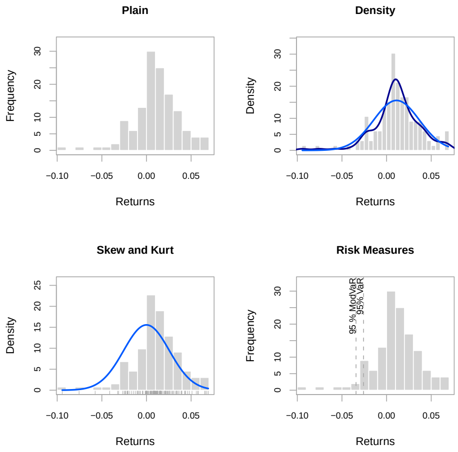
Compare Distributions.

```
> chart.Boxplot(managers[trailing36.rows, c(manager.col, peers.cols,
+     indexes.cols)], main = "Trailing 36-Month Returns")
```



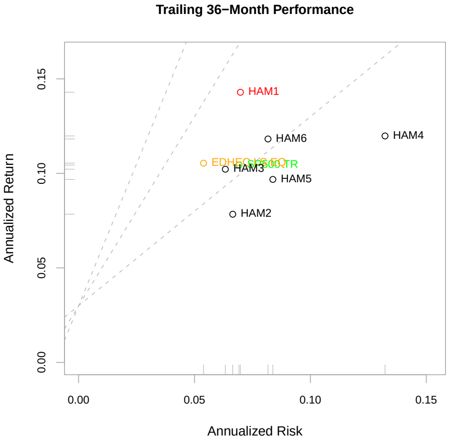
Compare Distributions.

```
> layout(rbind(c(1, 2), c(3, 4)))
> chart.Histogram(managers[, 1, drop = F], main = "Plain", methods = NULL)
> chart.Histogram(managers[, 1, drop = F], main = "Density", breaks = 40,
+     methods = c("add.density", "add.normal"))
> chart.Histogram(managers[, 1, drop = F], main = "Skew and Kurt",
+     methods = c("add.centered", "add.rug"))
> chart.Histogram(managers[, 1, drop = F], main = "Risk Measures",
+     methods = c("add.risk"))
```



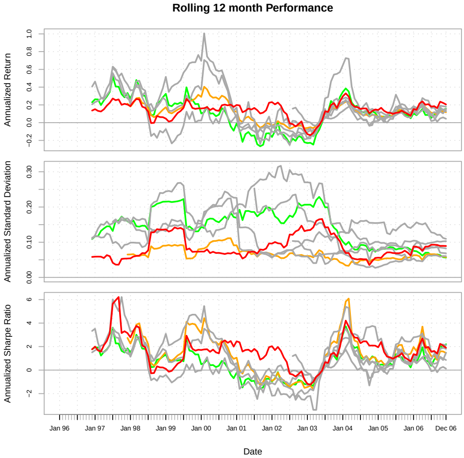
Show Relative Return and Risk.

```
> chart.RiskReturnScatter(managers[trailing36.rows, 1:8], Rf = 0.03/12,
+     main = "Trailing 36-Month Performance", colorset = c("red",
+         rep("black", 5), "orange", "green"))
```



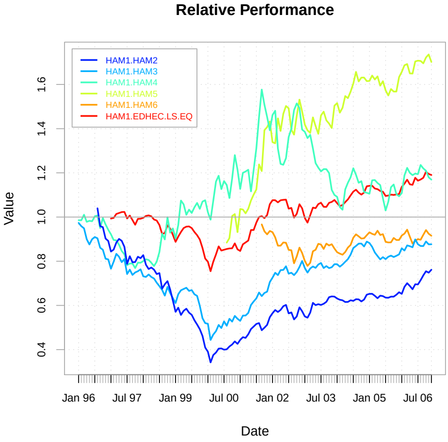
Examine Performance Consistency.

```
> charts.RollingPerformance(managers[, c(manager.col, peers.cols,
+     indexes.cols)], Rf = 0.03/12, colorset = c("red", rep("darkgray",
+     5), "orange", "green"), lwd = 2)
```



