SAS Performance Analytics Library

July 16, 2015

**Type** Package

**Title** Econometric tools for performance and risk analysis

**Version** 1.01

**Description**

A collection of econometric functions for performance and risk analysis. This library aims to aid practitioners and researchers in utilizing research in analysis of return streams. In general, it is most tested on returns (rather than price) data on a regular scale, but most functions will work with irregular return data as well. This package aims to replicate the Performance Analytics package available in R with a few minor tweaks to improve functionality, but otherwise adhering very closely to R Performance Analytics.

**Copyright** © 2015

**Authors** Dominic Pazzula

Carter Johnston

Ruicheng Ma

Qiyuan Yang

**Date/Publication** 2015-07-16

***SAS topics documented:***

[ActivePremium 11](#_Toc456877167)

[Adjusted\_SharpeRatio 12](#_Toc456877168)

[Appraisal\_Ratio 13](#_Toc456877169)

[Average\_Depth 15](#_Toc456877170)

[Average\_Length 16](#_Toc456877171)

[Average\_Recovery 17](#_Toc456877172)

[BernardoLedoit\_Ratio 18](#_Toc456877173)

[BetaCoMoments 20](#_Toc456877174)

[Bull\_Bear\_beta 21](#_Toc456877175)

[Burke\_Ratio 22](#_Toc456877176)

[Calmar\_Ratio 26](#_Toc456877177)

[CAPM\_Alpha\_Beta 27](#_Toc456877178)

[CAPM\_Epsilon 28](#_Toc456877179)

[CAPM\_JensenAlpha 30](#_Toc456877180)

[CDD 31](#_Toc456877181)

[Centered\_Moments 33](#_Toc456877182)

[Chart\_Autoregression 35](#_Toc456877183)

[Chart\_CaptureRatios 36](#_Toc456877184)

[Chart\_Correlation 38](#_Toc456877185)

[Chart\_CumulativeReturns 39](#_Toc456877186)

[Chart\_Drawdown 40](#_Toc456877187)

[Chart\_ECDF 41](#_Toc456877188)

[Chart\_Histogram 42](#_Toc456877189)

[Chart\_Regression 44](#_Toc456877190)

[Chart\_RelativePerformance 45](#_Toc456877191)

[Chart\_Scatter 46](#_Toc456877192)

[CoMoments 48](#_Toc456877193)

[create\_hash 49](#_Toc456877194)

[D\_Ratio 50](#_Toc456877195)

[download\_ff3 51](#_Toc456877196)

[download\_FRED 51](#_Toc456877197)

[download\_yahoo 52](#_Toc456877198)

[downside\_frequency 53](#_Toc456877199)

[downside\_risk 54](#_Toc456877200)

[Drawdowns 56](#_Toc456877201)

[Drawdown\_Deviation 57](#_Toc456877202)

[Fama\_Beta 58](#_Toc456877203)

[Find\_Drawdowns 59](#_Toc456877204)

[Geo\_Mean 61](#_Toc456877205)

[get\_stocks 62](#_Toc456877206)

[Hurst\_Index 63](#_Toc456877207)

[Information\_Ratio 64](#_Toc456877208)

[Kappa 66](#_Toc456877209)

[KellyRatio 67](#_Toc456877210)

[LPM 69](#_Toc456877211)

[Market\_Timing 71](#_Toc456877212)

[Martin\_Ratio 73](#_Toc456877213)

[max\_drawdown 74](#_Toc456877214)

[Mean\_Abs\_Deviation 76](#_Toc456877215)

[MSquared 77](#_Toc456877216)

[MSquared\_Excess 78](#_Toc456877217)

[Netselectivity 80](#_Toc456877218)

[Omega 82](#_Toc456877219)

[Omega\_SharpeRatio 83](#_Toc456877220)

[OmegaExcessReturn 85](#_Toc456877221)

[Pain\_Index 86](#_Toc456877222)

[Pain\_Ratio 88](#_Toc456877223)

[Prices 89](#_Toc456877224)

[Prospect\_Ratio 90](#_Toc456877225)

[Return\_Accumulate 91](#_Toc456877226)

[Return\_Annualized 92](#_Toc456877227)

[Return\_Annualized\_Excess 94](#_Toc456877228)

[Return\_Calculate 95](#_Toc456877229)

[Return\_Centered 96](#_Toc456877230)

[Return\_Cumulative 97](#_Toc456877231)

[Return\_Excess 99](#_Toc456877232)

[Return\_Relative 100](#_Toc456877233)

[Scalar\_Annualized 101](#_Toc456877234)

[Sharpe\_Ratio 102](#_Toc456877235)

[SharpeRatio\_Annualized 103](#_Toc456877236)

[simple\_normalize 105](#_Toc456877237)

[simple\_normalize\_by 106](#_Toc456877238)

[SkewnessKurtosisRatio 106](#_Toc456877239)

[Sort\_Drawdowns 107](#_Toc456877240)

[SortinoRatio 109](#_Toc456877241)

[Specific\_Risk 110](#_Toc456877242)

[Standard\_Deviation 112](#_Toc456877243)

[StdDev\_Annualized 113](#_Toc456877244)

[Sterling\_Ratio 114](#_Toc456877245)

[Systematic\_Risk 116](#_Toc456877246)

[Table\_Annualized\_Returns 117](#_Toc456877247)

[Table\_AutoCorrelation 119](#_Toc456877248)

[Table\_CalendarReturns 120](#_Toc456877249)

[Table\_CAPM 121](#_Toc456877250)

[Table\_CaptureRatios 123](#_Toc456877251)

[Table\_Correlation 124](#_Toc456877252)

[Table\_Distributions 126](#_Toc456877253)

[Table\_Drawdowns 127](#_Toc456877254)

[Table\_DrawdownsRatio 129](#_Toc456877255)

[Table\_HigherMoments 130](#_Toc456877256)

[Table\_InformationRatio 132](#_Toc456877257)

[Table\_SpecificRisk 133](#_Toc456877258)

[Table\_Stats 134](#_Toc456877259)

[Table\_UpDownRatios 136](#_Toc456877260)

[Table\_Variability 137](#_Toc456877261)

[Total\_Risk 139](#_Toc456877262)

[TrackingError 140](#_Toc456877263)

[Treynor\_Ratio 141](#_Toc456877264)

[Ulcer\_Index 143](#_Toc456877265)

[UpDownRatios 144](#_Toc456877266)

[upside\_frequency 146](#_Toc456877267)

[upside\_risk 147](#_Toc456877268)

[UpsidePotentialRatio 148](#_Toc456877269)

***List of Tables:***

[Table 1. Definition comparison of drawdown for DOW in Jan, 2005 23](#_Toc456607235)

[Table 2. Example of Sort\_Drawdowns output data set 94](#_Toc456607236)

[Table 3. Sample output from returns of 01/03/2005-12/31/2014 103](#_Toc456607237)

[Table 4. Sample output from returns of 01/03/2005-12/31/2014 105](#_Toc456607238)

[Table 5. Calendar returns sample output 106](#_Toc456607239)

[Table 6. Table CAPM sample output 108](#_Toc456607240)

[Table 7. Table CaptureRatios sample output 109](#_Toc456607241)

[Table 8. Table Correlation sample output 111](#_Toc456607242)

[Table 9. Table Correlation sample output 112](#_Toc456607243)

[Table 10. Table Drawdowns sample output 113](#_Toc456607244)

[Table 11. Table DrawdownsRatio sample output 115](#_Toc456607245)

[Table 12. Table HigherMoments sample output 117](#_Toc456607246)

[Table 13. Table InformationRatio sample output 118](#_Toc456607247)

[Table 14. Table InformationRatio sample output 119](#_Toc456607248)

[Table 15. Table SpecificRisk sample output 121](#_Toc456607249)

[Table 16. Table Variability sample output 122](#_Toc456607250)

[Table 17. Table Variability sample output 124](#_Toc456607251)

***List of Figures:***

[Figure 1. Cumulative return of GOOGLE from 01/03/2005 to 10/14/2005 22](#_Toc456607335)

[Figure 2. Cumulative return of GOOGLE from 01/03/2005 to 10/14/2005 22](#_Toc456607336)

**SAS Performance Analytics Library**

Econometric tools for performance and risk analysis.

**Description**

Working for the Financial Risk Group, the creators of the SAS Performance Analytics macro library saw an opportunity: to recreate the performance analytics package found in R in SAS to implement performance analysis tools needed for the FRG platform. Performance Analytics provides a SAS package of econometric functions for performance and risk analysis of financial instruments or portfolios. This packages aims to aid practitioners and researchers in using the latest research for analysis of a returns series.

We created this library to include functionality that appears in the R Performance Analytics package, which is taken from academic literature on performance analysis and risk. These tools had no functional equivalent in SAS previously. Generally, this package requires return data rather than price data. However, price data can be quickly converted to returns data with the use of the macro [Return\_Calculate](#name_Return_Calculate). Almost all of the macros in this library will work with annual, quarterly, monthly, or daily frequency. In the following summary, we attempt to provide an overview of the capabilities provided by SAS Performance Analytics. We hope that the accompanying library and documentation can fill the void when it comes to financial risk and performance analytics tools available to an analyst using the SAS system.

With the growing accessibility to alternative assets to the individual investor, demand has become ever higher for research and analysis tools in performance analytics. The simple tools that were appropriate in a relative investment world now seem inappropriate for investment returns in the current context. Risk measurement, which is inseparable from performance assessment, has become multi-dimensional and multi-moment all the while attempting to answer a very simple question: “What is my risk?” Portfolio construction and risk budgeting are then two sides of the same coin: optimizing a portfolio by maximizing return while minimizing volatility. With the increasing availability of complicated alternative investment strategies to investors, and the state of near perfect information, an engaging debate about performance analysis and evaluation is as crucial as ever.

Performance analytics does not guarantee a perfect portfolio immune to all risk. However, what it does offer is an accretion of evidence, organized to assist a decision maker in answering a specific question on a particular asset or portfolio. Using such tools to uncover information and ask better questions will create a better informed investor. Performance measurement starts with returns. However, the normalization inherent in calculating returns can be deceiving. It is important that returns be standardized because this “price per unit of investment” standardization is useful in comparing opportunity costs and because of the standardization’s useful statistical qualities. As a result, the Performance Analytics library focuses on standardized returns rather than prices [See [Return\_calculate](#name_Return_Calculate) for converting net asset values or prices into returns, either discrete or log based]. Many papers and theories refer to “excess returns”, or risk premium: we implement a simple function for aligning time series and calculating these excess returns in [Return\_excess](#name_Return_Excess). Returns and risk may be annualized as a way to simplify comparison over longer or unequal time periods. Although it requires a bit of estimating, such aggregation is popular because it offers a reference point for easy comparison. Examples of this estimation can be found in [Return\_annualized](#name_Return_Annualized), [StdDev\_annualized](#name_StdDevAnnualized), and [SharpeRatio\_annualized](#name_SharpeRatioAnnualized). Basic measures of performance tend to treat returns as independent observations. In this case, the entirety of the SAS base is applicable to such analysis. Some basic statistics are collected in [table.Stats](#name_tableStats).

These types of summary statistics and tables provide the bulk of the information an investor may want to analyze, and provides an organized way to view results of potentially thousands of periodic return data. Usually these statistics are the most “readable” when organized into a table of related statistics assembled for a particular purpose. A common offering of past returns organized by month and cumulated by calendar year is usually presented as a table, such as in [table\_CalendarReturns](#name_tableCalendarReturns). Adding benchmarks or peers alongside the annualized data is helpful for comparing returns in calendar years. Examples of other tables for comparison of related groupings of statistics discussed in this documentation:

[table.Stats](#name_tableStats) provides Basic statistics and stylized facts

[table.AnnualizedReturns](#name_tableAnnualizedReturns) Annualized return, standard deviation, and Sharpe ratio

[table.CalendarReturns](#name_tableCalendarReturns) Monthly and calendar year return table

[table.Correlation](#name_tableCorrelations) Comparison of correlations and significance statistics

[table.Autocorrelation](#name_tableAutoCorrelations) The first six autocorrelation coefficients and significance

[table.HigherMoments](#name_tableHigherMoments) Higher co-moments and beta co-moments

[table\_Distributions](#name_tableDistributions) provides distribution statistics

[table\_InformationRatio](#name_tableInformationRatio) Provides the information ratio as well as the tracking error and annualized tracking error.

[table\_SpecificRisk](#name_tableSpecificRisk) Table of specific risk, systematic risk, and total risk.

[table\_Variability](#name_tableVariability) Table of variability statistics from a returns data set.

Modern Portfolio Theory (MPT), although somewhat ironically outdated, is the collection of tools and techniques by which a risk-averse investor may construct an “optimal” portfolio. It was pioneered by Harry Markowitz in 1952 and encompasses CAPM, the efficient market hypothesis, and all forms of quantitative portfolio construction and optimization. The Capital Asset Pricing Model (CAPM), initially developed by William Sharpe in 1964, provides a justification for passive or index investing by proposing that assets that are not on the efficient frontier will either rise or fall in price until they are. The [CAPM­ alpha](#name_CAPMalphabeta) is the degree to which the asset’s returns are not due to the return that could be captured from the market as a whole. Conversely, the [CAPM beta](#name_CAPMalphabeta) describes the portions of the returns of the asset that could be directly attributed to the returns of a passive investment in the benchmark asset. CAPM is a market equilibrium model or a general equilibrium theory of the relation of prices to risk, but it is usually applied to partial equilibrium portfolios, which can create (sometimes serious) problems in valuation. The performance premium provided by an investment over a passive strategy (the benchmark) is provided by the [active premium](#name_activePremium), which is the investment’s annualized return minus the benchmark’s annualized return. A closely related measure is the [Tracking Error](#name_trackingError), which measures the unexplained portion of the investment’s performance relative to a benchmark. The [Information Ratio](#name_InformationRatio) of an investment in a MPT or CAPM framework is the Active Premium divided by the Tracking Error. The Information Ratio may be used to rank investments in a relative fashion. Research shows that relative rankings across multiple pricing methodologies may be positively correlated with each other and with expected returns. This is quite an important finding because it shows that multiple methods of predicting returns and risk which have underlying measures and factors which are not directly correlated to another measure or factor will still producesimilar quantile rankings. While analyzing an asset or portfolio using the performance analytics tools in this library does not guarantee greater returns, verifying the asset or portfolio over multiple measures for evidence will help prove a positive investment decision for the rational investor.

While we acknowledge that the library is currently incomplete, we hope to continue to append the library as much as we can and in due time find its functionality competitive with that of R Performance Analytics.

**Authors**

Dominic Pazzula

Carter Johnston

Ruicheng Ma

Qiyuan Yang

[*ActivePremium*](#ActivePremium_TOC) Calculate active premium

**Description**

Active premium is the asset’s annualized return minus benchmark’s annualized return.

**Usage**

**%**ActivePremium(returns, BM=)

%ActivePremium(returns, BM=, scale=, method=)

%ActivePremium(returns, BM=, scale=, method=, dateColumn=, outData=)

**Arguments**

returns - Required. Data set containing returns.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=1]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output data set with active premium. [Default= active\_premium]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[CoMoments](#name_CoMoments)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***ActivePremium***(prices, BM= SPY);

/\*Or\*/

/\*%ActivePremium(prices, BM= DOW, scale= 252, method= DISCRETE, dateColumn= Date, outData= active\_premium);\*/

*[Adjusted\_SharpeRatio](#adjustedSharpe_TOC)* Calculate Adjusted Sharpe Ratio

**Description**

Adjusted Sharpe Ratio adds skewness and kurtosis of return distribution on top of Sharpe Ratio. It can be derived from a Taylor series expansion of expected utility with an exponential utility function (see reference).

**Details**

Adjusted Sharpe Ratio can be calculated as:

Where is annualized Sharpe Ratio, is the skewness and is the kurtosis.

**Usage**

**%**Adjusted\_SharpeRatio(returns)

%Adjusted\_SharpeRatio(returns, Rf =, scale=)

%Adjusted\_SharpeRatio(returns, Rf=, scale=, VARDEF=, dateColumn=, outData=)

**Arguments**

returns - Required. Data set containing returns with option to include risk free rate variable.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output data set with Adjusted Sharpe ratios. [Default= adjusted\_SharpeRatio]

**Author**

Dominic Pazzula, Carter Johnston, Qiyuan Yang

**See Also**

[Sharpe\_Ratio](#name_SharpeRAtio) , [SharpeRatio\_annualized](#name_SharpeRatioAnnualized)

**References**

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.99

Pezier and White, *The Relative Merits of Investable Hedge Fund Indices and of Funds of Hedge Funds in Optimal Passive Portfolios*, 2006, p.15

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Adjusted\_SharpeRatio***(prices);

/\*Or\*/

/\*%Adjusted\_SharpeRatio(prices, Rf= 0.01/252, scale= 252, dateColumn= Date, outData= adjusted\_SharpeRatio);\*/

*[Appraisal\_Ratio](#appraisalRatio_TOC)*Calculate Appraisal Ratio

**Description**

Appraisal ratio (or Treynor-Black ratio, first suggested by Treynor and Black (1973)) is the ratio of Jensen’s alpha divided by specific risk. It is similar to Sharpe ratio however using Jensen’s alpha adjusted for systematic risk. The denominator is specific risk rather than total risk.

**Details**

Jensen’s alpha is the intercept of the regression equation in CAPM, which in effect is the excess return adjusted for systematic risk. The appraisal ratio measures the systematic risk-adjusted reward for each unit of specific risk.

Modified Jensen is Jensen’s alpha divided by beta. Note: Michael Jensen (1969) decribed beta as systematic risk. We refer systematic risk to Bacon’s (2008) definition, which is the product of beta by market risk. See [**Systematic\_Risk**](#name_SystematicRisk)**.**

Alternative modified Jensen is Jensen’s alpha divided by systematic risk.

Where is the Jensen’s alpha, is the specific risk, is regression beta, and is the systematic risk.

**Usage**

**%**Appraisal\_Ratio(returns, BM=, option=)

%Appraisal\_Ratio(returns, BM=, Rf=, scale=, option=)

%Appraisal\_Ratio(returns, BM=, Rf=, scale=, option=, method=, VARDEF=, dateColumn=, outData=)

**Arguments**

returns - Required. Data set containing returns with option to include risk free rate variable.

BM - required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

option - Required. {APPRAISAL, MODIFIED, ALTERNATIVE}. Choose “appraisal” to calculate the appraisal ratio, “modified” to calculate the modified Jensen’s alpha, or “alternative” to calculate alternative Jensen’s alpha.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=1]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output data set with Appraisal ratios. [Default= “appraisal\_ratio”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[CAPM\_JensenAlpha](#name_CAPMJensenalpha), [Specific\_Risk](#name_SpecificRisk), [CAPM\_Alpha\_Beta](#name_CAPMalphabeta), [Systematic\_Risk](#name_SystematicRisk)

**References**

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.77

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Appraisal\_Ratio***(prices, BM= SPY, Rf= 0, option= appraisal);

/\*Or\*/

/\*%Appraisal\_Ratio(prices, BM= SPY, Rf= 0.01/252, scale= 252, option= modified, method= DISCRETE, dateColumn= Date, outData= appraisal\_ratio);\*/

*[Average\_Depth](#AverageDepth_TOC)* calculate average depth of drawdown

**Description**

Calculate average depth of observed drawdowns over entire period.

