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

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

# SAS topics documented:

[ActivePremium……………………………………………………………………………………………………………………………8](#name_activePremium)

[Adjusted\_SharpeRatio…………………………………………………………………………………………………………………9](#name_Adjusted_SharpeRatio)

[Appraisal\_Ratio………………………………………………………………………………………………………………………..10](#name_appraisalRatio)

[Average\_Drawdown……………………………………………………………………………………………………….…………12](#name_averageDrawdown)

[Average\_Length…….……………………………………………………………………………………………………….…………13](#name_AverageLength)

[Average\_Recovery……………………………………………………………………………………………………….…………14](#name_AverageRecovery)

[BernardoLedoitRatio…………………………………………………………………………………………………….…………15](#name_BernardoLedoit)

[BetaC](#name_BetaCo)[oMoments…………………………………………………………………………………………………………….…………16](#name_BetaCo)

[Bull\_B](#name_BullBearbeta)[ear\_beta……………………………………………………………………………………………………………….…………17](#name_BullBearbeta)

[Burke\_Ratio…...…………………………………………………………………………………………………………….…………18](#name_BurkeRatio)

[Calmar\_Ratio…...…………………………………………………………………………………………………………….…………20](#name_CalmarRatio)

[CAPM\_alpha\_beta…………………………………………………………………………………………………………….………21](#name_CAPMalphabeta)

[CAPM\_epsilon………………………………………………………………………………………………………………….……….22](#name_CAPMepsilon)

[CAPM\_JensenAlpha………………………………………………………………………………………………………………….24](#name_CAPMJensenalpha)

[Centered\_moments………………………………………………………………………………………………………………….25](#name_CenteredMoments)

[Chart\_Autoregression…………………………………………………………………………………………………………….26](#ChartAutoregression_TOC)

[Chart\_Correlation……….…………………………………………………………………………………………………………….28](#ChartCorrelation_TOC)

[Chart\_CumulativeReturns……………………………………………………………………………………………………….29](#ChartCumulativeReturns_TOC)

[Chart\_Drawdown.…………………………………………………………………………………………………………………….30](#name_ChartDrawdown)

[Chart\_ECDF…..……………………………………………………………………………………………………………………….31](#name_ChartECDF)

[Chart\_Histogram……………………………………………………………………………………………………………………….32](#name_ChartHistogram)

[C](#name_ChartRegression)[hart\_Regression…………………………………………………………………………………………………………………….34](#name_ChartRegression)

[Chart\_RelativePerformance…………………………………………………………………………………………………….35](#name_ChartRelativePerformance)

[Chart\_Scatter……………………………………………………………………………………………………………………………36](#name_ChartScatter)

[CoMoments………………………………………………………………………………………………………………………………38](#name_CoMoments)

[Crea](#name_CreateHash)[te\_Hash……………………………………………………………………………………………………………………………39](#name_CreateHash)

[D\_Ratio……………………………………………………………………………………………………………………………………40](#name_DRatio)

[download\_ff3……………………………………………………………………………………………………………………………41](#name_Downloadff3)

[download\_FRED………………………………………………………………………………………………………………………42](#name_downloadFRED)

[download\_yahoo………………………………………………………………………………………………………………………42](#name_downloadyahoo)

[downside\_frequency…………………………………………………………………………………………………………………43](#name_DownsideFrequency)

[downside\_risk…………………………………………………………………………………………………………………………44](#name_DownsideRisk)

[Drawdowns….……………………………………………………………………………………………………………………………45](#name_Drawdowns)

[Drawdown\_Deviation………………………………………………………………………………………………………………47](#name_DrawdownDeviation)

[Fama\_Beta………………………………………………………………………………………………………………………………..48](#name_famabeta)

[Find\_Drawdowns.……………………………………………………………………………………………………………………..49](#name_FindDrawdowns)

[Geo\_Mean………………………………………………………………………………………………………………………………..50](#name_GeoMean)

[get](#name_GetStocks)[\_stocks………………………………………………………………………………………………………………………………..51](#name_GetStocks)

[Hurst\_Index](#name_HurstIndex)[……………………………………………………………………………………………………………………………..52](#name_HurstIndex)

[Information\_Ratio…………………………………………………………………………………………………………………….53](#name_InformationRatio)

