Presentation of Performance Attribution from Bacon

Vignette	Matthieu Lestel
Presentation	Joe Wayne Byers

March 19, 2020

Abstract

This vignette gives a brief overview of the functions developed in Bacon(2008) to evaluate the performance and risk of portfolios that are included in **PerformanceAnalytics** and how to use them. There are some tables at the end which give a quick overview of similar functions. The page number next to each function is the location of the function in Bacon(2008)

Table of Contents

Frequency (p.64)

Gives the period of the return distribution (ie 12 if monthly return, 4 if quarterly return).[1]

```
> data(portfolio_bacon);
> print(Frequency(portfolio_bacon[,1]));
[1] 12
```

[1] Expected = 12

Sharpe Ratio (p.64)

The Sharpe ratio is simply the return per unit of risk (represented by variability). In the classic case, the unit of risk is the standard deviation of the returns.

$$\frac{\overline{(R_a - R_f)}}{\sqrt{\sigma_{(R_a - R_f)}}}$$

```
> data(managers)
> SharpeRatio(managers[,1,drop=FALSE], Rf=.035/12, FUN="StdDev")

                                      HAM1
StdDev Sharpe (Rf=0.3%, p=95%): 0.3201889
```

Risk-adjusted return: MSquared (p.67)

M^2 is a risk adjusted return useful to judge the size of relative performance between differents portfolios. With it you can compare portfolios with different levels of risk.[2]

$$M^2 = r_P + SR * (\sigma_M - \sigma_P) = (r_P - r_F) * \frac{\sigma_M}{\sigma_P} + r_F$$

where r_P is the portfolio return annualized, σ_M is the market risk and σ_P is the portfolio risk.

```
> data(portfolio_bacon)
> print(MSquared(portfolio_bacon[,1], portfolio_bacon[,2]))

                    benchmark.return....
benchmark.return....              0.10062
```

[2] Expected = 0.1068

MSquared Excess (p.68)

M^2 excess is the quantity above the standard M. There is a geometric excess return which is better for Bacon and an arithmetic excess return.[3]

$$M^2 excess(geometric) = \frac{1 + M^2}{1 + b} - 1$$
$$M^2 excess(arithmetic) = M^2 - b$$

where M^2, is MSquared and b is the benchmark annualised return.

```
> data(portfolio_bacon)
> print(MSquaredExcess(portfolio_bacon[,1], portfolio_bacon[,2]))

                    benchmark.return....
benchmark.return....         -0.01553103

> print(MSquaredExcess(portfolio_bacon[,1], portfolio_bacon[,2],
+                      Method="arithmetic"))

                    benchmark.return....
benchmark.return....         -0.01736344
```

[3] Expected = -0.00998 and -0.011

Regression equation (p.71)

This is a regression equation

$$r_P = \alpha + \beta * b + \epsilon$$

Regression alpha (p.71)

"Alpha" purports to be a measure of a manager's skill by measuring the portion of the managers returns that are not attributable to "Beta", or the portion of performance attributable to a benchmark.

```
> data(managers)
> print(CAPM.alpha(managers[,1,drop=FALSE], managers[,8,drop=FALSE],
+                  Rf=.035/12))

[1] 0.005960609
```

Regression beta (p.71)

CAPM Beta is the beta of an asset to the variance and covariance of an initial portfolio. Used to determine diversification potential.

```
> data(managers)
> CAPM.beta(managers[, "HAM2", drop=FALSE], managers[, "SP500 TR", drop=FALSE], 
```

```
[1] 0.3383942
```

Regression epsilon (p.71)

The regression epsilon is an error term measuring the vertical distance between the return predicted by the equation and the real result.[4]

$$\epsilon_r = r_p - \alpha_r - \beta_r * b$$

where α_r is the regression alpha, β_r is the regression beta, r_p is the portfolio return and b is the benchmark return.

```
> data(managers)
> print(CAPM.epsilon(portfolio_bacon[,1], portfolio_bacon[,2]))

[1] -0.01313932
```

[4] Expected = -0.013

Jensen's alpha (p.72)

The Jensen's alpha is the intercept of the regression equation in the Capital Asset Pricing Model and is in effect the exess return adjusted for systematic risk.[5]

$$\alpha = r_p - r_f - \beta_p * (b - r_f)$$

where r_f is the risk free rate, β_r is the regression beta, r_p is the portfolio return and b is the benchmark return.

```
> data(portfolio_bacon)
> print(CAPM.jensenAlpha(portfolio_bacon[,1], portfolio_bacon[,2]))

[1] -0.01416944
```