**Details**

Recalculate drawdown depths for multiple assets, unlike AverageDrawdown.r which calls the Find\_Drawdowns() function. Option to choose the compounding method.

**Usage**

%Average\_Depth(returns)

%Average\_Depth(returns, method=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with average drawdowns. [Default="Avg\_DD"]

**Author**

Qiyuan Yang

**See Also**

[Average\_Length](#name_AverageLength), [Average\_Recovery](#AverageRecovery_TOC)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Average\_Depth***(prices);

/\*Or\*/

/\*%Average\_Depth(prices, method=LOG, dateColumn=DATE, outData=MyAveDepth);\*/

*[Average\_Length](#AverageLength_TOC)* calculate average length of drawdown

**Description**

Find the arithmetic mean value of drawdown length. Multiple assets can be calculated at the same time.

**Details**

In order to calculate average lengths from multiple asset drawdowns, Find\_Drawdowns() is not called in this macro. Instead, the drawdown length of each asset is recalculated. In R function AverageLength, the compounding method for return is defaulted as geometric and it is not an input option to change. In this macro, user can choose the compounding method.

**Usage**

%Average\_Length(returns)

%Average\_Length(returns, method=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with average length. [Default="AverageLength"]

**Author**

Ruicheng Ma

**See Also**

[Average\_Depth](#averageDrawdown_TOC),[Average\_Recovery](#name_AverageRecovery)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Average\_Length***(prices);

/\*Or\*/

/\*%Average\_Length(prices, method=LOG, dateColumn=DATE, outData=MyAverageLength);\*/

*[Average\_Recovery](#AverageRecovery_TOC)* calculate average recovery of drawdown

**Description**

Find the arithmetic mean value of drawdown recovery. Multiple assets can be calculated at the same time.

**Usage**

%Average\_Recovery(returns)

%Average\_Recovery(returns, method=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with average recovery. [Default="AverageRecovery"]

**Author**

Ruicheng Ma

**See Also**

[Average\_Depth](#name_AverageDepth), [Average\_Length](#name_AverageLength)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Average\_Recovery***(prices);

/\*Or\*/

/\*%Average\_Recovery(prices, method=LOG, dateColumn=DATE, outData=MyAverageRecovery);\*/

*[BernardoLedoit\_Ratio](#BernardoLedoitRatio_TOC)*  calculate total risk of returns

**Description**

BernardoLedoit ratio is the sum of positive returns divided by the sum of negative returns. It is a special case of the omega ratio.

**Details**

Statistical analysis of historical return distribution provides a measurement of portfolio performance. Most metrics care about the first two moments of the distribution, which are mean and variance. When it comes to non-normal distribution, skewness and kurtosis become more important for the assessment. Adjusted Sharpe Ratio is an example that takes the two into account. Omega ratio, containing all the information about the risk and return of a portfolio, will also address the concern about non-normal distribution. It is a probability weighted ratio of gains to losses, against a threshold. BernardoLedoit Ratio employs 0 as the threshold.

When there is no positive return, BernardoLedoit ratio becomes 0, and when there is no negative return BernardoLedoit ratio becomes infinity.

Where is the number of whole observations, is loss threshold. BernardoLedoit ratio is the omega ratio with .

**Usage**

%BernardoLedoit\_Ratio(returns)

%BernardoLedoit\_Ratio(returns, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with BernardoLedoit ratio. [Default="BLRatio"]

**Author**

Qiyuan Yang

**See Also**

[D\_Ratio](#name_DRatio)

**Reference**

Con Keating, William F. Shadwick, *An Introduction to Omega*, The Finance Development Centre, 2002.

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.95

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***BernardoLedoit\_Ratio***(prices);

/\*Or\*/

/\*%BernardoLedoit\_Ratio(prices, dateColumn=DATE, outData=MyBLRatio);\*/

[*BetaCoMoments*](#betaCo_TOC) calculate higher co-moment betas

**Description**

Calculate higher co-moment betas, or ‘systematic’ variance, skewness, and kurtosis matrices. Beta Covariance is equivalent to Covariance over Variance, Beta Coskewness is equivalent to Coskewness over skewness, and Beta CoKurtosis is equivalent to CoKurtosis over Kurtosis.

**Usage**

**%**BetaCoMoments(Returns)

%BetaCoMoments(Returns, dateColumn=)

%BetaCoMoments(Returns, dateColumn=, outBetaCoVar=, outBetaCoSkew=, outBetaCoKurt=)

**Arguments**

returns - Required. Data set containing returns.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outBetaCoVar - Optional. Output Beta Covariance matrix. [Default= “BetaM2”]

outBetaCoSkew - Optional. Output Beta Coskewness matrix. [Default= “BetaM3”]

outBetaCoKurt - Optional. Output Beta Cokurtosis matrix. [Default= “BetaM4”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[CoMoments](#name_CoMoments)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***BetaCoMoments***(prices);

/\*Or\*/

/\*%BetaCoMoments(prices, dateColumn= Date,outBetaCoVar=BetaM2, outBetaCoSkew= BetaM3, outBetaCoKurt=BetaM4);\*/

*[Bull\_Bear\_beta](#BullBearbeta_TOC)* calculate bull/bear beta from CAPM model

**Description**

The beta from CAPM model is now divided into two classes: bull beta when market return is positive, and bear beta when market return is negative. Bull/bear beta measures portfolio sensitivity to the market under different market performances.

**Usage**

%Bull\_Bear\_beta(returns=, BM=);

%Bull\_Bear\_beta(returns=, BM=, Rf=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set of bull and bear betas. [Default= "bull\_and\_bear"]

**Author**

Qiyuan Yang

**See Also**

[CAPM\_alpha\_beta](#name_CAPMalphabeta)

**References**

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.72

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Bull\_Bear\_beta***(prices, BM=IBM);

/\*Or\*/

/\*%Bull\_Bear\_beta(prices, BM=IBM, Rf=0, dateColumn=DATE, outData=MyBullBear);\*/

*[Burke\_Ratio](#BurkeRatio_TOC)* calculate Burke Ratio

**Description**

Calculate Burke Ratio with the option to output modified Burke Ratio.

**Details**

Burke Ratio is similar to Calmar Ratio and Sterling Ratio where risk is expressed in form of drawdowns. Burke Ratio has the difference of asset return and risk free rate as its numerator and square root of the sum of squared drawdowns as its denominator. It can thus be calculated as:

Where is observations of asset return, is the risk free rate for the peirod, is total number of drawdowns, is total number of observations from the data set, and is the tth drawdown.

If the number of drawdowns is not restricted to the largest drawdowns, Burke ratio and modified Burke ratio will generate same portfolio rankings.

**Note**

in Burke Ratio is different from “Drawdown” as defined in [Drawdowns](#name_Drawdowns) and [Martin\_Ratio](#name_MartinRatio).

Here, the returns are cumulated from the most recent non-negative return. The drawdown is the cumulated return before the next non-negative return. The shaded area in the following charts indicates the way drawdown is accounted. The drawdown is set as zero or missing where the return is non-negative.

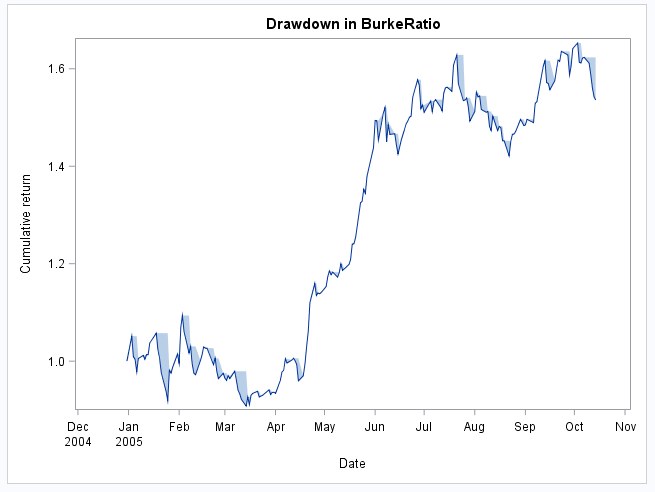


Figure . Cumulative return of GOOGLE from 01/03/2005 to 10/14/2005

Elsewhere in the package, is the drawdown since the historical peak, and is calculated as the cumulative return divided by maximum cumulative return minus 1. This definition of drawdown is shown as the shaded area in the following chart.

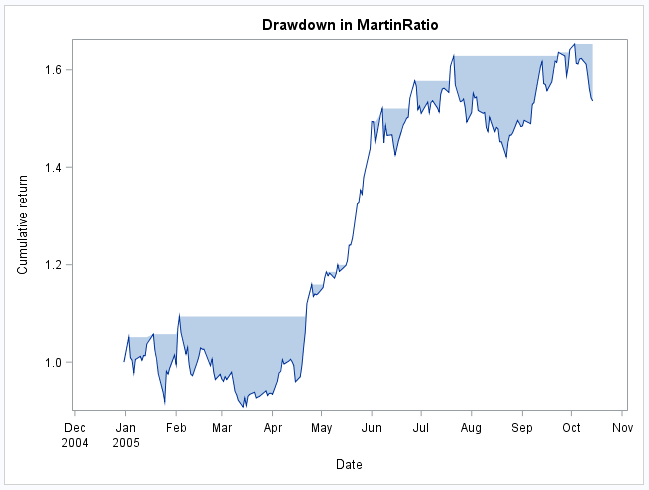


Figure . Cumulative return of GOOGLE from 01/03/2005 to 10/14/2005

Below is an example to illustrate the difference. The returns are calculated and cumulated in discrete method. The drawdown defined in Burke Ratio only exits in the observation right before a new non-negative return, otherwise it is set to missing.

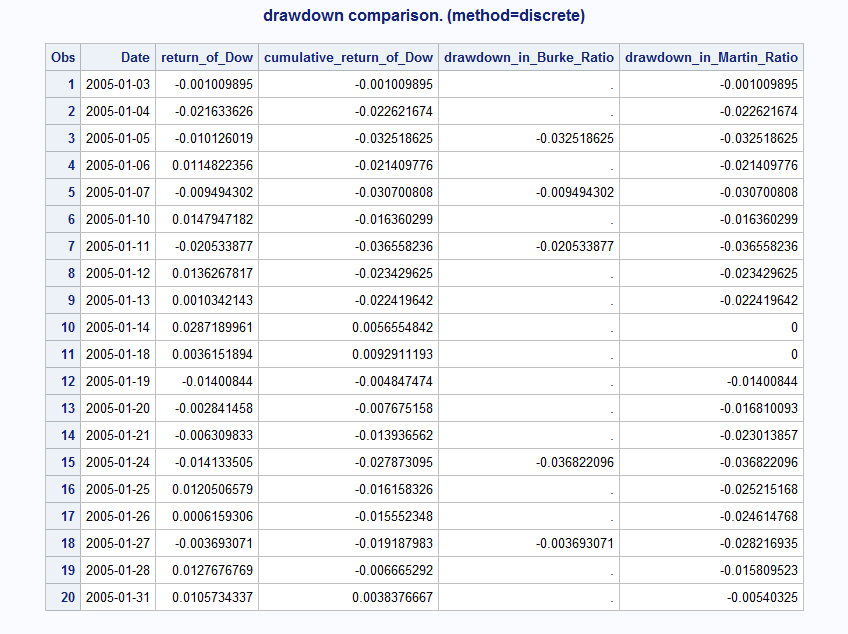
****

Table . Definition comparison of drawdown for DOW in Jan, 2005

**Usage**

%Burke\_Ratio(returns);

%Burke\_Ratio(returns, Rf=, scale=, method=, modified=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing return.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

modified - Optional. Option to calculate modified Burke Ratio. {TRUE, FALSE} [Default=FALSE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with Burke Ratios. [Default="BurkeRatio"]

**Author**

Qiyuan Yang

**See Also**

[Calmar\_Ratio](#name_CalmarRatio),[Sterling\_Ratio](#name_SterlingRatio), [Sharpe\_Ratio](#name_SharpeRAtio), [Martin\_Ratio](#name_MartinRatio)

**References**

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.90

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Burke\_Ratio***(prices);

/\*Or\*/

/\*%Burke\_Ratio(prices, Rf=0, scale=252, method=LOG, modified=TRUE, dateColumn=DATE, outData=MyBurkeRatio);\*/

*[Calmar\_Ratio](#CalmarRatio_TOC)*  calculate Calmar Ratio

**Description**

Calmar Ratio measures an asset’s return against its drawdown risk. Similar to Sharpe Ratio and Sterling Ratio, it is one of the statistics that measures return vs. risk.

**Details**

Calmar Ratio is calculated as annualized return divided by maximum drawdown.

Where is annualized return of asset, is the maximum drawdown.

**Usage**

%Calmar\_Ratio(returns);

%Calmar\_Ratio(returns, scale=, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with Calmar ratios. [Default="CalmarRatio"]

**Author**

Qiyuan Yang

**See Also**

[Drawdowns](#name_Drawdowns), [max\_drawdown](#name_MaxDrawdown), [Sharpe\_Ratio](#name_SharpeRAtio), [Sterling\_Ratio](#name_SterlingRatio)

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Calmar\_Ratio***(prices);

/\*Or\*/

/\*%Calmar\_Ratio(prices, scale=4, method=LOG, dateColumn=DATE, outData=MyCalmarRatio);\*/

*[CAPM\_Alpha\_Beta](#CAPMalphabeta_TOC)* calculate alpha and beta from CAPM model

**Description**

This macro calculates values of Alpha and Beta as defined by CAPM (single factor model), effectively combining the wrapper functions of CAPM.alpha and CAPM.beta in the R performance analytics package.

**Details**

Below is a revised regression equation of CAPM:

“Alpha” measures how much the portfolio outperforms the market or the suitable benchmark, due to the managing skill.

It should be noted that the classical CAPM model has been almost completely discredited by academics. However, it is a good example of a simple single factor model comparing an asset to an arbitrary benchmark.

The CAPM Beta is the beta of an asset to the variance and covariance of an initial portfolio. It is used to determine diversification potential. Beta is often thought of as the slope of the regression line used to determine the risk premium of a returns time series. Alpha is thought of as the intercept of this regression line.

Beta can be calculated as:

Where is asset return, is benchmark return, is the covariance of asset return and benchmark return, and 𝜎b is benchmark standard deviation.

**Usage**

**%**CAPM\_Alpha\_Beta (Returns, BM=, Rf=);

%CAPM\_Alpha\_Beta (Returns, BM=, Rf=, dateColumn=, outData=);

**Arguments**

returns - Required. Data set containing returns with option to include risk free rate variable.

BM - required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output data set with alpha and beta. [Default= “alphas\_and\_betas”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[CAPM\_Epsilon](#name_CAPMepsilon), [Bull\_Bear\_beta](#name_BullBearbeta)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***CAPM\_Alpha\_Beta***(prices, BM= SPY, Rf= **0.01**/**252**);

/\*Or\*/

/\*%CAPM\_Alpha\_Beta(prices, BM= SPY, Rf= IBM, dateColumn= Date, outData= alphas\_and\_betas);\*/

*[CAPM\_Epsilon](#CAPMepsilon_TOC)*find the error term of the regression

**Description**

The epsilon is the error term from the regression of CAPM between portfolio and benchmark. It measures the vertical distance between the two.

**Details**

Below is a revised regression equation of CAPM:

The regression epsilon is given by the following formula:

Where is the regression alpha, is the regression beta, is annualized return of difference between asset return and risk free rate, and is annualized return of difference between benchmark and risk free rate.

**Usage**

%CAPM\_Epsilon(Returns, BM=, Rf=, scale=);

%CAPM\_Epsilon(Returns, BM=, Rf=, scale=, dateColumn=, outData=);

**Arguments**

returns - Required. Data set containing returns with option to include risk free rate variable.

BM - required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output data set with values of epsilon. [Default= “epsilon”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[CAPM\_Alpha\_Beta](#name_CAPMalphabeta)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***CAPM\_Epsilon***(prices, BM= SPY, Rf= **0.01**/**252**);

/\*Or\*/

/\*%CAPM\_Epsilon(prices, BM= SPY, Rf= IBM, scale= 252, dateColumn= Date, outData= epsilon);\*/

*[CAPM\_JensenAlpha](#CAPMJensenalpha_TOC)* calculate Jensen’s alpha

**Description**

The Jensen’s Alpha is the ratio of active premium and the specific return. It is also the intercept of the regression equation of CAPM.

**Details**

Jensen alpha is calculated as:

Where is the regression beta, is annualized return of difference between asset return and risk free rate, and is annualized return of difference between benchmark and risk free rate.

**Usage**

%CAPM\_JensenAlpha(Returns, BM=);

%CAPM\_JensenAlpha(Returns, BM=, Rf=, scale=, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data set containing returns with option to include risk free rate variable.

BM - required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=1]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output data set with values of Jensen alphas. [Default= “Jensen\_Alpha”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[CAPM\_alpha\_beta](#name_CAPMalphabeta)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***CAPM\_JensenAlpha***(prices, BM= SPY);

/\*Or\*/

/\*%CAPM\_JensenAlpha(prices, BM= SPY, Rf= 0.01/252, scale= 252, method= DISCRETE, dateColumn= Date, outData= Jensen\_Alpha);\*/

[*CDD*](#CenteredMoments_TOC) calculate Condition Drawdown-at-Risk

**Description**

This macro calculates Conditional Drawdown-at-Risk (CDaR) which can be interpreted as the mean of the worst p% drawdowns.

**Details**

CDaR proposed by Chekhlov (2000) quantifies the frequency and magnitude of the portfolio drawdowns during a time interval. It is related to Value-at-Risk (VaR) and Conditional Value-at-Risk (CVaR). By definition, CDaR is based on CVaR only modified by treating drawdown as losses. Both CVaR and CDaR can be implemented in portfolio optimization problem which can be solved by linear programming.

Three method of calculating CDaR is applied in this macro in “option”.

For “option=SIMPLE”, CDaR is calculated as:

Where is the tth drawdown, is the p% percentile, n is total number of observations and d is the number of drawdowns worst than DaR.

For “option=MEAN”, CDaR is calculated as:

For “option=WEIGHT”, CDaR is calculated as:

Where is the probability that drawdowns do not exceed or equal to . is the average of the drawdowns strictly exceeding .

**Notes**

The R function CDD calculated p% percentile of the drawdown, but not the mean value as defined by CDaR. This macro will therefore generate different result from R function.

In addition, the drawdowns defined in CDaR is consistent with the definition in [Drawdowns](#name_Drawdowns) (see [Burke\_Ratio](#name_BurkeRatio)).

There exists 5 methods in SAS PROC UNIVARIATE procedure to calculate percentile. Option “pctldef” allows user to choose the calculation accordingly.

**Usage**

%cdd(Returns);

%cdd(Returns, invert=, p=, method=, option=, pctldef=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

invert - Optional. Option to invert CDaR. {TRUE, FALSE}. [Default=TRUE]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

option - Optional. Choose the method to calculate CDaR. {SIMPLE, MEAN, WEIGHT}. [Default=SIMPLE]

pctldef - Optional. Choose the method to calculate p% percentile. (See SAS reference about calculating percentiles). [Default=1]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

p - Optional. Confidence level. [Default=0.95]

outData - Optional. Output Data Set with drawdowns. [Default="CDD"]

**Author**

Ruicheng Ma

**See Also**

[Burke\_Ratio](#name_BurkeRatio), [Drawdowns](#name_Drawdowns)

**References**

S.Uryasev, *Conditional Value-at-Risk (CVaR): Algorithms and Applications*, Working Paper, University of Florida, 2000.

