Package 'PerformanceAnalytics'

October 27, 2009

Type Package

Title Econometric tools for performance and risk analysis.

Version 1.0.0

Date 2009-10-23

Author Peter Carl, Brian G. Peterson

Maintainer Brian G. Peterson <brian@braverock.com>

Description Library of econometric functions for performance and risk analysis. This library aims to aid practitioners and researchers in utilizing the latest research in analysis of non-normal return streams. In general, this library is most tested on return (rather than price) data on a monthly scale, but most functions will work with daily or irregular return data as well.

Depends R (>= 2.9.0), zoo, xts (>= 0.6-8)

Suggests Hmisc, MASS, tseries, quadprog, sn, robustbase, quantreg

License GPL

URL http://braverock.com/R/

Contributors A special thanks for contributions from Kris Boudt, Stefan Albrecht, Khahn Nygyen, Sankalp Upadhyay

Repository CRAN

Date/Publication 2009-10-27 19:46:00

R topics documented:

PerformanceAnalytics-package
Econometric tools for performance and risk analysis.

Description

PerformanceAnalytics provides an R library of econometric functions for performance and risk analysis of financial instruments or portfolios. This library aims to aid practitioners and researchers in using the latest research for analysis of both normally and non-normally distributed return streams.

We created this library to include functionality that has been appearing in the academic literature on performance analysis and risk over the past several years, but had no functional equivalent in R. In doing so, we also found it valuable to have wrappers for some functionality with good defaults and naming consistent with common usage in the finance literature.

In general, this library requires return (rather than price) data. Almost all of the functions will work with any periodicity, from annual, monthly, daily, to even minutes and seconds.

The following sections cover Time Series Data, Performance Analysis, Risk Analysis (with a separate treatment of VaR), Summary Tables of related statistics, Charts and Graphs, a variety of Wrappers and Utility functions, and some thoughts on work yet to be done.

In this summary, we attempt to provide an overview of the capabilities provided by *PerformanceAnalytics* and pointers to other literature and resources in R useful for performance and risk analysis. We hope that this summary and the accompanying package and documentation partially fill a hole in the tools available to a financial engineer or analyst.

Time Series Data

Not all, but many of the measures in this package require time series data. *PerformanceAnalytics* uses the xts package for managing time series data for several reasons. Besides being fast and efficient, xts includes functions that test the data for periodicity and draw attractive and readable time-based axes on charts. Another benefit is that xts provides compatability with Rmetrics' timeSeries, zoo and other time series classes, such that *PerformanceAnalytics* functions that return a time series will return the results in the same format as the object that was passed in. Jeff Ryan and Josh Ulrich, the authors of xts, have been extraordinarily helpful to the development of *PerformanceAnalytics* and we are very greatful for their contributions to the community. The xts package extends the excellent zoo package written by Achim Zeileis and Gabor Grothendieck. zoo provides more general time series support, whereas xts provides functionality that is specifically aimed at users in finance.

Users can easily load returns data as time series for analysis with *PerformanceAnalytics* by using the Return.read function. The Return.read function loads csv files of returns data where the data is organized as dates in the first column and the returns for the period in subsequent columns. See read.zoo and as.xts if more flexibility is needed.

The functions described below assume that input data is organized with asset returns in columns and dates represented in rows. All of the metrics in *PerformanceAnalytics* are calculated by column and return values for each column in the results. This is the default arrangement of time series data in xts.

Some sample data is available in the managers dataset. It is an xts object that contains columns of monthly returns for six hypothetical asset managers (HAM1 through HAM6), the EDHEC Long-Short Equity hedge fund index, the S&P 500 total returns, and total return series for the US Treasury 10-year bond and 3-month bill. Monthly returns for all series end in December 2006 and begin at different periods starting from January 1996. That data set is used extensively in our examples and should serve as a model for formatting your data.

For retrieving market data from online sources, see quantmod's getSymbols function for downloading prices and chartSeries for graphing price data. Also see the tseries library for the function get.hist.quote. Look at xts's to.period function to rationally coerce irregular price data into regular data of a specified periodicity. The aggregate function has methods for tseries and zoo timeseries data classes to rationally coerce irregular data into regular data of the correct periodicity.

Finally, see the function Return.calculate for calculating returns from prices.

Performance Analysis

The literature around the subject of performance analysis seems to have exploded with the popularity of alternative assets such as hedge funds, managed futures, commodities, and structured products. Simpler tools that may have seemed appropriate in a relative investment world seem inappropriate for an absolute return world. Risk measurement, which is nearly inseparable from performance assessment, has become multi-dimensional and multi-moment while trying to answer a simple question: "How much could I lose?" Portfolio construction and risk budgeting are two sides of the same coin: "How do I maximize my expected gain and avoid going broke?" But before we can approach those questions we first have to ask: "Is this something I might want in my portfolio?"

With the the increasing availability of complicated alternative investment strategies to both retail and institutional investors, and the broad availability of financial data, an engaging debate about performance analysis and evaluation is as important as ever. There won't be one *right* answer delivered in these metrics and charts. What there will be is an accretion of evidence, organized to *assist* a decision maker in answering a specific question that is pertinent to the decision at hand. Using such tools to uncover information and ask better questions will, in turn, create a more informed investor.

Performance measurement starts with returns. Traders may object, complaining that "You can't eat returns," and will prefer to look for numbers with currency signs. To some extent, they have a point - the normalization inherent in calculating returns can be deceiving. Most of the recent work in performance analysis, however, is focused on returns rather than prices and sometimes called "returns-based analysis" or RBA. This "price per unit of investment" standardization is important for two reasons - first, it helps the decision maker to compare opportunities, and second, it has some useful statistical qualities. As a result, the *PerformanceAnalytics* library focuses on returns. See Return.calculate for converting net asset values or prices into returns, either discrete or continuous. Many papers and theories refer to "excess returns": we implement a simple function for aligning time series and calculating excess returns in Return.excess.

We provide two functions to calculate weighted returns for a portfolio of assets. If you have a single weighting vector, or want the equal weighted portfolio, use Return.portfolio. If you have a portfolio that is periodically rebalanced, and multiple time periods with different weights, use Return.rebalancing. Both functions will subset the return series to only include returns for assets for which weights are provided.

Returns and risk may be annualized as a way to simplify comparison over longer time periods. Although it requires a bit of estimating, such aggregation is popular because it offers a reference point for easy comparison. Examples are in `Return.annualized`, `sd.annualized`, and `SharpeRatio.annualized`.

Basic measures of performance tend to treat returns as independent observations. In this case, the entirety of R's base is applicable to such analysis. Some basic statistics we have collected in `table.Stats` include:

`mean`	arithmetic mean
`mean.geometric`	geometric mean
`mean.stderr`	standard error of the mean (S.E. mean)
`mean.LCL`	lower confidence level (LCL) of the mean
`mean.UCL`	upper confidence level (UCL) of the mean
`quantile`	for calculating various quantiles of the distribution
`min`	minimum return
`max`	maximum return
`range`	range of returns
`length(R)`	number of observations
`sum(is.na(R))`	number of NA's

It is often valuable when evaluating an investment to know whether the instrument that you are examining follows a normal distribution. One of the first methods to determine how close the asset is to a normal or log-normal distribution is to visually look at your data. Both `chart.QQPlot` and `chart.Histogram` will quickly give you a feel for whether or not you are looking at a normally distributed return history. Differences between `var` and `SemiVariance` will help you identify `skewness` in the returns. Skewness measures the degree of asymmetry in the return distribution. Positive skewness indicates that more of the returns are positive, negative skewness indicates that more of the returns are negative. An investor should in most cases prefer a positively skewed asset to a similar (style, industry, region) asset that has a negative skewness.

Kurtosis measures the concentration of the returns in any given part of the distribution (as you should see visually in a histogram). The `kurtosis` function will by default return what is referred to as "excess kurtosis", where 0 is a normal distribution, other methods of calculating kurtosis than `method="excess"` will set the normal distribution at a value of 3. In general a rational investor should prefer an asset with a low to negative excess kurtosis, as this will indicate more predictable returns (the major exception is generally a combination of high positive skewness and high excess kurtosis). If you find yourself needing to analyze the distribution of complex or non-smooth asset distributions, the `nortest` package has several advanced statistical tests for analyzing the normality of a distribution.

Modern Portfolio Theory (MPT) is the collection of tools and techniques by which a risk-averse investor may construct an "optimal" portfolio. It was pioneered by Markowitz's ground-breaking 1952 paper *Portfolio Selection.* It also encompasses CAPM, below, 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 positing that assets that are not on the efficient frontier will either rise or fall in price until they are. The `CAPM.RiskPremium` is the measure of how much the asset's performance differs from the risk free rate. Negative Risk Premium generally

indicates that the investment is a bad investment, and the money should be allocated to the risk free asset or to a different asset with a higher risk premium. `CAPM.alpha` is the degree to which the assets returns are not due to the return that could be captured from the market. Conversely, `CAPM.beta` 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.

The Capital Market Line `CAPM.CML` relates the excess expected return on an efficient market portfolio to its risk (represented in CAPM by `sd`). The slope of the CML, `CAPM.CML.slope`, is the Sharpe Ratio for the market portfolio. The Security Market Line is constructed by calculating the line of `CAPM.RiskPremium` over `CAPM.beta`. For the benchmark asset this will be 1 over the risk premium of the benchmark asset. The slope of the SML, primarily for plotting purposes, is given by `CAPM.SML.slope`. 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.

One extension to the CAPM contemplates evaluating an active manager's ability to time the market. Two other functions apply the same notion of best fit to positive and negative market returns, separately. The `CAPM.beta.bull` is a regression for only positive market returns, which can be used to understand the behavior of the asset or portfolio in positive or 'bull' markets. Alternatively, `CAPM.beta.bear` provides the calculation on negative market returns. The `TimingRatio` uses the ratio of those to help assess whether the manager has shown evidence that of timing skill.

The performance premium provided by an investment over a passive strategy (the benchmark) is provided by `ActivePremium`, which is the investment's annualized return minus the benchmark's annualized return. A closely related measure is the `TrackingError`, which measures the unexplained portion of the investment's performance relative to a benchmark. The `InformationRatio` of an investment in a MPT or CAPM framework is the Active Premium divided by the Tracking Error. Information Ratio may be used to rank investments in a relative fashion.

We have also included a function to compute the `KellyRatio`. The Kelly criterion applied to position sizing will maximize log-utility of returns and avoid risk of ruin. For our purposes, it can also be used as a stack-ranking method like `InformationRatio` to describe the "edge" an investment would have over a random strategy or distribution.

These metrics and others such as `SharpeRatio`, `SortinoRatio`, `UpsidePotentialRatio`, Spearman rank correlation (see `rcorr`), etc., are all methods of rank-ordering relative performance. *Alexander and Dimitriu (2004) in "The Art of Investing in Hedge Funds"* show 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 that are not directly correlated to another measure or factor will still produce widely similar quantile rankings, so that the "buckets" of target instruments will have significant overlap. This observation specifically supports the point made early in this document regarding "accretion of the evidence" for a positive or negative investment decision.

Style Analysis

Style analysis is one way to help determine a fund's exposures to the changes in returns of major asset classes or other factors. *`PerformanceAnalytics`* has a few functions that calculate style weights using an asset class style model as described in detail in Sharpe (1992).

The following functions combine to calculate effective style weights and display the results in a bar chart. `chart.Style` calculates and displays style weights calculated over a single period.

chart.RollingStyle calculates and displays those weights in rolling windows through time. style.fit manages the calculation of the weights by method, and style.QPfit calculates the specific constraint case that requires quadratic programming.

There is a significant amount of academic literature on identifying and attributing sources of risk or returns. Much of it falls into the field of "factor analysis" where "risk factors" are used to retrospectively explain sources of risk, and through regression and other analytical methods *predict* future period returns and risk based on factor drivers. These are well covered in chapters on factor analysis in *Zivot and Wang(2006)* and also in the R functions factanal for basic factor analysis and princomp for Principal Component Analysis. The authors feel that financial engineers and analysts would benefit from some wrapping of this functionality focused on finance, but the capabilities already available from the base functions are quite powerful.

Risk Analysis

Many methods have been proposed to measure, monitor, and control the risks of a diversified portfolio. Perhaps a few definitions are in order on how different risks are generally classified. *Market Risk* is the risk to the portfolio from a decline in the market price of instruments in the portfolio. *Liquidity Risk* is the risk that the holder of an instrument will find that a position is illiquid, and will incur extra costs in unwinding the position resulting in a less favorable price for the instrument. In extreme cases of liquidity risk, the seller may be unable to find a buyer for the instrument at all, making the value unknowable or zero. *Credit Risk* encompasses *Default Risk*, or the risk that promised payments on a loan or bond will not be made, or that a convertible instrument will not be converted in a timely manner or at all. There are also *Counterparty Risks* in over the counter markets, such as those for complex derivatives. Tools have evolved to measure all these different components of risk. Processes must be put into place inside a firm to monitor the changing risks in a portfolio, and to control the magnitude of risks. For an extensive treatment of these topics, see Litterman, Gumerlock, et. al.(1998). For our purposes, *PerformanceAnalytics* tends to focus on market and liquidity risk.

The simplest risk measure in common use is volatility, usually modeled quantitatively with a univariate standard deviation on a portfolio. See sd. Volatility or Standard Deviation is an appropriate risk measure when the distribution of returns is normal or resembles a random walk, and may be annualized using sd.annualized, or the equivalent function sd.multiperiod for scaling to an arbitrary number of periods. Many assets, including hedge funds, commodities, options, and even most common stocks over a sufficiently long period, do not follow a normal distribution. For such common but non-normally distributed assets, a more sophisticated approach than standard deviation/volatility is required to adequately model the risk.

Markowitz, in his Nobel acceptance speech and in several papers, proposed that SemiVariance would be a better measure of risk than variance. See *Zin, Markowitz, Zhao (2006)*. This measure is also called SemiDeviation. The more general case is DownsideDeviation, as proposed by *Sortino and Price(1994)*, where the minimum acceptable return (MAR) is a parameter to the function. It is interesting to note that variance and mean return can produce a smoothly elliptical efficient frontier for portfolio optimization using solve.QP or portfolio.optim or MarkowitzPortfolio. Use of semivariance or many other risk measures will not necessarily create a smooth ellipse, causing significant additional difficulties for the portfolio manager trying to build an optimal portfolio. We'll leave a more complete treatment and implementation of portfolio optimization techniques for another time.

Another very widely used downside risk measures is analysis of drawdowns, or loss from peak value achieved. The simplest method is to check the maxDrawdown, as this will tell you the

worst cumulative loss ever sustained by the asset. If you want to look at all the drawdowns, you can `findDrawdowns` and `sortDrawdowns` in order from worst/major to smallest/minor. The `UpDownRatios` function will give you some insight into the impacts of the skewness and kurtosis of the returns, and letting you know how length and magnitude of up or down moves compare to each other. You can also plot drawdowns with `chart.Drawdown`.

One of the most commonly used and cited measures of the risk/reward tradeoff of an investment or portfolio is the `SharpeRatio`, which measures return over standard deviation. If you are comparing multiple assets using Sharpe, you should use `SharpeRatio.annualized`. It is important to note that William Sharpe now recommends `InformationRatio` preferentially to the original Sharpe Ratio. The `SortinoRatio` uses mean return over `DownsideDeviation` below the MAR as the risk measure to produce a similar ratio that is more sensitive to downside risk. Sortino later enhanced his ideas to use upside returns for the numerator and `DownsideDeviation` as the denominator in `UpsidePotentialRatio`. *Favre and Galeano(2002)* propose using the ratio of expected excess return over the Cornish-Fisher `VaR` to produce `SharpeRatio.modified`. `TreynorRatio` is also similar to the Sharpe Ratio, except it uses `CAPM.beta` in place of the volatility measure to produce the ratio of the investment's excess return over the beta.

One of the newer statistical methods developed for analyzing the risk of financial instruments is `Omega`. Omega analytically constructs a cumulative distribution function, in a manner similar to `chart.QQPlot`, but then extracts additional information from the location and slope of the derived function at the point indicated by the risk quantile that the researcher is interested in. Omega seeks to combine a large amount of data about the shape, magnitude, and slope of the distribution into one method. The academic literature is still exploring the best manner to use Omega in a risk measurement and control process, or in portfolio construction.