[5] Expected = -0.014

Systematic Risk (p.75)

Systematic risk as defined by Bacon(2008) is the product of beta by market risk. Be careful! It's not the same definition as the one given by Michael Jensen. Market risk is the standard deviation of the benchmark. The systematic risk is annualized.[6]

$$\sigma_s = \beta * \sigma_m$$

where σ_s is the systematic risk, β is the regression beta and σ_m is the market risk.

```
> data(portfolio_bacon)
> print(SystematicRisk(portfolio_bacon[,1], portfolio_bacon[,2]))
[1] 0.132806
```

[6] Expected = 0.013

Specific Risk (p.75)

Specific risk is the standard deviation of the error term in the regression equation.[7]

```
> data(portfolio_bacon)
> print(SpecificRisk(portfolio_bacon[,1],
+                    portfolio_bacon[,2]))
[1] 0.03293109
```

[7] Expected = 0.0329

Total Risk (p.75)

The square of total risk is the sum of the square of systematic risk and the square of specific risk. Specific risk is the standard deviation of the error term in the regression equation. Both terms are annualized to calculate total risk.[8]

$$TotalRisk = \sqrt{SystematicRisk^2 + SpecificRisk^2}$$

```
> data(portfolio_bacon)
> print(TotalRisk(portfolio_bacon[,1], portfolio_bacon[,2]))
[1] 0.136828
```

[8] Expected = 0.0134

Treynor ratio (p.75)

The Treynor ratio is similar to the Sharpe Ratio, except it uses beta as the volatility measure (to divide the investment's excess return over the beta).

$$TreynorRatio = \frac{\overline{(R_a - R_f)}}{\beta_{a,b}}(mean(Ra - Rf))/(Beta(Ra, Rb))$$

```
> data(managers)
> print(round(TreynorRatio(managers[,1,drop=FALSE], managers[,8,drop=FALSE],
+                               Rf=.035/12),4))
```

```
[1] 0.2528
```

Modified Treynor ratio (p.77)

To calculate modified Treynor ratio, we divide the numerator by the systematic risk instead of the beta.[9]

```
> data(portfolio_bacon)
> print(TreynorRatio(portfolio_bacon[,1], portfolio_bacon[,2],
+                    modified = TRUE))
```

```
[1] 0.7806747
```

[9] Expected = 1.677

Appraisal ratio (or Treynor-Black ratio) (p.77)

Appraisal ratio is the Jensen's alpha adjusted for specific risk. The numerator is divided by specific risk instead of total risk.[10]

$$Appraisalratio = \frac{\alpha}{\sigma_\epsilon}$$

where *alpha* is the Jensen's alpha and $\sigma_{epsilon}$ is the specific risk.

```
> data(portfolio_bacon)
> print(AppraisalRatio(portfolio_bacon[,1], portfolio_bacon[,2],
+                        method="appraisal"))
```

```
[1] -0.4302756
```

[10] Expected = -0.430

Modified Jensen (p.77)

Modified Jensen's alpha is Jensen's alpha divided by beta.

$$ModifiedJensen's alpha = \frac{\alpha}{\beta}$$

where *alpha* is the Jensen's alpha.

```
> data(portfolio_bacon)
> print(AppraisalRatio(portfolio_bacon[,1], portfolio_bacon[,2],
+                      method="modified"))

[1] -0.01418576
```

Fama decomposition (p.77)

Fama beta is a beta used to calculate the loss of diversification. It is made so that the systematic risk is equivalent to the total portfolio risk.[11]

$$\beta_F = \frac{\sigma_P}{\sigma_M}$$

where σ_P is the portfolio standard deviation and σ_M is the market risk.

```
> data(portfolio_bacon)
> print(FamaBeta(portfolio_bacon[,1], portfolio_bacon[,2]))

                            portfolio.monthly.return....
portfolio.monthly.return....                    1.030395
```

[11] Expected = 1.03

Selectivity (p.78)

Selectivity is the same as Jensen's alpha

$$Selectivity = r_p - r_f - \beta_p * (b - r_f)$$

where r_f is the risk free rate, β_r is the regression beta, r_p is the portfolio return and b is the benchmark return.[12]

```
> data(portfolio_bacon)
> print(Selectivity(portfolio_bacon[,1], portfolio_bacon[,2]))

[1] -0.01416944
```

[12] Expected = -0.0141

Net selectivity (p.78)

Net selectivity is the remaining selectivity after deducting the amount of return require to justify not being fully diversified.[13] If net selectivity is negative the portfolio manager has not justified the loss of diversification.

$$Netselectivity = \alpha - d$$

where α is the selectivity and d is the diversification.

```
> data(portfolio_bacon)
> print(NetSelectivity(portfolio_bacon[,1], portfolio_bacon[,2]))

                             portfolio.monthly.return....
portfolio.monthly.return....                   -0.0178912
```

[13] Expected = -0.017

Tracking error (p.78)

A measure of the unexplained portion of performance relative to a benchmark.
Tracking error is calculated by taking the square root of the average of the squared deviations between the investment's returns and the benchmark's returns, then multiplying the result by the square root of the scale of the returns.