A.Chekhlov, S.Uryasev, M.Zabarankin, *Porfolio Optimization with Drawdown Constraints*, 2003

S.Johri, *Porfolio Optimization with Hedge Funds: Conditional Value At Risk and Conditional Draw-Down at Risk for Portfolio Optimization With Alternative Investments*, Swiss Federal Institute of Technology, 2004

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***cdd***(prices);

/\*Or\*/

/\*%cdd(prices, invert=TRUE, p=0.9, method=DISCRETE, option=MEAN, pctldef=1, dateColumn=DATE, outData=MyCDD);\*/

*[Centered\_Moments](#CenteredMoments_TOC)* calculate centered moments

**Description**

This macro is used internally by SAS Performance Analytics to calculate centered moments for a multivariate distribution as well as the standardized moments of a portfolio distribution. However, it can be called independently for those who wish to calculate centered moments directly.

**Details**

The *n*-th centered moment is calculated as:

Centered\_Moments returns values of the centered variance, centered skewness, and centered kurtosis in separate tables.

**Usage**

%Centered\_Moments(Returns);

%Centered\_Moments(Returns, dateColumn=, outCenteredVar=, outCenteredSkew=, outCenteredKurt=);

**Arguments**

returns - Required. Data set containing returns.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outCenteredVar - Optional. Output data set of centered variance. [Default= “centered\_Var”]

outCenteredSkew - Optional. Output data set of centered skewness. [Default= “centered\_Skew”]

outCenteredKurt - Optional. Output data set of centered kurtosis. [Default= “centered\_Kurt”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Return\_Centered](#name_Return_Centered)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Centered\_Moments***(prices);

/\*Or\*/

/\*%Centered\_Moments (prices, dateColumn= Date, outCenteredVar= centered\_Var, outCenteredSkew= centered\_Skew, outCenteredKurt= centered\_Kurt);\*/

[*Chart\_Autoregression*](#ChartAutoregression_TOC)  A series of auto-regression charts

**Description**

Create a series of auto-regression charts for analysis using a return data set. This macro is in tandem with chart.ACFplus from R library. Besides the default setting of plotting ACF and PACF plots, this macro includes many other plots for users.

**Usage**

%Chart\_Autoregression(returns, asset=, lag=)

%Chart\_Autoregression(returns, asset=, lag=, title= , ALL=, dateColumn=)

**Arguments**

returns - Required. Data Set containing returns.

asset - Required. Specifies the variable or asset to be plotted.

lag - Required. Specifies the amount of lags to plot in each chart.

title - Optional. Title for Charts. Default= AutoRegression Analysis for &asset

ALL - Optional. Option to plot all charts available via Proc Timeseries for analysis. [Default= FALSE]

ACF - Optional. Option to plot an ACF chart for the specified lag. [Default= TRUE]

PACF - Optional. Option to plot a PACF chart for the specified lag. [Default= TRUE]

WN - Optional. Option to plot White Noise charts for the specified lag. [Default= FALSE]

IACF - Optional. Option to plot Inverse ACF charts for the specified lag. [Default= FALSE]

RESIDUAL - Optional. Option to plot Residual charts for the specified lag. [Default= FALSE]

SeasonalAdjusted - Optional. Option to plot a Seasonal adjusted chart for the specified lag. [Default= FALSE]

SeasonalComponent - Optional. Option to plot a Seasonal component chart for the specified lag. [Default= FALSE]

SeasonalCycle - Optional. Option to plot a Seasonal cycle chart for the specified lag. [Default= FALSE]

TrendComponent - Optional. Option to plot a trend component chart for the specified lag. [Default= FALSE]

TrendCycleComponent - Optional. Option to plot a trend cycle component chart for the specified lag. [Default= FALSE]

TrendCycleSeasonal - Optional. Option to plot a Seasonally adjusted trend cycle chart for the specified lag. [Default= FALSE]

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= Date]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_Autoregression***(prices,asset=IBM,lag=5,ALL=TRUE);

/\*Or\*/

/\*%Chart\_Autoregression(prices, asset=IBM, lag=5, title=My Autogression, ACF=FALSE, PACF=FALSE, WN=TRUE, RESIDUAL=TRUE, dateColumn=DATE);\*/

*[Chart\_CaptureRatios](#ChartCaptureRatios_TOC)* create a chart of capture ratios

**Description**

The benchmark in the chart is shown as a straight line with up ratio versus down ratio. Other assets are shown by scatter plot.

**Usage**

%Chart\_CaptureRatios(returns, BM=)

%Chart\_CaptureRatios(returns, BM=, title=, legend\_pos=, grid=, color=, linecolot=, size= )

**Arguments**

returns - Required. Data Set containing returns and benchmark.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

title - Optional. Title for chart. [Default= Drawback Chart for &asset]

AddName - Optional. Option to add name to data points. {TRUE, FALSE} [Default=TRUE]

legend\_pos - Optional. Position of key legend. See SAS KEYLEGEND statement for more information. [Default= BOTTOMLEFT]

grid - Optional. Overlay grid lines on both axiss. [Default= TRUE]

xlabel - Optional. Specifies x label. [Default=DownCapture]

ylabel - Optional. Specifies y label. [Default=UpCapture]

transparency - Optional. Specifies the level of transparency for data points. [Default= 0.2]

color - Optional. Change the color of the scatter plot points. [Default= cornflowerblue]

linecolor - Optional. Change the color of the reference line. [Default= cornflowerblue]

linepattern - Optional. Change the pattern of the reference line. (see Line Attributes and Patterns) [Default= shortdash]

linetransparency - Optional. Specifies the level of transparency for reference line. [Default= 0.2]

size - Optional. Change the size (in pixels) of the plot points. [Default= 6]

symbol - Optional. Change the symbol of the scatter plot points. See list of possible symbols at SAS product documentation (markerattrs symbol). [Default= circle]

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

**Author**

Ruicheng Ma

**See Also**

[Table\_CaptureRatios](#name_TableCaptureRatios), [UpDownRatios](#name_UpDownRatios)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_CaptureRatios***(prices, BM=SPY);

/\*Or\*/

/\*%Chart\_CaptureRatios(prices, BM=SPY, title= Capture Ratio, AddName=TRUE, legend\_pos=BOTTOMRIGHT, grid=TRUE, xlabel=Down, ylabel=Up, transparency=0.1, color=red, dateColumn=DATE);\*/

[*Chart\_Correlation*](#ChartCorrelation_TOC) Correlation matrix of returns

**Description**

Create a table showing the correlation of returns between every two assets.

**Usage**

%Chart\_Correlation(returns)

%Chart\_Correlation(returns, title=, histogram=, histogramDensity=, color=, symbol=, size= )

**Arguments**

returns - Required. Data Set containing returns.

title - Optional. Title for histogram. [Default= Portfolio Asset Correlations]

histogram - Optional. Option to insert histograms for each asset along the diagonal of the plot matrix. [Default= FALSE]

histogramDensity - Optional. Selects a type of density to overlay on histograms along the diagonal. {Normal, Kernel} [Default= Normal]

color - Optional. Change the color of the scatter plot points. [Default= cornflowerblue]

symbol - Optional. Change the symbol of the scatter plot points. [Default= circle]. See list of possible symbols at SAS product documentation (markerattrs symbol)

size - Optional. Change the size (in pixels) of the plot points. [Default= 6]

ellipse - Option to add a predictive ellipse to scatter plots. {True, False}. [Default= FALSE]

ellipseType - Optional. If ellipse is overlayed, specifies type. {mean, predicted} [Default= predicted]

alpha - Optional. If ellipse is overlayed, specifies value of alpha for predictive bands. [Default= 0.05]

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_Correlation***(prices);

/\*Or\*/

/\*%Chart\_Correlation(prices, title= Correlation Chart, histogram= TRUE, histogramDensity=kernel, color=red, symbol= star, size=8, ellipse=TRUE, ellipseType=mean, dateColumn=DATE);\*/

[*Chart\_CumulativeReturns*](#ChartCumulativeReturns_TOC) plot cumulative return

**Description**

Create a chart of cumulative returns for all asset in the given data set. An option of “wealth index” is available.

**Usage**

%Chart\_CumulativeReturns(returns)

%Chart\_CumulativeReturns(returns, title=, method=, WealthIndex=, principal=, grid=, Interval=, dateColumn=)

**Arguments**

returns - Required. Data Set containing returns.

title - Optional. Title for chart. [Default= Cumulative Returns]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

WealthIndex - Optional. Specifies that the value of a dollar in the first sample is $1, therefore charting the value of the returns per dollar over time. {TRUE, FALSE} [Default= FALSE]

principal - Optional. Specify the principal value to cumulate, only works when WealthIndex=TRUE. [Default=1]

grid - Optional. Overlay grid lines on the returns axis. [Default= TRUE]

Interval - Optional. Specifies the frequency of grid lines overlayed on the returns axis. [Default= 1 (100%)]

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

**Author**

Dominic Pazzula, Carter Johnston, Qiyuan Yang

**See Also**

[Return\_Cumulative](#name_Return_Cumulative)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_CumulativeReturns***(prices);

/\*Or\*/

/\*%Chart\_CumulativeReturns(prices, title= My Cumulative Returns, method=LOG, WealthIndex=TRUE, principal=100000, grid=FALSE, dateColumn=DATE);\*/

*[Chart\_Drawdown](#ChartDrawdown_TOC)* chart of drawdowns

**Description**

Create a chart that displays multiple asset drawdowns through time.

**Usage**

%Chart\_Drawdown(returns, asset=)

%Chart\_Drawdown(returns, asset=, method=, title=, grid=, Interval=, linecolor=, legend\_pos=, dateColumn=)

**Arguments**

returns - Required. Data Set containing returns.

asset - Required. Name of the variable to plot drawdown chart for. Asset names are separated by space. {e.g. asset=IBM GE DOW}

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

title - Optional. Title for chart. [Default= Drawback Chart for &asset]

grid - Optional. Overlay grid lines on the returns axis. [Default= TRUE]

Interval - Optional. Specifies the frequency of grid lines overlayed on the returns axis. [Default= -0.1 (-10%)]

linecolor - Optional. Specifies the color of the lines. See SAS COLOR NAMES for reference. {e.g. for three assets, linecolor=GOLD BLACK RED} [Default: automatically assigned by SAS]

legend\_pos - Optional. Position of key legend. See SAS KEYLEGEND statement for more information. [Default= BOTTOMLEFT]

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

**Author**

Ruicheng Ma

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_Drawdown***(prices, asset=IBM GE DOW);

/\*Or\*/

/\*%Chart\_Drawdown(prices, asset=IBM GE DOW, method=LOG, title= my chart, grid=TRUE, Interval=-0.2, linecolor=GOLD OLIVE SALMON, legend\_pos=BOTTOM, dateColumn=DATE);\*/

*[Chart\_ECDF](#ChartECDF_TOC)* CDF of asset return in comparison with a Normal CDF

**Description**

Create a chart for the CDF of asset return in comparison with a fitted normal density CDF.

**Usage**

%Chart\_ECDF(returns)

%Chart\_ECDF(returns, title=, dateColumn=)

**Arguments**

returns - Required. Data Set containing returns.

title- Optional. Title for chart. [Default= Empirical CDF]

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_ECDF***(prices);

/\*Or\*/

/\*%Chart\_ECDF(prices, title= CDF Comparison, dateColumn=DATE);\*/

*[Chart\_Histogram](#ChartHistogram_TOC)*a chart of histograms

**Description**

Given the returns, create the histogram chart with fitting curve for density and normal distribution.

**Usage**

%Chart\_Histogram(Returns, asset, title=);

%Chart\_Histogram(Returns, asset, scale=, title=, bindwidth=, density=, color=, dateColumn=)

**Arguments**

returns - Required. Data Set containing returns.

asset - Required. Specifies the variable name of benchmark asset or index in the returns data set.

scale - Optional. Specifies whether the y-axis should go by probability or frequency. {count, percent, proportion}, [Default= count]

title - Optional. Title for histogram. [Default= asset returns]

bindwidth - Optional. Specifies the range of returns to select for each bar. [Default= 0.001]

density - Optional. Overlay a normal density curve on top of the histogram for comparison. [Default= TRUE]. If true, [TRUE=TURE]. {TRUE, FALSE}.

color - Optional. Change the color of the histogram bins. [Default= cornflowerblue]

densitycolor - Optional. Change the color of the density line. [Default= red]

histogramTransparency - Optional. Change the transparency of the histogram bins. [Default= 0.8]

keepOutliers - Optional. Delete outlier returns from the histogram within the range of Q1- 1.5IQR and Q3+1.5IQR. [Default= TRUE]. {TRUE, FALSE}

qqplot - Optional. Display a QQ Plot in addition to the histogram. [Default= FALSE]. {TRUE,FALSE}

rug - Optional. Display a fringe plot overlayed onto the histogram. [Default= FALSE]. {TRUE,FALSE}

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_Histogram***(prices, asset= IBM);

/\*Or\*/

/\*%Chart\_Histogram(prices, asset= IBM, scale=percent, title= My Histogram, density= TRUE, dateColumn= Date);\*/

*[Chart\_Regression](#ChartRegression_TOC)* Chart of simple regression

**Description**

Create a chart of simple regression between an asset return and a benchmark, given a return data set. It has the option to overlay a Loess line, and the option to apply linear or quadratic regression.

**Usage**

%Chart\_Regression(returns, xvar=, yvar=)

%Chart\_Regression(returns, xvar=, yvar=, title=, ExcessReturns=, loess=, dateColumn=)

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

xvar - Required. Specifies the variable or asset to be plotted on the x-axis.

yvar - Required. Specifies the variable or asset to be plotted on the y-axis (a benchmark asset).

title - Optional. Title for Scatter Plot. [Default= xvar versus yvar Regression Plot]

ExcessReturns - Optional. Option to plot returns in excess of the benchmark or a risk free rate. {TRUE, FALSE} [Default= FALSE]

Rf - Optional. If excessReturns is true, then specifies the risk free rate as a number or as a benchmark asset {Rf= 0.05, Rf= SPY} [Default= 0]

grid - Optional. Overlay a grid aligned with the points on the x and y axis. {TRUE,FALSE} [Default= TRUE]

transparency - Optional. Specifies the level of transparency for data symbols. [Default= 0.35]

color - Optional. Change the color of the scatter plot points. [Default= cornflowerblue]

symbol - Optional. Change the symbol of the scatter plot points. See list of possible symbols at SAS product documentation (markerattrs symbol). [Default= circle]

size - Optional. Change the size (in pixels) of the plot points. [Default= 6]

loess - Optional. To overlay a loess fit to the scatter plot for comparison. Logical, {TRUE, FALSE}. [Default= FALSE].

cl - Optional. Option to create confidence limits for the regression line. {CLM, CLI}. [Default= CLI]

degree - Optional. Specifies linear or quadratic fit. For linear, degree=1, for quadratic, degree=2. [Default= 1]

alpha - Optional. If ellipse is overlayed, specifies value of alpha for predictive bands. [Default= 0.05]

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_Regression***(prices, xvar=IBM, yvar=GE);

/\*Or\*/

/\*%Chart\_Regression(prices, xvar=IBM, yvar=GE, title= My simple Regression, loess=TRUE, degree=2, dateColumn=DATE);\*/

*[Chart\_RelativePerformance](#ChartRelativePerformance_TOC)* Chart of relative performance

**Description**

Compare the relative performance of an asset to a benchmark and create the chart.

**Usage**

%Chart\_RelativePerformance(returns)

%Chart\_RelativePerformance(returns, Rf=, title=, method=, dateColumn=)

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

title - Optional. Title for chart. [Default= Relative Performance Against &Rf]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_RelativePerformance***(prices);

/\*Or\*/

/\*%Chart\_RelativePerformance(prices, Rf=0.0005, title= My Relative Performance, method=LOG, dateColumn=DATE);\*/

*[Chart\_Scatter](#ChartScatter_TOC)* a scatter chart

**Description**

Create a chart of scatter that shows the simple correlation between two asset returns with the option to overlay a regression line.

**Usage**

%Chart\_Scatter(returns, xvar=, yvar=,)

%Chart\_Scatter(returns, xvar=, yvar=, title=, grid=, regLine=, dateColumn=)

**Arguments**

returns - Required. Data Set containing returns.

xvar - Required. Specifies the variable or asset to be plotted on the x-axis.

yvar - Required. Specifies the variable or asset to be plotted on the y-axis.

title - Optional. Title for Scatter Plot. [Default= &xvar versus &yvar Scatter]

grid - Optional. Overlay a grid aligned with the points on the x and y axis. {TRUE, FALSE} [Default= FALSE]

transparency - Optional. Specifies the level of transparency for data symbols. [Default= 0.35]

color - Optional. To change the color of the scatter plot points. [Default= cornflowerblue]

symbol - Optional. To change the symbol of the scatter plot points. See list of possible symbols at SAS product documentation (markerattrs symbol). [Default= circle]

size - Optional. To change the size (in pixels) of the plot points. [Default= 6]

regLine - Optional. Overlay a regression line on the scatter plot. {TRUE, FALSE}. [Default= FALSE]

cl - Optional. If regLine= TRUE, option to create confidence limits for the regression line. {CLM, CLI}. [Default= CLI]

degree - Optional. If regLine= TRUE, specifies linear or quadratic fit. For linear, degree=1, for quadratic, degree=2. [Default= 1]

ellipse - Optional. Add a predictive ellipse to scatter plots. {True, False}. [Default= FALSE]

EllipseType - Optional. If ellipse is overlayed, specifies type. {mean, predicted}. [Default= predicted]

alpha - Optional. If ellipse is overlayed, specifies value of alpha for predictive bands. [Default= 0.05]

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Chart\_Scatter***(prices, xvar=IBM, yvar=GE);

/\*Or\*/

/\*%Chart\_Scatter(prices, xvar=IBM, yvar=GE, title= My Scatter Chart, grid=TRUE, color=red, regLine=True, cl=CLM, ellipse=TRUE, dateColumn=DATE);\*/

*[CoMoments](#CoMoments_TOC)* calculate co-moments from multiple asset returns

**Description**

For multiple assets, calculate coskewness and cokurtosis. This data is input into two separate matrices. CoMoments is an internal macro used in table\_HigherMoments, but can be exposed if the user wishes to see the output directly.

**Details**

The individual elements of the co-skewness matrix can be obtained as:

Similarly, the individual elements of the co-kurtosis matrix can be obtained as:

**Usage**

%CoMoments(returns);

%CoMoments(returns, dateColumn=, outCoSkew=, outCoKurt=);

**Arguments**

returns - Required. Data Set containing returns.

dateColumn - Optional. Specifies the date column for returns in the data set. [Default= DATE]

outCoSkew - Optional. Output co-skewness matrix. [Default= “M3”]

outCoKurt - Optional. Output co-kurtosis matrix. [Default= “M4”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Return\_Centered](#name_Return_Centered), [BetaCoMoments](#name_BetaCo)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***Return\_Calculate***(prices);

%***CoMoments***(prices); /\*Or\*/

/\*%CoMoments (prices, dateColumn= Date, outCoSkew= M3, outCoKurt= M4);\*/

*[create\_hash](#CreateHash_TOC)* Declare a hash object

**Description**

A helper macro to declare a hash object to be used in DATA STEP. See SAS reference on “DECLARE statement, Hash and Hash Iterator Objects”.