Any risk measure should be viewed with suspicion if there are not a large number of historical observations of returns for the asset in question available. Depending on the measure, the number of observations required will vary greatly from a statistical standpoint. As a heuristic rule, ideally you will have data available on how the instrument performed through several economic cycles and shocks. When such a long history is not available, the investor or researcher has several options. A full discussion of the various approaches is beyond the scope of this introduction, so we will merely touch on several areas that an interested party may wish to explore in additional detail. Examining the returns of assets with a similar style, industry, or asset class to which the asset in question is highly correlated and shares other characteristics can be quite informative. Factor analysis may be used to uncover specific risk factors where transparency is not available. Various resampling (see `tsbootstrap`) and simulation methods are available in R to construct an artificially long distribution for testing. If you use a method such as Monte Carlo simulation or the bootstrap, it is often valuable to use `chart.Boxplot` to visualize the different estimates of the risk measure produced by the simulation, to see how small (or wide) a range the estimates cover, and thus gain a level of confidence with the results. Proceed with extreme caution when your historical data is lacking. Problems with lack of historical data are a major reason why many institutional investors will not invest in an alternative asset without several years of historical return data available.

Value at Risk - VaR

Traditional mean-VaR: In the early 90's, academic literature started referring to "value at risk", typically written as VaR. Take care to capitalize VaR in the commonly accepted manner, to avoid confusion with var (variance) and VAR (vector auto-regression). With a sufficiently large data set, you may choose to use a non-parametric VaR estimation method using the historical distribution and

the probability quantile of the distribution calculated using qnorm. The negative return at the correct quantile (usually 95% or 99%), is the non-parametric VaR estimate. J.P. Morgan's RiskMetrics parametric mean-VaR was published in 1994 and this methodology for estimating parametric mean-VaR has become what people are generally referring to as "VaR" and what we have implemented as VaR with method="historical". See *Return to RiskMetrics: Evolution of a Standard* at http://www.riskmetrics.com/r2rovv.html. Parametric traditional VaR does a better job of accounting for the tails of the distribution by more precisely estimating the tails below the risk quantile. It is still insufficient if the assets have a distribution that varies widely from normality. That is available in VaR with method="gaussian".

The R package VaR contains methods for simulating and estimating lognormal VaR.norm and generalized Pareto VaR.gpd distributions to overcome some of the problems with nonparametric or parametric mean-VaR calculations on a limited sample size. There is also a VaR.backtest function to apply simulation methods to create a more robust estimate of the potential distribution of losses. The VaR package also provides plots for its functions.

Modified Cornish-Fisher VaR: The limitations of traditional mean-VaR are all related to the use of a symmetrical distribution function. Use of simulations, resampling, or Pareto distributions all help in making a more accurate prediction, but they are still flawed for assets with significantly non-normal (skewed and/or kurtotic) distributions. *Huisman (1999)* and *Favre and Galleano (2002)* propose to overcome this extensively documented failing of traditional VaR by directly incorporating the higher moments of the return distribution into the VaR calculation.

This new VaR measure incorporates skewness and kurtosis via an analytical estimation using a Cornish-Fisher (special case of a Taylor) expansion. The resulting measure is referred to variously as "Cornish-Fisher VaR" or "Modified VaR". We provide this measure in function VaR with method="modified". Modified VaR produces the same results as traditional mean-VaR when the return distribution is normal, so it may be used as a direct replacement. Many papers in the finance literature have reached the conclusion that Modified VaR is a superior measure, and may be substituted in any case where mean-VaR would previously have been used.

Conditional VaR and Expected Shortfall: We have implemented Conditional Value at Risk, also called Expected Shortfall (not to be confused with shortfall probability, which is much less useful), in function ES. Expected Shortfall attempts to measure the magnitude of the average loss exceeding the traditional mean-VaR. Expected Shortfall has proven to be a reasonable risk predictor for many asset classes. We have provided traditional historical, Gaussian and modified Cornish-Fisher measures of Expected Shortfall by using method="historical", method="gaussian" or method="modified". See *Uryasev(2000)* and *Sherer and Martin(2005)* for more information on Conditional Value at Risk and Expected Shortfall. Please note that your milage will vary; expect that values obtained from the normal distribution may differ radically from the real situation, depending on the assets under analysis.

Multivariate extensions to risk measures: We have extened all moments calculations to work in a multivariate portfolio context. In a portfolio context the multivariate moments are generally to be preferred to their univariate counterparts, so that all information is available to subsequent calculations. Both the VaR and ES functions allow calculation of metrics in a portfolio context when weights and a portfolio_method are passed into the function call.

Marginal, Incremental, and Component VaR: Marginal VaR is the difference between the VaR of the portfolio without the asset in question and the entire portfolio. The VaR function calculates Marginal VaR for all instruments in the portfolio if you set method="marginal". Marginal VaR as provided here may use traditional mean-VaR or Modified VaR for the calculation. Per

Artzner,et.al.(1997) properties of a coherent risk measure include subadditivity (risks of the portfolio should not exceed the sum of the risks of individual components) as a significantly desirable trait. VaR measures, including Marginal VaR, on individual components of a portfolio are *not* subadditive.

Clearly, a general subadditive risk measure for downside risk is required. In Incremental or Component VaR, the Component VaR value for each element of the portfolio will sum to the total VaR of the portfolio. Several EDHEC papers suggest using Modified VaR instead of mean-VaR in the Incremental and Component VaR calculation. We have succeeded in implementing Component VaR and ES calculations that use Modified Cornish-Fisher VaR, historical decomposition, and kernel estimators. You may access these with `VaR` or `ES` by setting the appropriate `portfolio_method` and `method` arguments.

The `chart.VaRSensitivity` function creates a chart of Value-at-Risk or Expected Shortfall estimates by confidence interval for multiple methods. Useful for comparing a calculated VaR or ES method to the historical VaR or ES, it may also be used to visually examine whether the VaR method "breaks down" or gives nonsense results at a certain threshold.

Which VaR measure to use will depend greatly on the portfolio and instruments being analyzed. If there is any generalization to be made on VaR measures, we agree with *Bali and Gokcan(2004)* who conclude that "the VaR estimations based on the generalized Pareto distribution and the Cornish-Fisher approximation perform best".

Moments and Co-moments

Analysis of financial time series often involves evaluating their mathematical moments. While `var` and `cov` for variance has always been available, as well as `skewness` and `kurtosis` (which we have extended to make multivariate and multi-column aware), a larger suite of multivariate moments calculations was not available in R. We have now implemented multivariate moments and co-moments and their beta or systematic co-moments in *`PerformanceAnalytics`*.

Ranaldo and Favre (2005) define coskewness and cokurtosis as the skewness and kurtosis of a given asset analysed with the skewness and kurtosis of the reference asset or portfolio. The co-moments are useful for measuring the marginal contribution of each asset to the portfolio's resulting risk. As such, co-moments of an asset return distribution should be useful as inputs for portfolio optimization in addition to the covariance matrix. Functions include `CoVariance`, `CoSkewness`, `CoKurtosis`.

Measuring the co-moments should be useful for evaluating whether or not an asset is likely to provide diversification potential to a portfolio. But the co-moments do not allow the marginal impact of an asset on a portfolio to be directly measured. Instead, *Martellini and Zieman (2007)* develop a framework that assesses the potential diversification of an asset relative to a portfolio. They use higher moment betas to estimate how much portfolio risk will be impacted by adding an asset.

Higher moment betas are defined as proportional to the derivative of the covariance, coskewness and cokurtosis of the second, third and fourth portfolio moment with respect to the portfolio weights. A beta that is less than 1 indicates that adding the new asset should reduce the resulting portfolio's volatility and kurtosis, and to an increase in skewness. More specifically, the lower the beta the higher the diversification effect, not only in terms of normal risk (i.e. volatility) but also the risk of assymetry (skewness) and extreme events (kurtosis). See the functions for `BetaCoVariance`, `BetaCoSkewness`, and `BetaCoKurtosis`.

Robust Data Cleaning

The functions `Return.clean` and `clean.boudt` implement statistically robust data cleaning methods tuned to portfolio construction and risk analysis and prediction in financial time series while trying to avoid some of the pitfalls of standard robust statistical methods.

The primary value of data cleaning lies in creating a more robust and stable estimation of the distribution generating the large majority of the return data. The increased robustness and stability of the estimated moments using cleaned data should be used for portfolio construction. If an investor wishes to have a more conservative risk estimate, cleaning may not be indicated for risk monitoring.

In actual practice, it is probably best to back-test the out-of-sample results of both cleaned and uncleaned series to see what works best when forecasting risk with the particular combination of assets under consideration.

Summary Tabular Data

Summary statistics are then the necessary aggregation and reduction of (potentially thousands) of periodic return numbers. Usually these statistics are most palatable 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`. Adding benchmarks or peers alongside the annualized data is helpful for comparing returns in calendar years.

When we started this project, we debated whether such tables would be broadly useful or not. No reader is likely to think that we captured the precise statistics to help their decision. We merely offer these as a starting point for creating your own. Add, subtract, do whatever seems useful to you. If you think that your work may be useful to others, please consider sharing it so that we may include it in a future version of this package.

Other tables for comparison of related groupings of statistics discussed elsewhere:

`table.Stats`	Basic statistics and stylized facts
`table.TrailingPeriods`	Statistics and stylized facts compared over different trailing periods
`table.AnnualizedReturns`	Annualized return, standard deviation, and Sharpe ratio
`table.CalendarReturns`	Monthly and calendar year return table
`table.CAPM`	CAPM-related measures
`table.Correlation`	Comparison of correlalations and significance statistics
`table.DownsideRisk`	Downside risk metrics and statistics
`table.Drawdowns`	Ordered list of drawdowns and when they occurred
`table.Autocorrelation`	The first six autocorrelation coefficients and significance
`table.HigherMoments`	Higher co-moments and beta co-moments
`table.Arbitrary`	Combines a function list into a table

Charts and Graphs

Graphs and charts can also help to organize the information visually. Our goal in creating these charts was to simplify the process of creating well-formatted charts that are used often in performance analysis, and to create high-quality graphics that may be used in documents for consumption by non-analysts or researchers. R's graphics capabilities are substantial, but the simplicity of the output of R default graphics functions such as `plot` does not always compare well against graphics delivered with commercial asset or performance analysis from places such as MorningStar or

PerTrac.

The cumulative returns or wealth index is usually the first thing displayed, even though neither conveys much information. See chart.CumReturns. Individual period returns may be helpful for identifying problematic periods, such as in chart.Bar. Risk measures can be helpful when overlaid on the period returns, to display the bounds at which losses may be expected. See chart.BarVaR and the prior section on Risk Analysis. More information can be conveyed when such charts are displayed together, as in charts.PerformanceSummary, which combines the performance data with detail on downside risk (see chart.Drawdown).

The relative performance through time of two assets can be plotted with chart.RelativePerformance. This plot displays the ratio of the cumulative performance at each point in time and makes periods of under- or out-performance easy to see. The value of the chart is less important than the slope of the line. If the slope is positive, the first asset is outperforming the second, and vice verse. Affectionately known as the Canto chart, it was used effectively in Canto (2006).

Two-dimensional charts can also be useful while remaining easy to understand. chart.Scatter is a utility scatter chart with some additional attributes that are used in chart.RiskReturnScatter. Overlaying Sharpe ratio lines or boxplots helps to add information about relative performance along those dimensions.

For distributional analysis, a few graphics may be useful. chart.Boxplot is an example of a graphic that is difficult to create in Excel and is under-utilized as a result. A boxplot of returns is, however, a very useful way to instantly observe the shape of large collections of asset returns in a manner that makes them easy to compare to one another. chart.Histogram and chart.QQPlot are two charts originally found elsewhere and now substantially expanded in *PerformanceAnalytics*.

Rolling performance is typically used as a way to assess stability of a return stream. Although perhaps it doesn't get much credence in the financial literature as it derives from work in digital signal processing, many practitioners find it a useful way to examine and segment performance and risk periods. See chart.RollingPerformance, which is a way to display different metrics over rolling time periods. chart.RollingMean is a specific example of a rolling mean and standard error bands. A group of related metrics is offered in charts.RollingPerformance. These charts use utility functions such as rollapply.

chart.SnailTrail is a scatter chart that shows how rolling calculations of annualized return and annualized standard deviation have proceeded through time where the color of lines and dots on the chart diminishes with respect to time. chart.RollingCorrelation shows how correlations change over rolling periods. chart.RollingRegression displays the coefficients of a linear model fitted over rolling periods. A group of charts in charts.RollingRegression displays alpha, beta, and R-squared estimates in three aligned charts in a single device.

chart.StackedBar creates a stacked column chart with time on the horizontal axis and values in categories. This kind of chart is commonly used for showing portfolio 'weights' through time, although the function will plot any values by category.

We have been greatly inspired by other peoples' work, some of which is on display at http://addictedtor.free.fr/. Particular inspiration came from Dirk Eddelbuettel and John Bollinger for their work at http://addictedtor.free.fr/graphiques/RGraphGallery.php?graph=65. Those interested in price charting in R should also look at the quantmod package.

Wrapper and Utility Functions

R is a very powerful environment for manipulating data. It can also be quite confusing to a user more accustomed to Excel or even MatLAB. As such, we have written some wrapper functions that may aid you in coercing data into the correct forms or finding data that you need to use regularly. To simplify the management of multiple-source data stored in R in multiple data formats, we have provided `checkData`. This function will attempt to coerce data in and out of R's multitude of mostly fungible data classes into the class required for a particular analysis. The data-coercion function has been hidden inside the functions here, but it may also save you time and trouble in your own code and functions as well.

R's built-in `apply` function in enormously powerful, but is can be tricky to use with timeseries data, so we have provided wrapper functions to `apply.fromstart` and `apply.rolling` to make handling of "from inception" and "rolling window" calculations easier.

Further Work

We have attempted to standardize function parameters and variable names, but more work exists to be done here.

Any comments, suggestions, or code patches are invited.

If you've implemented anything that you think would be generally useful to include, please consider donating it for inclusion in a later version of this package.

Acknowledgments

Data series `edhec` used in *`PerformanceAnalytics`* and related publications with the kind permission of the EDHEC Risk and Asset Management Research Center.
`http://www.edhec-risk.com/indexes/pure_style`

Kris Boudt was instrumental in our research on component risk for portfolios with non-normal distributions, and is responsible for much of the code for multivariate moments and co-moments.

Jeff Ryan and Josh Ulrich are active participants in the R finance community and created `xts`, upon which much of PerformanceAnalytics depends.

Prototypes of the drawdowns functionality were provided by Sankalp Upadhyay, and modified with permission. Stephan Albrecht provided detailed feedback on the Getmansky/Lo Smoothing Index. Diethelm Wuertz provided prototypes of modified VaR and skewness and kurtosis functions (and is of course the maintainer of the RMetrics suite of pricing and optimization functions). Any errors are, of course, our own.

Thanks to Joe Wayne Byers and Dirk Eddelbuettel for comments on early versions of these functions, and to Khanh Nguyen and Ryan Sheftel for careful testing and detailed problem reports.

Thanks to the R-SIG-Finance community without whom this package would not be possible. We are indebted to the R-SIG-Finance community for many helpful suggestions, bugfixes, and requests.

Author(s)

Brian G. Peterson
Peter Carl

Maintainer: Brian G. Peterson ⟨brian@braverock.com⟩

References

Amenc, N. and Le Sourd, V. *Portfolio Theory and Performance Analysis*. Wiley. 2003.
Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004.
Canto, V. *Understanding Asset Allocation*. FT Prentice Hall. 2006.
Lhabitant, F. *Hedge Funds: Quantitative Insights*. Wiley. 2004.
Litterman, R., Gumerlock R., et. al. *The Practice of Risk Management: Implementing Processes for Managing Firm-Wide Market Risk*. Euromoney. 1998.
Martellini, Lionel, and Volker Ziemann. *Improved Forecasts of Higher-Order Comoments and Implications for Portfolio Selection.* EDHEC Risk and Asset Management Research Centre working paper. 2007.
Ranaldo, Angelo, and Laurent Favre Sr. *How to Price Hedge Funds: From Two- to Four-Moment CAPM*. SSRN eLibrary. 2005. Murrel, P. *R Graphics*. Chapman and Hall. 2006.
Ruppert, D. *Statistics and Finance, an Introduction*. Springer. 2004.
Scherer, B. and Martin, D. *Modern Portfolio Optimization*. Springer. 2005.
Shumway, R. and Stoffer, D. *Time Series Analysis and it's Applications, with R examples*, Springer, 2006.
Tsay, R. *Analysis of Financial Time Series*. Wiley. 2001.
Zin, Markowitz, Zhao (2006) http://papers.ssrn.com/sol3/papers.cfm?abstract_id=910640
Zivot, E. and Wang, Z. *Modeling Financial Time Series with S-Plus: second edition*. Springer. 2006.