[K](#name_Kappa)[appa………………….…………………………………………………………………………………………………………………….55](#name_Kappa)

[KellyRatio…….……………………………………………………………………………………………………………………………56](#name_KellyRatio)

[LPM……….…….……………………………………………………………………………………………………………………………57](#name_LPM)

[Martin\_Ratio…….………………………………………………………………………………………………………………………59](#name_MartinRatio)

[max\_drawdown………………………………………………………………………………………………………………………60](#name_MaxDrawdown)

[Mean\_Abs\_Deviation………………………………………………………………………………………………………………61](#name_MeanAbsDeviation)

[MSquared…………………………………………………………………………………………………………………………………62](#name_MSquared)

[Pain\_Index………………………………………………………………………………………………………………………………..64](#name_PainIndex)

[Pain\_Ratio………………………………………………………………………………………………………………………………..65](#name_PainRatio)

[Prices………………………………………………………………………………………………………………………………………..66](#name_prices)

[Prospect\_Ratio…](#name_ProspectRatio)[….……………………………………………………………………………………………………………………67](#name_ProspectRatio)

[Return\_Annualized……………………………………………………………………………………………………………………68](#name_Return_Annualized)

[Return\_Calculate………………………………………………………………………………………………………………………69](#name_Return_Calculate)

[Return\_Centered……………………………………………………………………………………………………………………...70](#name_Return_Centered)

[Return\_Cumulative…………………………………………………………………………………………………………………..71](#name_Return_Cumulative)

[Return\_Excess……………………………………………………………………………………………………………………….….72](#name_Return_Excess)

[Sharpe\_Ratio………………………………………………………………………………………………………………………..…..73](#name_SharpeRAtio)

[SharpeRatio\_Annualized………………………………………………………………………………………………………..…75](#name_SharpeRatioAnnualized)

[Simple\_Normalize……………………………………………………………………………………………………………..…76](#name_SimpleNormalize)

[Simple\_Normalize\_By………………………………………………………………………………………………………..…77](#name_SimpleNormalizeBy)

[Sort\_Drawdowns.](#name_SortDrawdowns)[……………………………………………………………………………………………………………………..77](#name_SortDrawdowns)

[SortinoRatio……………………………………………………………………………………………………………………………..79](#name_SortinoRatio)

[Specific\_Risk……………………………………………………………………………………………………………………………..80](#name_SpecificRisk)

[Standard\_Deviation…………………………………………………………………………………………………………………..81](#name_StandardDeviation)

[StdDev\_Annualized………………………………………………………………………………………………………………..…83](#name_StdDevAnnualized)

[Ster](#name_SterlingRatio)[ling\_Ratio……….………………………………………………………………………………………………………………..…84](#name_SterlingRatio)

[Systematic\_Risk………………………………………………………………………………………………………………..………85](#name_SystematicRisk)

[Table\_Annualized\_Returns…………………………………………………………………………………………..………….86](#name_tableAnnualizedReturns)

[Table\_AutoCorrelations…………………………………………………………………………..……………………….………88](#name_tableAutoCorrelations)

[Table\_CalendarReturns…………………………………………………………………………………………………….………89](#name_tableCalendarReturns)

[Table\_](#name_tableCAPM)[CAPM……..………………………………………………………………………………………………………………..……90](#name_tableCAPM)

[Table\_CaptureRatios](#name_TableCaptureRatios)[…………………………………………………………………………………………………………..……91](#name_TableCaptureRatios)

[Table\_Correlation………………………………………………………………………………………………………………..……92](#name_tableCorrelations)

[Table\_Distributions……………………………………………………………………………………………………………..……93](#name_tableDistributions)

[Table\_Drawdowns……..](#name_TableDrawdowns)[……………………………………………………………………………………………………….…….94](#name_TableDrawdowns)

[Table\_DrawdownsRatio…………………………………………………………………………………………………….…….95](#name_tableDrawdownsRatio)

[Table\_HigherMoments……………………………………………………………………………………………………….…….97](#name_tableHigherMoments)

[Table\_InformationRatio……………………………………………………………………………………………………………98](#name_tableInformationRatio)

[Table\_SpecificRisk…………………………………………………………………………………………………………………….99](#name_tableSpecificRisk)