**Usage**

%create\_hash(name=, key=, data\_vars=, dataset=)

**Arguments**

name - Required. Specifies the name of the hash object. {ie. name = my\_hash}

key - Required. Lookup keys to initialize hash object. {ie. key = id}

data\_vars - Required. Specifies the data variables which is to be munipulated. {ie. data\_vars = salary}

dataset - Required. Name of the data set. {ie. dataset = "my\_data\_set"}

**Author**

Dominic Pazzula, Carter Johnston

**Example**

**data** group;

input sex $ groups $;

datalines;

M A

F B

;

**run**;

**data** all(drop=rc);

length groups $1;

set sashelp.class;

if \_n\_ = 1 then do;

%***create\_hash***(name=my\_hash,key=sex,data\_vars=groups,dataset=”group”);

end;

if my\_hash.find() = 0;

**run**;

*[D\_Ratio](#DRatio_TOC)* calculate total risk of returns

**Description**

D ratio is the sum of positive returns divided by the sum of negative returns, with the frequencies being considered.

**Details**

When there is no positive return, D ratio becomes infinity, and when there is no negative return D ratio becomes 0.

Where is the number of whole observations, is the number of negative returns, is the number of positive returns.

It is very similar to BernarLedoit Ratio, only inversed and replaces with and .

**Usage**

%D\_Ratio(returns)

%D\_Ratio(returns, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with d ratio. [Default="DRatio"]

**Author**

Qiyuan Yang

**See Also**

[BernardoLedoit\_Ratio](#name_BernardoLedoit)

**References**

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.95

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***D\_Ratio***(prices);

/\*Or\*/

/\*%D\_Ratio(prices, dateColumn=DATE, outData=MyDRatio);\*/

*[download\_ff3](#Downloadff3_TOC)* download ff3 factors

**Description**

A helper macro that downloads up-to-date daily Fama French 3 factors from online source. The data starts from Jul 1st, 1926.

**Usage**

%download\_ff3(outData=);

**Arguments**

outData - Required. Output data set with ff3 factors. [Default= “ff3”]

**Author**

Dominic Pazzula, Carter Johnston

**Example**

%download\_ff3(outData=my\_ff3);

*[download\_FRED](#downloadFRED_TOC)* download FRED time series data

**Description**

A helper macro that downloads time series data from FRED.

**Usage**

%download\_FRED(symbol, from=, to=);

**Arguments**

symbol - Required. Sticker of one index. {ie.symbol=DGS10}

from - Required. Starting date (inclusive). {ie. 31DEC2004} [Default = 1 year before today's date]

to - Required. Ending data (inclusive). {ie. 01JAN2015} [Default = 1 day before today's date]

**Author**

Qiyuan Yang

**Example**

%download\_FRED(symbol=DGS10);

*[download\_yahoo](#downloadyahoo_TOC)* download stock data from yahoo

**Description**

A helper macro that downloads stock price data from yahoo and calculate returns.

**Usage**

%download\_yahoo(symbol, from=, to=);

%download\_yahoo(symbol, from=, to=, keepPrice=, LogReturn=,PriceColumn=);

**Arguments**

symbol - Required. Sticker of one stock. {ie.symbol=IBM}

from - Optional. Starting date (inclusive). {ie. 31DEC2004} [Default = 1 year before today's date]

to - Optional. Ending data (inclusive). {ie. 01JAN2015} [Default = 1 day before today's date]

keepPrice - Optional. Specify whether to keep the price data. {0,1} [Default = 0]

LogReturn - Optional. Compound or single returns. {0,1} [Default = 1]

PriceColumn - Optional. Specify the kind of price to be kept. [Default = adj\_close]

**Author**

Dominic Pazzula

**Example**

%download\_yahoo(symbol=IBM, from=31DEC2004,to=01JAN2015,keepPrice=1, LogReturn=0, priceColumn= adj\_close);

*[downside\_frequency](#DownsideFrequency_TOC)* calculate downside frequency

**Description**

Calculate the frequency of returns smaller than the Minimum Acceptable Return (MAR). The MAR could be the risk free rate, the benchmark or other rate required by manager.

**Details**

Downside frequency and downside deviation ([downside\_risk](#name_DownsideRisk)) help to measure the frequency and magnitude of failure.

Where is the number of returns smaller than MAR and is the number of whole observations.

**Usage**

%downside\_frequency(returns)

%downside\_frequency(returns, MAR=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. [Default=0]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with downside frequency. [Default="DownsideFrequency"]

**Author**

Qiyuan Yang

**See Also**

[downside\_risk](#name_DownsideRisk), [upside\_risk](#name_UpsideRisk), [upside\_frequency](#name_UpsideFrequency)

**References**

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.94

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***downside\_frequency***(prices);

/\*Or\*/

/\*%downside\_frequency(prices, MAR=0, dateColumn=DATE, outData=MyDownsideFrequency);\*/

*[downside\_risk](#DownsideRisk_TOC)* calculate total downside statistics of risk

**Description**

Calculate downside risk, variance and potential. All three are metrics of variability of performance under a target rate.

**Details**

We take the subset of returns that are smaller than the Minimum Acceptable Return (proposed by Sharpe) to calculate the statistics. The downside variance is the square of downside risk and downside potential is the average sum of returns below MAR.

Where can be the number of whole observations or the number of observations with return smaller than MAR. The user has the option to choose wether the subgroup is taken as the denominator.

**Usage**

%downside\_risk(returns, option=)

%downside\_risk(returns, MAR=, option=, group=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. [Default=0]

option- Required. {RISK, VARIANCE, POTENTIAL}. Choose "RISK" to calculate the downside risk, "VARIANCE" to calculate downside variance, or "POTENTIAL" to calculate downside potential.

group - Optional. Specifies to choose full observations or subset observations as 'n' in the divisor. {FULL, SUBSET} [Default=FULL]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. output Data Set with downside risks. [Default="DownsideRisk"]

**Author**

Qiyuan Yang

**See Also**

[downside\_frequency](#name_DownsideFrequency), [upside\_risk](#name_UpsideRisk), [upside\_frequency](#name_UpsideFrequency)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***downside\_risk***(prices, option=RISK);

/\*Or\*/

/\*%downside\_risk(prices, MAR=0, option=RISK, group=FULL, dateColumn=DATE, outData=MyDownsideRisk);\*/

*[Drawdowns](#Drawdowns_TOC)* calculate drawdowns

**Description**

Calculate the decline of asset return from historical peaks for periodical time series. Expressed as percentages of losses from peak returns.

**Details**

A drawdown is the decline of an investment since its most recent peak price. If the return of the investment is positive, the drawdown is zero. The calculation of drawdown is expressed as:

See [Burke\_Ratio](#name_BurkeRatio) for more information about drawdown.

**Usage**

%Drawdowns(returns);

%Drawdowns(returns, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with drawdowns. [Default="drawdownPeak"]

**Author**

Qiyuan Yang

**References**

[Burke\_Ratio](#name_BurkeRatio)

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Drawdowns***(prices);

/\*Or\*/

/\*%Drawdowns(prices, method=LOG, dateColumn=DATE, outData=MyDrawdownPeak);\*/

*[Drawdown\_Deviation](#DrawdownDev_TOC)* calculate standard deviation of drawdowns

**Description**

Given a return data set, this macro calculates drawdowns first and then output the standard deviation of drawdowns.

**Details**

See [Burke\_Ratio](#name_BurkeRatio) for definition of drawdown.

The standard deviation of drawdown is calculated as:

Where is the tth drawdown depth, is the total number of drawdowns, and is the total number of observations.

**Usage**

%Drawdown\_Deviation(returns)

%Drawdown\_Deviation(returns, method=, dateColumn=, outData)

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with drawdown deviation. [Default="DD\_dev "]

**Author**

Qiyuan Yang

**References**

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.88

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Drawdown\_Deviation***(prices);

/\*Or\*/

/\*%Drawdown\_Deviation(prices, method=LOG, dateColumn=DATE, outData=MyDDdev);\*/

*[Fama\_Beta](#famabeta_TOC)*calculateFama beta of the returns

**Description**

Fama beta is the effective beta required so that the systematic risk is equivalent to the total portfolio risk. It is the measure of return required to justify the loss of diversification.

**Details**

Portfolio manegers seek to give up diversification for additional return by taking specific risk. If there is no specific risk, the total risk equals to the systematic risk. Specific risk is the error term in the CAPM regression equation. Systematic risk, by the definition in this package, is beta multiplied by market risk. Total risk is systematic risk plus specific risk.

Fama beta is calculated as:

Where is the asset standard deviation and is the market risk.

**Usage**

%Fama\_beta(Returns, BM=);

%Fama\_beta(Returns, BM=, dateColumn=, outData=);

**Arguments**

returns - Required. Data set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData- Optional. Output data set with values of Fama beta. [Default= “fama\_beta”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Specific\_Risk](#name_SpecificRisk), [Systematic\_Risk](#name_SystematicRisk), [Total\_Risk](#name_TotalRisk)

**References**

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.78

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***Return\_Calculate***(prices);

%***Fama\_beta***(prices, BM= SPY);

/\*Or\*/

/\*%Fama\_beta(prices,BM= SPY, dateColumn= Date, outData= fama\_beta);\*/

*[Find\_Drawdowns](#FindDrawdowns_TOC)* find starting and ending points of drawdown, as well as length of the interval

**Description**

This macro finds the points in a data set which indicate the starting point, ending point, length, and depth of drawdowns.

**Details**

In the output, variable ‘trough’ indicates the point where the max drawdown of the particular interval occurs. Variables ‘peaktotrough’ and ‘recovery’ indicate the number of points before and after ‘trough’. The output data set is similar to that of Sort\_Drawdowns, but not sorted. See [Sort\_Drawdowns](#name_SortDrawdowns) for visual illustration.

**Usage**

%Find\_Drawdowns(returns, asset=);

%Find\_Drawdowns(returns, asset=, method=, dateColumn=, SortDrawdown=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

asset - Required. Name of the variable to find drawdown interval for.

method - Optional. Specifies either DISCRETE or LOG chaining method. {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

SortDrawdown - Optional. Specifies whether to sort the data on drawdown depth. {TRUE, FALSE} [Default=FALSE]

outData - Optional. Output Data Set with drawdowns. [Default="FindDrawdowns"]

**Author**

Ruicheng Ma

**See Also**

[Drawdowns](#name_Drawdowns), [max\_drawdown](#MaxDrawdown_TOC), [Sharpe\_Ratio](#name_SharpeRAtio), [Calmar\_Ratio](#name_CalmarRatio), [Sort\_Drawdowns](#name_SortDrawdowns)

**References**

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.88

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Find\_Drawdowns***(prices, asset=IBM);

/\*Or\*/

/\*%Find\_Drawdowns(prices, asset=IBM, method=LOG, dateColumn=DATE, SortDrawdown=FALSE, outData=MyFindDrawdowns);\*/

*[Geo\_Mean](#GeoMean_TOC)*calculate the geometric mean of the observation series

**Description**

The geometric mean is a measure of central tendency, using multiplication instead of the traditional addition to summarize data values. Geometric means are useful summaries for highly skewed data. Do not use a geometric mean if there are negative or zero values in the data set.

**Details**

Geo\_Mean is intended as a wrapper function to be used inside other macros that require it. However, the user can call Geo\_Mean if they so choose. The geometric mean is given by one of two formulas:

Where is the return and is the number of total obervations.

**Usage**

%Geo\_Mean (returns, BM=, Rf=, scale=);

%Geo\_Mean(returns, BM=, Rf=, scale=, dateColumn=, outData=);

**Arguments**

Returns - Required. Data Set containing returns.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output data set with of systematic risk. [Default= “\_geoMean”]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Geo\_Mean***(prices);

/\*Or\*/

/\*%Geo\_Mean(prices, dateColumn= Date, outData= \_geoMean);\*/

*[get\_stocks](#GetStocks_TOC)* download daily stock price and return data

**Description**

A helper macro that downloads daily stock price and return data from yahoo.finance, with the option to keep the price.

**Usage**

%get\_stocks(stocks=, from=, to=);

%get\_stocks(stocks=, from=, to=, keepPrice=, LogReturn=, PriceColumn=, outReturns=)

**Arguments**

stocks - Required. Tickers of the stocks. {ie.stocks=IBM GE}

from - Optional. Starting date (inclusive). {ie. 31DEC2004} [Default = 1 year before today's date]

to - Optional. Ending data (inclusive). {ie. 01JAN2015} [Default = 1 day before today's date]

keepPrice - Optional. Specify whether to keep the price data. {0, 1} [Default = 0]

LogReturn - Optional. Compound or single returns. {0, 1} [Default = 1]

PriceColumn - Optional. Specify the kind of price to be kept. [Default = adj\_close]

outReturns - Optional. Output data set with returns. [Default =”returns”]

**Author**

Dominic Pazzula, Carter Johnston

**Example**

%let stocks = IBM GE DOW GOOGL SPY;

%***get\_stocks***(&stocks);

/\*Or\*/

/\*%get\_stocks(&stocks, from=31DEC2010, to=01JAN2016, keepPrice=1, outReturns=StockReturn);\*/

*[Hurst\_Index](#HurstIndex_TOC)* calculate the Hurst Index

**Description**

The study of Hurst Index is also referred to as rescaled range (R/S) analysis, because Hurst (1940s) used rescaled range to compare different types of time series. It is used to measure whether the returns are mean-reverting, totally random, or persistent.

**Details**

The Effective Market Hypothesis (EMH) assumes that the stock price is the reflection of all available information, and future price depends on only new information. Stock prices follow random walk and is unrelated to previous activities. Peters (1991) applied Hurst Index proposed by Hurst into capital market. However affected by time frequency and industry section, Peters found that H is usually greater than 0.5 for individual stocks, and even higher for the market as a whole. The result shows that stock market behaved as “biased random walk”. In other words, the observations are not independent.

1>H>0.5 means the return series are persistent, H=0.5 totally random, 0<H<0.5 mean reversed. When H is large the cumulative return curve will look smooth, which indicates its predictivity. When H is small, the curve will be more jagged.

Hurst Index is another meansure of risk different from deviation or variance.

The Hurst Index can be calculated as:

Where is the maximum value of asset return, is the minimum value of asset return, and is standard deviation of asset return.s

**Usage**

%Hurst\_Index(returns)

%Hurst\_Index(returns, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. output Data Set with hurst index. [Default="HurstIndex"]

**Author**

Qiyuan Yang

**References**

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.104.

Edgar Peters, *Chaos and Order in the Capital Markets*, 1991, chapter 7&8.

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Hurst\_Index***(prices);

/\*Or\*/

/\*%Hurst\_Index(prices, dateColumn=DATE, outData=MyHurstIndex);\*/

*[Information\_Ratio](#InformationRatio_TOC)* calculate Information Ratio

**Description**

Information Ratio is a measure of risk-adjusted return of the portfolio.

**Details**

The Information Ratio is similar to the Sharpe Ratio. However, the Sharpe Ratio is the “excess” return of an asset over the risk free rate divided by standard deviation of the return, and Information Ratio is the “excess” return over a benchmark divided by tracking error. Normally, annualized excess return and annualized tracking errors are used for the calculation.

The information Ratio is given by the following formula:

Where is asset return, is benchmark return, is the annualized tracking error.

**Usage**

%Information\_Ratio(returns, BM=, scale=);

%Information\_Ratio(returns,BM=, scale=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with information ratio. [Default="Info\_Ratio"]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[TrackingError](#name_trackingError)

**References**

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.80.

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Information\_Ratio***(prices, BM= SPY, scale= **252**);

/\*Or\*/

/\*%Information\_Ratio(prices, BM= SPY, scale= 252, dateColumn= Date, outData= Info\_Rat);\*/

*[Kappa](#Kappa_TOC)* calculate Kappa of returns

**Description**

Kappa is a downside risk-adjusted performance measure proposed by Kaplan and Knowles (2004). It is a generalized measure that Sortino Ratio and Omega are special case of Kappa, and both two does not assumes return to be normally distributed.

**Details**

For n=1 Kappa is the Sharpe-omega ratio and for n=2 Kappa is the Sortino Ratio.

While K2 depends in part on semi-variance, and K1 depends on semi-mean, there is no corresponding distribution moment for Kappa variants with non-integer parameter values, nor is the relationship between K3 and a notional “semi-skewness” statistic, or K4 and a “semi-kurtosis” statistic.

The nth lower partial moment ([LPM](#name_LPM)) is defined as:

Where Rt is the tth return,MAR is Minimum Acceptable Return as a threshold, n is the degree of the moment. m is the number of whole observations.

Kappa is calculated as:

Where Ra is the asset return, n is the coefficient of Kappa. MAR can be a variable in the input data set.

**Usage**

%Kappa(returns)

%Kappa(returns, MAR=, L=, group=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. [Default=0]

L - Optional. The exponential coefficient of Kappa. [Default=1]

group - Optional. Specifies to choose full observations or subset observations in the divisor. {FULL, SUBSET} [Default=FULL]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with Kappa. [Default="Kappa"]

**Author**

Ruicheng Ma

**See Also**

[SortinoRatio](#name_SortinoRatio), [LPM](#name_LPM)

**References**

P. D. Kaplan and J. A. Knowles, *Kappa: A Generalized Downside Risk-Adjusted Performance Measure*, 2004.

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Kappa***(prices);

/\*Or\*/

/\*%Kappa(prices, MAR=0, L=2, group=FULL, dateColumn=DATE, outData=MyKappa);\*/

*[KellyRatio](#KellyRatio_TOC)* calculate Kelly Ratio

**Description**

The Kelly Criterion was identified by John Kelly. It can be expressed as the expected excess return of a strategy divided by the expected variance of the excess return.

**Details**

Kelly Ratio is calculated as the average excess return divided by the variance of return.

Where is the observations of asset return, is the risk free rate for period, is the variance of excess return. Option to calculate half Kelly ratio.

**Usage**

%KellyRatio(returns)

%KellyRatio(returns, Rf=, option=, VARDEF=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

option - Optional. Option to use half-Kelly. [Default=HALF]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with Kelly ratio. [Default="KellyRatio"]

**Author**

Ruicheng Ma

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***KellyRatio***(prices);

/\*Or\*/

/\*%KellyRatio(prices, Rf=0, option=HALF, dateColumn=DATE, outData=MyKellyRatio);\*/

*[LPM](#LPM_TOC)* calculate lower partial moments.

**Description**

Similar to standard deviation, lower partial moments (LPM) is another series of risk measures with pre-specified threshold and a risk-preference indicator ‘n’. ‘n’ larger than 1 means investors are risk averse, less than 1 risk seeking, equal to 1 risk neutral.

**Details**

Moments describe the shape of a set of points. A general definition of moment can be expressed as:

Where is a random variable with cumulative distribution function and reference level MAR.

The first moment around zero is the mean and the second moment around mean is the variance. The special case when equals to the mean is the central moment.