See Also

CRAN task view on Empirical Finance
http://cran.r-project.org/src/contrib/Views/Econometrics.html

Grant Farnsworth's Econometrics in R
http://cran.r-project.org/doc/contrib/Farnsworth-EconometricsInR.pdf

Collection of R charts and graphs
http://addictedtor.free.fr/graphiques/

`ActivePremium` *Active Premium*

Description

The return on an investment's annualized return minus the benchmark's annualized return.

Active Premium = Investment's annualized return - Benchmark's annualized return

Usage

```
ActivePremium(Ra, Rb, scale = NA)
```

Arguments

Ra	return vector of the portfolio
Rb	return vector of the benchmark asset
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)

Author(s)

Peter Carl

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.

See Also

InformationRatio TrackingError Return.annualized

Examples

```
data(managers)
ActivePremium(managers[, "HAM1", drop=FALSE], managers[, "SP500.TR", drop=FALSE])
ActivePremium(managers[,1,drop=FALSE], managers[,8,drop=FALSE])
ActivePremium(managers[,1:6], managers[,8,drop=FALSE])
ActivePremium(managers[,1:6], managers[,8:7,drop=FALSE])
```

---

apply.fromstart *calculate a function over an expanding window always starting from the beginning of the series*

Description

A function to calculate a function over an expanding window from the start of the timeseries. This wrapper allows easy calculation of "from inception" statistics.

Usage

```
apply.fromstart(R, FUN = "mean" , gap = 1, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
FUN	any function that can be evaluated using a single set of returns (e.g., rolling beta won't work, but Return.annualized will)
gap	the number of data points from the beginning of the series required to "train" the calculation
...	any other passthru parameters

Author(s)

Peter Carl

See Also

rollapply

Examples

```
data(managers)
apply.fromstart(managers[,1,drop=FALSE], FUN="mean", width=36)
```

---

apply.rolling *calculate a function over a rolling window*

Description

Creates a results timeseries of a function applied over a rolling window.

Wrapper function for rollapply to hide some of the complexity of managing single-column zoo objects.

Usage

```
apply.rolling(R, width, trim = TRUE, gap = 12, by = 1, FUN = "mean", ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
width	number of periods to apply rolling function window over
gap	numeric number of periods from start of series to use to train risk calculation
trim	TRUE/FALSE, whether to keep alignment caused by NA's
FUN	any function that can be evaluated using a single set of returns (e.g., rolling beta won't work, but Return.annualized will)
by	calculate FUN for trailing width points at every by-th time point.
...	any other passthru parameters

Value

A timeseries in a zoo object of the calculation results

Author(s)

Peter Carl

See Also

apply
rollapply

Examples

```
data(managers)
apply.rolling(managers[,1,drop=FALSE], FUN="mean", width=36)
```

Beta Co-Moments *Functions to calculate systematic or beta co-moments of return series*

Description

calculate higher co-moment betas, or 'systematic' variance, skewness, and kurtosis

Usage

```
BetaCoVariance(Ra, Rb)
BetaCoSkewness(Ra, Rb, test = FALSE)
BetaCoKurtosis(Ra, Rb)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	an xts, vector, matrix, data frame, timeSeries or zoo object of index, benchmark, or secondary asset returns to compare against
test	condition not implemented yet

Details

The co-moments, including covariance, coskewness, and cokurtosis, do not allow the marginal impact of an asset on a portfolio to be directly measured. Instead, Martellini and Zieman (2007) develop a framework that assesses the potential diversification of an asset relative to a portfolio. They use higher moment betas to estimate how much portfolio risk will be impacted by adding an asset, in terms of symmetric risk (i.e., volatility), in asymmetry risk (i.e., skewness), and extreme risks (i.e. kurtosis). That allows them to show that adding an asset to a portfolio (or benchmark) will reduce the portfolio's variance to be reduced if the second-order beta of the asset with respect to the portfolio is less than one. They develop the same concepts for the third and fourth order moments. The authors offer these higher moment betas as a measure of the diversification potential of an asset.

Higher moment betas are defined as proportional to the derivative of the covariance, coskewness and cokurtosis of the second, third and fourth portfolio moment with respect to the portfolio weights. The beta co-variance is calculated as:

$$\beta_{a,b}^{(2)} = \frac{CoV(R_a, R_b)}{\mu^{(2)}(R_b)}$$

Beta co-skewness is given as:

$$\beta_{a,b}^{(3)} = \frac{CoS(R_a, R_b)}{\mu^{(3)}(R_b)}$$

Beta co-kurtosis is:

$$\beta_{a,b}^{(4)} = \frac{CoK(R_a, R_b)}{\mu^{(4)}(R_b)}$$

where the n-th centered moment is calculated as

$$\mu^{(n)}(R) = E[(R - E(R))^n]$$

A beta is greater than one indicates that no diversification benefits should be expected from the introduction of that asset into the portfolio. Conversely, a beta that is less than one indicates that adding the new asset should reduce the resulting portfolio's volatility and kurtosis, and to an increase in skewness. More specifically, the lower the beta the higher the diversification effect on normal risk (i.e. volatility). Similarly, since extreme risks are generally characterised by negative skewness and positive kurtosis, the lower the beta, the higher the diversification effect on extreme risks (as reflected in Modified Value-at-Risk or ER).

The addition of a small fraction of a new asset to a portfolio leads to a decrease in the portfolio's second moment (respectively, an increase in the portfolio's third moment and a decrease in the portfolio's fourth moment) if and only if the second moment (respectively, the third moment and fourth moment) beta is less than one (see Martellini and Ziemann (2007) for more details).

For skewness, the interpretation is slightly more involved. If the skewness of the portfolio is negative, we would expect an increase in portfolio skewness when the third moment beta is lower than one. When the skewness of the portfolio is positive, then the condition is that the third moment beta is greater than, as opposed to lower than, one.

Author(s)

Kris Boudt, Peter Carl, Brian Peterson

References

Boudt, Kris, Brian G. Peterson, and Christophe Croux. 2008. Estimation and Decomposition of Downside Risk for Portfolios with Non-Normal Returns. Journal of Risk. Winter.

Martellini, Lionel, and Volker Ziemann. 2007. Improved Forecasts of Higher-Order Comoments and Implications for Portfolio Selection. EDHEC Risk and Asset Management Research Centre working paper.

See Also

CoMoments

Examples

```
data(managers)

BetaCoVariance(managers[, "HAM2", drop=FALSE], managers[, "SP500 TR", drop=FALSE])
BetaCoSkewness(managers[, "HAM2", drop=FALSE], managers[, "SP500 TR", drop=FALSE])
```

```
BetaCoKurtosis(managers[, "HAM2", drop=FALSE], managers[, "SP500 TR", drop=FALSE])
BetaCoKurtosis(managers[,1:6], managers[,8,drop=FALSE])
BetaCoKurtosis(managers[,1:6], managers[,8:7])
```

`CalmarRatio` *calculate a Calmar or Sterling reward/risk ratio*

Description

Calmar and Sterling Ratios are yet another method of creating a risk-adjusted measure for ranking investments similar to the `SharpeRatio`.

Usage

```
CalmarRatio(R, scale = NA)
SterlingRatio(R, scale = NA, excess=.1)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`scale`	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
`excess`	for Sterling Ratio, excess amount to add to the max drawdown, traditionally and default .1 (10%)

Details

Both the Calmar and the Sterling ratio are the ratio of annualized return over the absolute value of the maximum drawdown of an investment. The Sterling ratio adds an excess risk measure to the maximum drawdown, traditionally and defaulting to 10%.

It is also traditional to use a three year return series for these calculations, although the functions included here make no effort to determine the length of your series. If you want to use a subset of your series, you'll need to truncate or subset the input data to the desired length.

Many other measures have been proposed to do similar reward to risk ranking. It is the opinion of this author that newer measures such as Sortino's `UpsidePotentialRatio` or Favre's `SharpeRatio.modified` are both "better" measures, and should be preferred to the Calmar or Sterling Ratio.

Author(s)

Brian G. Peterson

References

Bacon, Carl. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004.

See Also

Return.annualized,
maxDrawdown,
SharpeRatio.modified,
UpsidePotentialRatio

Examples

```
data(managers)
CalmarRatio(managers[,1,drop=FALSE])
CalmarRatio(managers[,1:6])
SterlingRatio(managers[,1,drop=FALSE])
SterlingRatio(managers[,1:6])
```

CAPM.alpha *calculate CAPM alpha*

Description

This is a wrapper for calculating a CAPM alpha.

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

Usage

```
CAPM.alpha(Ra, Rb, Rf = 0)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
Rf	risk free rate, in same period as your returns

Author(s)

Peter Carl

References

Sharpe, W.F. Capital Asset Prices: A theory of market equilibrium under conditions of risk. *Journal of finance*, vol 19, 1964, 425-442.
Ruppert, David. *Statistics and Finance, an Introduction*. Springer. 2004.

See Also

CAPM.beta CAPM.utils

Examples

```
# First we load the data
    data(managers)
    CAPM.alpha(managers[,1,drop=FALSE], managers[,8,drop=FALSE], Rf=.035/12)
    CAPM.alpha(managers[,1,drop=FALSE], managers[,8,drop=FALSE], Rf = managers[,10,drop=FALS
    CAPM.alpha(managers[,1:6], managers[,8,drop=FALSE], Rf=.035/12)
    CAPM.alpha(managers[,1:6], managers[,8,drop=FALSE], Rf = managers[,10,drop=FALSE])
    CAPM.alpha(managers[,1:6], managers[,8:7,drop=FALSE], Rf=.035/12)
    CAPM.alpha(managers[,1:6], managers[,8:7,drop=FALSE], Rf = managers[,10,drop=FALSE])
```

CAPM.beta *calculate CAPM beta*

Description

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

This function uses a linear intercept model to achieve the same results as the symbolic model used by BetaCoVariance

Usage

```
CAPM.beta(Ra, Rb, Rf = 0)
CAPM.beta.bull(Ra, Rb, Rf = 0)
CAPM.beta.bear(Ra, Rb, Rf = 0)
TimingRatio(Ra, Rb, Rf = 0)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
Rf	risk free rate, in same period as your returns

Details

$$\beta_{a,b} = \frac{CoV_{a,b}}{\sigma_a} = \frac{\sum((R_a - \bar{R_a})(R_b - \bar{R_b}))}{\sum(R_a - \bar{R_a})^2}$$

Ruppert(2004) reports that this equation will give the estimated slope of the linear regression of R_a on R_b and that this slope can be used to determine the risk premium or excess expected return (see Eq. 7.9 and 7.10, p. 230-231).

Two other functions apply the same notion of best fit to positive and negative market returns, separately. The CAPM.beta.bull is a regression for only positive market returns, which can be used to understand the behavior of the asset or portfolio in positive or 'bull' markets. Alternatively, CAPM.beta.bear provides the calculation on negative market returns.

The TimingRatio can help assess whether the manager is a good timer of asset allocation decisions. The ratio, which is calculated as

$$TimingRatio = \frac{\beta^+}{\beta^-}$$

is best when greater than one in a rising market and less than one in a falling market.

Author(s)

Peter Carl

References

Sharpe, W.F. Capital Asset Prices: A theory of market equilibrium under conditions of risk. *Journal of finance*, vol 19, 1964, 425-442.
Ruppert, David. *Statistics and Finance, an Introduction*. Springer. 2004.
Bacon, Carl. *Practical portfolio performance measurement and attribution*. Wiley. 2004.

See Also

BetaCoVariance CAPM.alpha CAPM.utils

Examples

```
data(managers)
    CAPM.alpha(managers[,1,drop=FALSE], managers[,8,drop=FALSE], Rf=.035/12)
    CAPM.alpha(managers[,1,drop=FALSE], managers[,8,drop=FALSE], Rf = managers[,10,drop=FALS
    CAPM.alpha(managers[,1:6], managers[,8,drop=FALSE], Rf=.035/12)
    CAPM.alpha(managers[,1:6], managers[,8,drop=FALSE], Rf = managers[,10,drop=FALSE])
    CAPM.alpha(managers[,1:6], managers[,8:7,drop=FALSE], Rf=.035/12)
    CAPM.alpha(managers[,1:6], managers[,8:7,drop=FALSE], Rf = managers[,10,drop=FALSE])
    CAPM.beta(managers[, "HAM2", drop=FALSE], managers[, "SP500 TR", drop=FALSE], Rf = manag
    CAPM.beta.bull(managers[, "HAM2", drop=FALSE], managers[, "SP500 TR", drop=FALSE], Rf =
    CAPM.beta.bear(managers[, "HAM2", drop=FALSE], managers[, "SP500 TR", drop=FALSE], Rf =
    TimingRatio(managers[, "HAM2", drop=FALSE], managers[, "SP500 TR", drop=FALSE], Rf = man
    chart.Regression(managers[, "HAM2", drop=FALSE], managers[, "SP500 TR", drop=FALSE], Rf
```

CAPM.utils *utility functions for CAPM CML, SML, and RiskPremium*

Description

The Capital Asset Pricing Model, from which the popular SharpeRatio is derived, is a theory of market equilibrium. These utility functions provide values for various measures proposed in the CAPM.

Usage

```
CAPM.CML.slope(Rb, Rf = 0 )
CAPM.CML(Ra, Rb, Rf = 0)
CAPM.RiskPremium(Ra, Rf = 0)
CAPM.SML.slope(Rb, Rf = 0)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
Rf	risk free rate, in same period as your returns

Details

The CAPM provides a justification for passive or index investing by positing that assets that are not on the efficient frontier will either rise or lower in price until they are on the efficient frontier of the market portfolio.

The CAPM Risk Premium on an investment is the measure of how much the asset's performance differs from the risk free rate. Negative Risk Premium generally indicates that the investment is a bad investment, and the money should be allocated to the risk free asset or to a different asset with a higher risk premium.

The Capital Market Line relates the excess expected return on an efficient market portfolio to it's Risk. The slope of the CML is the Sharpe Ratio for the market portfolio. The Security Market line is constructed by calculating the line of Risk Premium over `CAPM.beta`. For the benchmark asset this will be 1 over the risk premium of the benchmark asset. The CML also describes the only path allowed by the CAPM to a portfolio that outperforms the efficient frontier: it describes the line of reward/risk that a leveraged portfolio will occupy. So, according to CAPM, no portfolio constructed of the same assets can lie above the CML.

Probably the most complete criticism of CAPM in actual practice (as opposed to structural or theory critiques) is that it posits a market equilibrium, but is most often used only in a partial equilibrium setting, for example by using the S&P 500 as the benchmark asset. A better method of using and testing the CAPM would be to use a general equilibrium model that took global assets from all asset classes into consideration.

Chapter 7 of Ruppert(2004) gives an extensive overview of CAPM, its assumptions and deficiencies.

`CAPM.RiskPremium` is the premium returned to the investor over the risk free asset

$$\overline{(R_a - R_f)}$$

`CAPM.CML` calculates the expected return of the asset against the benchmark Capital Market Line

`CAPM.CML.slope` calculates the slope of the Capital Market Line for looking at how a particular asset compares to the CML

`CAPM.SML.slope` calculates the slope of the Security Market Line for looking at how a particular asset compares to the SML created by the benchmark

Author(s)

Brian G. Peterson

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.
Sharpe, W.F. Capital Asset Prices: A theory of market equilibrium under conditions of risk. *Journal of finance*, vol 19, 1964, 425-442.
Ruppert, David. *Statistics and Finance, an Introduction*. Springer. 2004.

See Also

CAPM.beta CAPM.alpha SharpeRatio InformationRatio TrackingError ActivePremium

Examples

```
data(managers)
CAPM.CML.slope(managers["SP500 TR",drop=FALSE], managers[,10,drop=FALSE])
CAPM.CML(managers["HAM1",drop=FALSE], managers["SP500 TR",drop=FALSE], Rf=0)
CAPM.RiskPremium(managers["SP500 TR",drop=FALSE], Rf=0)
CAPM.RiskPremium(managers["HAM1",drop=FALSE], Rf=0)
CAPM.SML.slope(managers["SP500 TR",drop=FALSE], Rf=0)
# should create plots like in Ruppert 7.1 7.2
```

---

chart.ACF *Create ACF chart or ACF with PACF two-panel chart*

Description

Creates an ACF chart or a two-panel plot with the ACF and PACF set to some specific defaults.

Usage

```
chart.ACF(R, maxlag = NULL, elementcolor = "gray", main = NULL, ...)
chart.ACFplus(R, maxlag = NULL, elementcolor = "gray", main = NULL, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
maxlag	the number of lags to calculate for, optional
elementcolor	the color to use for chart elements, defaults to "gray"
main	title of the plot; uses the column name by default.
...	any other passthru parameters

Note

Inspired by the website: http://www.stat.pitt.edu/stoffer/tsa2/Rcode/acf2.R "...here's an R function that will plot the ACF and PACF of a time series at the same time on the SAME SCALE, and it leaves out the zero lag in the ACF: acf2.R. If your time series is in x and you want the ACF and PACF of x to lag 50, the call to the function is acf2(x,50). The number of lags is optional, so acf2(x) will use a default number of lags [sqrt(n) + 10, where n is the number of observations]."