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

```
> data(managers)
> TrackingError(managers[,1,drop=FALSE], managers[,8,drop=FALSE])

[1] 0.1131667
```

Information ratio (p.80)

The Active Premium divided by the Tracking Error.

$InformationRatio = ActivePremium/TrackingError$

This relates the degree to which an investment has beaten the benchmark to the consistency with which the investment has beaten the benchmark.

```
> data(managers)
> InformationRatio(managers[,"HAM1",drop=FALSE],
+                  managers[, "SP500 TR", drop=FALSE])

[1] 0.3604125
```

Skewness (p.83)

Measures the deformation from a normal deformation.

$$Skewness = \frac{1}{n} * \sum_{i=1}^{n} (\frac{r_i - \overline{r}}{\sigma_P})^3$$

where n is the number of return, $\overline{r}$ is the mean of the return distribution, σ_P is the standard deviation of the distribution and σ_{S_P} is the sample standard deviation of the distribution.

```
> data(managers)
> skewness(managers)
```

```
               HAM1    HAM2      HAM3       HAM4       HAM5       HAM6
Skewness -0.6588445 1.45804 0.7908285 -0.4310631 0.07380869 -0.2799993
         EDHEC LS EQ   SP500 TR  US 10Y TR  US 3m TR
Skewness  0.01773013 -0.5531032 -0.4048722 -0.328171
```

Sample skewness (p.84)[14]

$$SampleSkewness = \frac{n}{(n-1)*(n-2)} * \sum_{i=1}^{n} (\frac{r_i - \overline{r}}{\sigma_{S_P}})^3$$

where n is the number of return, $\overline{r}$ is the mean of the return distribution, σ_P is the standard deviation of the distribution and σ_{S_P} is the sample standard deviation of the distribution.

```
> data(portfolio_bacon)
> print(skewness(portfolio_bacon[,1], method="sample"))

[1] -0.09398414
```

[14] Expected = -0.09

Kurtosis (p.84)

Kurtosis measures the weight or returns in the tails of the distribution relative to standard deviation.[15]

$$Kurtosis(moment) = \frac{1}{n} * \sum_{i=1}^{n} (\frac{r_i - \overline{r}}{\sigma_P})^4$$

where n is the number of return, $\overline{r}$ is the mean of the return distribution, σ_P is standard deviation of the distribution and σ_{S_P} is sample standard deviation of the distribution.

```
> data(portfolio_bacon)
> print(kurtosis(portfolio_bacon[,1], method="moment"))
```

```
[1] 2.432454
```

[15] Expected = 2.43

Excess kurtosis (p.85)[16]

$$ExcessKurtosis = \frac{1}{n} * \sum_{i=1}^{n} (\frac{r_i - \overline{r}}{\sigma_P})^4 - 3$$

where n is the number of return, $\overline{r}$ is the mean of the return distribution, σ_P is standard deviation of the distribution and σ_{S_P} is the sample standard deviation of the distribution.

```
> data(portfolio_bacon)
> print(kurtosis(portfolio_bacon[,1], method="excess")) #expected -0.57

[1] -0.5675462
```

[16] Expected = -0.57

Sample kurtosis (p.85)[17]

$$Samplekurtosis = \frac{n*(n+1)}{(n-1)*(n-2)*(n-3)} * \sum_{i=1}^{n} (\frac{r_i - \overline{r}}{\sigma_{S_P}})^4$$

where n is the number of return, $\overline{r}$ is the mean of the return distribution, σ_P is the standard deviation of the distribution and σ_{S_P} is the sample standard deviation of the distribution.

```
> data(portfolio_bacon)
> print(kurtosis(portfolio_bacon[,1], method="sample"))
```

```
[1] 3.027405
```