[Table\_Stats……………………………………………………………………………………………………………………………100](#name_tableStats)

[Table\_Variability…………………………………………………………………………………………………………………….101](#name_tableVariability)

[Table\_UpDownRatios](#name_TableUpDownRatios)[……………………………………………………………………………………………………….……102](#name_TableUpDownRatios)

[Total\_Risk……………………………………………………………………………………………………………………….………103](#name_TotalRisk)

[TrackingError…………………………………………………………………………………………………………………….……104](#name_trackingError)

[Treynor\_Ratio………………………………………………………………………………………………………………….……106](#name_TreynorRatio)

[Ulcer\_Index…………………………………………………………………………………………………………………….……107](#name_UlcerIndex)

[UpDownRatios…...…………………………………………………………………………………………………………….……109](#name_UpDownRatios)

[upside\_frequency…………………………………………………………………………………………………………….……110](#name_UpsideFrequency)

[upside\_risk……………………………………………………………………………………………………………………….……111](#name_UpsideRisk)

[UpsidePotentialRatio……………………………………………………………………………………………………….……113](#name_UpSidePotentialRatio)

**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

**[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 SR is annualized Sharpe Ratio, S is the skewness and K 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

**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 α is the Jensen’s alpha, is the specific risk, 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\_Drawdown](#averageDrawdown_TOC)** calculate average drawdown

**Description**

Calculate average depth of observed drawdowns over entire period.

**Details**

The average depth is calculated as:

Where d is total number of drawdowns and Dt is the tth drawdown.

**Usage**

%Average\_Drawdown(returns)

%Average\_Drawdown(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](#AverageLength_TOC), [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\_Drawdown***(prices);

/\*Or\*/

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

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

**Description**

Finds 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, Qiyuan Yang

**See Also**

[Average\_Drawdown](#averageDrawdown_TOC),[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\_Length***(prices);

/\*Or\*/

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

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

**Description**

Finds 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, Qiyuan Yang

**See Also**

[Average\_Drawdown](#averageDrawdown_TOC), [Average\_Length](#AverageLength_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\_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 n is the number of whole observations, RT is loss threshold. BernardoLedoit ratio is the omega ratio with RT=0.

**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:

In above formulas, d is total number of drawdowns, n is total number of observations from the data set, and Dt 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.

**Usage**

%Burke\_Ratio(returns);

%Burke\_Ratio(returns, Rf=, scale=, asset=, 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)

**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.

**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**

“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:

**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 the portfolio return and b is the benchmark return.

**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 risk free rate, is the regression beta, is the portfolio return and b is the benchmark return.

The way to calculate Jensen’s alpha in the macro is to find from regression, annualize portfolio and benchmark return and , and follow the above formula to find it.

**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);\*/

**[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\_Autor****egression**](#name_ChartAutoregression)  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\_Co****rrelation**](#name_ChartCorrelation) Correlation matrix of returns

**Description**

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

**Usage**

%Chart\_Correlation(returns, title=, histogram=)

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

**Arguments**

returns - Required. Data Set containing returns.

title - Required. 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](#name_ChartCumulativeReturns)** 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=, 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]

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

**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, 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, 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);

%***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 n is the number of whole observations, nd is the number of negative returns, nu is the number of positive returns. It is very similar to BernarLedoit Ratio, only inversed and replace n with nd or nu.

**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 m is the number of returns smaller than MAR and n 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 n 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. Therefore, the calculation of drawdown is expressed as:

**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

**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**

The standard deviation of drawdown is calculated as:

Where d is the total number of drawdowns, Dt is the tth drawdown, and n 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 portfolio 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.

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 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’.

**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, Qiyuan Yang

**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:

*Geometric Mean=*

or

Where is the return and n is the number of total obervations. is Euler’s number.

**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:

m=[max(

Where r is the returns and n is the number of observations.

**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 overa benchmark divided by tracking error.

The information Ratio is given by the following formula:

*Information Ratio*=

where

**Usage**

%InformationRatio(returns, BM=, scale=);

%InformationRatio(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)

**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).

**Details**

Where rp is the mean of returns, n is the number of whole observations, l is the coefficient of Kappa, and Rt is the tth return in the time series.