That description made a lot of sense, so it's implemented here for both the ACF alone and the ACF with the PACF.

Author(s)

Peter Carl

See Also

plot

Examples

```
data(edhec)
chart.ACFplus(edhec[,1,drop=FALSE])
```

---

chart.Bar *wrapper for barchart of returns*

Description

A wrapper to create a chart of periodic returns in a bar chart. This is a difficult enough graph to read that it doesn't get much use. Still, it is useful for viewing a single set of data.

Usage

```
chart.Bar(R, legend.loc = NULL, colorset = (1:12), ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center
colorset	color palette to use, set by default to rational choices
...	any other passthru parameters, see plot

Details

This is really a wrapper for chart.TimeSeries, so several other attributes can also be passed.

Creates a plot of time on the x-axis and vertical lines for each period to indicate value on the y-axis.

Author(s)

Peter Carl

See Also

chart.TimeSeries
plot

Examples

```
data(edhec)
chart.Bar(edhec[,"Funds of Funds"], main="Monthly Returns")
```

---

chart.BarVaR *Periodic returns in a bar chart with risk metric overlay*

Description

Plots the periodic returns in a bar chart overlayed with a risk metric calculation.

Usage

```
chart.BarVaR(R, width = 0, gap = 12, methods = c("none", "ModifiedVaR", "GaussianVa
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
width	periods specified for rolling-period calculations. Note that VaR and Std Dev with width=0 are calculated from the start of the timeseries
gap	numeric number of periods from start of series to use to train risk calculation
methods	any of *StdDev*, *HistoricalVaR*, *GaussianVaR*, or *ModifiedVaR*, defaults to HistoricalVaR and ModifiedVaR
p	confidence level for VaR or ModifiedVaR calculation, default is .99
ylim	set the y-axis limit, same as in plot
lwd	set the line width, same as in plot
lty	set the line type, same as in plot
all	if TRUE, calculates risk lines for each column given in R. If FALSE, only calculates the risk line for the first column
show.clean	if TRUE and a method for 'clean' is specified, overlays the actual data with the "cleaned" data. See Return.clean for more detail
clean	the method to use to clean outliers from return data prior to risk metric estimation. See Return.clean and VaR for more detail
show.horizontal	if TRUE, shows a line across the timeseries at the value of the most recent VaR estimate, to help the reader evaluate the number of exceptions thus far

show.symmetric
: if TRUE and the metric is symmetric, this will show the metric's positive values as well as negative values, such as for method "StdDev".

legend.loc
: legend location, such as in chart.TimeSeries

ypad
: adds a numerical padding to the y-axis to keep the data away when legend.loc="bottom". See examples below.

legend.cex
: sets the legend text size, such as in chart.TimeSeries

colorset
: color palette to use, set by default to rational choices

...
: any other passthru parameters to chart.TimeSeries

Details

Note that StdDev and VaR are symmetric calculations, so a high and low measure will be plotted. ModifiedVaR, on the other hand, is assymetric and only a lower bound will be drawn.

Creates a plot of time on the x-axis and vertical lines for each period to indicate value on the y-axis. Overlays a line to indicate the value of a risk metric calculated at that time period.

Author(s)

Peter Carl

See Also

chart.TimeSeries
plot
VaR
VaR
Return.clean

Examples

```
data(managers)
# plain
chart.BarVaR(managers[,1,drop=FALSE], main="Monthly Returns")

# with risk line
chart.BarVaR(managers[,1,drop=FALSE], methods="HistoricalVaR", main="... with Empirical VaR

# with lines for all managers in the sample
chart.BarVaR(managers[,1:6], methods="GaussianVaR", all=TRUE, lty=1, lwd=2, colorset= c("red

# with multiple methods
chart.BarVaR(managers[,1,drop=FALSE],methods=c("HistoricalVaR", "ModifiedVaR", "GaussianVaR"

# cleaned up a bit
chart.BarVaR(managers[,1,drop=FALSE],methods=c("HistoricalVaR", "ModifiedVaR", "GaussianVaR"

# with 'cleaned' data for VaR estimates
chart.BarVaR(managers[,1,drop=FALSE],methods=c("HistoricalVaR", "ModifiedVaR"), lwd=2, ypad=
```

```
# Cornish Fisher VaR estimated with cleaned data, with horizontal line to show exceptions
chart.BarVaR(managers[,1,drop=FALSE],methods="ModifiedVaR", lwd=2, ypad=.01, clean="boudt",
```

`chart.Boxplot` *box whiskers plot wrapper*

Description

A wrapper to create box and whiskers plot with some defaults useful for comparing distributions.

Usage

```
chart.Boxplot(R, horizontal = TRUE, names = TRUE, as.Tufte = FALSE, sort.by = c(NUI
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`horizontal`	TRUE/FALSE plot horizontal (TRUE) or vertical (FALSE)
`names`	logical. if TRUE, show the names of each series
`as.Tufte`	logical. default FALSE. if TRUE use method derived for Tufte for limiting chartjunk
`sort.ascending`	logical. If TRUE sort the distributions by ascending `sort.by`
`sort.by`	one of "NULL", "mean", "median", "variance"
`colorset`	color palette to use, set by default to rational choices
`symbol.color`	draws the symbols described in `mean.symbol`,`median.symbol`,`outlier.symbol` in the color specified
`mean.symbol`	symbol to use for the mean of the distribution
`median.symbol`	symbol to use for the median of the distribution
`outlier.symbol`	symbol to use for the outliers of the distribution
`show.data`	numerical vector of column numbers to display on top of boxplot, default NULL
`add.mean`	logical. if TRUE, show a line for the mean of all distributions plotted
`xlab`	set the x-axis label, same as in `plot`
`main`	set the chart title, same as in `plot`
`element.color`	specify the color of chart elements. Default is "darkgray"
`...`	any other passthru parameters

Details

We have also provided controls for all the symbols and lines in the chart. One default, set by `as.Tufte=TRUE`, will strip chartjunk and draw a Boxplot per recommendations by Edward Tufte. It can also be useful when comparing several series to sort them in order of ascending or descending "mean", "median", "variance" by use of `sort.by` and `sort.ascending=TRUE`.

Value

box plot of returns

Author(s)

Peter Carl

References

Tufte, Edward R. *The Visual Display of Quantitative Information*. Graphics Press. 1983. p. 124-129

See Also

`boxplot`

Examples

```
data(edhec)
chart.Boxplot(edhec)
chart.Boxplot(edhec,as.Tufte=TRUE)
```

---

chart.CaptureRatios

Chart of Capture Ratios against a benchmark

Description

Scatter plot of Up Capture versus Down Capture against a benchmark

Usage

```
chart.CaptureRatios(Ra, Rb, main = "Capture Ratio", add.names = TRUE, xlab = "Downs
```

Arguments

`Ra`	Returns to test, e.g., the asset to be examined
`Rb`	Returns of a benchmark to compare the asset with
`main`	Set the chart title, same as in `plot`
`add.names`	Plots the row name with the data point. Default TRUE. Can be removed by setting it to NULL

xlab	Set the x-axis label, as in plot
ylab	Set the y-axis label, as in plot
colorset	Color palette to use, set by default to "black"
symbolset	From pch in plot. Submit a set of symbols to be used in the same order as the data sets submitted
legend.loc	Places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
xlim	set the x-axis limit, same as in plot
ylim	set the y-axis limit, same as in plot
cex.axis	The magnification to be used for axis annotation relative to the current setting of 'cex', same as in plot.
cex.legend	The magnification to be used for sizing the legend relative to the current setting of 'cex'.
cex.main	The magnification to be used for sizing the title relative to the current setting of 'cex'.
cex.lab	The magnification to be used for x and y labels relative to the current setting of 'cex'.
element.color	Specify the color of the box, axes, and other chart elements. Default is "darkgray"
benchmark.color	Specify the color of the benchmark reference and crosshairs. Default is "darkgray"
...	Any other passthru parameters to plot

Details

Scatter plot shows the coordinates of each set of returns' Up and Down Capture against a benchmark. The benchmark value is by definition plotted at (1,1) with solid crosshairs. A diagonal dashed line with slope equal to 1 divides the plot into two regions: above that line the UpCapture exceeds the DownCapture, and vice versa.

Author(s)

Peter Carl

See Also

plot,
par,
UpDownRatios,
table.UpDownRatios

Examples

```
data(managers)
chart.CaptureRatios(managers[,1:6], managers[,7,drop=FALSE])
```

---

chart.Correlation *correlation matrix chart*

Description

Visualization of a Correlation Matrix. On top the (absolute) value of the correlation plus the result of the cor.test as stars. On botttom, the bivariate scatterplots, with a fitted line

Usage

```
chart.Correlation(R, histogram = TRUE, ...)
```

Arguments

R	data for the x axis, can take matrix,vector, or timeseries
histogram	TRUE/FALSE whether or not to display a histogram
...	any other passthru parameters into pairs

Note

based on plot at http://addictedtor.free.fr/graphiques/sources/source_137.R

Author(s)

Peter Carl

See Also

table.Correlation

Examples

```
data(managers)
chart.Correlation(managers[,1:8], histogram=TRUE, pch="+")
```

`chart.CumReturns` *Cumulates and graphs a set of periodic returns*

Description

Chart that cumulates the periodic returns given and draws a line graph of the results as a "wealth index".

Usage

```
chart.CumReturns(R, wealth.index = FALSE, legend.loc = NULL, colorset =(1:12), begi
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`wealth.index`	if `wealth.index` is `TRUE`, shows the "value of $1", starting the cumulation of returns at 1 rather than zero
`legend.loc`	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
`colorset`	color palette to use, set by default to rational choices
`begin`	Align shorter series to: • first - prior value of the first column given for the reference or longer series or, • axis - the initial value (1 or zero) of the axis.
`...`	any other passthru parameters

Details

Cumulates the return series and displays either as a wealth index or as cumulative returns.

Author(s)

Peter Carl

References

Bacon, Carl. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004.

See Also

`chart.TimeSeries`
`plot`

Examples

```
data(edhec)
chart.CumReturns(edhec[,"Funds of Funds",drop=FALSE],main="Cumulative Returns")
chart.CumReturns(edhec[,"Funds of Funds",drop=FALSE],wealth.index=TRUE, main="Value of \$1")
```

---

chart.Drawdown *Time series chart of drawdowns through time*

Description

A time series chart demonstrating drawdowns from peak equity attained through time, calculated from periodic returns.

Usage

```
chart.Drawdown(R, legend.loc = NULL, colorset = (1:12), ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
colorset	color palette to use, set by default to rational choices
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
...	any other passthru parameters

Details

Any time the cumulative returns dips below the maximum cumulative returns, it's a drawdown. Drawdowns are measured as a percentage of that maximum cumulative return, in effect, measured from peak equity.

Author(s)

Peter Carl

See Also

plot
chart.TimeSeries
findDrawdowns
sortDrawdowns
maxDrawdown
table.Drawdowns
table.DownsideRisk

Examples

```
data(edhec)
chart.Drawdown(edhec[,c(1,2)], main="Drawdown from Peak Equity Attained", legend.loc="bottom
```

---

chart.ECDF *Create an ECDF overlaid with a Normal CDF*

Description

Creates an emperical cumulative distribution function (ECDF) overlaid with a cumulative distribution function (CDF)

Usage

```
chart.ECDF(R, main = "Empirical CDF", xlab = "x", ylab = "F(x)", colorset = c("blac
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
main	set the chart title, same as in plot
xlab	set the x-axis label, same as in plot
ylab	set the y-axis label, same as in plot
colorset	color palette to use, defaults to c("black", "#005AFF"), where first value is used to color the step function and the second color is used for the fitted normal
lwd	set the line width, same as in plot
xlim	set the x-axis limit, same as in plot
ylim	set the y-axis limit, same as in plot
element.color	specify the color of chart elements. Default is "darkgray"
lty	set the line type, same as in plot
...	any other passthru parameters to plot

Details

The empirical cumulative distribution function (ECDF for short) calculates the fraction of observations less or equal to a given value. The resulting plot is a step function of that fraction at each observation. This function uses ecdf and overlays the CDF for a fitted normal function as well. Inspired by a chart in Ruppert (2004).

Author(s)

Peter Carl

References

Ruppert, David. *Statistics and Finance, an Introduction*. Springer. 2004. Ch. 2 Fig. 2.5

http://www.stat.tamu.edu/~ljin/Finance/chapter2/Fig2_5.txt

See Also

plot, ecdf

Examples

```
data(edhec)
chart.ECDF(edhec[, 1, drop=FALSE])
```

---

chart.Events *Plots a time series with event dates aligned*

Description

Creates a time series plot where events given by a set of dates are aligned, with the adjacent prior and posterior time series data plotted in order. The x-axis is time, but relative to the date specified, e.g., number of months preceeding or following the events.

Usage

```
chart.Events(R, dates, prior = 12, post = 12, main = NULL, xlab = NULL, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
dates	a list of dates (e.g., `c("09/03","05/06")`) formatted the same as in R. This function matches the re-formatted row or index names (dates) with the given list, so to get a match the formatting needs to be correct.
prior	the number of periods to plot prior to the event. Interpreted as a positive number.
post	the number of periods to plot following to the event. Interpreted as a positive number.
main	set the chart title, same as in plot
xlab	set the x-axis label, same as in plot
...	any other passthru parameters to the plot function

Details

This is a chart that is commonly used for event studies in econometrics, usually with recession dates, to demonstrate the path of a time series going into and coming out of an event. The time axis is simply the number of periods prior and following the event, and each line represents a different event. Note that if time periods are close enough together and the window shown is wide enough, the data will appear to repeat. That can be confusing, but the function does not currently allow for different windows around each event.

Author(s)

Peter Carl

See Also

chart.TimeSeries,
plot,
par

Examples

```
data(managers)
R = Drawdowns(managers[,2,drop=FALSE])
n = table.Drawdowns(managers[,2,drop=FALSE])
chart.Events(Drawdowns(managers[,2,drop=FALSE]), dates = n$Trough, prior=max(na.omit(n$"To T
```

---

chart.Histogram *histogram of returns*

Description

Create a histogram of returns, with optional curve fits for density and normal. This is a wrapper function for hist, see the help for that function for additional arguments you may wish to pass in.

Usage

```
chart.Histogram(R, breaks="FD", main = NULL, xlab = "Returns", ylab = "Frequency",
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
breaks	one of: • a vector giving the breakpoints between histogram cells, • a single number giving the number of cells for the histogram, • a character string naming an algorithm to compute the number of cells (see 'Details'), • a function to compute the number of cells. In the last three cases the number is a suggestion only. see hist for details, default "FD"
methods	what to graph, one or more of:
add.density	to display the density plot
add.normal	to display a fitted normal distibution line over the mean
add.centered	to display a fitted normal line over zero
add.rug	to display a rug of the observations
add.risk	to display common risk metrics

add.qqplot to display a small qqplot in the upper corner of the histogram plot

p confidence level for calculation, default p=.99

probability logical; if TRUE, the histogram graphic is a representation of frequencies, the counts component of the result; if FALSE, probability densities, component density, are plotted (so that the histogram has a total area of one). Defaults to TRUE if and only if breaks are equidistant (and probability is not specified). see hist

show.outliers logical; if TRUE (the default), the histogram will show all of the data points. If FALSE, it will show only the first through the fourth quartile and will exclude outliers.

main set the chart title, same as in plot

ylab set the y-axis label, same as in plot

xlab set the x-axis label, same as in plot

ylim set the y-axis limits, same as in plot

border.col color to use for the border

xlim set the x-axis limit, same as in plot

lwd set the line width, same as in plot

colorset color palette to use, set by default to rational choices

element.color provides the color for drawing chart elements, such as the box lines, axis lines, etc. Default is "darkgray"

note.lines draws a vertical line through the value given.

note.labels adds a text label to vertical lines specified for note.lines.

note.cex The magnification to be used for note line labels relative to the current setting of 'cex'.

note.color specifies the color(s) of the vertical lines drawn.

cex.legend The magnification to be used for sizing the legend relative to the current setting of 'cex'.

cex.axis The magnification to be used for axis annotation relative to the current setting of 'cex', same as in plot.

cex.lab The magnification to be used for x- and y-axis labels relative to the current setting of 'cex'.

cex.main The magnification to be used for the main title relative to the current setting of 'cex'.

xaxis if true, draws the x axis

yaxis if true, draws the y axis

... any other passthru parameters to plot

Details

The default for breaks is "FD". Other names for which algorithms are supplied are "Sturges" (see nclass.Sturges), "Scott", and "FD" / "Freedman-Diaconis" (with corresponding functions nclass.scott and nclass.FD). Case is ignored and partial matching is used. Alternatively, a function can be supplied which will compute the intended number of breaks as a function of R.