LPM, first defined by Fishburn (1977), simply examines the moment of degree n below a certain threshold . The LPM can be calculated as following:

Where is the degree of the moment, is either the number of all observations or observations less than MAR. Option to specify whether to calculate LPM about the mean under the threshold, the mean of all observations, or threshold.

**Usage**

%LPM(returns)

%LPM(returns, n=, group=, MAR=, about\_mean=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

n - Optional. The n-th moment to return. It can be used as an indicator of risk preference. Risk averse behavior is signified by n>1, whereas risk seeking behavior is indicated by n<1. [Default=2]

group - Optional. Specifies to choose full observations or subset observations in the divisor. {FULL, SUBSET} [Default=FULL]

MAR - Optional. Minimum Acceptable Return. A reference point to be compared. The reference. point may be the mean or some specified threshold.Default=0

about\_mean - Optional. Specify whether to calculate LPM about the mean under the threshold, the mean of all observations, or threshold. {UNDER, ALL, NULL}. [Default=NULL]

dateColumn - Optional. Date column in Data Set. Default=DATE

outData - Optional. Output Data Set with lower partial moments. [Default="lpm"]

**Author**

Qiyuan Yang

**See Also**

[Kappa](#name_Kappa)

**Reference**

Huffman S.P. & Moll C.R., *The impact of Asymmetry on Expected Stock Returns: An Investigation of Alternative Risk Measures*, Algorithmic Finance 1, 2011 p. 79-93

Alexander Wojt, *Portfolio Selection and Lower Partial Moments*, Ph.D. Thesis, Royal Institute of Technology, Sweden, 2009

P. D. Kaplan and J. A. Knowles, *Kappa: A Generalized Downside Risk-Adjusted Performance Measure*, 2004.

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***LPM***(prices);

/\*Or\*/

/\*%LPM(prices, n=1, group=SUBSET, threshold=0, about\_mean=UNDER, dateColumn=DATE, outData=MyLPM);\*/

[*Market\_Timing*](#LPM_TOC) calculate market timing ability

**Description**

This macro measures market timing ability of manager. Option to choose Treynor-Mazuy or Merton-Henriksson market timing model.

**Details**

Portfolio managers have two types of ability: selectivity (stock-picking) ability and timing ability. The “selectivity” ability is the ability of investing in stocks with positive “alpha”, for example, the Jensen alpha. The “timing” ability is the one of adjusting portfolio beta in respond to forecast of benchmark excess return, aka beta is an increasing function of benchmark excess return.

The Treynor-Mazuy model (1966) is know as “magnitude” timer. Manager forecasts the magnitude of excess return of the benchmark which implies that the manager has a linear beta response function:

The market timing is a quadratic pact added in the basic CAPM:

Where is excess return of asset, is excess return of benchmark, is defined in basic CAPM, represents the market timing ability.

A statistically significant positive value would imply positive market timing skill, which means if a manager can forecast market returns, he will hold greater proportion of the market portfolio when the return of the market is high and a smaller proportion when the return is low.

The Merton-Henriksson model (1988) is known as “direction” timer. Manager forecasts the direction of excess return of benchmark:

Where is 1 if forecast for is positive (benchmark beats risk-free), and 0 otherwise.

Then Merton-Henriksson regression model is:

The Merton-Henriksson model measures the excess return obtained by the manager that cannot be replicated by a mix of options and market portfolio, which represents market timing ability.

**Usage**

%Market\_Timing (returns, BM=, option=)

%Market\_Timing (returns, BM=, Rf=, option=, dateColum=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. Default=0

option- Required. Specify the model between Treynor-Mazuy and Henriksson-Merton models. {TM, HM}. Default="TM".

dateColumn - Optional. Date column in Data Set. Default=DATE

outData - outData - Optional. Output Data Set of asset market timing. Default= "market\_timing"

**Author**

Qiyuan Yang

**See Also**

CAPM\_alpha\_beta

**Reference**

Roy D. Henriksson, *Market Timing and Mutual Fund Perforamnce: An Empirical Investigation*, The Journal of Business, Vol. 57, No. 1, Part 1. (Jan., 1984), pp. 73-96

Roy D. Henriksson and Robert C. Merton, "On Market Timing and Investment Performance. II. Statistical Procedures for Evaluating Forecast Skills," Journal of Business, vol.54, October 1981, pp.513-533

J. L. Treynor and K. Mazuy, "Can Mutual Funds Outguess the Market?" Harvard Business Review, vol44, 1966, pp. 131-136

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Market\_Timing***(prices, BM=SPY, option=HM);

/\*Or\*/

/\*% Market\_Timing(prices, BM=SPY, Rf=0.01/252, option=TM, dateColum=DATE, outData=MyMktTiming);\*/

*[Martin\_Ratio](#MartinRatio_TOC)* calculate Martin Ratio

**Description**

Martin Ratio is also known as Ulcer Performance Index (UPI). Standard deviation is sometimes rendered as a poor risk measure, while Ulcer Index (UI) serves as an alternative. Martin Ratio is the Sharpe Ratio with SD replaced by UI.

**Details**

Martin Ratio is also similar to modified Burke Ratio but the definition of drawdown is a bit different. See [Burke\_Ratio](#name_BurkeRatio) for more information. The return in the numerator needs to be annualized first.

The Ulcer Index and Martin Ratio can be calculated as:

Where is drawdowns calculated by Drawdowns macro, is number of observations of , is observations of asset return, and is the risk free rate for period.

See [Burke\_Ratio](#name_BurkeRatio) for more information about drawdowns.

**Usage**

%Martin\_Ratio(returns);

%Martin\_Ratio(returns, Rf=, scale=, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with martin ratio. [Default="MartinRatio"]

**Author**

Qiyuan Yang

**See Also**

[Burke\_Ratio](#name_BurkeRatio), [Ulcer\_Index](#name_UlcerIndex)

**References**

P.G.Martin, *An Alternative Approach to the Measurement of Investment Risk & Risk-Adjusted Peformance*, Peter Martin’s Ulcer Index page, 1987.

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.91.

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Martin\_Ratio***(prices);

/\*Or\*/

/\*%Martin\_Ratio(prices, Rf=0, method=LOG, dateColumn=DATE, outData=MyMartinRatio);\*/

*[max\_drawdown](#MaxDrawdown_TOC)* calculate drawdowns

**Description**

Find the worst drawdown for the entire period for every asset. Drawdown is measured as a percentage loss of cumulative return to the most recent peak point. The ‘invert’ option allows user to output drawdown as either positive or negative numbers to serve different purposes. The default option invert=TRUE will provide the drawdown as a positive number.

**Usage**

%max\_drawdowns(returns);

%max\_drawdowns(returns, method=, invert=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

invert - Optional. Specifies whether to invert the drawdown measure. [Default=TRUE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with maximum drawdowns. [Default="max\_dd"]

**Author**

Qiyuan Yang

**See Also**

[Drawdowns](#name_Drawdowns)

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***max\_drawdown***(prices);

/\*Or\*/

/\*%max\_drawdown(prices, method=LOG, invert=FALSE, dateColumn=DATE, outData=MyMaxDrawdowns);\*/

*[Mean\_Abs\_Deviation](#MeanAbsDeviation_TOC)* calculate mean absolute deviation

**Description**

Calculate mean absolute deviation. It is defined as the sum of absolute value of difference between the returns and average return divided by total number.

**Details**

Where is return observation of asset, is the mean return of the asset, and is number of observations.

Mean absolute deviation, variance and standard deviation are three related measures used to calculate variability (or dispersion) of returns from the average or mean return.

**Usage**

%Mean\_Abs\_Deviation(returns=);

%Mean\_Abs\_Deviation(returns=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with mean absolute deviation. [Default="mean\_abs\_dev"]

**Author**

Qiyuan Yang

**See Also**

[Standard\_Deviation](#name_StandardDeviation)

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Mean\_Abs\_Deviation***(prices);

/\*Or\*/

/\*%Mean\_Abs\_Deviation(prices, dateColumn=DATE, outData=MyMeanAbsDev);\*/

*[MSquared](#MSquared_TOC)* calculate M squared

**Description**

M squared measures the adjusted asset return based on the same benchmark risk.

**Details**

Where = risk free for the period, = annualized return of return in excess of risk free rate, = annualized return of benchmark, = annualized standard deviation of asset, = annualized standard deviation of benchmark, and = annualized Sharpe ratio of asset.

M Squared is extremely useful for comparing portfolios with different levels of risk. Original M Squared calculates the return of asset when it burdens the same risk level with benchmark. Net M Squared measures the outperformance over the benchmark in terms of return based on risk level of benchmark.

The statistic is called M2 because it was first proposerd by the partnership of Leah Modigliani(1997) and her grandfather Professor Franco Modigliani.

**Usage**

%MSquared (returns, BM=);

%MSquared(returns, BM=, Rf=, scale=, method=, NET=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

NET - Optional. Specify whether report the value add over the benchmark. {FALSE, TRUE}. Default= FALSE.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set of MSquared. [Default= "MSquared"]

**Author**

Dominic Pazzula, Carter Johnston, Qiyuan Yang

**Notes**

The authors would like to express that this macro is distinct from the function given by R performance analytics due to disagreements with its implementation. The definition of M squared is the Sharpe Ratio multiplied by the standard deviation of the benchmark plus the risk free rate. When annualizing M squared, the geometric average ought to be calculated as the geometric average of , not the geometric average of . For this reason, results between the macro and its equivalent in R will be different.

**See Also**

[MSquared\_Excess](#name_Msquared_Excess)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***MSquared***(prices, BM= SPY, scale= **252**);

/\*Or\*/

/\*%MSquared(prices, BM= SPY, Rf= 0.01/252, scale= 252, method= DISCRETE, dateColumn= Date, outData= MSquared);\*/

*[MSquared\_Excess](#MSquared_Excess_TOC)* calculate M squared excess return

**Description**

M squared excess return measures the quantity above benchmark. As normal excess returns, there are geometric excess and arithmetic excess.

**Details**

Where = annualized return of benchmark.

**Usage**

%MSquared\_Excess(returns, BM=, option);

%MSquared\_Excess(returns, BM=, Rf=, scale=, method=, option=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

option - Required. Specify whether report the geometric or arithmetic excess M Squared. {GEOMETRIC, ARITHMETIC}.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set of MSquared excess return. [Default= "MSquaredExcess"]

**Author**

Qiyuan Yang

**See Also**

[MSquared](#name_MSquared)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***MSquared\_Excess***(prices, BM= SPY, scale= **252**, option=geometric);

/\*Or\*/

/\*%MSquared\_Excess(prices, BM= SPY, Rf= 0.01/252, scale= 252, method= DISCRETE, option=arithmetic, dateColumn= Date, outData= MyMSquaredExcess);\*/

*[Netselectivity](#PainIndex_TOC)* calculate net selectivity

**Description**

Net selectivity measures the excess return of the portfolio over the benchmark after adjusted for diversification.

**Details**

The required return to justify not being fully diversified can be calculated as:

Where is the Fama Beta and is the CAPM beta. is the annualized benchmark return, is the risk free rate.

Net selectivity indicates the remaining return after deducting d. It is calculated as:

Where is Jensen’s alpha.

**Note**

The R function only use discrete method to annualize the benchmark return. In this macro, it can either be discrete or log method. If “Rf” is an input number, it should be an annualized risk free rate; If “Rf” is a column, then the difference of annualized benchmark return and annualized risk free rate is taken for the calculation.

**Usage**

%Netselectivity(returns, BM=);

%Netselectivity(returns, BM=, Rf=, scale=, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. Default=DATE

outData - Optional. Output Data Set with Kappa. [Default="NetSelectivity"]

**Author**

Ruicheng Ma

**See Also**

[CAPM\_JensenAlpha](#name_CAPMJensenalpha), [Fama\_Beta](#name_famabeta), [CAPM\_alpha\_beta](#name_CAPMalphabeta)

**References**

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.78.

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***netselectivity***(prices);

/\*Or\*/

/\*%netselectivity(prices, BM=SPY, Rf=0.05, method=DISCRETE, dateColumn=DATE, outData=MyNetSelectivity);\*/

*Omega* calculate Omega ratio for return series

**Description**

As a way to capture all the higher moments of the returns distribution, it involves partitioning returns into loss and gain above and below a return threshold and then considering the probability weighted ratio of returns above and below the partitioning.

BernardoLedoit ratio is a special case of the Omega ratio.

**Details**

Omega could be considered as a Sharper ratio, or the successor to Jensen's alpha. Omega takes the value 1 when r is the mean return. The slope of the Omega indicates risk: the steeper it is, the less the possibility of extreme returns, aka less risk.

Statistical analysis of historical return distribution provides a measurement of portfolio performance. Most metrics care about the first two moments of the distribution, which are mean and variance. When it comes to non-normal distribution, skewness and kurtosis become more important for the assessment. Adjusted Sharpe Ratio is an example that takes the two into account. Omega ratio, containing all the information about the risk and return of a portfolio, will also address the concern about non-normal distribution. It is a probability weighted ratio of gains to losses, against a threshold. BernardoLedoit Ratio employs 0 as the threshold.

Where is the number of whole observations, is loss threshold.

**Usage**

%Omega(returns)

%Omega(returns, MAR=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. A reference point to be compared. The reference point may be the mean or some specified threshold.Default=0

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with BernardoLedoit ratio. [Default="BLRatio"]

**Author**

Qiyuan Yang

**See Also**

[BernardoLedoit\_Ratio](#name_BernardoLedoit)

**Reference**

Con Keating, William F. Shadwick, *An Introduction to Omega*, The Finance Development Centre, 2002.

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.94

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Omega***(prices);

/\*Or\*/

/\*%Omega(prices, MAR= 0.01/252, dateColumn=DATE, outData=MyOmega);\*/

*Omega\_SharpeRatio* calculate Omega Sharpe ratio for return series

**Description**

Omega Sharpe ratio is converted from Omega ratio to a ranking statistic in similar form to the Sharpe ratio. It can be simplified as Omega minus 1.

**Details**

Omega Sharpe Ratio can be calculated as:

Where is the number of whole observations, is loss threshold, and is Omega ratio.

The average asset return less the target return is equal to upside potential less downside potential.

**Usage**

%Omega\_SharpeRatio(returns)

%Omega\_SharpeRatio(returns, MAR=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. A reference point to be compared. The reference point may be the mean or some specified threshold.Default=0

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with BernardoLedoit ratio. [Default="BLRatio"]

**Author**

Qiyuan Yang

**See Also**

[Omega](#name_Omega)

**Reference**

Con Keating, William F. Shadwick, *An Introduction to Omega*, The Finance Development Centre, 2002.

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.95

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Omega\_SharpeRatio***(prices);

/\*Or\*/

/\*%Omega\_SharpeRatio(prices, MAR= 0.01/252, dateColumn=DATE, outData=MyOmegaSharpe);\*/

*OmegaExcessReturn* calculate omega excess return for return series

**Description**

Omega excess return (Sortino *et al.*, 1997) (not related to the omega ratio) is another form of downside risk-adjusted return.

**Details**

The downside risk-adjusted benchmark return is calculated by multiplying the downside variance of the style benchmark by 3 times the style beta.

The 3 is arbitrary and assumes the investor requires 3 units of return for 1 unit of variance. The style beta is ratio of downside risk of portfolio divided by the downside risk of the style benchmark.

Where is the annualized return of asset, is annualized standard deviation of asset, and is annualized standard deviation of benchmark.

**Usage**

%OmegaExcessReturn(returns, BM=)

%OmegaExcessReturn(returns, BM= , MAR= , scale= , method=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

MAR - Optional. Minimum Acceptable Return. A reference point to be compared. The reference point may be the mean or some specified threshold.Default=0

scale - Required. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. Default=DISCRETE

dateColumn - Optional. Date column in Data Set. Default=DATE

outData - Optional. Output Data Set with omega excess returns. Default="omegaexcess".

**Author**

Qiyuan Yang

**See Also**

[MSquared\_Excess](#name_Msquared_Excess)

**Reference**

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.103

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***OmegaExcessReturn***(prices);

/\*Or\*/

/\*% OmegaExcessReturn (prices, BM=SPY, scale=252, MAR= 0.01/252, dateColumn=DATE, outData=MyOmegaExcess);\*/

*[Pain\_Index](#PainIndex_TOC)* calculate Pain Index

**Description**

Pain Index is also a risk measure, similar to standard deviation, tracking error, only in terms of losses. Proposed by Thomas Becker, it is the mean value of drawdowns. It is notable that to calculate the mean value, the sum of absolute value of drawdowns is divided by the total number of observations, rather than the number of drawdowns.

**Details**

The Pain Index reflects the depth, the duration, and the frequency of losses. Mathematically, it can be expressed as:

Therefore, it is visually the area between drawdown curve and x axis (see [chart\_drawdown](#name_ChartDrawdown)). In discrete form, Pain Index can be written as:

Where is drawdowns calculated by Drawdowns macro, is number of observations of .

See [Burke\_Ratio](#name_BurkeRatio) for more information about drawdowns.

**Usage**

%Pain\_Index(returns);

%Pain\_Index(returns, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with pain index. [Default="PainIndex"]

**Author**

Qiyuan Yang

**References**

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.89.

Marc Odo, *Pain Index and Pain Ratio*, Zephyr Associates. Inc, 2011

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Pain\_Index***(prices);

/\*Or\*/

/\*%Pain\_Index(prices, method=LOG, dateColumn=DATE, outData=MyPainIndex);\*/

*[Pain\_Ratio](#PainRatio_TOC)* calculate Pain Ratio

**Description**

Pain Ratio is calculated by taking the difference between asset return and risk free rate and divided by Pain Index.

**Details**

Pain Ratio is the pain index as the risk metric incorporated with return metric. While Sharpe Ratio can be interpreted as the amount of return per unit of volatility, Pain Ratio is the amount of return per unit of loss.

The Pain Index and Pain Ratio can be calculated as:

Where is drawdowns calculated by Drawdowns macro, is number of observations of , is the asset return and is the risk free rate for period.

**Usage**

Pain\_Ratio %Pain\_Ratio(returns);

%Pain\_Ratio(returns, Rf=, scale=, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

Rf - Optional. the value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with pain ratio. [Default="PainRatio"]

**Author**

Qiyuan Yang

**See Also**

[Sharpe\_Ratio](#name_SharpeRAtio), [Information\_Ratio](#name_InformationRatio), [Sortino\_Ratio](#name_SortinoRatio), [Burke\_Ratio](#name_BurkeRatio), [Martin\_Ratio](#name_MartinRatio)

**References**

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.91.

Marc Odo, *Pain Index and Pain Ratio*, Zephyr Associates. Inc, 2011

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Pain\_Ratio***(prices);

/\*Or\*/

/\*%Pain\_Ratio(prices, Rf=0, scale=252, method=LOG, dateColumn=DATE, outData=MyPainRatio);\*/

*[Prices](#prices_TOC)* Selected price series example data

**Description**

An example price timeseries data set produced by %get\_Stocks which was used to test the majority of the macros in this library, as well as the data set used in examples in this documentation.

**Usage**

%Return\_Calculate(prices);

**Format**

SAS data set

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

*[Prospect\_Ratio](#ProspectRatio_TOC)* calculate the Prospect Ratio.