[17] Expected = 3.03

Sample excess kurtosis (p.85)[18]

$$Sampleexcesskurtosis = \frac{n*(n+1)}{(n-1)*(n-2)*(n-3)} * \sum_{i=1}^{n}(\frac{r_i - \overline{r}}{\sigma_{S_P}})^4 - \frac{3*(n-1)^2}{(n-2)*(n-3)}$$

where n is the number of return, $\overline{r}$ is the mean of the return distribution, σ_P is the standard deviation of the distribution and σ_{S_P} is the sample standard deviation of the distribution.

```
> data(portfolio_bacon)
> print(kurtosis(portfolio_bacon[,1], method="excess"))

[1] -0.5675462
```

[18] Expected = -0.57

Pain index (p.89)

The pain index is the mean value of the drawdowns over the entire analysis period. The measure is similar to the Ulcer index except that the drawdowns are not squared. Also, it's different than the average drawdown, in that the numerator is the total number of observations rather than the number of drawdowns. Visually, the pain index is the area of the region that is enclosed by the horizontal line at zero percent and the drawdown line in the Drawdown chart.[19]

$$Painindex = \sum_{i=1}^{n} \frac{\mid D'_i \mid}{n}$$

where n is the number of observations of the entire series, D'_i is the drawdown since previous peak in period i.

```
> data(portfolio_bacon)
> print(PainIndex(portfolio_bacon[,1]))
```

```
           portfolio.monthly.return....
Pain Index                    0.0390113
```

[19] Expected = 0.04

Calmar ratio (p.89)

Calmar ratio is another method of creating a risk-adjusted measure for ranking investments similar to the Sharpe ratio.

Sterling ratio (p.89)

Sterling ratio is another method of creating a risk-adjusted measure for ranking investments similar to the Sharpe ratio.

```
> data(managers)
> SterlingRatio(managers[,1,drop=FALSE])
                                  HAM1
Sterling Ratio (Excess = 10%) 0.5462542
```

Burke ratio (p.90)

To calculate Burke ratio, we take the difference between the portfolio return and the risk free rate and we divide it by the square root of the sum of the square of the drawdowns.[20]

$$BurkeRatio = \frac{r_P - r_F}{\sqrt{\sum_{t=1}^{d} {D_t}^2}}$$

where d is number of drawdowns, r_P is the portfolio return, r_F is the risk free rate and D_t the t^{th} drawdown.

```
> data(portfolio_bacon)
> print(BurkeRatio(portfolio_bacon[,1]))
[1] 0.7447309
```

[20] Expected = 0.74

Modified Burke ratio (p.91)

To calculate the modified Burke ratio, we just multiply the Burke ratio by the square root of the number of observations.[21]

$$ModifiedBurkeRatio = \frac{r_P - r_F}{\sqrt{\sum_{t=1}^{d} \frac{{D_t}^2}{n}}}$$

where n is the number of observations of the entire series, d is number of drawdowns, r_P is the portfolio return, r_F is the risk free rate and D_t the t^{th} drawdown. The denominator in the modified Burke ratio is the Drawdown Deviation.

```
> data(portfolio_bacon)
> print(BurkeRatio(portfolio_bacon[,1], modified = TRUE))
[1] 3.648421
```

[21] Expected = 3.65

Martin ratio (p.91)

To calculate Martin ratio, we divide the difference of the portfolio return and the risk free rate by the Ulcer index.[22]

$$Martinratio = \frac{r_P - r_F}{\sqrt{\sum_{i=1}^{n} \frac{{D'_i}^2}{n}}}$$

where r_P is the annualized portfolio return, r_F is the risk free rate, n is the number of observations of the entire series, D'_i is the drawdown since previous peak in period i. The denominator in the Martin ration is the Ulcer Index.

```
> data(portfolio_bacon)
> print(MartinRatio(portfolio_bacon[,1]))
```

```
               portfolio.monthly.return....
Ulcer Index                         1.70772
```

[22] Expected = 1.70

Pain ratio (p.91)

To calculate Pain ratio, we divide the difference of the portfolio return and the risk free rate by the Pain index.[23]

$$Painratio = \frac{r_P - r_F}{\sum_{i=1}^{n} \frac{|D'_i|}{n}}$$

where r_P is the annualized portfolio return, r_F is the risk free rate, n is the number of observations of the entire series, D'_i is the drawdown since previous peak in period i. The denominator in the Pain ratio is the Pain Index.

```
> data(portfolio_bacon)
> print(PainRatio(portfolio_bacon[,1]))

           portfolio.monthly.return....
Pain Index                     2.657647
```

[23] Expected = 2.66

Downside risk (p.92)