Note

Code inspired by a chart on:
http://zoonek2.free.fr/UNIX/48_R/03.html

Author(s)

Peter Carl

See Also

hist

Examples

```
data(edhec)
chart.Histogram(edhec[,'Equity Market Neutral',drop=FALSE])

# version with more breaks and the standard close fit density distribution
chart.Histogram(edhec[,'Equity Market Neutral',drop=FALSE], breaks=40, methods = c("add.

chart.Histogram(edhec[,'Equity Market Neutral',drop=FALSE], methods = c( "add.density",

# version with just the histogram and normal distribution centered on 0
chart.Histogram(edhec[,'Equity Market Neutral',drop=FALSE], methods = c( "add.density",

# add a rug to the previous plot for more granularity on precisely where the distributic
chart.Histogram(edhec[,'Equity Market Neutral',drop=FALSE], methods = c( "add.centered",

# now show a qqplot to give us another view on how normal the data are
chart.Histogram(edhec[,'Equity Market Neutral',drop=FALSE], methods = c( "add.centered",

# add risk measure(s) to show where those are in relation to observed returns
chart.Histogram(edhec[,'Equity Market Neutral',drop=FALSE], methods = c("add.density", "
```

---

chart.QQPlot *Plot a QQ chart*

Description

Plot the return data against any theoretical distribution.

Usage

```
chart.QQPlot(R, distribution = "norm", ylab = NULL, xlab = paste(distribution, "Qua
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
distribution	root name of comparison distribution - e.g., 'norm' for the normal distribution; 't' for the t-distribution. See examples for other ideas.
xlab	set the x-axis label, as in plot
ylab	set the y-axis label, as in plot
main	set the chart title, same as in plot
las	set the direction of axis labels, same as in plot
envelope	confidence level for point-wise confidence envelope, or 'FALSE' for no envelope.
labels	vector of point labels for interactive point identification, or 'FALSE' for no labels.
col	color for points and lines; the default is the *second* entry in the current color palette (see 'palette' and 'par').
lwd	set the line width, as in plot
pch	symbols to use, see also plot
cex	symbols to use, see also plot
line	'"quartiles"' to pass a line through the quartile-pairs, or '"robust"' for a robust-regression line; the latter uses the 'rlm' function in the 'MASS' package. Specifying 'line = "none"' suppresses the line.
element.color	provides the color for drawing chart elements, such as the box lines, axis lines, etc. Default is "darkgray"
cex.legend	The magnification to be used for sizing the legend relative to the current setting of 'cex'
cex.axis	The magnification to be used for axis annotation relative to the current setting of 'cex'
cex.lab	The magnification to be used for x- and y-axis labels relative to the current setting of 'cex'
cex.main	The magnification to be used for the main title relative to the current setting of 'cex'.
...	any other passthru parameters to the distribution function

Details

A Quantile-Quantile (QQ) plot is a scatter plot designed to compare the data to the theoretical distributions to visually determine if the observations are likely to have come from a known population. The empirical quantiles are plotted to the y-axis, and the x-axis contains the values of the theorical model. A 45-degree reference line is also plotted. If the empirical data come from the population with the choosen distribution, the points should fall approximately along this reference line. The larger the departure from the reference line, the greater the evidence that the data set have come from a population with a different distribution.

Author(s)

John Fox, ported by Peter Carl

References

main code forked/borrowed/ported from the excellent:
Fox, John (2007) *car: Companion to Applied Regression*
http://www.r-project.org, http://socserv.socsci.mcmaster.ca/jfox/

See Also

qqplot
qq.plot
plot

Examples

```
library(MASS)
data(managers)
x = checkData(managers[,2, drop = FALSE], na.rm = TRUE, method = "vector")
#layout(rbind(c(1,2),c(3,4)))
# Panel 1, Normal distribution
chart.QQPlot(x, main = "Normal Distribution", distribution = 'norm', envelope=0.95)
# Panel 2, Log-Normal distribution
fit = fitdistr(1+x, 'lognormal')
chart.QQPlot(1+x, main = "Log-Normal Distribution", envelope=0.95, distribution='lnorm')#, m
## Not run:
# Panel 3, Skew-T distribution
library(sn)
fit = st.mle(y=x)
chart.QQPlot(x, main = "Skew T Distribution", envelope=0.95, distribution = 'st', location =
#Panel 4: Stable Parietian
library(fBasics)
fit.stable = stableFit(x,doplot=FALSE)
chart.QQPlot(x, main = "Stable Paretian Distribution", envelope=0.95, distribution = 'stable
## End(Not run)
```

---

chart.Regression *Takes a set of returns and relates them to a market benchmark in a scatterplot*

Description

Uses a scatterplot to display the relationship of a set of returns to a market benchmark. Fits a linear model and overlays the resulting model. Also overlays a Loess line for comparison.

Usage

```
chart.Regression(Ra, Rb, Rf, excess.returns = FALSE, reference.grid = TRUE, main =
```

Arguments

Ra	a vector of returns to test, e.g., the asset to be examined
Rb	a matrix, data.frame, or timeSeries of benchmark(s) to test the asset against
Rf	risk free rate, in same period as the returns
excess.returns	logical; should excess returns be used?
reference.grid	if true, draws a grid aligned with the points on the x and y axes
main	set the chart title, same as in plot
ylab	set the y-axis title, same as in plot
xlab	set the x-axis title, same as in plot
xlim	set the x-axis limit, same as in plot
colorset	color palette to use
symbolset	symbols to use, see also 'pch' in plot
element.color	provides the color for drawing chart elements, such as the box lines, axis lines, etc. Default is "darkgray"
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
ylog	Not used
fit	for values of "loess", "linear", or "conditional", plots a line to fit the data. Conditional lines are drawn separately for positive and negative benchmark returns. "Quadratic" is not yet implemented.
span	passed to loess line fit, as in loess.smooth
degree	passed to loess line fit, as in loess.smooth
family	passed to loess line fit, as in loess.smooth
ylim	set the y-axis limit, same as in plot
evaluation	passed to loess line fit, as in loess.smooth
cex	set the cex size, same as in plot
legend.cex	set the legend size
lwd	set the line width for fits, same as in lines
...	any other passthru parameters to plot

Author(s)

Peter Carl

References

Chapter 7 of Ruppert(2004) gives an extensive overview of CAPM, its assumptions and deficiencies.

See Also

plot

Examples

```
data(managers)
chart.Regression(managers[, 1:2, drop = FALSE], managers[, 8, drop = FALSE], Rf = managers[,
```

---

chart.RelativePerformance

relative performance chart between multiple return series

Description

Plots a time series chart that shows the ratio of the cumulative performance for two assets at each point in time and makes periods of under- or out-performance easier to see.

Usage

```
chart.RelativePerformance(Ra, Rb, main = "Relative Performance", xaxis = TRUE, colo
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
main	set the chart title, same as in plot
xaxis	if true, draws the x axis
colorset	color palette to use, set by default to rational choices
elementcolor	provides the color for drawing less-important chart elements, such as the box lines, axis lines, etc. replaces darken
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
ylog	TRUE/FALSE set the y-axis to logarithmic scale, similar to plot, default FALSE
cex.legend	the magnification to be used for sizing the legend relative to the current setting of 'cex'.
lty	set the line type, same as in plot
...	any other passthru parameters

Details

To show under- and out-performance through different periods of time, a time series view is more helpful. The value of the chart is less important than the slope of the line. If the slope is positive, the first asset (numerator) is outperforming the second, and vice versa. May be used to look at the returns of a fund relative to each member of the peer group and the peer group index. Alternatively, it might be used to assess the peers individually against an asset class or peer group index.

Author(s)

Peter Carl

See Also

Return.relative

Examples

```
data(managers)
chart.RelativePerformance(managers[, 1:6, drop=FALSE], managers[, 8, drop=FALSE], colorset=r
```

---

chart.RiskReturnScatter

scatter chart of returns vs risk for comparing multiple instruments

---

Description

A wrapper to create a scatter chart of annualized returns versus annualized risk (standard deviation) for comparing manager performance. Also puts a box plot into the margins to help identify the relative performance quartile.

Usage

```
chart.RiskReturnScatter(R, Rf = 0, main = "Annualized Return and Risk", add.names =
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rf	risk free rate, in same period as your returns
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
geometric	generate geometric (TRUE) or simple (FALSE) returns, default TRUE
main	set the chart title, same as in plot
add.names	plots the row name with the data point. default TRUE. Can be removed by setting it to NULL
xlab	set the x-axis label, as in plot
ylab	set the y-axis label, as in plot
method	if set as "calc", then the function will calculate values from the set of returns passed in. If method is set to "nocalc" then we assume that R is a column of return and a column of risk (e.g., annualized returns, annualized risk), in that order. Other method cases may be set for different risk/return calculations.

add.sharpe	this draws a Sharpe ratio line that indicates Sharpe ratio levels of c(1,2,3). Lines are drawn with a y-intercept of the risk free rate and the slope of the appropriate Sharpe ratio level. Lines should be removed where not appropriate (e.g., sharpe.ratio = NULL).
add.boxplots	TRUE/FALSE adds a boxplot summary of the data on the axis
colorset	color palette to use, set by default to rational choices
symbolset	from pch in plot, submit a set of symbols to be used in the same order as the data sets submitted
element.color	provides the color for drawing chart elements, such as the box lines, axis lines, etc. Default is "darkgray"
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
xlim	set the x-axis limit, same as in plot
ylim	set the y-axis limit, same as in plot
cex.axis	The magnification to be used for axis annotation relative to the current setting of 'cex', same as in plot.
cex.legend	The magnification to be used for sizing the legend relative to the current setting of 'cex'.
cex.main	The magnification to be used for sizing the title relative to the current setting of 'cex'.
cex.lab	The magnification to be used for x and y labels relative to the current setting of 'cex'.
...	any other passthru parameters to plot

Note

Code inspired by a chart on: http://zoonek2.free.fr/UNIX/48_R/03.html

Author(s)

Peter Carl

Examples

```
data(edhec)
chart.RiskReturnScatter(edhec, Rf = .04/12)
chart.RiskReturnScatter(edhec, Rf = .04/12, add.boxplots = TRUE)
```

`chart.RollingCorrelation` *chart rolling correlation fo multiple assets*

Description

A wrapper to create a chart of rolling correlation metrics in a line chart

Usage

```
chart.RollingCorrelation(Ra, Rb, width = 12, xaxis = TRUE, legend.loc = NULL, color
```

Arguments

`Ra`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rb`	return vector of the benchmark asset
`width`	number of periods to apply rolling function window over
`xaxis`	if true, draws the x axis
`legend.loc`	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
`colorset`	color palette to use, set by default to rational choices
`na.pad`	TRUE/FALSE If TRUE it adds any times that would not otherwise have been in the result with a value of NA. If FALSE those times are dropped.
`...`	any other passthru parameters

Author(s)

Peter Carl

Examples

```
# First we get the data
data(managers)
chart.RollingCorrelation(managers[, 1:6, drop=FALSE], managers[, 8, drop=FALSE], colorset=ri
```

chart.RollingMean *chart the rolling mean return*

Description

A wrapper to create a rolling mean return chart with 95% confidence bands.

Usage

```
chart.RollingMean(R, width = 12, xaxis = TRUE, ylim = NULL, na.pad =
FALSE, lwd = c(2, 1, 1), ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
width	number of periods to apply rolling function window over
xaxis	if true, draws the x axis
ylim	set the y-axis limit, same as in plot
na.pad	TRUE/FALSE If TRUE it adds any times that would not otherwise have been in the result with a value of NA. If FALSE those times are dropped.
lwd	set the line width, same as in plot. Specified in order of the main line and the two confidence bands.
...	any other passthru parameters

Author(s)

Peter Carl

Examples

```
data(edhec)
chart.RollingMean(edhec[, 9, drop = FALSE])
```

chart.RollingPerformance *wrapper to create a chart of rolling performance metrics in a line chart*

Description

A wrapper to create a chart of rolling performance metrics in a line chart

Usage

```
chart.RollingPerformance(R, width = 12, xaxis = TRUE, legend.loc = NULL, colorset =
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
width	number of periods to apply rolling function window over
FUN	any function that can be evaluated using a single set of returns (e.g., rolling CAPM.beta won't work, but Return.annualized will)
na.pad	TRUE/FALSE If TRUE it adds any times that would not otherwise have been in the result with a value of NA. If FALSE those times are dropped.
ylog	TRUE/FALSE set the y-axis to logarithmic scale, similar to plot, default FALSE
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
main	set the chart title, same as in plot
event.lines	If not null, vertical lines will be drawn to indicate that an event happened during that time period. event.lines should be a list of dates (e.g., c("09/03","05/06")) formatted the same as date.format. This function matches the re-formatted row names (dates) with the events.list, so to get a match the formatting needs to be correct.
event.labels	if not null and event.lines is not null, this will apply a list of text labels (e.g., c("This Event", "That Event") to the vertical lines drawn
event.color	draws the event described in event.labels in the color specified
period.areas	these are shaded areas described by start and end dates in the same format as the date.format. This is provided as a list of pairs, e.g., list(c("10/26","11/27"), c("08/29","03/33")) See the examples below.
period.color	draws the shaded region described by period.areas in the color specified
type	set the chart type, same as in plot
xaxis	if true, draws the x axis
ylab	set the y-axis label, same as in plot
xlab	set the x-axis label, same as in plot
xlim	set the x-axis limit, same as in plot
ylim	set the y-axis limit, same as in plot
pch	symbols to use, see also plot
lty	set the line type, same as in plot
lwd	set the line width, same as in plot
colorset	color palette to use, set by default to rational choices
bg	same as in plot
cex	set the y-axis limit, same as in plot
log	not used
sub	same as in plot
ann	same as in plot
axes	same as in plot

frame.plot	same as in plot
panel.first	same as in plot
panel.last	same as in plot
asp	same as in plot
cex.axis	The magnification to be used for axis annotation relative to the current setting of 'cex', same as in plot.
cex.legend	The magnification to be used for sizing the legend relative to the current setting of 'cex'.
cex.labels	The magnification to be used for event line labels relative to the current setting of 'cex'.
major.ticks	Should major tickmarks be drawn and labeled, default 'auto'
minor.ticks	Should minor tickmarks be drawn, default TRUE
grid.color	sets the color for the reference grid
grid.lty	defines the line type for the grid
element.color	provides the color for drawing chart elements, such as the box lines, axis lines, etc. Default is "darkgray"
...	any other passthru parameters to rollapply

Author(s)

Peter Carl

See Also

charts.RollingPerformance, rollapply

Examples

```
data(edhec)
chart.RollingPerformance(edhec[, 1:3], width = 24)
chart.RollingPerformance(edhec[, 1:3], FUN = 'mean', width = 24, colorset = rich8equal, lwd
chart.RollingPerformance(edhec[, 1:3], FUN = 'SharpeRatio.annualized', width = 24, colorset
```

---

chart.RollingRegression

A wrapper to create charts of relative regression performance through time

---

Description

A wrapper to create a chart of relative regression performance through time

A group of charts in charts.RollingRegression displays alpha, beta, and R-squared estimates in three aligned charts in a single device.

Usage

```
chart.RollingRegression(Ra, Rb, width = 12, Rf = 0, attribute = c("Beta", "Alpha",
chart.RollingQuantileRegression(Ra, Rb, width = 12, Rf = 0, attribute = c("Beta", "

charts.RollingRegression(Ra, Rb, width = 12, Rf = 0, main = NULL, legend.loc = NULL
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
Rf	risk free rate, in same period as your returns
width	number of periods to apply rolling function window over
attribute	one of "Beta","Alpha","R-Squared" for which attribute to show
main	set the chart title, same as in plot
event.labels	TRUE/FALSE whether or not to display lines and labels for historical market shock events
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
na.pad	TRUE/FALSE If TRUE it adds any times that would not otherwise have been in the result with a value of NA. If FALSE those times are dropped.
...	any other passthru parameters to chart.TimeSeries

Details

The attribute parameter is probably the most confusing. In mathematical terms, the different choices yeild the following:

Alpha - shows the y-intercept
Beta - shows the slope of the regression line
R-Squared - shows the degree of fit of the regression to the data

chart.RollingQuantileRegression uses rq rather than lm for the regression, and may be more robust to outliers in the data.