**Description**

This is a Sharpe type ratio that penalizes losses more than rewarding gains, by assign negative gains more weight. According to Watanabe (2006), people have a tendency to feel loss more than gain, a phenomenon described by Prospect Theory.

**Details**

The Prospect Ratio can be calculated as:

Where is Miminum Acceptable Raturn, is the number of observations and is the downside risk.

**Usage**

%Prospect\_Ratio(returns)

%Prospect\_Ratio(returns, MAR=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. [Default=0]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with prospect ratio. [Default="ProspectRatio"]

**Author**

Qiyuan Yang

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Prospect\_Ratio***(prices);

/\*Or\*/

/\*%Prospect\_Ratio(prices, MAR=0, dateColumn=DATE, outData=MyProspectRatio);\*/

*[Return\_Accumulate](#Return_Accumulate_TOC)* calculate aggregated periodic return

**Description**

Calculate aggregated returns from a lower level to a higher level. For example, input daily returns and output aggregated monthly, quarterly, or yearly return. Option to update the table in place or create a new output.

**Details**

Calculate cumulative return based on specified time frequency.

**Usage**

%Return\_Accumulate (returns);

%Return\_ Accumulate (returns, method=, toFreq=, dateColumn=, updateInPlace=, outData=);

**Arguments**

returns - Required. Data Set containing returns

method - Optional. {LOG, DISCRETE} -- compound or simple returns. Default=DISCRETE

toFreq - Optional. {DAY|DAILY, MONTH|MTH|MONTHLY, QUARTER|QTR|QUARTERLY, YEAR|YR|YEARLY} Default = MONTH

dateColumn - Optional. Date column in Data Set. Default=DATE

updateInPlace - Optional. {TRUE, FALSE} -- update the &returns Data Set in place. Default=TRUE

outData - Optional. Output Data Set with returns. Only used if updateInPlace=FALSE Default="agg\_returns"

**Author**

Dominic Pazzula

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***return\_accumulate***(prices, method= DISCRETE, toFreq=MONTH)

/\*Or\*/

/\*% return\_accumulate(prices, method= LOG, toFreq= YEAR, dateColumn= Date, updateInPlace= FALSE, outData= MyAggReturns);\*/

*[Return\_Annualized](#Return_Annualized_TOC)* calculate annualized returns

**Description**

An annualized return is convenient for comparing with different length history. For discrete returns, an annualized return is the geometric mean based on the number of periods in a year; for log returns, an annualized return is the arithmetic mean multiply by the number of periods in one year.

**Details**

For simple (or discrete) returns:

For compound (or log) returns:

Where = return observation of asset, = the number of periods in one year , and = the number of return observations.

**Usage**

%Return\_Annualized (returns, scale=);

%Return\_Annualized(returns, scale=, method=, dateColumn, outData=);

**Arguments**

returns - Required. Data Set containing returns.

scale - Required. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with annualized returns. [Default="annualized\_returns"]

**Author**

Dominic Pazzula, Carter Johnston, Qiyuan Yang

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Return\_Annualized***(prices, scale= **252**, method= DISCRETE)

/\*Or\*/

/\*%Return\_Annualized(prices, scale= 252, method= LOG, dateColumn= Date, outData= annualized\_returns);\*/

[*Return\_Annualized*](#Return_Annualized_TOC)*\_Excess* calculate annualized excess returns

**Description**

Calculate the difference in performance between asset and benchmark, which is convenient to make a comparison.

**Details**

There are two common measures of the excess return, "arithmetic" and "geometric".

Where is annualized return of asset, and is annualized return of benchmark.

**Usage**

%return\_annualized\_excess (returns, BM=, option=);

%return\_annualized\_excess (returns, BM=, scale=, method=, option=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

scale - Required. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

option - Required. Specify whether report the geometric or arithmetic annualized excess return. {GEOMETRIC, ARITHMETIC}.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set of annualized excess return. Default= "Annualized\_Excess".

**Author**

Qiyuan Yang

**Reference**

Carl Bacon, *Practical Portfolio Performance Measurement and Attribution*, second edition 2008, p.51-52

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***return\_annualized\_excess*** (prices, BM=SPY, option=Geometric)

/\*Or\*/

/\*% return\_annualized\_excess (prices, BM=SPY, scale= 252, method= DISCRETE, option= Arithmetic, dateColumn= Date, outData= MyExcessReturn);\*/

*[Return\_Calculate](#Return_Calculate_TOC)* calculate asset returns

**Description**

Calculate simple (or discrete) and compound (or log) returns from a series of prices. Option to update the table in place or create a new output.

**Details**

For simple (or discrete) returns:

For compound (or log) returns:

Where = price at time t, = price at time t-1.

The price data needs to be regular to use the macro. Prices can be for any time scale as long as the time frequency is consistent in the data. If the time frequency is inconsistent, this macro will not be able to detect the difference and adjust accordingly.

The calculation does not take into account corporation actions, such as stock-split, dividens, time or money weighting. This macro assumes fully adjusted close prices as input. The default compounding method is discrete as most other macros.

**Usage**

%Return\_Calculate (prices);

%Return\_Calculate(prices, method=, dateColumn=, updateInPlace=, outData=);

**Arguments**

prices - Required. Data Set containing prices.

method - Optional. Compound or simple returns. {LOG, DISCRETE} [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

updateInPlace - Optional. Update the &prices Data Set in place. {TRUE, FALSE} [Default=TRUE]

outData - Optional. Output Data Set with returns. Only used if updateInPlace=FALSE. [Default="returns"]

**Author**

Dominic Pazzula

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

/\*Or\*/

/\*%Return\_Calculate(prices, method= DISCRETE, dateColumn= Date, updateInPlace= TRUE, outData= returns);\*/

*[Return\_Centered](#Return_Centered_TOC)* calculate centered returns

**Description**

Calculate the values of centered returns from a series of returns.

**Details**

The centered return is calculated as return minus the expected return, or mean, of an instrument. It is used in the calculations of higher moments.

**Usage**

%Return\_Centered (returns);

%Return\_Centered(returns, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output data set with centered returns. [Default= “centered\_returns”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[CoMoments](#name_CoMoments), [table\_HigherMoments](#name_tableHigherMoments)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Return\_Centered***(prices);

/\*Or\*/

/\*%Return\_Centered(prices, dateColumn= Date, outData= centered\_returns);\*/

*[Return\_Cumulative](#Return_Cumulative_TOC)* calculate a compounded cumulative return

**Description**

Calculate the cumulative return over full time period, applying to simple (or discrete) and compound (or log) return.

**Details**

For simple (or discrete) returns:

For compound (or log) returns:

Where = return observation of asset.

**Usage**

%Return\_Cumulative (returns);

%Return\_Cumulative(returns, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with cumulative returns. [Default="cumulative\_returns"]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Return\_Annualized](#name_Return_Annualized)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Return\_Cumulative***(prices);

/\*Or\*/

/\*%Return\_Cumulative(prices, method= DISCRETE, dateColumn= Date, outData= cumulative\_returns);\*/

*[Return\_Excess](#Return_Excess_TOC)* Calculate the excess return against a given rate

**Description**

Take the differences between the returns and a number or a column indicating the risk free rate or a benchmark.

**Details**

If a single number of risk free rate is used, all return observations will be substracted by that number. If the rate is a column in the same data set as returns, row-by-row substraction will be conducted. The Rf parameter is named after the risk free rate, and any timeseries is allowed. This macro simply conducts the calculation, and user could choose either risk free rate or benchmark asset as Rf for specific purposes.

**Usage**

%Return\_Excess (returns, Rf=);

%Return\_Excess(returns, Rf=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with risk premium. [Default="risk\_premium"]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Return\_Calculate](#name_Return_Calculate)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Return\_Excess***(prices, Rf= 0.01/252);

/\*Or\*/

/\*%Return\_Excess(prices, Rf= SPY, dateColumn= Date, outData= risk\_premium);\*/

[*Return\_Relative*](#Return_Excess_TOC) Calculate relative cumulative performance

**Description**

Calculate relative cumulative performance for assets over benchmark through time.

**Details**

For simple (or discrete) returns:

For compound (or log) returns:

Where is return observations of asset, and is return observations of benchmark.

**Usage**

%return\_relative(returns, BM=);

%return\_relative(returns, BM=, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. Default=DISCRETE

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with relative cumulative performance. Default=["relative\_cum"]

**Author**

Qiyuan Yang

**See Also**

Return\_Cumulative

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***return\_relative***(prices, BM=SPY);

/\*Or\*/

/\*%return\_relative(prices, BM=SPY, method=DISCRETE, dateColumn= Date, outData=relativePerformance);\*/

*[Scalar\_Annualized](#ScalarAnnualized_TOC)* annualize a scalar

**Description**

This helper function annualize a value with specified method by user.

**Usage**

%Scalar\_Annualized(value=, scale=, method=, type=);

**Details**

For type=VALUE, method =DISCRETE:

For type=VALUE, method=LOG:

For type=STD:

**Arguments**

value - Required. Value to be annualized

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

type - Optional. Specifies if the value is a {VALUE, STD}. VALUE are annualized using METHOD. STD are annualized by sqrt(SCALE)

**Author**

Dominic Pazzula

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

%let x=%***scalar\_annualized***(0.0005,scale=252);

%put &x;

*[Sharpe\_Ratio](#Sharpe_Ratio_TOC)*  calculate a traditional Sharpe Ratio of return over standard deviation.

**Description**

The Sharpe ratio is simply the return (or reward) per unit of variability (or risk). The original Sharpe ratio uses standard deviation of returns as measurement of risk.

**Details**

Where = return observation of asset, = risk free rate for the period, and = standard deviation of returns.

The Sharpe Ratio, in its original form, is the expected excess return over the standard deviation of returns. The higher the Sharpe ratio, the better the combined performance of “risk” and “reward”. It is worth mentioning that William Sharpe, the originator of the Sharpe ratio, now recommends the Information ratio preferentially to the original Sharpe Ratio.

It should be noted that the equivalent function in R performance analytics allows the user to calculate modified Sharpe Ratios with Value at Risk (VaR) or Expected Shortfall (ES) instead of Standard Deviation. We are working on adding these options to %Sharpe\_Ratio and will hopefully have this updated soon.

**Usage**

%Sharpe\_Ratio (returns, Rf=);

%Sharpe\_Ratio(returns, Rf=, VARDEF=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with Sharpe ratios. [Default="SharpeRatio"]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Information\_Ratio](#name_InformationRatio), [SharpeRatio\_Annualized](#name_SharpeRatioAnnualized), [Adjusted\_SharpeRatio](#name_Adjusted_SharpeRatio)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Sharpe\_Ratio***(prices, Rf= 0.01/252);

/\*Or\*/

/\*%Sharpe\_Ratio(prices, Rf= SPY, dateColumn= Date, outData= SharpeRatio);\*/

*[SharpeRatio\_Annualized](#SharpeRatioAnnualized_TOC)* calculate annualized Sharpe ratio

**Description**

The annualized Sharpe ratio is calculated using both annualized returns and annualized standard deviation to measure risk-adjusted returns.

**Details**

The Sharpe ratio is simply the return (or reward) per unit of variability (or risk). The original Sharpe ratio uses standard deviation of returns as measurement of risk. This macro annualizes this ratio based on the scale parameter which specifies the number of periods in one year.

For simple (or discrete) returns:

For compound (or log) returns:

Where is the observations of asset return, is risk free rate for the period, is standard deviation of returns, is the number of periods in one year, and is number of observations.

Using an annualized Sharpe ratio is useful for comparison of multiple return streams where length of holdings are not necessarily equal.

**Usage**

%SharpeRatio\_Annualized (returns, Rf=, scale=);

%SharpeRatio\_Annualized(returns, Rf=, scale=, method=, VARDEF=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=0]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with annualized Sharpe Ratio. [Default="Annualized\_SharpeRatio"]

**Author**

Dominic Pazzula, Carter Johnston, Qiyuan Yang

**See Also**

[Sharpe\_Ratio](#name_SharpeRAtio), [Adjusted\_SharpeRatio](#name_Adjusted_SharpeRatio)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***SharpeRatio\_Annualized***(prices, Rf= 0.01/252, scale= 252);

/\*Or\*/

/\*%SharpeRatio\_Annualized(prices, Rf= SPY, scale= 252, method= DISCRETE, dateColumn= Date, outData= SharpeRatio);\*/

*[simple\_normalize](#SimpleNormalize_TOC)* assign weight to each value

**Description**

Calculate the weight of each observation under a variable with the option of choosing a total weight. The original values from input dataset will be replaced by the weight.

**Usage**

%simple\_normalize(data=, var=, sum=);

**Arguments**

data - Required. Data set containing required variable.

var - Required. Name of the variable to be calculated.

sum - Optional. The total weight of the variable. [Default = 1]

**Author**

Dominic Pazzula, Carter Johnston

**Example**

**data** airlines;

set sashelp.airline;

**run**;

%***simple\_normalize***(airlines, var=air, sum=1);

*[simple\_normalize\_by](#SimpleNormalizeBy_TOC)* assign weight to each value by group

**Description**

Calculate the weight of each observation under a variable by groups.

**Usage**

%simple\_normalize\_by(data=, var=, by=);

**Arguments**

data - Required. Data set that contains the data of interest. {ie. data = my\_data\_set}

var - Required. Name of the variable to be manipulated.

by - Required. Specifies the variable that divides data into subgroups.

**Author**

Dominic Pazzula, Carter Johnston

**Example**

**data** electric;

set sashelp.electric;

keep revenue year;

format revenue best.;

**run**;

%***simple\_normalize\_by***(electric,var=revenue,by=year);

*SkewnessKurtosisRatio* calculate skewness-kurtosis ratio of return series

**Description**

Calculate skewness-kurtosis ratio of return seires.

**Details**

Where is skewness of returns, and is kurtosis of returns, which is different from excess kurtosis.

**Usage**

%SkewnessKurtosisRatio (returns)

%SkewnessKurtosisRatio (returns, VARDEF=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} Default= DF.

dateColumn - Optional. Date column in Data Set. Default=DATE

outData - Optional. Output Data Set with skewness-kurtosis ratio. Default="SKratio".

**Author**

Qiyuan Yang

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***SkewnessKurtosisRatio*** (prices);

/\*Or\*/

/\*% SkewnessKurtosisRatio (prices, VARDEF=N, dateColumn=DATE, outData=MySKratio);\*/

*[Sort\_Drawdowns](#SortDrawdowns_TOC)* Sort data based on drawdown return

**Description**

Given the drawdowns data from Find\_Drawdowns macro, sort return variable from lowest to highest.

**Details**

The output data set contains statistics of “return” as drawdown depth, “begin” as starting period, “trough” as period when drawdown reaches the lowest value, “end” as ending period, “length” as duration, “peaktotrough” as length before trough, “recovery” as length after trough. Below is an example for IBM with the worst ten draws.

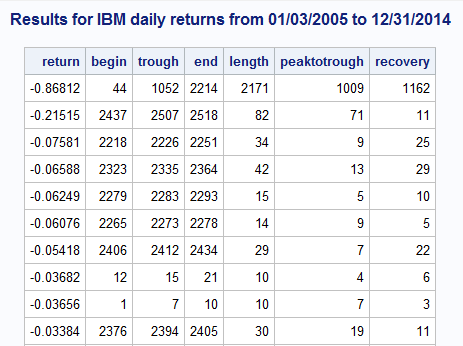


Table . Example of Sort\_Drawdowns output data set

**Usage**

%Sort\_Drawdowns(returns, asset=);

%Sort\_Drawdowns(returns, asset=, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

asset - Required. Name of the variable to find drawdown interval for.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with drawdowns. [Default="SortDrawdown"]

**Author**

Ruicheng Ma

**See Also**

[Drawdowns](#name_Drawdowns), [max\_drawdown](#name_MaxDrawdown), [Find\_Drawdowns](#name_FindDrawdowns), [Table\_Drawdowns](#name_TableDrawdowns)

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Sort\_Drawdowns***(prices, asset=IBM);

/\*Or\*/

/\*%Sort\_Drawdowns(prices, asset=IBM, method=LOG, dateColumn=DATE, outData=MySortDrawdowns);\*/

*[SortinoRatio](#SortinoRatio_TOC)* calculate Sortino Ratio

**Description**

Sortino Ratio is a variant of Sharpe Ratio. It is the excess return divided by only downside deviation of risk (see [downside\_risk](#name_DownsideRisk)). Proposed by Sortino, risk should be measured in regards to the returns not meeting the target rate, therefore Miminum Acceptable Return (MAR) is applied.

**Details**

The Sortino Ratio is used to make comparison between portfolios or funds, similar to Sharpe Ratio. A higher value indicates better performance. Comparison between Sortino Ratio and Sharpe Ratio gives information of how much the portfolio’s volatility is related to upside risk and downside risk.

Where is the observation of asset return, and is the downside risk under MAR . MAR can be an variable in the input data set.

**Usage**

%SortinoRatio(returns)

%SortinoRatio(returns, MAR=, group=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. [Default=0]

group - Optional. Specifies to choose full observations or subset observations as 'n' in the divisor. {FULL, SUBSET} [Default=FULL]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with Sortino Ratio. [Default="SortinoRatio"]

**Author**

Ruicheng Ma

**See Also**

[Sharpe\_Ratio](#name_SharpeRAtio), [Kappa](#name_Kappa), [downside\_risk](#name_DownsideRisk), [Information\_Ratio](#name_InformationRatio)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***SortinoRatio***(prices);

/\*Or\*/

/\*%SortinoRatio(prices, MAR=0, group=FULL, dateColumn=DATE, outData=MySortinoRatio);\*/

*[Specific\_Risk](#SpecificRisk_TOC)* **specific risk of the return distribution**

**Description**

Specific risk is calculated by taking the standard deviation of the error term of the CAPM regressing equation. Specific risk is not the same as market risk, which is the standard deviation of the benchmark asset or index.

**Details**

Specific risk can be thought of as the standard deviation of the regression epsilon, or it can be thought of as the square root of total risk squared minus systematic risk squared.

Where is the regression epsilon.

**Usage**

%Specific\_Risk (returns, BM=, Rf=, scale=);

%Specific\_Risk(returns, BM=, Rf=, scale=, dateColumn=, outData=);

**Arguments**

returns - Required. Data set containing returns with option to include risk free rate variable.

BM - required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

VARDEF - Pptional. Specifies the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData- Optional. Output data set with specific risk values. [Default= “Risk\_Specific”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Systematic\_Risk](#name_SystematicRisk), [table\_SpecificRisk](#name_tableSpecificRisk), [Total\_Risk](#name_TotalRisk)

**References**

Carl Bacon, *Practical portfolio performance measurement and attribution*, second edition 2008, p.75.

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Specific\_Risk***(prices, BM= SPY, Rf= 0.01/252, scale= 252);

/\*Or\*/

/\*%Specific\_Risk(prices, BM= SPY, Rf= IBM, scale= 252, dateColumn= Date, outData= Risk\_specific);\*/

*[Standard\_Deviation](#StandardDeviation_TOC)* **calculate standard deviation for each variable**

**Description**

Calculate standard deviation for each variable in the input data set. This macro contains the option to annualize the standard deviation given the extra parameter, scale.

**Details**

The standard deviation is calculated using the following formula:

Where = return observation of asset, = the mean return of the asset, and = number of observations.