Downside deviation, similar to semi deviation, eliminates positive returns when calculating risk. Instead of using the mean return or zero, it uses the Minimum Acceptable Return as proposed by Sharpe (which may be the mean historical return or zero). It measures the variability of underperformance below a minimum targer rate. The downside variance is the square of the downside potential.[24]

$$DownsideDeviation(R, MAR) = \delta_{MAR} = \sqrt{\sum_{t=1}^{n} \frac{min[(R_t - MAR), 0]^2}{n}}$$

$$DownsideVariance(R, MAR) = \sum_{t=1}^{n} \frac{min[(R_t - MAR), 0]^2}{n}$$

$$DownsidePotential(R, MAR) = \sum_{t=1}^{n} \frac{min[(R_t - MAR), 0]}{n}$$

where n is either the number of observations of the entire series or the number of observations in the subset of the series falling below the MAR.

```
> data(portfolio_bacon)
> MAR = 0.5
> DownsideDeviation(portfolio_bacon[,1], MAR)

[1] 0.492524

> DownsidePotential(portfolio_bacon[,1], MAR)

[1] 0.491
```

[24] Expected = 0.493 and 0.491

UpsideRisk (p.92)

Upside Risk is the similar of semideviation taking the return above the Minimum Acceptable Return instead of using the mean return or zero.[25]

$$UpsideRisk(R, MAR) = \sqrt{\sum_{t=1}^{n} \frac{max[(R_t - MAR), 0]^2}{n}}$$

$$UpsideVariance(R, MAR) = \sum_{t=1}^{n} \frac{max[(R_t - MAR), 0]^2}{n}$$

$$UpsidePotential(R, MAR) = \sum_{t=1}^{n} \frac{max[(R_t - MAR), 0]}{n}$$

where n is either the number of observations of the entire series or the number of observations in the subset of the series falling below the MAR.

```
> data(portfolio_bacon)
> MAR = 0.005
> print(UpsideRisk(portfolio_bacon[,1], MAR, stat="risk"))

[1] 0.02937332

> print(UpsideRisk(portfolio_bacon[,1], MAR, stat="variance"))

[1] 0.0008627917

> print(UpsideRisk(portfolio_bacon[,1], MAR, stat="potential"))

[1] 0.01770833
```

25 Expected = 0.02937, 0.08628 and 0.01771

Downside frequency (p.94)

To calculate Downside Frequency, we take the subset of returns that are less than the target (or Minimum Acceptable Returns (MAR)) returns and divide the length of this subset by the total number of returns.[26]

$$DownsideFrequency(R, MAR) = \sum_{t=1}^{n} \frac{min[(R_t - MAR), 0]}{R_t * n}$$

where n is the number of observations of the entire series.

```
> data(portfolio_bacon)
> MAR = 0.005
> print(DownsideFrequency(portfolio_bacon[,1], MAR))
[1] 0.4583333
```

[26] Expected = 0.458

Bernardo and Ledoit ratio (p.95)

To calculate Bernardo and Ledoit ratio, we take the sum of the subset of returns that are above 0 and we divide it by the opposite of the sum of the subset of returns that are below 0.[27]

$$BernardoLedoitRatio(R) = \frac{\frac{1}{n}\sum_{t=1}^{n} max(R_t, 0)}{\frac{1}{n}\sum_{t=1}^{n} max(-R_t, 0)}$$

where n is the number of observations of the entire series.

```
> data(portfolio_bacon)
> print(BernardoLedoitRatio(portfolio_bacon[,1]))
[1] 1.779783
```

[27] Expected = 1.78

d ratio (p.95)

The d ratio is similar to the Bernado Ledoit ratio but inverted and taking into account the frequency of positive and negative returns. It has values between zero and infinity. It can be used to rank the performance of portfolios. The lower the d ratio the better the performance, a value of zero indicating there are no returns less than zero and a value of infinity indicating there are no returns greater than zero.[28]

$$DRatio(R) = \frac{n_d * \sum_{t=1}^{n} max(-R_t, 0)}{n_u * \sum_{t=1}^{n} max(R_t, 0)}$$

where n is the number of observations of the entire series, n_d is the number of observations less than zero, n_u is the number of observations greater than zero.

```
> data(portfolio_bacon)
> print(DRatio(portfolio_bacon[,1]))
```

```
[1] 0.4013329
```

[28] Expected = 0.401

Omega-Sharpe ratio (p.95)