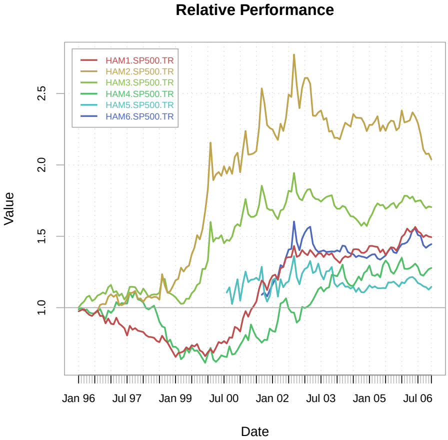
Display Relative Performance.

```
> chart.RelativePerformance(managers[, manager.col, drop = FALSE],
+     managers[, c(peers.cols, 7)], colorset = tim8equal[-1], lwd = 2,
+     legend.loc = "topleft")
```



Compare to a Benchmark.

```
> chart.RelativePerformance(managers[, c(manager.col, peers.cols)],
+     managers[, 8, drop = F], colorset = rainbow8equal, lwd = 2,
+     legend.loc = "topleft")
```



Compare to a Benchmark.

```
> table.CAPM(managers[trailing36.rows, c(manager.col, peers.cols)],
+     managers[trailing36.rows, 8, drop = FALSE], Rf = managers[trailing36.rows,
+         Rf.col, drop = F])
                        HAM1 to SP500 TR HAM2 to SP500 TR HAM3 to SP500 TR
Alpha                             0.0051           0.0020           0.0020
Beta                              0.6267           0.3223           0.6320
Beta+                             0.8227           0.4176           0.8240
Beta-                             1.1218          -0.0483           0.8291
R-squared                         0.3829           0.1073           0.4812
Annualized Alpha                  0.0631           0.0247           0.0243
Correlation                       0.6188           0.3276           0.6937
Correlation p-value               0.0001           0.0511           0.0000
Tracking Error                    0.0606           0.0426           0.0042
Active Premium                    0.0373          -0.0254          -0.0021
Information Ratio                 0.6157          -0.5973          -0.5051
Treynor Ratio                     0.1741           0.1437           0.1101
                        HAM4 to SP500 TR HAM5 to SP500 TR HAM6 to SP500 TR
Alpha                             0.0009           0.0002           0.0022
Beta                              1.1282           0.8755           0.8150
Beta+                             1.8430           1.0985           0.9993
Beta-                             1.2223           0.5283           1.1320
R-squared                         0.3444           0.5209           0.4757
Annualized Alpha                  0.0109           0.0030           0.0271
Correlation                       0.5868           0.7218           0.6897
Correlation p-value               0.0002           0.0000           0.0000
Tracking Error                    0.0353           0.0105           0.0236
Active Premium                    0.0149          -0.0075           0.0134
Information Ratio                 0.4232          -0.7121           0.5684
Treynor Ratio                     0.0768           0.0734           0.1045
```

Calculate Returns.

- The single-period arithmetic return, or simple return, can be calculated as

$$R_t = \frac{P_t}{P_{t-1}} - 1 = \frac{P_t - P_{t-1}}{P_{t-1}} \tag{1}$$

- Simple returns, cannot be added together. A multiple-period simple return is calculated as:

$$R_t = \frac{P_t}{P_{t-k}} - 1 = \frac{P_t - P_{t-k}}{P_{t-k}} \tag{2}$$

- The natural logarithm of the simple return of an asset is referred to as the continuously compounded return, or *log return*:

$$r_t = ln(1 + R_t) = ln\frac{P_t}{P_{t-1}} = p_t - p_{t-1} \tag{3}$$

- Calculating log returns from simple gross return, or vice versa:

$$r_t = ln(1 + R_t), R_t = exp(r_t) - 1. \tag{4}$$

- *Return.calculate* or *CalculateReturns* (now deprecated) may be used to compute discrete and continuously compounded returns for data containing asset prices.

table.CAPM underlying techniques

- Return.annualized — Annualized return using

$$prod(1+R_a)^{\frac{scale}{n}} - 1 = \sqrt[n]{prod(1+R_a)^{scale}} - 1 \tag{5}$$