Equations above calculate standard deviation based on the periodicity of the data used—daily, monthly, quarterly, or yearly.

The annualized standard deviation is calculated as:

**Usage**

%Standard\_Deviation (returns);

%Standard\_Deviation(returns, annualized=, scale=, VARDEF=, dateColumn=, outData=);

**Arguments**

Returns - Required. Data Set containing returns.

annualized - Optional. Option to annualize the standard deviation. {TRUE, FALSE} [Default= FALSE]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default= 1]

VARDEF - optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output data set with standard deviations. [Default= “StdDev”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[StdDev\_Annualized](#name_StdDevAnnualized)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Standard\_Deviation***(prices);

/\*Or\*/

/\*%Standard\_Deviation(prices, dateColumn= Date, outData= StdDev);\*/

*[StdDev\_Annualized](#StdDevAnnualized_TOC)* calculate annualized standard deviation

**Description**

Calculate the annualized standard deviation based on the number of periods in a year.

**Details**

For comparison, standard deviation is normally annualized for presentation purposes. To annualized standard deviation we need to multiply by the square root of the number of periods in a year.

**Usage**

%StdDev\_Annualized (returns, scale=);

%StdDev\_Annualized(returns, scale=, VARDEF=, dateColumn=, outData=);

**Arguments**

returns - required. Data Set containing returns.

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

annualized - Optional. Annualize the standard deviation. {TRUE, FALSE} [Default= FALSE]

VARDEF - Optional. Specifies the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData- Optional. Output data set with annualized standard deviation. [Default= “annualized\_StdDev”]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Standard\_Deviation](#name_StandardDeviation)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***StdDev\_Annualized***(prices, scale=252);

/\*Or\*/

/\*%StdDev\_Annualized(prices, scale= 252, dateColumn= Date, outData= annualized\_StdDev);\*/

*[Sterling\_Ratio](#SterlingRatio_TOC)* calculate Sterling Ratio

**Description**

Sterling Ratio measures an asset’s return against its drawdown risk like Calmar Ratio. The difference is Sterling Ratio adds an excess risk measure to the drawdown in the denominator.

**Details**

There exist different methods to calculate Sterling Ratio. Our definition is consistent with the one in R package.

Sterling Ratio is calculated as follows.

**Usage**

%Sterling\_Ratio(returns);

%Sterling\_Ratio(returns, scale=, method=, excess=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

excess - Optional. The yield of risk-free investment compared by any investment with a return stream. [Default=0.1]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with Sterling ratios. [Default="SterlingRatio"]

**Author**

Qiyuan Yang

**See Also**

[Drawdowns](#name_Drawdowns), [max\_drawdown](#name_MaxDrawdown), [Sharpe\_Ratio](#name_SharpeRAtio), [Calmar\_Ratio](#name_CalmarRatio)

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Sterling\_Ratio***(prices);

/\*Or\*/

/\*%Sterling\_Ratio(prices, scale=252, method=LOG, excess=0.2, dateColumn=DATE, outData=MySterlingRatio);\*/

*[Systematic\_Risk](#SystematicRisk_TOC)*calculate systematic risk

**Description**

The systematic risk calculated here uses the definition of the product of beta from CAPM and market risk. Market risk is the standard deviation of the benchmark.

**Details**

Where is systematic risk, is the beta from regression equation of CAPM, and is the benchmark risk.

**Usage**

%Systematic\_Risk (Returns, BM=, Rf=, scale=);

%Systematic\_Risk(Returns, BM=, Rf=, scale=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set of systematic risk. [Default="Risk\_systematic"]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Specific\_Risk](#name_SpecificRisk), [table\_SpecificRisk](#name_tableSpecificRisk), [Appraisal\_Ratio](#name_appraisalRatio)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Systematic\_Risk***(prices, BM= SPY, Rf= 0.01/252, scale=252);

/\*Or\*/

/\*%Systematic\_Risk(prices, BM= SPY, Rf= IBM, scale= 252, dateColumn= Date, outData= Risk\_systematic);\*/

*[Table\_Annualized\_Returns](#tableAnnualizedReturns_TOC)*create a table of annualized statistics

**Description**

This macro returns a table of annualized statistics including annualized return, annualized standard deviation, and annualized Sharpe Ratio.

**Details**

Here’s an example of the form of output data set.

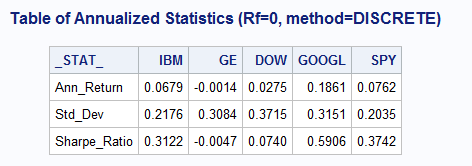


Table . Sample output from returns of 01/03/2005-12/31/2014

**Usage**

%table\_Annualized\_Returns (returns, Rf=, scale=);

%table\_Annualized\_Returns(returns, Rf=, scale=, method=, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

digits - Optional. Specifies the amount of digits to display in output. [Default= 4]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set of annualized returns statistics. [Default="annualized\_table"]

printTable - Optional. Option to print table. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Return\_Annualized](#name_Return_Annualized), [Standard\_Deviation](#name_StandardDeviation), [SharpeRatio\_Annualized](#name_SharpeRatioAnnualized)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_Annualized\_Returns(***prices, Rf= 0.01/252, scale=252);

/\*Or\*/

/\*%table\_Annualized\_Returns(prices, Rf= SPY, scale= 252,method= DISCRETE, dateColumn= DATE, outData= annualized\_table, printTable= PRINT);

*[Table\_AutoCorrelation](#tableAutoCorrelations_TOC)* create a table of autocorrelation values and Q-statistics

**Description**

Creates a table consisting of p-valuesand corresponding significance levels as measured by the Ljung-Box test Q-statistic for each asset.

More information on the Ljung-Box test and the accompanying Q-test can be found here:

<https://en.wikipedia.org/wiki/Ljung%E2%80%93Box_test>

**Details**

This is an example of output format.

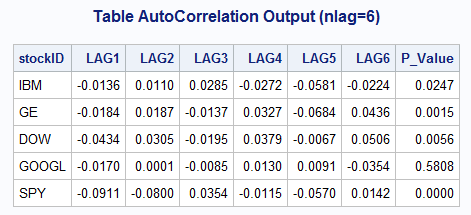


Table . Sample output from returns of 01/03/2005-12/31/2014

**Usage**

%Table\_AutoCorrelation (Returns, Rf=, scale=);

%Table\_AutoCorrelation(Returns, Rf=, scale=, method=, annualized=, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data Set containing returns of the portfolio.

nlag - Required. Specifies the number of lags to perform (and number of columns). The value of lag should be at least p+d+q based from the model ARIMA(p, d, q). "table\_autocorrelation" will not return a p-value if lag is less than this value.

digits - Optional. Specifies the amount of digits to display in output. [Default= 4]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Specifies name of output Data Set of autocorrelations. [Default="AutoCorrelations"]

printTable- Optional. Option to print table. {PRINT, NOPRINT}. [Default= NOPRINT]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_AutoCorrelation(***prices, nlag= 6);

/\*Or\*/

/\*%table\_AutoCorrelation(prices, nlag= 6, dateColumn= DATE, outData=AutoCorrelations, printTable= PRINT);

*[Table\_CalendarReturns](#tableCalendarReturns_TOC)* Create a table of calendar returns

**Description**

Creates a table containing yearly and monthly returns.

**Details**

To mimic results from R performance analytics, the user should specify one asset by name to be printed in the parameters of the macro. If no asset name is found, the table will show results for all assets in order by asset name.

Below is a sample output of DOW from 01/03/2005 to 12/31/2014.

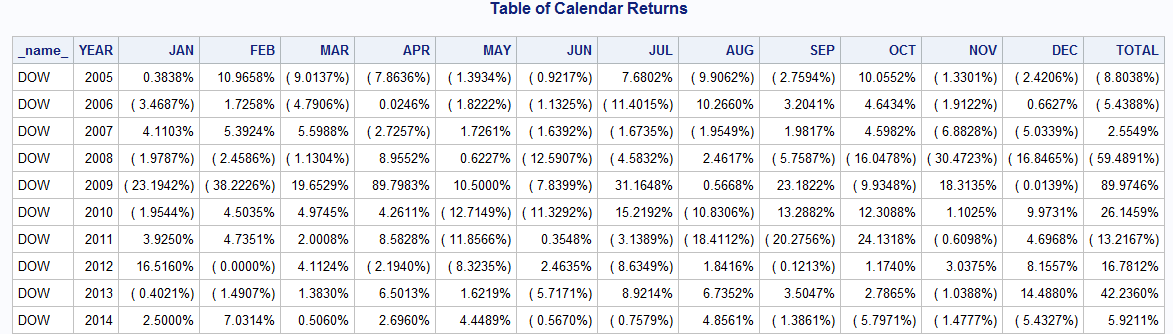


Table . Calendar returns sample output

**Usage**

%table\_CalendarReturns(Returns);

%table\_CalendarReturns(Returns, method=, dateColumn=, outData=, printTable=, name=);

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

digits - Optional. Specifies the amount of digits to display in output. [Default= 4]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set of calendar returns. [Default="Calendar\_Returns"]

printTable - Optional. Option to print returns of all or one asset. {PRINT, NOPRINT}. [Default= NOPRINT]

asset - Required. Name of the variable to print if printTable= PRINT.

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_CalendarReturns***(prices);

/\*Or\*/

/\*%table\_CalendarReturns(prices, method= DISCRETE, dateColumn= DATE, outData= Calendar\_Returns, printTable= PRINT, name= IBM);

*[Table\_CAPM](#table_CAPM_TOC)* table containing a series of measures of asset excess returns against a benchmark

**Description**

The excess return of assets are used to find a series of measures against a benchmark, based on CAPM model. The market benchmark comes from the input data set and its variable name needs to be specified.

**Details**

Below is a sample output with benchmark as SPY, from 01/03/2005 to 12/31/2014.

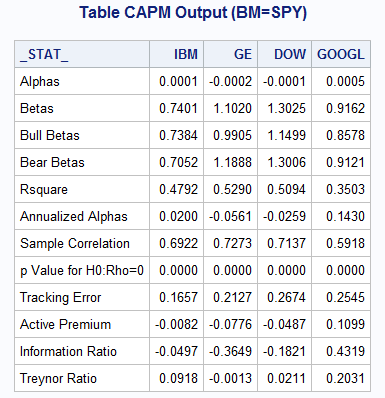


Table . Table CAPM sample output

**Usage**

%table\_CAPM(returns=, BM=);

%table\_CAPM(returns=, BM=, scale=, digits=, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data set containing returns of the portfolio.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year. {daily=252, monthly=12, quarterly=4, yearly=1} [Default=1]

digits - Optional. Specifies number of digits displayed in the output.

VARDEF - Optional. Specifies the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Specifies name of output Data Set of correlations. [Default="CAPM"]

printTable - Optional. Option to print output table. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Qiyuan Yang

**See Also**

[CAPM\_alphta\_beta](#name_CAPMalphabeta), [Bull\_Bear\_beta](#name_BullBearbeta), [TrackingError](#name_trackingError), [Information\_Ratio](#name_InformationRatio), [Treynor\_Ratio](#name_TreynorRatio).

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_CAPM***(prices, BM=IBM);

/\*Or\*/

/\*%table\_CAPM(prices, BM=IBM, scale=4, outData=MyCAPMTable, dateColumn=DATE, printTable=PRINT);\*/

*[Table\_CaptureRatios](#tableCaptureRatios_TOC)* create a table of up/down capture ratios

**Description**

This macro creates a table of capture ratios with the option to print it. See [UpDownRatios](#name_UpDownRatios) for definition of capture ratio.

**Details**

Below is an example of capture ratios with return data of 01/03/2005-12/31/2014.

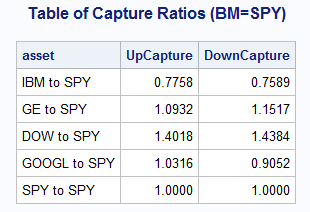


Table . Table CaptureRatios sample output

**Usage**

%Table\_CaptureRatios(returns, BM=)

%Table\_CaptureRatios(returns, BM=, digits=, dateColumn=, outData=, printTable=)

**Arguments**

returns - Required. Data Set containing returns and benchmark.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

digits - Optional. Specifies number of digits displayed in the output. [Default=4]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with up-down capture ratios. [Default="TableCaptureRatios"]

printTable - Optional. Option to print table. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Ruicheng Ma

**See Also**

[UpDownRatios](#name_UpDownRatios), [Table\_UpDownRatios](#name_TableUpDownRatios)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Table\_CaptureRatios***(prices, BM=SPY);

/\*Or\*/

/\*%Table\_CaptureRatios(prices, BM=SPY, dateColumn=DATE, outData=MyTableCaptureRatios, printTable=PRINT);\*/

*[Table\_Correlation](#tableCorrelations_TOC)* calculate correlations between assets and benchmark

**Description**

Calculates the correlation, significance, and confidence intervals for correlation estimates with an asset or benchmark index.

**Details**

Below is an example of the output with return data from 01/03/2005 to 12/31/2014

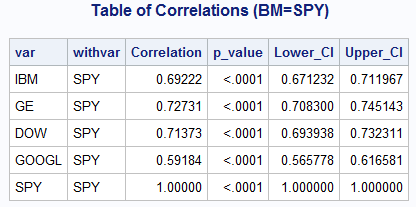


Table . Table Correlation sample output

**Usage**

%table\_Correlation(returns, returnsCompare=);

%table\_Correlation(returns, returnsCompare=, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data Set containing returns of the portfolio.

returnsCompare - Required. Specifies the variable to compute correlations against.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Specifies name of output Data Set of correlations. [Default="Correlations"]

printTable - Optional. Option to print output table. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_Correlation(***prices, returnsCompare= GE);

/\*Or\*/

/\*%table\_Correlation(prices,returnsCompare= SPY, dateColumn= DATE, outData=Correlations, printTable= PRINT);

*[Table\_Distributions](#Table_Distributions_TOC)*Distribtuions summary: statistics and stylized facts

**Description**

Create a table of statistics related to distribution. An example shows the list of the statistics in the “details” section.

**Details**

The VARDEF option is set for calculating annuzlied standard deviation only. As for skewness and kurtosis, the output table contains both sample and moment statistics. Excess kurtosis is kurtosis minus 3.

For sample skewness, this macro uses a different calculation (Fisher, 1973) from R. There exists tiny differences in the result.

See SAS documentations of UNIVARIATE procedure for more information about the algorithms.

Below is an example of the distribution table.

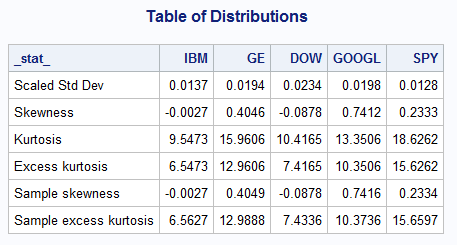


Table . Table Correlation sample output

**Usage**

%table\_Distributions(returns);

%table\_Distributions(returns, dateColumn=, outData=, digits=, scale=, printTable=);

**Arguments**

returns - Required. Data Set containing returns.

dateColumn - Optional. Specifies the date column in the data set. [Default= DATE]

outData - Optional. Output Data Set with distribution statistics. [Default= “distribution\_table”]

digits - Optional. Specifies the amount of digits to display in output. [Default= 4]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

printTable - Optional. Option to print table. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_Distributions(***prices);

/\*Or\*/

/\*%table\_Distributions(prices, dateColumn= DATE, outData=distribution\_table, digits= 8, scale= 252, printTable= PRINT);

*[Table\_Drawdowns](#tableDrawdowns_TOC)* list a table of worst drawdowns

**Description**

Create a table with user specified number of rows to display the statistics of worst drawdowns.

**Details**

Below is an example of the drawdowns table of IBM, from 01/03/2004 to 12/31/2014. The missing value in “EndDate” and “Recovery” indicates the most recent drawdown, which is not recovered yet.

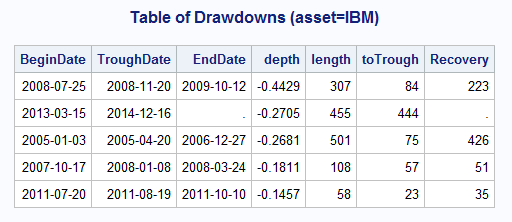


Table . Table Drawdowns sample output

**Usage**

%Table\_Drawdowns(returns, asset=);

%Table\_Drawdowns(returns, asset=, TOP=, method=, digits=, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data Set containing returns.

asset - Required. Name of the variable to find drawdown interval for.

TOP - Optional. The number of the drawdowns with worst depth to include. If the given number is larger than the number of total observations, all drawdowns will be displayed. [Default=5]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

digits - Optional. Specifies number of digits displayed in the output.

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with drawdowns. [Default="TableDrawdowns"]

printTable - Optional. Option to print table. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Ruicheng Ma

**See Also**

[Drawdowns](#name_Drawdowns), [max\_drawdown](#name_MaxDrawdown), [Find\_Drawdowns](#name_FindDrawdowns), [Sort\_Drawdowns](#name_SortDrawdowns)

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Table\_Drawdowns***(prices, asset=IBM, TOP=5);

/\*Or\*/

/\*%Table\_Drawdowns(prices, asset=IBM, TOP=10, method=LOG, digits=5, dateColumn=DATE, outData=MyTableDrawdowns, printTable=PRINT);\*/

*[Table\_DrawdownsRatio](#tableDrawdownsRatio_TOC)* a table of drawdown-related ratios

**Description**

The table contains Calmar Ratio, Sterling Ratio, Burke Ratio, Pain Index, Pain Ratio, Ulcer Index, Martin Ratio.

**Details**

Below is an example of drawdowns ratio table.

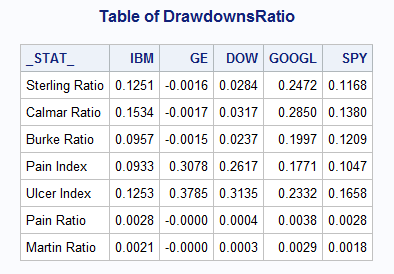


Table . Table DrawdownsRatio sample output

**Usage**

%Table\_DrawdownsRatio(returns);

%Table\_DrawdownsRatio(returns, Rf=, scale=, method=, digits=, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data Set containing returns of the portfolio.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year. {daily=252, monthly=12, quarterly=4, yearly=1}. [Default=1]

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

digits - Optional. Specifies number of digits displayed in the output. [Default=4]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Specifies name of output Data Set of drawdown ratios. [Default="table\_DrawdownsRatio"]

printTable - Optional. Option to print output table. {PRINT, NOPRINT}. [Default= NOPRINT]

**Author**

Qiyuan Yang

**See Also**

[Burke\_Ratio](#name_BurkeRatio), [Calmar\_Ratio](#name_CalmarRatio), [Sterling\_Ratio](#name_SterlingRatio), [Pain\_Index](#name_PainIndex), [Pain\_Ratio](#name_PainRatio), [Ulcer\_Index](#name_UlcerIndex) [Martin\_Ratio](#name_MartinRatio)

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Table\_DrawdownsRatio***(prices);

/\*Or\*/

/\*%Table\_DrawdownsRatio(prices, Rf=0, scale=252, method=LOG, digits=4, dateColumn=DATE, outData=MyTable, printTable=PRINT);\*/

*[Table\_HigherMoments](#tableHigherMoments_TOC)* create a table of higher moments and co-moments

**Description**

Create a summary of moments and co-moments with higher order.