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

**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 as 'n' 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, Qiyuan Yang

**See Also**

[SortinoRatio](#name_SortinoRatio)

**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, which 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 the excess return.

**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, 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);

%***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**

The LPM can be calculated as:

Where Xit is the return of asset i at time t, n is the degree of the moment, m is either the number of all observations or observations less than 𝜏. 𝜏 can be the threshold or the mean value.

**Usage**

%LPM(returns)

%LPM(returns, n=, group=, threshold=, 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]

threshold - Optional. 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

**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

**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);\*/

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

**Description**

Martin Ratio is calculated by taking the difference between asset return and risk free rate and divided by Ulcer Index.

**Details**

The Ulcer Index and Martin Ratio can be calculated as:

**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

**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**

Finds the worst drawdown for the whole time period for every asset. The drawdown is calculated based on cumulative return. The ‘invert’ option allows user to output drawdown as either positive or negative numbers to serve different purposes.

**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**

**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

**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 portfolio returns adjusted for its risk related to benchmark.

**Details**

**Usage**

%MSquared (returns, BM=);

%MSquared(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=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 of MSquared. [Default= "MSquared"]

**Author**

Dominic Pazzula, Carter Johnston

**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.

**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);\*/

**[Pain\_Index](#PainIndex_TOC)** calculate Pain Index

**Description**

Pain Index 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 can be calculated as:

Where n denotes the number of observations from the data set, and Dt is the tth drawdown.

**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

**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**

The Pain Index and Pain Ratio can be calculated as:

**Usage**

%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

**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 MAR is Miminum Acceptable Raturn, n 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\_Annualized](#Return_Annualized_TOC) calculate annualized returns

**Description**

With annualized returns, it is easier to compare assets with different time intervals of returns.

**Details**

For discrete compounding, the annualized return is:

where scale is the number of periods in a year, and n is the total number of observations in terms of periods.

For simple arithmetic returns, the formula is simply .

**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

**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\_Calculate](#Return_Calculate_TOC)** calculate asset returns

**Description**

Calculate simple or compound returns from a series of prices. Option to update the table in place or create new output.

**Details**

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

**See Also**

[Return\_Cumulative](#name_Return_Cumulative)

**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**

Calculates the cumulative return over a period of time, producing simple or geometric returns.

**Details**

The product of all individual period returns is

If calculating the cumulative return using the simple arithmetic method, cumulative returns are equivalent to the sum of all returns.

**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 returns of an asset in excess of the given risk free rate

**Description**

Calculates the returns of an asset in excess of the given “risk free rate” for the period.

**Details**

Ideally, the risk free rate will be for each period in which returns data is available, but a single average risk free rate for the period will work with this macro. While the Rf parameter is named after the risk free rate, any timeseries is allowed. A common alteration would be to find the excess returns over a benchmark asset or index.

**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);\*/

**[Sharpe\_Ratio](#Sharpe_Ratio_TOC)** calculate a traditional Sharpe Ratio of return over standard deviation.

**Description**

The Sharpe ratio is simply the return per unit of risk (represented by variability). The original Sharpe ratio uses standard deviation of returns as unit of risk.

**Details**

The Sharpe Ratio, in its original form, is the expected excess return over the standard deviation of excess return. The higher the Sharpe ratio, the better the combined performance of “risk” and return. 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=, 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)

**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 Sharpe ratio is a risk-adjusted measure of return that uses standard deviation to represent risk. The annualized Sharpe ratio is calculated using both annualized returns and annualized standard deviation to measure return.

**Details**

The Sharpe ratio is simply the return per unit of risk (represented by variance). The higher the Sharpe ratio, the better the combined performance of “risk” and return. This macro annualizes this ratio based on the scale parameter which specifies the number of periods for which there is data in one year.

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=, 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

**See Also**

[Sharpe\_Ratio](#name_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);

**[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.

**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, Qiyuan Yang

**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 improves Sharpe Ratio by taking only downside deviation of risk. 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**

Where Ra is the asset return, and δMAR is the downside risk.

**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, Qiyuan Yang

**See Also**

Sharpe\_Ratio, Kappa, downside\_risk, Information\_Ratio

**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 the standard deviation of the error term in the regression equation. Specific risk is not the same as market risk. Market risk 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.