- TreynorRatio — ratio of asset's Excess Return to Beta β of the benchmark

$$\frac{(\overline{R_a - R_f})}{\beta_{a,b}} \tag{6}$$

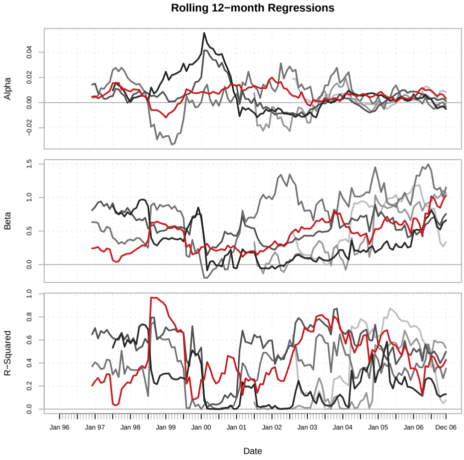
- ActivePremium — investment's annualized return minus the benchmark's annualized return
- Tracking Error — A measure of the unexplained portion of performance relative to a benchmark, given by

$$TrackingError = \sqrt{\sum \frac{(R_a - R_b)^2}{len(R_a)\sqrt{scale}}} \tag{7}$$

- InformationRatio — ActivePremium/TrackingError

Compare to a Benchmark.

```
> charts.RollingRegression(managers[, c(manager.col, peers.cols),
+     drop = FALSE], managers[, 8, drop = FALSE], Rf = 0.03/12,
+     colorset = redfocus, lwd = 2)
```



Calculate Downside Risk.

```
> table.DownsideRisk(managers[, 1:6], Rf = 0.03/12)

                                HAM1    HAM2    HAM3    HAM4    HAM5    HAM6
Semi Deviation                0.0270  0.0258  0.0319  0.0576  0.0456  0.0260
Gain Deviation                0.0169  0.0347  0.0290  0.0311  0.0313  0.0149
Loss Deviation                0.0211  0.0107  0.0191  0.0365  0.0324  0.0128
Downside Deviation (MAR=10%)  0.0273  0.0226  0.0313  0.0585  0.0464  0.0253
Downside Deviation (Rf=3%)    0.0281  0.0190  0.0295  0.0562  0.0463  0.0238
Downside Deviation (0%)       0.0291  0.0171  0.0291  0.0548  0.0451  0.0229
Maximum Drawdown             -0.1518 -0.2399 -0.2894 -0.2874 -0.3405 -0.0788
Historical VaR (95%)         -0.0258 -0.0294 -0.0425 -0.0799 -0.0733 -0.0341
Historical ES (95%)          -0.0513 -0.0331 -0.0555 -0.1122 -0.1023 -0.0392
Modified VaR (95%)           -0.0342 -0.0276 -0.0368 -0.0815 -0.0676 -0.0298
Modified ES (95%)            -0.0610 -0.0614 -0.0440 -0.1176 -0.0974 -0.0390
```

Semivariance and Downside Deviation

- Downside Deviation as proposed by Sharpe is a generalization of semivariance which calculates bases on the deviation below a Minimumn Acceptable Return(MAR)

$$\delta_{MAR} = \sqrt{\frac{\sum_{t=1}^{n}(R_t - MAR)^2}{n}} \tag{8}$$

- Downside Deviation may be used to calculate semideviation by setting MAR=mean(R) or may also be used with MAR=0
- Downside Deviation (and its special cases semideviation and semivariance) is useful in several performance to risk ratios, and in several portfolio optimization problems.

Value at Risk

- Value at Risk (VaR) has become a required standard risk measure recognized by Basel II and MiFID
- Traditional mean-VaR may be derived historically, or estimated parametrically using

$$z_c = q_p = qnorm(p) \tag{9}$$

$$VaR = \bar{R} - z_c \cdot \sqrt{\sigma} \tag{10}$$

- Even with robust covariance matrix or Monte Carlo simulation, mean-VaR is not reliable for non-normal asset distributions
- For non-normal assets, VaR estimates calculated using GPD (as in VaR.GPD) or Cornish Fisher perform best
- Modified Cornish Fisher VaR takes higher moments of the distribution into account:

$$z_{cf} = z_c + \frac{(z_c^2 - 1)S}{6} + \frac{(z_c^3 - 3z_c)K}{24} + \frac{(2z_c^3 - 5z_c)S^2}{36} \tag{11}$$

$$modVaR = \bar{R} - z_{cf}\sqrt{\sigma} \tag{12}$$

- Modified VaR also meets the definition of a coherent risk measure per Artzner,et.al.(1997)