Note

Most inputs are the same as "plot" and are principally included so that some sensible defaults could be set.

Author(s)

Peter Carl

See Also

lm
rq

Examples

```
# First we load the data
data(managers)
chart.RollingRegression(managers[, 1, drop=FALSE], managers[, 8, drop=FALSE], Rf = .04/12)
charts.RollingRegression(managers[, 1:6], managers[, 8, drop=FALSE], Rf = .04/12, colorset =
```

`chart.Scatter` *wrapper to draw scatter plot with sensible defaults*

Description

Draws a scatter chart. This is another chart "primitive", since it only contains a set of sensible defaults.

Usage

```
chart.Scatter(x, y, reference.grid = TRUE, main = "Title", ylab = NULL, xlab = NULL
```

Arguments

`x`	data for the x axis, can take matrix,vector, or timeseries
`y`	data for the y axis, can take matrix,vector, or timeseries
`reference.grid`	if true, draws a grid aligned with the points on the x and y axes
`main`	set the chart title, same as in `plot`
`ylab`	set the y-axis label, as in `plot`
`xlab`	set the x-axis label, as in `plot`
`xlim`	set the x-axis limit, same as in `plot`
`ylim`	set the y-axis limit, same as in `plot`
`colorset`	color palette to use, set by default to rational choices
`symbolset`	from `pch` in `plot`, submit a set of symbols to be used in the same order as the data sets submitted
`element.color`	provides the color for drawing chart elements, such as the box lines, axis lines, etc. Default is "darkgray"
`cex.lab`	The magnification to be used for x- and y-axis labels relative to the current setting of 'cex'
`cex.axis`	The magnification to be used for axis annotation relative to the current setting of 'cex', same as in `plot`.
`cex.main`	The magnification to be used for the main title relative to the current setting of 'cex'.
`cex.legend`	The magnification to be used for sizing the legend relative to the current setting of 'cex'.
`...`	any other passthru parameters

Note

Most inputs are the same as "plot" and are principally included so that some sensible defaults could be set.

Author(s)

Peter Carl

See Also

plot

Examples

```
data(edhec)
chart.Scatter(edhec[,1],edhec[,2])
```

---

chart.SnailTrail *chart risk versus return over rolling time periods*

Description

A chart that shows rolling calculations of annualized return and annualized standard deviation have proceeded through time. Lines and dots are darker for more recent time periods.

Usage

```
chart.SnailTrail(R, Rf = 0, main = "Annualized Return and Risk", add.names = c("all
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rf	risk free rate, in same period as your returns
main	set the chart title, same as in plot
add.names	plots the row name with the data point. default TRUE. Can be removed by setting it to NULL
xlab	set the x-axis label, as in plot
ylab	set the y-axis label, as in plot
add.sharpe	this draws a Sharpe ratio line that indicates Sharpe ratio levels of c(1,2,3). Lines are drawn with a y-intercept of the risk free rate and the slope of the appropriate Sharpe ratio level. Lines should be removed where not appropriate (e.g., sharpe.ratio = NULL).
colorset	color palette to use, set by default to rational choices
symbolset	from pch in plot, submit a set of symbols to be used in the same order as the data sets submitted

cex.legend	The magnification to be used for sizing the legend relative to the current setting of 'cex'.
element.color	provides the color for drawing chart elements, such as the box lines, axis lines, etc. Default is "darkgray"
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
xlim	set the x-axis limit, same as in plot
ylim	set the y-axis limit, same as in plot
width	number of periods to apply rolling calculations over, sometimes referred to as a 'window'
stepsize	the frequency with which to make the rolling calculation
lty	set the line type, same as in plot
lwd	set the line width, same as in plot
cex.lab	The magnification to be used for sizing the label relative to the current setting of 'cex', similar to plot.
cex.main	The magnification to be used for sizing the main chart relative to the current setting of 'cex', as in plot.
cex.axis	The magnification to be used for sizing the axis text relative to the current setting of 'cex', similar to plot.
cex.text	The magnification to be used for sizing the text relative to the current setting of 'cex', similar to plot.
...	any other passthru parameters

Author(s)

Peter Carl

References

~put references to the literature/web site here ~

See Also

chart.RiskReturnScatter

Examples

```
data(managers)
chart.SnailTrail(managers[,c("HAM2","SP500 TR"),drop=FALSE], width=36, stepsize=12, colorset
```

`chart.StackedBar` *create a stacked bar plot*

Description

This creates a stacked column chart with time on the horizontal axis and values in categories. This kind of chart is commonly used for showing portfolio 'weights' through time, although the function will plot any values by category.

Usage

```
chart.StackedBar(w, colorset = NULL, space = 0.2, cex.axis=0.8, cex.legend = 0.8, c
```

Arguments

`w`	a matrix, data frame or zoo object of values to be plotted. Rownames should contain dates or period labels; column names should indicate categories. See examples for detail.
`colorset`	color palette to use, set by default to rational choices
`space`	the amount of space (as a fraction of the average bar width) left before each bar, as in `barplot`. Default is 0.2.
`cex.legend`	The magnification to be used for sizing the legend relative to the current setting of 'cex', similar to `plot`.
`cex.labels`	The magnification to be used for event line labels relative to the current setting of 'cex'.
`cex.lab`	The magnification to be used for x- and y-axis labels relative to the current setting of 'cex'.
`cex.main`	The magnification to be used for the chart title relative to the current setting of 'cex'.
`xaxis`	If true, draws the x axis
`ylab`	Set the y-axis label, same as in `plot`
`date.format`	Re-format the dates for the xaxis; the default is "%m/%y"
`major.ticks`	Should major tickmarks be drawn and labeled, default 'auto'
`minor.ticks`	Should minor tickmarks be drawn, default TRUE
`xaxis.labels`	Allows for non-date labeling of date axes, default is NULL
`cex.axis`	The magnification to be used for sizing the axis text relative to the current setting of 'cex', similar to `plot`.
`las`	sets the orientation of the axis labels, as described in `par`. Defaults to '3'.
`legend.loc`	places a legend into a location on the chart similar to `chart.TimeSeries`. The default, "under," is the only location currently implemented for this chart. Use 'NULL' to remove the legend.

element.color
: provides the color for drawing less-important chart elements, such as the box lines, axis lines, etc.

unstacked
: logical. If set to 'TRUE' *and* only one row of data is submitted in 'w', then the chart creates a normal column chart. If more than one row is submitted, then this is ignored. See examples below.

xlab
: the x-axis label, which defaults to 'NULL'.

ylim
: set the y-axis limit, same as in plot

...
: arguments to be passed to barplot.

Details

This function is a wrapper for barplot but adds three additional capabilities. First, it calculates and sets a bottom margin for long column names that are rotated vertically. That doesn't always result in the prettiest chart, but it does ensure readable labels.

Second, it places a legend "under" the graph rather than within the bounds of the chart (which would obscure the data). The legend is created from the column names. The default is to create the legend when there's more than one row of data being presented. If there is one row of data, the chart may be "unstacked" and the legend removed.

Third, it plots or stacks negative values from an origin of zero, similar to the behavior of barchart from the 'lattice' package.

Note

The 'w' attribute is so named because this is a popular way to show portfolio weights through time. That being said, this function isn't limited to any particular values and doesn't provide any normalization, so that the chart can be used more generally.

The principal drawback of stacked column charts is that it is very difficult for the reader to judge size of 2nd, 3rd, etc., data series because they do not have common baseline. Another is that with a large number of series, the colors may be difficult to discern. As alternatives, Cleveland advocates the use of trellis like displays, and Tufte advocates the use of small multiple charts.

Author(s)

Peter Carl

References

Cleveland, W.S. (1994), The Elements of Graphing Data, Summit, NJ: Hobart Press.

Tufte, Edward R. (2001) The Visual Display of Quantitative Information, 2nd edition. The Graphics Press, Cheshire, Connecticut. See http://www.edwardtufte.com for this and other references.

See Also

barplot, par

Examples

```
data(weights)
head(weights)

# With the legend "under" the chart
chart.StackedBar(weights, date.format="%Y", cex.legend = 0.7, colorset=rainbow12equal)

# Without the legend
chart.StackedBar(weights, colorset=rainbow12equal, legend.loc=NULL)

# for one row of data, use 'unstacked' for a better chart
chart.StackedBar(weights[1,,drop=FALSE], unstacked=TRUE, las=3)
```

---

chart.Style *calculate and display effective style weights*

Description

Functions that calculate effective style weights and display the results in a bar chart. `chart.Style` calculates and displays style weights calculated over a single period. `chart.RollingStyle` calculates and displays those weights in rolling windows through time. `style.fit` manages the calculation of the weights by method. `style.QPfit` calculates the specific constraint case that requires quadratic programming.

Usage

```
chart.Style(R.fund, R.style, method = c("constrained", "unconstrained", "normalized

chart.RollingStyle(R.fund, R.style, method = c("constrained","unconstrained","norma

style.fit(R.fund, R.style, model=FALSE, method = c("constrained", "unconstrained",

style.QPfit(R.fund, R.style, model = FALSE, leverage = FALSE, ...)
```

Arguments

`R.fund`	matrix, data frame, or zoo object with fund returns to be analyzed
`R.style`	matrix, data frame, or zoo object with style index returns. Data object must be of the same length and time-aligned with R.fund
`method`	specify the method of calculation of style weights as "constrained", "unconstrained", or "normalized". For more information, see `style.fit`
`leverage`	logical, defaults to 'FALSE'. If 'TRUE', the calculation of weights assumes that leverage may be used. For more information, see `style.fit`
`model`	logical. If 'model' = TRUE in `style.QPfit`, the full result set is shown from the output of `solve.QP`.

selection	either "none" (default) or "AIC". If "AIC", then the function uses a stepwise regression to identify find the model with minimum AIC value. See step for more detail.
unstacked	logical. If set to 'TRUE' *and* only one row of data is submitted in 'w', then the chart creates a normal column chart. If more than one row is submitted, then this is ignored. See examples below.
space	the amount of space (as a fraction of the average bar width) left before each bar, as in barplot. Default for chart.RollingStyle is 0; for chart.Style the default is 0.2.
main	set the chart title, same as in plot
width	number of periods or window to apply rolling style analysis over
ylim	set the y-axis limit, same as in plot
...	for the charting functions, these are arguments to be passed to barplot. These can include further arguments (such as 'axes', 'asp' and 'main') and graphical parameters (see 'par') which are passed to 'plot.window()', 'title()' and 'axis'. For the calculation functions, these are ignored.

Details

These functions calculate style weights using an asset class style model as described in detail in Sharpe (1992). The use of quadratic programming to determine a fund's exposures to the changes in returns of major asset classes is usually refered to as "style analysis".

The "unconstrained" method implements a simple factor model for style analysis, as in:

$$Ri = bi1 * F1 + bi2 * F2 + ... + bin * Fn + ei$$

where Ri represents the return on asset i, Fj represents each factor, and ei represents the "non-factor" component of the return on i. This is simply a multiple regression analysis with fund returns as the dependent variable and asset class returns as the independent variables. The resulting slope coefficients are then interpreted as the fund's historic exposures to asset class returns. In this case, coefficients do not sum to 1.

The "normalized" method reports the results of a multiple regression analysis similar to the first, but with one constraint: the coefficients are required to add to 1. Coefficients may be negative, indicating short exposures. To enforce the constraint, coefficients are normalized.

The "constrained" method includes the constraint that the coefficients sum to 1, but adds that the coefficients must lie between 0 and 1. These inequality constraints require a quadratic programming algorithm using solve.QP from the 'quadprog' package, and the implementation is discussed under style.QPfit. If set to TRUE, "leverage" allows the sum of the coefficients to exceed 1.

According to Sharpe (1992), the calculation for the constrained case is represented as:

$$minvar(Rf - sum[wi * R.si]) = minvar(F - w * S)$$

$$s.t. sum[wi] = 1; wi > 0$$

Remembering that:

$$\sigma(aX + bY) = a^2\sigma(X) + b^2\sigma(Y) + 2abcov(X, Y) = \sigma(R.f) + w' * V * w - 2 * w' * cov(R.f, R.s)$$

we can drop $var(Rf)$ as it isn't a function of weights, multiply both sides by 1/2:

$$= min(1/2)w' * V * w - C'w$$

$$s.t.w' * e = 1, w_i > 0$$

Which allows us to use `solve.QP`, which is specified as:

$$min(-d'b + 1/2b'Db)$$

and the constraints

$$A'b >= b.0$$

so: b is the weight vector, D is the variance-covariance matrix of the styles d is the covariance vector between the fund and the styles

The chart functions then provide a graphical summary of the results. The underlying function, `style.fit`, provides the outputs of the analysis and more information about fit, including an R-squared value.

Styles identified in this analysis may be interpreted as an average of potentially changing exposures over the period covered. The function `chart.RollingStyle` may be useful for examining the behavior of a manager's average exposures to asset classes over time, using a rolling-window analysis.

The chart functions plot a column chart or stacked column chart of the resulting style weights to the current device. Both `style.fit` and `style.QPfit` produce a list of data frames containing 'weights' and 'R.squared' results. If 'model' = TRUE in `style.QPfit`, the full result set is shown from the output of `solve.QP`.

Note

None of the functions `chart.Style`, `style.fit`, and `style.QPfit` make any attempt to align the two input data series. The `chart.RollingStyle`, on the other hand, does merge the two series and manages the calculation over common periods.

Author(s)

Peter Carl

References

Sharpe, W. Asset Allocation: Management Style and Performance Measurement Journal of Portfolio Management, 1992, 7-19. See http://www.stanford.edu/~wfsharpe/art/sa/sa.htm

See Also

`barplot`, `par`

Examples

```
data(edhec)
data(managers)
style.fit(managers[97:132,2,drop=FALSE],edhec[85:120,], method="constrained", leverage=FALSE
chart.Style(managers[97:132,2,drop=FALSE],edhec[85:120,], method="constrained", leverage=FAL
chart.RollingStyle(managers[,2,drop=FALSE],edhec[,1:11], method="constrained", leverage=FALS
```

---

chart.TimeSeries *Creates a time series chart with some extensions.*

Description

Draws a line chart and labels the x-axis with the appropriate dates. This is really a "primitive", since it extends the base `plot` and standardizes the elements of a chart. Adds attributes for shading areas of the timeline or aligning vertical lines along the timeline. This function is intended to be used inside other charting functions.

Usage

```
chart.TimeSeries(R, auto.grid = TRUE, xaxis = TRUE, yaxis = TRUE, yaxis.right = FAI
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`auto.grid`	if true, draws a grid aligned with the points on the x and y axes
`grid.color`	sets the color for the reference grid
`grid.lty`	defines the line type for the grid
`xaxis`	if true, draws the x axis
`yaxis`	if true, draws the y axis
`yaxis.right`	if true, draws the y axis on the right-hand side of the plot
`type`	set the chart type, same as in `plot`
`lty`	set the line type, same as in `plot`
`lwd`	set the line width, same as in `plot`
`main`	set the chart title, same as in `plot`
`ylab`	set the y-axis label, same as in `plot`
`xlab`	set the x-axis label, same as in `plot`
`date.format`	re-format the dates for the xaxis; the default is "%m/%y"
`xlim`	set the x-axis limit, same as in `plot`
`ylim`	set the y-axis limit, same as in `plot`
`event.lines`	If not null, vertical lines will be drawn to indicate that an event happened during that time period. `event.lines` should be a list of dates (e.g., `c("09/03","05/06")`) formatted the same as date.format. This function matches the re-formatted row names (dates) with the events.list, so to get a match the formatting needs to be correct.