The Omega-Sharpe ratio is a conversion of the omega ratio to a ranking statistic in familiar form to the Sharpe ratio. To calculate the Omega-Sharpe ratio, we subtract the target (or Minimum Acceptable Returns (MAR)) return from the portfolio return and we divide it by the opposite of the Downside Deviation.[29]

$$OmegaSharpeRatio(R, MAR) = \frac{r_p - r_t}{\sum^n_{t=1} \frac{max(r_t - r_i, 0)}{n}}$$

where n is the number of observations of the entire series.

```
> data(portfolio_bacon)
> MAR = 0.005
> print(OmegaSharpeRatio(portfolio_bacon[,1], MAR))

[1] 0.2917933
```

[29] Expected = 0.29

Sortino ratio (p.96)

Sortino proposed an improvement on the Sharpe Ratio to better account for skill and excess performance by using only downside semivariance as the measure of risk.

$$SortinoRatio = \frac{(\overline{R_a - MAR})}{\delta_{MAR}}$$

where δ_{MAR} is the Downside Deviation.

```
> data(managers)
> round(SortinoRatio(managers[, 1]),4)

                          HAM1
Sortino Ratio (MAR = 0%) 0.7649
```

Kappa (p.96)

Introduced by Kaplan and Knowles (2004), Kappa is a generalized downside risk-adjusted performance measure. To calculate it, we take the difference of the mean of the distribution to the target and we divide it by the l-root of the lth lower partial moment. To calculate the lth lower partial moment, we take the subset of returns below the target and we sum the differences of the target to these returns. We then return return this sum divided by the length of the whole distribution.[30]

$$Kappa(R, MAR, l) = \frac{r_p - MAR}{\sqrt[l]{\frac{1}{n} * \sum_{t=1}^{n} max(MAR - R_t, 0)^l}}$$

```
> data(portfolio_bacon)
> MAR = 0.005
> l = 2
> print(Kappa(portfolio_bacon[,1], MAR, l)) #expected 0.157

[1] 0.1566371
```

[30] Expected = 0.157

Upside potential ratio (p.97)

Sortino proposed an improvement on the Sharpe Ratio to better account for skill and excess performance by using only downside semivariance as the measure of risk. That measure is the Sortinon ratio. This function, Upside Potential Ratio, was a further improvement, extending the measurement of only upside on the numerator, and only downside of the denominator of the ratio equation.[31]

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

where δ_{MAR} is the Downside Deviation.

```
> data(edhec)
> UpsidePotentialRatio(edhec[, 6], MAR=.05/12)

                             Event Driven
Upside Potential (MAR = 0.4%)    0.5376613
```

[31] Expected = 5 percent/yr MAR

Volatility skewness (p.97)

Volatility skewness is a similar measure to omega but using the second partial moment. It's the ratio of the upside variance compared to the downside variance.[32]

$$VolatilitySkewness(R, MAR) = \frac{\sigma_U^2}{\sigma_D^2}$$

where σ_U is the Upside risk and σ_D is the Downside Risk.

```
> data(portfolio_bacon)
> MAR = 0.005
> print(VolatilitySkewness(portfolio_bacon[,1], MAR,
+                              stat="volatility"))

[1] 1.323046
```

[32] Expected = 1.32

Variability skewness (p.98)

Variability skewness is the ratio of the upside risk compared to the downside risk.[33]

$$VariabilitySkewness(R, MAR) = \frac{\sigma_U}{\sigma_D}$$

where σ_U is the Upside risk and σ_D is the Downside Risk.

```
> data(portfolio_bacon)
> MAR = 0.005
> print(VolatilitySkewness(portfolio_bacon[,1], MAR,
+                                  stat="variability"))
[1] 1.150238
```

[33] Expected = 1.15

Adjusted Sharpe ratio (p.99)

Adjusted Sharpe ratio was introduced by Pezier and White (2006) to adjusts for skewness and kurtosis by incorporating a penalty factor for negative skewness and excess kurtosis.[34]

$$AdjustedSharpeRatio = SR * [1 + (\frac{S}{6}) * SR - (\frac{K-3}{24}) * SR^2]$$

where SR is the sharpe ratio with data annualized, S is the skewness and K is the kurtosis.

```
> data(portfolio_bacon)
> print(AdjustedSharpeRatio(portfolio_bacon[,1]))

                              portfolio.monthly.return....
Annualized Sharpe Ratio (Rf=0%)                  0.7591435
```

[34] Expected = 0.81

Skewness-kurtosis ratio (p.99)

Skewness-Kurtosis ratio is the division of Skewness by Kurtosis. It is used in conjunction with the Sharpe ratio to rank portfolios. The higher the rate the better.[35]

$$SkewnessKurtosisRatio(R, MAR) = \frac{S}{K}$$

where S is the skewness and K is the Kurtosis.

```
> data(portfolio_bacon)
> print(SkewnessKurtosisRatio(portfolio_bacon[,1]))
[1] -0.03394204
```