**Details**

Below is an example of higher moments table (with data of GE and IBM).

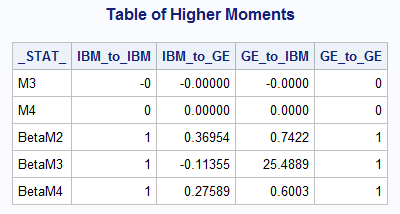


Table . Table HigherMoments sample output

**Usage**

%table\_HigherMoments(returns);

%table\_HigherMoments(returns, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data Set containing returns.

dateColumn - Optional. Specifies the date column in the returns data set. [Default= DATE]

outData - Optional. Output table name. [Default= “Higher\_Moments”]

printTable - Optional. Option to print the data set. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[BetaCoMoments](#name_BetaCo), [CoMoments](#name_CoMoments)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_HigherMoments(***prices);

/\*Or\*/

/\*%table\_HigherMoments(prices, dateColumn= DATE, outData=Higher\_Moments, printTable= PRINT);

*[Table\_InformationRatio](#tableInformationRatio_TOC)*Create a table of information ratio

**Description**

The table contains the information ratio and related statistics.

**Details**

Below is an example of the information ratio table (BM=SPY).

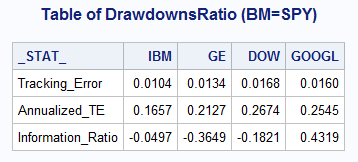


Table . Table InformationRatio sample output

**Usage**

%table\_InformationRatio(returns, BM=);

%table\_InformationRatio(returns, BM=, scale=, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data Set containing returns.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

digits - Optional. Specifies the amount of digits to display in output. [Default= 4]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

outData - Optional. Output Data Set with information ratio and tracking error. [Default="table\_InformationRatio"]

printTable - Optional. Option to print the output data set. {PRINT, NOPRINT}. [Default= NOPRINT]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Information\_Ratio](#name_InformationRatio), [TrackingError](#name_trackingError)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_InformationRatio(***prices, BM= SPY);

/\*Or\*/

/\*%table\_InformationRatio(prices, BM= DOW, scale= 252, dateColumn= DATE, outData=table\_InformationRatio, printTable= PRINT);

*[Table\_SpecificRisk](#tableSpecificRisk_TOC)*create a table of specific risk

**Description**

The output table contains specific risk, systematic risk, and total risk of an asset or financial instrument. Total risk is assumed to be the standard deviation in this table.

**Details**

Below is an example of the specific risk table (BM=SPY Rf=0).

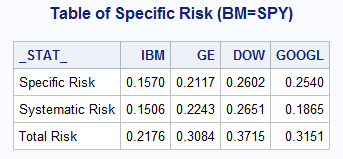


Table . Table InformationRatio sample output

**Usage**

%table\_SpecificRisk(returns, BM=, Rf=);

%table\_SpecificRisk(returns, BM=, Rf=, scale=, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

digits - Optional. Specifies the amount of digits to display in output. [Default= 4]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set of specific risk. [Default="table\_SpecificRisk"]

printTable - Optional. Option to print output data set. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Specific\_Risk](#name_SpecificRisk), [Systematic\_Risk](#name_SystematicRisk), [Standard\_Deviation](#name_StandardDeviation)

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_SpecificRisk(***prices, BM= SPY);

/\*Or\*/

/\*%table\_SpecificRisk(prices, BM= DOW, Rf= 0.01/252, scale= 252, dateColumn= DATE, outData=table\_SpecificRisk, printTable= PRINT);

*[Table\_Stats](#tableStats_TOC)*create a table of return statistics

**Description**

The table contains statistics of the return time series data. The list can be found in the following table.

**Details**

Below is an example of the stats table.

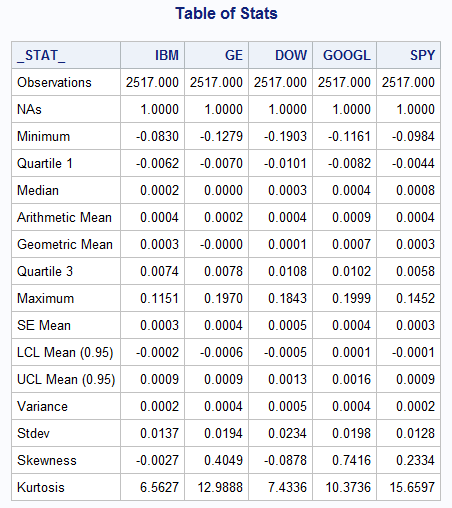


Table . Table SpecificRisk sample output

**Usage**

%table\_Stats(returns);

%table\_Stats(returns, alpha=, outData=, dateColumn=, digits=, printTable=);

**Arguments**

returns - Required. Data Set containing returns.

alpha - Optional. Significance level. Specifies the level of significance for the mean. [Default= 0.05]

outData - Optional. Output Data Set with related statistics. [Default= “Stats”]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

digits - Optional. Specifies the number of digits to display in the output table. [Default= 4]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

printTable - Optional. Option to print table. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_Stats(***prices);

/\*Or\*/

/\*%table\_Stats(prices, alpha= 0.05, outData= Stats, dateColumn= DATE, digits= 4,printTable= PRINT);

[*Tabl**e\_UpDownRatios*](#tableUpDownRatios_TOC) create a table of up/down capture/number/percent ratios

**Description**

This macro creates a table of capture/number/percent ratios with the option to print it.

**Details**

Below is an example of the variability table.

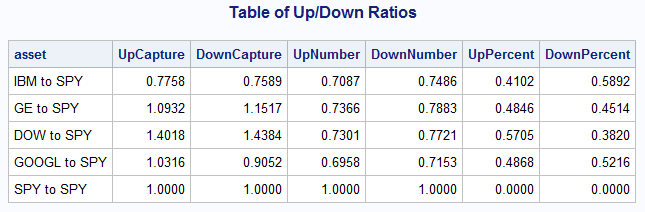


Table . Table Variability sample output

**Usage**

%Table\_UpDownRatios(returns, BM=)

%Table\_UpDownRatios(returns, BM=, digits=, dateColumn=, outData=, printTable=)

**Arguments**

returns - Required. Data Set containing returns and benchmark.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

digits - Optional. Specifies number of digits displayed in the output. [Default=4]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data set with up-down ratios. [Default="TableUpDownRatios"]

printTable - Optional. Option to print table. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Ruicheng Ma

**See Also**

[UpDownRatios](#name_UpDownRatios), [Table\_CaptureRatios](#name_TableCaptureRatios)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Table\_UpDownRatios***(prices, BM=SPY);

/\*Or\*/

/\*%Table\_UpDownRatios(prices, BM=SPY, dateColumn=DATE, outData=MyTableUpDowneRatios, printTable=PRINT);\*/

*[Table\_Variability](#tableVariability_TOC)*create a table of variability

**Description**

The variability in the table consists of mean absolute deviation, monthly and annualized standard deviation.

**Details**

Below is an example of the variability table.

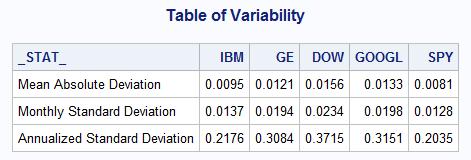


Table . Table Variability sample output

**Usage**

%table\_Variability(returns, scale=);

%table\_Variability(returns, scale=, dateColumn=, outData=, printTable=);

**Arguments**

returns - Required. Data Set containing returns.

scale - Required. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

digits - Optional. Specifies the amount of digits to display in output. [Default= 4]

dateColumn - Optional. Specifies the date column in the returns data set. [Default= DATE]

outData - Optional. Output Data Set with variability statistics. [Default= “variability\_table”]

printTable - Optional. Option to print output data set. {PRINT, NOPRINT} [Default= NOPRINT]

**Author**

Dominic Pazzula, Carter Johnston

**Examples**

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***table\_Variability(***prices, scale= 252);

/\*Or\*/

/\*%table\_Variability(prices, scale= 252,dateColumn= DATE, outData= variability\_table, printTable= PRINT);

*[Total\_Risk](#TotalRisk_TOC)* calculate total risk of returns

**Description**

Total risk is the squared root of the sum of squared systematic risk and squared specific risk. Systematic risk is the product of beta and market risk, and specific risk is the standard deviation of the error term in the regression equation. Both terms need to be annualized first.

**Details**

**Usage**

%Total\_Risk(returns, BM=)

%Total\_Risk(returns, BM=, Rf=, scale=, VARDEF=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}. [Default=1]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set of total risk. [Default="Risk\_total"]

**Author**

Qiyuan Yang

**See Also**

[Systematic\_Risk](#name_SystematicRisk), [Specific\_Risk](#name_SpecificRisk)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Total\_Risk***(prices, BM=SPY, scale=252);

/\*Or\*/

/\*%Total\_Risk(prices, BM=SPY, Rf=0, scale=252, dateColumn=DATE, outData=MyTotalRisk);\*/

*[TrackingError](#trackingError_TOC)*calculate the tracking error

**Description**

The tracking error is the standard deviation of the arithmetic excess return.

**Details**

This macro has the option to calculate annualized tracking error which is used in calculating information ratio. Also, to calculate the standard deviation of the excess return, the variance divisor can be chosen as n or n-1.

Where Ra,t is the tth asset return, Rb,t is the tth benchmark return. is the tth excess return, is the arithmetic mean of excess return, and n is the total number of observations.

**Usage**

%TrackingError(returns, BM=, scale=);

%TrackingError(returns, BM=, annualized=, scale=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

annualized - Optional. Specifies whether to return annualized tracking error rather than tracking error. {TRUE,FALSE}. [Default= FALSE]

scale - Optional. Option if annualized= TRUE, the number of periods in a year (ie daily scale= 252, monthly scale= 12, quarterly scale= 4). [Default= 1]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with tracking error. [Default="tracking\_error"]

**Author**

Dominic Pazzula, Carter Johnston

**See Also**

[Information\_Ratio](#name_InformationRatio)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***TrackingError***(prices, BM= SPY);

/\*Or\*/

/\*%TrackingError(prices, BM= SPY, scale= 252, annualized= TRUE, dateColumn= Date, outData= tracking\_error);\*/

*[Treynor\_Ratio](#TreynorRatio_TOC)* calculate Treynor Ratio of return over beta or systematic risk

**Description**

Calculates the regular Treynor Ratio, which is the asset excess return divided by asset beta. Asset beta comes from CAPM model and a market benchmark needs to be specified. Option to calculate modified Treynor Ratio is available and the denominator will be replaced by systematic risk.

**Details**

Treynor Ratio is very similar to Sharpe Ratio in a way that both ratios measure portfolio performance with respect to return and risk. Treynor Ratio uses the relative market risk and is thus more suitable for well-diversified portfolio. Sharpe Ratio uses the standard deviation and can be applied to all portfolios.

The systematic risk is defined by Bacon (2008) as the product of beta by market risk.

Where is the asset return, is the risk free rate for period, is regression beta, and is the systematic risk.

**Usage**

%Treynor\_Ratio(returns=, BM=);

%Treynor\_Ratio(returns=, BM=, scale=, method=, modified=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns with option to include risk free rate variable.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

Rf - Optional. The value or variable representing the risk free rate of return. [Default=0]

scale - Optional. Number of periods in a year {any positive integer, ie daily scale= 252, monthly scale= 12, quarterly scale= 4}.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

VARDEF - Optional. Specify the variance divisor, DF, degree of freedom, n-1; N, number of observations, n. {N, DF} [Default= DF]

modified - Optional. Specifies either regular or modified Treynor Ratio {FALSE, TRUE}. [Default=FALSE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with Treynor ratios. [Default="TreynorRatio"]

**Author**

Qiyuan Yang

**See Also**

[Sharpe Ratio](#name_SharpeRAtio), [Systematic Risk](#name_SystematicRisk), [CAPM\_alpha\_beta](#name_CAPMalphabeta).

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Treynor\_Ratio***(prices, BM=IBM);

/\*Or\*/

/\*%Treynor\_Ratio(prices, BM=IBM, Rf=0, scale=1, method=LOG, modified=FALSE, dateColumn=DATE, outData=MyTreynorRatio);\*/

*[Ulcer\_Index](#UlcerIndex_TOC)* calculate the Ulcer Index

**Description**

Calculate Ulcer Index devised by Perter Martin in 1987. Different from standard deviation that takes into account both up and down movements, Ulcer Index takes downside drawdowns.

**Details**

The drawdowns used in Ulcer ratio are calculated by Drawdowns macro. It is the cumulative return divided by historical peak cumulative return minus 1, or 0 if current cumulative return is the historical peak.

Ulcer ratio as a risk measure is used in calculation of Martin Ratio.

The Ulcer Index can be calculated as:

Where is drawdowns calculated by Drawdowns macro, is number of observations of .

See [Burke\_Ratio](#name_BurkeRatio) for more information about drawdowns.

**Usage**

%Ulcer\_Index(returns);

%Ulcer\_Index(returns, method=, dateColumn=, outData=);

**Arguments**

returns - Required. Data Set containing returns.

method - Optional. Specifies either DISCRETE or LOG chaining method {DISCRETE, LOG}. [Default=DISCRETE]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with ulcer index. [Default="UlcerIndex"]

**Author**

Qiyuan Yang

**See Also**

[Martin\_Ratio](#name_MartinRatio)

**References**

P.G.Martin, *An Alternative Approach to the Measurement of Investment Risk & Risk-Adjusted Peformance*, Peter Martin’s Ulcer Index page, 1987.

**Example**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***Ulcer\_Index***(prices);

/\*Or\*/

/\*%Ulcer\_Index(prices, method=LOG, dateColumn=DATE, outData=MyUlcerIndex);\*/

*[UpDownRatios](#UpDownRatios_TOC)* calculate capture/number/percent ratios in up/down market

**Description**

These three metrics measure the performance of asset when market is up or down, against a benchmark asset. To be consistent with R function, the returns are compounded arithmetically for calculating capture ratio.

**Details**

The metrics are calculated as follows:

Capture Ratio is the sum of asset returns when market is up/down, divided by the sum of benchmark returns when market is up/down.

Number Ratio is the number of periods the asset is up/down when market is up/down, divided by number of periods the market is up/down.

Percentage Ratio is the number of periods the asset outperforms the market when market is up/down, divided by the number of periods the market is up/down.

**Usage**

%UpDownRatios(returns, BM=)

%UpDownRatios(returns, BM=, option=, side=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns and benchmark.

BM - Required. Specifies the variable name of benchmark asset or index in the returns data set.

option - Optional. Specifies which ratio to be calculated. If not specified, all three ratios will be displayed. {CAPTURE, NUMBER, PERCENTAGE} [Defaulted as blank]

side - Optional. Specifies up/down market statistics. If not specified, both up and down market will be calculated. {UP, DOWN} [Defaulted as blank]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with up-down ratios. [Default="UpDownRatios"]

**Author**

Ruicheng Ma

**See Also**

[Table\_CaptureRatios](#name_TableCaptureRatios), [Table\_UpDownRatios](#name_TableUpDownRatios)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***UpDownRatios***(prices, BM=SPY);

/\*Or\*/

/\*%UpDownRatios(prices, BM=SPY, option=CAPTURE, side=UP, dateColumn=DATE, outData=MyUpDownRatio);\*/

*[upside\_frequency](#UpsideFrequency)* calculate upside frequency

**Description**

Calculate the frequency of returns greater than the Minimum Acceptable Return (MAR).

**Details**

Where is the number of returns greater than MAR and is the number of whole observations.

**Usage**

%upside\_frequency(returns)

%upside\_frequency(returns, MAR=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. [Default=0]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with upside frequency. [Default="UpsideFrequency"]

**Author**

Qiyuan Yang

**See Also**

[upside\_risk](#name_UpsideRisk), [downside\_risk](#name_DownsideRisk), [downside\_frequency](#name_DownsideFrequency)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***upside\_frequency***(prices);

/\*Or\*/

/\*%upside\_frequency(prices, MAR=0, dateColumn=DATE, outData=MyUpsideFrequency);\*/

*[upside\_risk](#UpsideRisk_TOC)* calculate total upside statistics of risk

**Description**

Calculate upside risk, variance and potential. All three are metrics of variability of performance over a target rate.

**Details**

We take the subset of returns that are greater than the Minimum Acceptable Return (proposed by Sharpe) to calculate the statistics.

Where can be the number of whole observations or the number of observations with return larger than MAR. The user has the option to choose wether the subgroup is taken as the denominator.

**Usage**

%upside\_risk(returns, option=)

%upside\_risk(returns, MAR=, option=, group=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. [Default=0]

option- Required. {RISK, VARIANCE, POTENTIAL}. Choose "RISK" to calculate the upside risk, "VARIANCE" to calculate upside variance, or "POTENTIAL" to calculate upside potential.

group - Optional. Specifies to choose full observations or subset observations as 'n' in the divisor. {FULL, SUBSET} [Default=FULL]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. output Data Set with upside risks. [Default="UpsideRisk"]

**Author**

Qiyuan Yang

**See Also**

[upside\_frequency](#name_UpsideFrequency), [downside\_risk](#name_DownsideRisk), [downside\_frequency](#name_DownsideFrequency)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***upside\_risk***(prices, option=RISK);

/\*Or\*/

/\*%upside\_risk(prices, MAR=0, option=RISK, group=FULL, dateColumn=DATE, outData=MyUpsideRisk);\*/

*[UpsidePotentialRatio](#UpsidePotentialRatio_TOC)*  calculate upside potential ratio

**Description**

Downside frequency is the proportion of returns that are smaller than the Minimum Acceptable Return (MAR) in the data set.

**Details**

Where is the tth return and is the Minimum Acceptable Raturn, and is either the number of total observations or the number of observations in the subset that are greater than MAR. is the number of total observations or the number of observations in the subset that are smaller than MAR.

**Usage**

%UpsidePotentialRatio(returns)

%UpsidePotentialRatio(returns, MAR=, group=, dateColumn=, outData=)

**Arguments**

returns - Required. Data Set containing returns.

MAR - Optional. Minimum Acceptable Return. [Default=0]

group - Optional. Specifies to choose full observations or subset observations as 'n' in the divisor. {FULL, SUBSET} [Default=FULL]

dateColumn - Optional. Date column in Data Set. [Default=DATE]

outData - Optional. Output Data Set with upside potential ratio. [Default="UPR"]

**Author**

Qiyuan Yang

**See Also**

[upside\_frequency](#name_UpsideFrequency), [upside\_risk](#name_UpsideRisk), [downside\_risk](#name_DownsideRisk), [downside\_frequency](#name_DownsideFrequency)

**Examples**

%let dir=C:\SVN\SAS\_Perf\_Anly;

libname input "&dir";

/\*Include SASPerformanceAnalytics\*/

%include "&dir\macro\\*.sas" /nosource;

**data** prices;

set input.prices;

**run**;

%***return\_calculate***(prices);

%***UpsidePotentialRatio***(prices);

/\*Or\*/

/\*%UpsidePotentialRatio(prices, MAR=0, group=SUBSET, dateColumn=DATE, outData=MyUPR);\*/