`event.labels` if not null and event.lines is not null, this will apply a list of text labels (e.g., `c("This Event", "That Event")` to the vertical lines drawn. See the example below.

`period.areas` these are shaded areas described by start and end dates in the same format as the date.format. This is provided as a list of pairs, e.g., `list(c("10/26","11/27"), c("08/29","03/33"))` See the examples below.

`event.color` draws the event described in `event.labels` in the color specified

`period.color` draws the shaded region described by `period.areas` in the color specified

`colorset` color palette to use, set by default to rational choices

`pch` symbols to use, see also `plot`

`element.color` provides the color for drawing chart elements, such as the box lines, axis lines, etc. Default is "darkgray"

`legend.loc` places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.

`ylog` TRUE/FALSE set the y-axis to logarithmic scale, similar to `plot`, default FALSE

`date.format.in` allows specification of other date formats in the data object, defaults to "%Y-%m-%d"

`cex.axis` The magnification to be used for axis annotation relative to the current setting of 'cex', same as in `plot`.

`cex.legend` The magnification to be used for sizing the legend relative to the current setting of 'cex'.

`cex.labels` The magnification to be used for event line labels relative to the current setting of 'cex'.

`cex.lab` The magnification to be used for x- and y-axis labels relative to the current setting of 'cex'.

`cex.main` The magnification to be used for the chart title relative to the current setting of 'cex'.

`major.ticks` Should major tickmarks be drawn and labeled, default 'auto'

`minor.ticks` Should minor tickmarks be drawn, default TRUE

`xaxis.labels` Allows for non-date labeling of date axes, default is NULL

`...` any other passthru parameters

Author(s)

Peter Carl

See Also

`plot`, `par`, `axTicksByTime`

Examples

```
# These are start and end dates, formatted the same way as the default axis labels
cycles.dates = list(
    c("Oct 26","Nov 27"),
    c("Aug 29","Mar 33"),
    c("May 37","Jun 38"),
    c("Feb 45","Oct 45"),
    c("Nov 48","Oct 49"),
    c("Jul 53","May 54"),
    c("Aug 57","Apr 58"),
    c("Apr 60","Feb 61"),
    c("Dec 69","Nov 70"),
    c("Nov 73","Mar 75"),
    c("Jan 80","Jul 80"),
    c("Jul 81","Nov 82"),
    c("Jul 90","Mar 91"),
    c("Mar 01","Nov 01"))
# Event lists - FOR BEST RESULTS, KEEP THESE DATES IN ORDER
risk.dates = c(
    "Oct 87",
    "Feb 94",
    "Jul 97",
    "Aug 98",
    "Oct 98",
    "Jul 00",
    "Sep 01")
risk.labels = c(
    "Black Monday",
    "Bond Crash",
    "Asian Crisis",
    "Russian Crisis",
    "LTCM",
    "Tech Bubble",
    "Sept 11")
data(edhec)

R=edhec[,"Funds of Funds",drop=FALSE]
Return.cumulative = cumprod(1+R) - 1
chart.TimeSeries(Return.cumulative)
chart.TimeSeries(Return.cumulative, colorset = "darkblue", legend.loc = "bottomright", peric
```

---

chart.VaRSensitivity

show the sensitivity of Value-at-Risk or Expected Shortfall estimates

---

Description

Creates a chart of Value-at-Risk and/or Expected Shortfall estimates by confidence interval for multiple methods.

Usage

```
chart.VaRSensitivity(R, methods = c("GaussianVaR", "ModifiedVaR", "HistoricalVaR","
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
methods	one or more calculation methods indicated "GaussianVaR", "ModifiedVaR", "HistoricalVaR", "GaussianES", "ModifiedES", "HistoricalES". See VaR or ES for more detail.
clean	method for data cleaning through Return.clean. Current options are "none" or "boudt" or "geltner".
elementcolor	the color used to draw chart elements. The default is "darkgray"
reference.grid	if true, draws a grid aligned with the points on the x and y axes
ylab	set the y-axis label, same as in plot
xlab	set the x-axis label, same as in plot
type	set the chart type, same as in plot
lty	set the line type, same as in plot
lwd	set the line width, same as in plot
colorset	color palette to use, set by default to rational choices
pch	symbols to use, see also plot
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
cex.legend	The magnification to be used for sizing the legend relative to the current setting of 'cex'.
main	set the chart title, same as in plot
...	any other passthru parameters

Details

This chart shows estimated VaR along a series of confidence intervals for selected calculation methods. Useful for comparing a method to the historical VaR calculation.

Author(s)

Peter Carl

References

Boudt, K., Peterson, B. G., Croux, C., 2008. Estimation and Decomposition of Downside Risk for Portfolios with Non-Normal Returns. Journal of Risk, forthcoming.

See Also

VaR
ES

Examples

```
data(managers)
chart.VaRSensitivity(managers[,1,drop=FALSE], methods=c("HistoricalVaR", "ModifiedVaR", "Gau
```

`charts.PerformanceSummary` *Create combined wealth index, period performance, and drawdown chart*

Description

For a set of returns, create a wealth index chart, bars for monthly performance, and underwater chart for drawdown.

Usage

```
charts.PerformanceSummary(R, Rf = 0, main = NULL, methods = c("ModifiedVaR",
                "HistoricalVaR"), width = 0, event.labels = NULL, ylog
                = FALSE, wealth.index = FALSE, gap = 12, begin =
                c("first", "axis"), legend.loc = "topleft", ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rf`	risk free rate, in same period as your returns
`main`	set the chart title, as in `plot`
`methods`	Used to select the risk parameter to use in the chart.BarVaR. May be any of: • None - does not add a line, • ModifiedVaR - uses Cornish-Fisher modified VaR, • GaussianVaR - uses traditional Value at Risk, • HistoricalVaR - calculates historical Value at Risk, • StdDev - monthly standard deviation of trailing 12 month returns.
`begin`	Align shorter series to: • first - prior value of the first column given for the reference or longer series or, • axis - the initial value (1 or zero) of the axis. passthru to `chart.CumReturns`
`event.labels`	TRUE/FALSE whether or not to display lines and labels for historical market shock events
`wealth.index`	if `wealth.index` is `TRUE`, shows the "value of $1", starting the cumulation of returns at 1 rather than zero
`width`	number of periods to apply rolling function window over
`gap`	numeric number of periods from start of series to use to train risk calculation

ylog	TRUE/FALSE set the y-axis to logarithmic scale, similar to plot, default FALSE
legend.loc	sets the legend location in the top chart. Can be set to NULL or nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
...	any other passthru parameters

Note

Most inputs are the same as "plot" and are principally included so that some sensible defaults could be set.

Author(s)

Peter Carl

See Also

chart.CumReturns
chart.BarVaR
chart.Drawdown

Examples

```
data(edhec)
charts.PerformanceSummary(edhec[,c(1,13)])
```

---

charts.RollingPerformance
rolling performance chart

Description

A wrapper to create a rolling annualized returns chart, rolling annualized standard deviation chart, and a rolling annualized sharpe ratio chart.

Usage

```
charts.RollingPerformance(R, width = 12, Rf = 0, main = NULL, trim = TRUE,
                          event.labels = NULL, legend.loc = NULL, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
width	number of periods to apply rolling function over
Rf	risk free rate, in same period as your returns
main	set the chart title, same as in plot

trim	TRUE/FALSE, whether to keep alignment caused by NA's
event.labels	TRUE/FALSE whether or not to display lines and labels for historical market shock events
legend.loc	places a legend into one of nine locations on the chart: bottomright, bottom, bottomleft, left, topleft, top, topright, right, or center.
...	any other passthru parameters

Author(s)

Peter Carl

See Also

chart.RollingPerformance

Examples

```
data(managers)
charts.RollingPerformance(managers[,1:8], Rf=managers[,10,drop=FALSE], colorset=tim8equal, m
```

---

checkData *check input data type and format and coerce to the desired output type*

Description

This function was created to make the different kinds of data classes at least *seem* more fungible. It allows the user to pass in a data object without being concerned that the function requires a matrix, data.frame, or timeSeries object. By using this, the function "knows" what data format it has to work with.

Usage

```
checkData(x, method = c("xts", "zoo", "data.frame", "matrix","vector"), na.rm = TRU
```

Arguments

x	a vector, matrix, data.frame, xts, timeSeries or zoo object to be checked and coerced
na.rm	TRUE/FALSE Remove NA's from the data? used only with 'vector'
quiet	TRUE/FALSE if false, it will throw warnings when errors are noticed, default TRUE
method	type of coerced data object to return, one of c("zoo","matrix","vector"), default "zoo"
...	any other passthru parameters

Author(s)

Peter Carl

Examples

```
data(edhec)
x = checkData(edhec)
class(x)
head(x)
tail(x)
# Note that passing in a single column loses the row and column names
x = checkData(edhec[,1])
class(x)
head(x)
# Include the "drop" attribute to keep row and column names
x = checkData(edhec[,1,drop=FALSE])
class(x)
head(x)
x = checkData(edhec, method = "matrix")
class(x)
head(x)
x = checkData(edhec[,1], method = "vector")
class(x)
head(x)
```

---

clean.boudt *clean extreme observations in a time series to to provide more robust risk estimates*

Description

Robustly clean a time series to reduce the magnitude, but not the number or direction, of observations that exceed the $1 - \alpha\%$ risk threshold.

Usage

```
clean.boudt(R, alpha = 0.01, trim = 0.001)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
alpha	probability to filter at 1-alpha, defaults to .01 (99%)
trim	where to set the "extremeness" of the Mahalanobis distance

Details

Many risk measures are calculated by using the first two (four) moments of the asset or portfolio return distribution. Portfolio moments are extremely sensitive to data spikes, and this sensitivity is only exacerbated in a multivariate context. For this reason, it seems appropriate to consider estimates of the multivariate moments that are robust to return observations that deviate extremely from the Gaussian distribution.

There are two main approaches in defining robust alternatives to estimate the multivariate moments by their sample means (see e.g. Maronna[2006]). One approach is to consider a more robust estimator than the sample means. Another one is to first clean (in a robust way) the data and then take the sample means and moments of the cleaned data.

Our cleaning method follows the second approach. It is designed in such a way that, if we want to estimate downside risk with loss probability α, it will never clean observations that belong to the $1-\alpha$ least extreme observations. Suppose we have an n-dimensional vector time series of length T: $r_1, ..., r_T$. We clean this time series in three steps.

1. *Ranking the observations in function of their extremeness.* Denote μ and Σ the mean and covariance matrix of the bulk of the data and let $\lfloor \cdot \rfloor$ be the operator that takes the integer part of its argument. As a measure of the extremeness of the return observation r_t, we use its squared Mahalanobis distance $d_t^2 = (r_t - \mu)' \Sigma^{-1} (r_t - \mu)$. We follow Rousseeuw(1985) by estimating μ and Σ as the mean vector and covariance matrix (corrected to ensure consistency) of the subset of size $\lfloor (1-\alpha)T \rfloor$ for which the determinant of the covariance matrix of the elements in that subset is the smallest. These estimates will be robust against the α most extreme returns. Let $d_{(1)}^2, ..., d_{(T)}^2$ be the ordered sequence of the estimated squared Mahalanobis distances such that $d_{(i)}^2 \leq d_{(i+1)}^2$.
2. *Outlier identification.* Return observations are qualified as outliers if their estimated squared Mahalanobis distance d_t^2 is greater than the empirical $1-\alpha$ quantile $d_{(\lfloor (1-\alpha)T \rfloor)}^2$ and exceeds a very extreme quantile of the Chi squared distribution function with n degrees of freedom, which is the distribution function of d_t^2 when the returns are normally distributed. In this application we take the 99.9% quantile, denoted $\chi_{n,0.999}^2$.
3. *Data cleaning.* Similarly to Khan(2007) we only clean the returns that are identified as outliers in step 2 by replacing these returns r_t with

$$r_t \sqrt{\frac{\max(d_{(\lfloor (1-\alpha)T \rfloor)}^2, \chi_{n,0.999}^2)}{d_t^2}}$$

   The cleaned return vector has the same orientation as the original return vector, but its magnitude is smaller. Khan(2007) calls this procedure of limiting the value of d_t^2 to a quantile of the χ_n^2 distribution, "multivariate Winsorization'.

Note that the primary value of data cleaning lies in creating a more robust and stable estimation of the distribution generating the large majority of the return data. The increased robustness and stability of the estimated moments utilizing cleaned data should be used for portfolio construction. If a portfolio manager wishes to have a more conservative risk estimate, cleaning may not be indicated for risk monitoring. It is also important to note that the robust method proposed here does not remove data from the series, but only decreases the magnitude of the extreme events. It may also be appropriate in practice to use a cleaning threshold somewhat outside the VaR threshold that the manager wishes to consider. In actual practice, it is probably best to back-test the results of both

cleaned and uncleaned series to see what works best with the particular combination of assets under consideration.

Value

cleaned data matrix

Note

This function and much of this text was originally written for Boudt, et. al, 2008

Author(s)

Kris Boudt, Brian G. Peterson

References

Boudt, K., Peterson, B. G., Croux, C., 2008. Estimation and Decomposition of Downside Risk for Portfolios with Non-Normal Returns. Journal of Risk, forthcoming.

Khan, J. A., S. Van Aelst, and R. H. Zamar (2007). Robust linear model selection based on least angle regression. Journal of the American Statistical Association 102.

Maronna, R. A., D. R. Martin, and V. J. Yohai (2006). Robust Statistics: Theory and Methods. Wiley.

Rousseeuw, P. J. (1985). Multivariate estimation with high breakdown point. In W. Grossmann, G. Pflug, I. Vincze, and W. Wertz (Eds.), Mathematical Statistics and Its Applications, Volume B, pp. 283?297. Dordrecht-Reidel.

See Also

`Return.clean`

Co-Moments *Functions for calculating comoments of financial time series*

Description

calculates coskewness and cokurtosis as the skewness and kurtosis of two assets with reference to one another.

Usage

```
CoVariance(Ra,Rb)
CoSkewness(Ra,Rb)
CoKurtosis(Ra,Rb)
```

Arguments

`Ra`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`Rb`	an xts, vector, matrix, data frame, timeSeries or zoo object of index, benchmark, portfolio, or secondary asset returns to compare against

Details

Ranaldo and Favre (2005) define coskewness and cokurtosis as the skewness and kurtosis of a given asset analysed with the skewness and kurtosis of the reference asset or portfolio. Adding an asset to a portfolio, such as a hedge fund with a significant level of coskewness to the portfolio, can increase or decrease the resulting portfolio's skewness. Similarly, adding a hedge fund with a positive cokurtosis coefficient will add kurtosis to the portfolio.

The co-moments are useful for measuring the marginal contribution of each asset to the portfolio's resulting risk. As such, comoments of asset return distribution should be useful as inputs for portfolio optimization in addition to the covariance matrix. Martellini and Ziemann (2007) point out that the problem of portfolio selection becomes one of selecting tangency points in four dimensions, incorporating expected return, second, third and fourth centered moments of asset returns.

Even outside of the optimization problem, measuring the co-moments should be a useful tool for evaluating whether or not an asset is likely to provide diversification potential to a portfolio, not only in terms of normal risk (i.e. volatility) but also the risk of assymetry (skewness) and extreme events (kurtosis).