[35] Expected = -0.034

Prospect ratio (p.100)

Prospect ratio is a ratio used to penalise loss since most people feel loss greater than gain.[36]

$$ProspectRatio(R) = \frac{\frac{1}{n} * \sum_{i=1}^{n}(Max(r_i,0) + 2.25 * Min(r_i,0) - MAR)}{\sigma_D}$$

where n is the number of observations of the entire series, MAR is the minimum acceptable return and σ_D is the downside risk.

```
> data(portfolio_bacon)
> MAR = 0.05
> print(ProspectRatio(portfolio_bacon[,1], MAR))
[1] -0.1347065
```

[36] Expected = -0.134

M Squared for Sortino (p.102)

M squared for Sortino is a M^2 calculated for Downside risk instead of Total Risk.[37]

$$M_S^2 = r_P + Sortinoratio * (\sigma_{DM} - \sigma_D)$$

where M_S^2 is MSquared for Sortino, r_P is the annualised portfolio return, σ_{DM} is the benchmark annualised downside risk and D is the portfolio annualised downside risk.

```
> data(portfolio_bacon)
> MAR = 0.005
> print(M2Sortino(portfolio_bacon[,1], portfolio_bacon[,2], MAR))

                          portfolio.monthly.return....
Sortino Ratio (MAR = 0.5%)                   0.1034799
```

[37] Expected = -0.1035

Omega excess return (p.103)

Omega excess return is another form of downside risk-adjusted return. It is calculated by multiplying the downside variance of the style benchmark by 3 times the style beta.

$$\omega = r_P - 3 * \beta_S * \sigma_{MD}^2$$

where ω is omega excess return, β_S is style beta, σ_D is the portfolio annualised downside risk and σ_{MD} is the benchmark annualised downside risk.[38]

```
> data(portfolio_bacon)
> MAR = 0.005
> print(OmegaExcessReturn(portfolio_bacon[,1], portfolio_bacon[,2], MAR))
[1] 0.08053795
```

[38] Expected = 0.0805

Variability risk

Table of Mean absolute difference, Monthly standard deviation and annualised standard deviation.

```
> data(managers)
> table.Variability(managers[,1:8])
```

```
                           HAM1   HAM2   HAM3   HAM4   HAM5   HAM6 EDHEC LS EQ
Mean Absolute deviation 0.0182 0.0268 0.0268 0.0410 0.0329 0.0187      0.0159
monthly Std Dev         0.0256 0.0367 0.0365 0.0532 0.0457 0.0238      0.0205
Annualized Std Dev      0.0888 0.1272 0.1265 0.1843 0.1584 0.0825      0.0708
                        SP500 TR
Mean Absolute deviation   0.0333
monthly Std Dev           0.0433
Annualized Std Dev        0.1500
```

Specific risk

Table of specific risk, systematic risk and total risk.

```
> data(managers)
> table.SpecificRisk(managers[,1:8], managers[,8])
```

	HAM1	HAM2	HAM3	HAM4	HAM5	HAM6	EDHEC LS EQ	SP500 TR
Specific Risk	0.0664	NA	0.0946	0.1521	NA	NA	NA	0.00
Systematic Risk	0.0586	0.0515	0.0836	0.1032	0.0477	0.0486	0.0503	0.15
Total Risk	0.0886	NA	0.1262	0.1838	NA	NA	NA	0.15

Information risk

Table of Tracking error, Annualised tracking error and Information ratio.

```
> data(managers)
> table.InformationRatio(managers[,1:8], managers[,8])
```

```
                            HAM1   HAM2   HAM3   HAM4   HAM5   HAM6 EDHEC LS EQ
Tracking Error            0.0327 0.0443 0.0334 0.0461 0.0520 0.0326      0.0326
Annualised Tracking Error 0.1132 0.1534 0.1159 0.1597 0.1800 0.1128      0.1130
Information Ratio         0.3604 0.5060 0.4701 0.1549 0.1212 0.6723      0.2985
                          SP500 TR
Tracking Error                   0
Annualised Tracking Error        0
Information Ratio              NaN
```