**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)

**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) calculates standard deviation for univariate and multivariate series**

**Description**

Calculates the standard deviation, or “risk”, for univariate and multivariate series. 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 n= number of observations.

The annualized standard deviation is calculated as:

**Usage**

%Standard\_Deviation (returns);

%Standard\_Deviation(returns, annualized=, scale=, 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 a multiperiod or annualized standard deviation

**Description**

Calculate the standard deviation of a set of observations and estimate the annualized standard deviation based on the number of periods in a year.

**Details**

To normalize standard deviation across multiple periods, we multiply by the square root of the number of periods we wish to calculate over. To annualize standard deviation, we multiply by the square root of the number of periods per year.

**Usage**

%StdDev\_Annualized (returns, scale=);

%StdDev\_Annualized(returns, scale=, 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**

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)** Systematic risk of the return distribution

**Description**

Systematic risk is defined as the product of beta and market risk. Market risk is the standard deviation of the benchmark asset or index. The systematic risk is then annualized for comparison purposes.

**Details**

Where is systematic risk, is the regression beta, and is the market 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)

**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)** Annualized returns summary: statistics and stylized facts

**Description**

This macro returns a table containing annualized return, annualized standard deviation, and annualized Sharpe ratio.

**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)** table for calculating autocorrelation coefficients and significance

**Description**

Produces a table consisting of autocorrelation coefficients *ρ* and 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>

**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)** Monthly and calendar year return table

**Description**

Returns a table of returns formatted with years in rows, months in columns, and a total return in the last column.

**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.

**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.

**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.

**Usage**

%Table\_CaptureRatios(returns)

%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, Qiyuan Yang

**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 of multiple assets

**Description**

Calculates the correlation, significance, and confidence intervals for correlation estimates with an asset or benchmark index.

**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**

Table containing, standard deviation, skewness, sample standard deviation, kurtosis, excess kurtosis, sample skewness, and sample excess kurtosis.

**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.

**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, Qiyuan Yang

**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.

**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, Rf=, scale=, method=, digits=, dateColumn=, outData=, printTable=);

/\*Or\*/

/\*%Table\_DrawdownsRatio(prices, Rf=0, scale=252, method=LOG, digits=4, dateColumn=DATE, outData=MyTable, printTable=PRINT);\*/

[Table\_HigherMoments](#tableHigherMoments_TOC) Higher Moments summary: Statistics and stylized facts

**Description**

Summary of the higher moments and co-moments of the return distribution. Used to determine diversification potential. Also called “systematic” moments.

**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)** Information Ratio summary: Statistics and stylized facts

**Description**

Table of the Tracking error, annualized tracking error, and information ratio.

**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)** Specific Risk summary: Statistics and stylized facts

**Description**

Table of 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.

**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)** Returns summary: Statistics and stylized facts

**Description**

Returns a table of basic statistics on a data set that match the period of the data passed in (e.g. monthly returns will return monthly statistics and daily returns will return daily stats…)

**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);

**[Table\_Variability](#tableVariability_TOC)** Variability summary: Statistics and stylized facts

**Description**

Table of mean absolute deviation, monthly standard deviation, and annualized standard deviation.

**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);

[**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.

**Usage**

%Table\_UpDownRatios(returns)

%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, Qiyuan Yang

**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);\*/

**[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 tracking error of returns against a benchmark

**Description**

The tracking error is the measure of the unexplained portion of performance relative to a benchmark asset or index.

**Details**

The tracking error is calculated by taking the square root of the average of the squared deviations between the investment’s returns and the benchmark’s returns, then multiplying the result by the square root of the scale of returns.

where

**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.

**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 Pain 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 Pain Index can be calculated as:

Where n denotes the number of observations from the data set, and Dt is the tth drawdown.

**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

**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 and 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. [Defaulted as blank]

side - Optional. Specifies up/down market statistics. If not specified, both up and down market will be calculated. [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, Qiyuan Yang

**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 m is the number of returns greater than MAR and n 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 n can be the number of whole observations or the number of observations with return greater than MAR.

**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 Rt is the tth return and MAR is the Minimum Acceptable Raturn, and nu is either the number of total observations or the number of observations in the subset that are greater than MAR. nd 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);\*/