Risk/Reward Ratios in *PerformanceAnalytics*

- SharpeRatio — return per unit of risk represented by variance, may also be annualized by

$$\frac{\sqrt[n]{prod(1+R_a)^{scale}}-1}{\sqrt{scale}\cdot\sqrt{\sigma}} \tag{13}$$

- Sortino Ratio — improvement on Sharpe Ratio utilizing downside deviation as the measure of risk

$$\frac{(\overline{R_a - MAR})}{\delta_{MAR}} \tag{14}$$

- Calmar and Sterling Ratios — ratio of annualized return (Eq. 1) over the absolute value of the maximum drawdown
- Sortino's Upside Potential Ratio — upside semdiviation from MAR over downside deviation from MAR

$$\frac{\sum_{t=1}^{n}(R_t - MAR)}{\delta_{MAR}} \tag{15}$$

- Favre's modified Sharpe Ratio — ratio of excess return over Cornish-Fisher VaR

$$\frac{(\overline{R_a - R_f})}{modVaR_{R_a,p}} \tag{16}$$

Summary

- Performance and risk analysis are greatly facilitated by the use of charts and tables.
- The display of your infomation is in many cases as important as the analysis.
- *PerformanceAnalytics* contains several tool for measuring and visualizing data that may be used to aid investment decision making.

- Further Work
  - Additional parameterization to make charts and tables more useful.
  - Pertrac or Morningstar-style sample reports.
  - Functions and graphics for more complicated topics such as factor analysis and optimization.

Install PerformanceAnalytics.

- As of version 0.9.4, PerformanceAnalytics is available in CRAN
- Version 0.9.5 was released at the beginning of July
- Install with:
  `> install.packages("PerformanceAnalytics")`
- Required packages include `Hmisc`, `zoo`, and Rmetrics packages such as `fExtremes`.
- Load the library into your active R session using:
  `> library("PerformanceAnalytics").`

Load and Review Data.

```
> data(managers)
> head(managers)

              HAM1 HAM2    HAM3    HAM4 HAM5 HAM6 EDHEC LS EQ SP500 TR
1996-01-30  0.0074   NA  0.0349  0.0222   NA   NA          NA   0.0340
1996-02-28  0.0193   NA  0.0351  0.0195   NA   NA          NA   0.0093
1996-03-30  0.0155   NA  0.0258 -0.0098   NA   NA          NA   0.0096
1996-04-29 -0.0091   NA  0.0449  0.0236   NA   NA          NA   0.0147
1996-05-30  0.0076   NA  0.0353  0.0028   NA   NA          NA   0.0258
1996-06-29 -0.0039   NA -0.0303 -0.0019   NA   NA          NA   0.0038
           US 10Y TR US 3m TR
1996-01-30   0.00380  0.00456
1996-02-28  -0.03532  0.00398
1996-03-30  -0.01057  0.00371
1996-04-29  -0.01739  0.00428
1996-05-30  -0.00543  0.00443
1996-06-29   0.01507  0.00412
```

Set Up Data for Analysis.

```
> dim(managers)

[1] 132  10

> managers.length = dim(managers)[1]
> colnames(managers)

 [1] "HAM1"         "HAM2"         "HAM3"         "HAM4"         "HAM5"
 [6] "HAM6"         "EDHEC LS EQ" "SP500 TR"    "US 10Y TR"    "US 3m TR"

> manager.col = 1
> peers.cols = c(2, 3, 4, 5, 6)
> indexes.cols = c(7, 8)
> Rf.col = 10
> trailing12.rows = ((managers.length - 11):managers.length)
> trailing12.rows

 [1] 121 122 123 124 125 126 127 128 129 130 131 132

> trailing36.rows = ((managers.length - 35):managers.length)
> trailing60.rows = ((managers.length - 59):managers.length)
> frInception.rows = (length(managers[, 1]) - length(managers[,
+     1][!is.na(managers[, 1])]) + 1):length(managers[, 1])
```

Draw a Performance Summary Chart.

```
> charts.PerformanceSummary(managers[, c(manager.col, indexes.cols)],
+     colorset = rich6equal, lwd = 2, ylog = TRUE)
```