Distributions

Table of Monthly standard deviation, Skewness, Sample standard deviation, Kurtosis, Excess kurtosis, Sample Skweness and Sample excess kurtosis.

```
> data(managers)
> table.Distributions(managers[,1:8])
```

```
                           HAM1   HAM2   HAM3    HAM4   HAM5    HAM6 EDHEC LS EQ
monthly  Std Dev         0.0256 0.0367 0.0365  0.0532 0.0457  0.0238      0.0205
Skewness                -0.6588 1.4580 0.7908 -0.4311 0.0738 -0.2800      0.0177
Kurtosis                 5.3616 5.3794 5.6829  3.8632 5.3143  2.6511      3.9105
Excess kurtosis          2.3616 2.3794 2.6829  0.8632 2.3143 -0.3489      0.9105
Sample skewness         -0.6741 1.4937 0.8091 -0.4410 0.0768 -0.2936      0.0182
Sample excess kurtosis   2.5004 2.5270 2.8343  0.9437 2.5541 -0.2778      1.0013
                        SP500 TR
monthly  Std Dev          0.0433
Skewness                 -0.5531
Kurtosis                  3.5598
Excess kurtosis           0.5598
Sample skewness          -0.5659
Sample excess kurtosis    0.6285
```

Drawdowns

Table of Calmar ratio, Sterling ratio, Burke ratio, Pain index, Ulcer index, Pain ratio and Martin ratio.

```
> data(managers)
> table.DrawdownsRatio(managers[,1:8])
```

	HAM1	HAM2	HAM3	HAM4	HAM5	HAM6	EDHEC LS EQ	SP500 TR
Sterling ratio	0.5463	0.5139	0.3884	0.3136	0.0847	0.7678	0.5688	0.1768
Calmar ratio	0.9062	0.7281	0.5226	0.4227	0.1096	1.7425	1.0982	0.2163
Burke ratio	0.6593	0.8970	0.6079	0.1998	0.1008	1.0788	0.8452	0.2191
Pain index	0.0157	0.0642	0.0674	0.0739	0.1452	0.0183	0.0178	0.1226
Ulcer index	0.0362	0.1000	0.1114	0.1125	0.1828	0.0299	0.0325	0.1893
Pain ratio	8.7789	2.7187	2.2438	1.6443	0.2570	7.4837	6.6466	0.7891
Martin ratio	3.7992	1.7473	1.3572	1.0798	0.2042	4.5928	3.6345	0.5112

Downside risk

Table of Monthly downside risk, Annualised downside risk, Downside potential, Omega, Sortino ratio, Upside potential, Upside potential ratio and Omega-Sharpe ratio.

```
> data(managers)
> table.DownsideRiskRatio(managers[,1:8])
```

```
                          HAM1   HAM2   HAM3   HAM4   HAM5   HAM6 EDHEC LS EQ
monthly downside risk   0.0145 0.0116 0.0174 0.0341 0.0304 0.0121      0.0098
Annualised downside risk 0.0504 0.0401 0.0601 0.1180 0.1054 0.0421      0.0341
Downside potential      0.0051 0.0061 0.0079 0.0159 0.0145 0.0054      0.0041
Omega                   3.1907 3.3041 2.5803 1.6920 1.2816 3.0436      3.3186
Sortino ratio           0.7649 1.2220 0.7172 0.3234 0.1343 0.9102      0.9691
Upside potential        0.0162 0.0203 0.0203 0.0269 0.0186 0.0165      0.0137
Upside potential ratio  0.7503 2.2078 1.0852 0.8009 0.7557 1.0003      1.1136
Omega-sharpe ratio      2.1907 2.3041 1.5803 0.6920 0.2816 2.0436      2.3186
                        SP500 TR
monthly downside risk     0.0283
Annualised downside risk  0.0980
Downside potential        0.0132
Omega                     1.6581
Sortino ratio             0.3064
Upside potential          0.0218
Upside potential ratio    0.7153
Omega-sharpe ratio        0.6581
```

Sharpe ratio

Table of Annualized Return, Annualized Std Dev, and Annualized Sharpe.

```
> data(managers)
> table.AnnualizedReturns(managers[,1:8])

                           HAM1   HAM2   HAM3   HAM4   HAM5   HAM6 EDHEC LS EQ
Annualized Return        0.1375 0.1747 0.1512 0.1215 0.0373 0.1373      0.1180
Annualized Std Dev       0.0888 0.1272 0.1265 0.1843 0.1584 0.0825      0.0708
Annualized Sharpe (Rf=0%) 1.5491 1.3732 1.1955 0.6592 0.2356 1.6642      1.6657
                         SP500 TR
Annualized Return          0.0967
Annualized Std Dev         0.1500
Annualized Sharpe (Rf=0%)  0.6449
```