Author(s)

Kris Boudt, Peter Carl, Brian Peterson

References

Boudt, Kris, Brian G. Peterson, and Christophe Croux. 2008. Estimation and Decomposition of Downside Risk for Portfolios with Non-Normal Returns. Journal of Risk. Winter.

Martellini, Lionel, and Volker Ziemann. 2007. Improved Forecasts of Higher-Order Comoments and Implications for Portfolio Selection. EDHEC Risk and Asset Management Research Centre working paper.

Ranaldo, Angelo, and Laurent Favre Sr. 2005. How to Price Hedge Funds: From Two- to Four-Moment CAPM. SSRN eLibrary.

Scott, Robert C., and Philip A. Horvath. 1980. On the Direction of Preference for Moments of Higher Order than the Variance. Journal of Finance 35(4):915-919.

See Also

`BetaCoSkewness`
`BetaCoKurtosis`
`BetaCoMoments`

Examples

```
data(managers)
CoVariance(managers[, "HAM2", drop=FALSE], managers[, "SP500.TR", drop=FALSE])
CoSkewness(managers[, "HAM2", drop=FALSE], managers[, "SP500.TR", drop=FALSE])
CoKurtosis(managers[, "HAM2", drop=FALSE], managers[, "SP500.TR", drop=FALSE])
```

DownsideDeviation *downside risk (deviation, variance) of the return distribution*

Description

Downside deviation, semideviation, and semivariance are measures of downside risk.

Usage

```
DownsideDeviation(R, MAR = 0, method=c("subset","full"))
SemiDeviation(R)
SemiVariance(R)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
MAR	Minimum Acceptable Return, in the same periodicity as your returns
method	one of "full" or "subset", indicating whether to use the length of the full series or the length of the subset of the series below the MAR as the denominator, defaults to "subset"

Details

Downside deviation, similar to semi deviation, eliminates positive returns when calculating risk. Instead of using the mean return or zero, it uses the Minimum Acceptable Return as proposed by Sharpe (which may be the mean historical return or zero).

To calculate it, we take the subset of returns that are less than the target (or Minimum Acceptable Returns (MAR)) returns and take the differences of those to the target. We sum the squares and divide by the total number of returns to get a below-target semi-variance.

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

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

SemiDeviation or SemiVariance is a popular alternative downside risk measure that may be used in place of standard deviation or variance. SemiDeviation and SemiVariance are implemented as a wrapper of DownsideDeviation with MAR=mean(R).

In many functions like Markowitz optimization, semideviation may be substituted directly, and the covariance matrix may be constructed from semideviation or the vector of returns below the mean rather than from variance or the full vector of returns.

In semideviation, by convention, the value of n is set to the full number of observations. In semivariance the the value of n is set to the subset of returns below the mean. It should be noted that while this is the correct mathematical definition of semivariance, this result doesn't make any sense if you are also going to be using the time series of returns below the mean or below a MAR to construct a semi-covariance matrix for portfolio optimization.

Sortino recommends calculating downside deviation utilizing a continuous fitted distribution rather than the discrete distribution of observations. This would have significant utility, especially in cases of a small number of observations. He recommends using a lognormal distribution, or a fitted distribution based on a relevant style index, to construct the returns below the MAR to increase the confidence in the final result. Hopefully, in the future, we'll add a fitted option to this function, and would be happy to accept a contribution of this nature.

Author(s)

Peter Carl, Brian G. Peterson

References

Sortino, F. and Price, L. Performance Measurement in a Downside Risk Framework. *Journal of Investing*. Fall 1994, 59-65.

Plantinga, A., van der Meer, R. and Sortino, F. The Impact of Downside Risk on Risk-Adjusted Performance of Mutual Funds in the Euronext Markets. July 19, 2001. Available at SSRN: http://ssrn.com/abstract=277352

http://www.sortino.com/htm/performance.htm see especially end note 10

http://en.wikipedia.org/wiki/Semivariance

Examples

```
data(managers)
sd(managers[,1:6], na.rm=TRUE)
DownsideDeviation(managers[,1:6])  # MAR 0%
DownsideDeviation(managers[,1:6], MAR = .04/12) #MAR 4%
SemiDeviation(managers[,1,drop=FALSE])
SemiDeviation(managers[,1:6])
SemiVariance (managers[,1,drop=FALSE])
SemiVariance (managers[,1:6]) #calculated using method="subset"
```

edhec *EDHEC-Risk Hedge Fund Style Indices*

Description

EDHEC composite hedge fund style index returns.

Usage

```
edhec
```

Format

CSV conformed into an xts object with monthly observations

Details

EDHEC Data used in PerformanceAnalytics and related publications with the kind permission of the EDHEC Risk and Asset Management Research Center.

The 'edhec' data set included with PerformanceAnalytics will be periodically updated (typically annually) to include additional observations. If you intend to use this data set in automated tests, please be sure to subset your data like `edhec[1:120,]` to use the first ten years of observations.

From the EDHEC website: "The EDHEC Risk and Asset Management Research Centre plays a noted role in furthering applied financial research and systematically highlighting its practical uses. As part of its philosophy, the centre maintains a dialogue with professionals which benefits the industry as a whole. At the same time, its proprietary R&D provides sponsors with an edge over competition and joint ventures allow selected partners to develop new business opportunities.

To further assist financial institutions and investors implement the latest research advances in order to meet the challenges of the changing asset management landscape, the centre has spawned two consultancies and an executive education arm. Clients of these derivative activities include many of the leading organisations throughout Europe."

see http://www.edhec-risk.com/about_us

Source

http://www.edhec-risk.com/indexes/pure_style

References

About EDHEC Alternative Indexes. December 16, 2003. EDHEC-Risk.
http://www.edhec-risk.com/indexes/pure_style/about

Vaissie Mathieu. A Detailed Analysis of the Construction Methods and Management Principles of Hedge Fund Indices. October 2003. EDHEC.
http://www.edhec-risk.com/site_edhecrisk/public/indexes/EDHEC_Publications/RISKReview10727051880657935 13

Examples

```
data(edhec)

#preview the data
head(edhec)

#summary period statistics
summary(edhec)

#cumulative index returns
tail(cumprod(1+edhec),1)
```

`ES` *calculates Expected Shortfall(ES) (or Conditional Value-at-Risk(CVaR) for univariate and component, using a variety of analytical methods.*

Description

Calculates Expected Shortfall(ES) (also known as) Conditional Value at Risk(CVaR) for univariate, component, and marginal cases using a variety of analytical methods.

Usage

```
ES(R, p = 0.95, method = c("modified", "gaussian", "historical", "kernel"), clean =
```

Arguments

`R`	a vector, matrix, data frame, timeSeries or zoo object of asset returns
`p`	confidence level for calculation, default p=.99
`method`	one of "modified","gaussian","historical", "kernel", see Details.
`clean`	method for data cleaning through `Return.clean`. Current options are "none", "boudt", or "geltner".
`portfolio_method`	one of "single","component","marginal" defining whether to do univariate, component, or marginal calc, see Details.
`weights`	portfolio weighting vector, default NULL, see Details
`mu`	If univariate, mu is the mean of the series. Otherwise mu is the vector of means of the return series , default NULL, , see Details
`sigma`	If univariate, sigma is the variance of the series. Otherwise sigma is the covariance matrix of the return series , default NULL, see Details
`m3`	If univariate, m3 is the skewness of the series. Otherwise m3 is the coskewness matrix of the returns series, default NULL, see Details
`m4`	If univariate, m4 is the excess kurtosis of the series. Otherwise m4 is the cokurtosis matrix of the return series, default NULL, see Details

`invert`	TRUE/FALSE whether to invert the VaR measure. see Details.
`operational`	TRUE/FALSE, default TRUE, see Details.
`...`	any other passthru parameters

Background

This function provides several estimation methods for the Expected Shortfall (ES) (also called Conditional Value at Risk (CVaR)) of a return series and the Component ES of a portfolio. At a preset probability level denoted c, which typically is between 1 and 5 per cent, the ES of a return series is the negative value of the expected value of the return when the return is less than its c-quantile. Unlike value-at-risk, conditional value-at-risk has all the properties a risk measure should have to be coherent and is a convex function of the portfolio weights (Pflug, 2000). With a sufficiently large data set, you may choose to estimate ES with the sample average of all returns that are below the c empirical quantile. More efficient estimates of VaR are obtained if a (correct) assumption is made on the return distribution, such as the normal distribution. If your return series is skewed and/or has excess kurtosis, Cornish-Fisher estimates of ES can be more appropriate. For the ES of a portfolio, it is also of interest to decompose total portfolio ES into the risk contributions of each of the portfolio components. For the above mentioned ES estimators, such a decomposition is possible in a financially meaningful way.

Estimation of ES of a univariate return series

The ES at a probability level p (e.g. 95%) is the negative value of the expected value of the return when the return is less than its $c = 1-p$ quantile. In a set of returns for which sufficently long history exists, the per-period ES can be estimated by the negative value of the sample average of all returns below the quantile. This method is also sometimes called "historical ES", as it is by definition *ex post* analysis of the return distribution, and may be accessed with `method="historical"`.

When you don't have a sufficiently long set of returns to use non-parametric or historical ES, or wish to more closely model an ideal distribution, it is common to us a parmetric estimate based on the distribution. Parametric ES does a better job of accounting for the tails of the distribution by more precisely estimating shape of the distribution tails of the risk quantile. The most common estimate is a normal (or Gaussian) distribution $R \sim N(\mu, \sigma)$ for the return series. In this case, estimation of ES requires the mean return $\bar{R}$, the return distribution and the variance of the returns σ. In the most common case, parametric VaR is thus calculated by

$$\sigma = variance(R)$$

$$ES = -\bar{R} + \sqrt{\sigma} \cdot \frac{1}{c}\phi(z_c)$$

where z_c is the c-quantile of the standard normal distribution. Represented in R by `qnorm(c)`, and may be accessed with `method="gaussian"`. The function ϕ is the Gaussian density function.

The limitations of Gaussian ES are well covered in the literature, since most financial return series are non-normal. Boudt, Peterson and Croux (2008) provide a modified ES calculation that takes the higher moments of non-normal distributions (skewness, kurtosis) into account through the use of a Cornish-Fisher expansion, and collapses to standard (traditional) Gaussian ES if the return stream follows a standard distribution. More precisely, for a loss probability c, modified ES is defined

as the negative of the expected value of all returns below the c Cornish-Fisher quantile and where the expectation is computed under the second order Edgeworth expansion of the true distribution function.

Modified expected shortfall should always be higher than modified Value at Risk. Due to estimation problems, this might not always be the case. Set Operational = TRUE to replace modified ES with modified VaR in the (exceptional) case where the modified ES is smaller than modified VaR.

Component ES

By setting `portfolio_method="component"` you may calculate the ES contribution of each element of the portfolio. The return from the function in this case will be a list with three components: the univariate portfolio ES, the scalar contribution of each component to the portfolio ES (these will sum to the portfolio ES), and a percentage risk contribution (which will sum to 100%).

Both the numerical and percentage component contributions to ES may contain both positive and negative contributions. A negative contribution to Component ES indicates a portfolio risk diversifier. Increasing the position weight will reduce overall portoflio ES.

If a weighting vector is not passed in via `weights`, the function will assume an equal weighted (neutral) portfolio.

Multiple risk decomposition approaches have been suggested in the literature. A naive approach is to set the risk contribution equal to the stand-alone risk. This approach is overly simplistic and neglects important diversification effects of the units being exposed differently to the underlying risk factors. An alternative approach is to measure the ES contribution as the weight of the position in the portfolio times the partial derivative of the portfolio ES with respect to the component weight.

$$C_i\text{ES} = w_i \frac{\partial \text{ES}}{\partial w_i}.$$

Because the portfolio ES is linear in position size, we have that by Euler's theorem the portfolio VaR is the sum of these risk contributions. Scaillet (2002) shows that for ES, this mathematical decomposition of portfolio risk has a financial meaning. It equals the negative value of the asset's expected contribution to the portfolio return when the portfolio return is less or equal to the negative portfolio VaR:

$$C_i\text{ES} == -E\left[w_i r_i | r_p \leq -\text{VaR}\right]$$

For the decomposition of Gaussian ES, the estimated mean and covariance matrix are needed. For the decomposition of modified ES, also estimates of the coskewness and cokurtosis matrices are needed. If r denotes the $Nx1$ return vector and mu is the mean vector, then the $N \times N^2$ coskewness matrix is

$$m3 = E\left[(r-\mu)(r-\mu)' \otimes (r-\mu)'\right]$$

The $N \times N^3$ co-kurtosis matrix is

$$m_4 = E\left[(r-\mu)(r-\mu)' \otimes (r-\mu)' \otimes (r-\mu)'\right]$$

where $\otimes$ stands for the Kronecker product. The matrices can be estimated through the functions `skewness.MM` and `kurtosis.MM`. More efficient estimators were proposed by Martellini and Ziemann (2007) and will be implemented in the future.

As discussed among others in Cont, Deguest and Scandolo (2007), it is important that the estimation of the ES measure is robust to single outliers. This is especially the case for modified VaR and

its decomposition, since they use higher order moments. By default, the portfolio moments are estimated by their sample counterparts. If `clean="boudt"` then the $1-p$ most extreme observations are winsorized if they are detected as being outliers. For more information, see Boudt, Peterson and Croux (2008) and `Return.clean`. If your data consist of returns for highly illiquid assets, then `clean="geltner"` may be more appropriate to reduce distortion caused by autocorrelation, see `Return.Geltner` for details.

Note

The option to `invert` the ES measure should appease both academics and practitioners. The mathematical definition of ES as the negative value of extreme losses will (usually) produce a positive number. Practitioners will argue that ES denotes a loss, and should be internally consistent with the quantile (a negative number). For tables and charts, different preferences may apply for clarity and compactness. As such, we provide the option, and set the default to TRUE to keep the return consistent with prior versions of PerformanceAnalytics, but make no value judgement on which approach is preferable.

Author(s)

Brian G. Peterson and Kris Boudt

References

Boudt, Kris, Peterson, Brian, and Christophe Croux. 2008. Estimation and decomposition of downside risk for portfolios with non-normal returns. 2008. The Journal of Risk, vol. 11, 79-103.

Cont, Rama, Deguest, Romain and Giacomo Scandolo. Robustness and sensitivity analysis of risk measurement procedures. Financial Engineering Report No. 2007-06, Columbia University Center for Financial Engineering.

Laurent Favre and Jose-Antonio Galeano. Mean-Modified Value-at-Risk Optimization with Hedge Funds. Journal of Alternative Investment, Fall 2002, v 5.

Martellini, Lionel, and Volker Ziemann. Improved Forecasts of Higher-Order Comoments and Implications for Portfolio Selection. 2007. EDHEC Risk and Asset Management Research Centre working paper.

Pflug, G. Ch. Some remarks on the value-at-risk and the conditional value-at-risk. In S. Uryasev, ed., Probabilistic Constrained Optimization: Methodology and Applications, Dordrecht: Kluwer, 2000, 272-281.

Scaillet, Olivier. Nonparametric estimation and sensitivity analysis of expected shortfall. Mathematical Finance, 2002, vol. 14, 74-86.

See Also

`VaR`
`SharpeRatio.modified`
`chart.VaRSensitivity`
`VaR.gpd`
`VaR.norm`
`VaR.backtest`
`Return.clean`

Examples

```
data(edhec)

# first do normal ES calc
ES(edhec, p=.95, method="historical")

# now use Gaussian
ES(edhec, p=.95, method="gaussian")

# now use modified Cornish Fisher calc to take non-normal distribution into account
ES(edhec, p=.95, method="modified")

# now use p=.99
ES(edhec, p=.99)
# or the equivalent alpha=.01
ES(edhec, p=.01)

# now with outliers squished
ES(edhec, clean="boudt")

# add Component ES for the equal weighted portfolio
ES(edhec, clean="boudt", portfolio_method="component")
```

---

findDrawdowns *Find the drawdowns and drawdown levels in a timeseries.*

Description

`findDrawdowns` will find the starting period, the ending period, and the amount and length of the drawdown.

Often used with `sortDrawdowns` to get the largest drawdowns.

`Drawdowns` will calculate the drawdown levels as percentages, for use in `chart.Drawdown`.

Usage

```
findDrawdowns(R)
Drawdowns(R)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns

Details

Returns an unordered list:

return depth of drawdown

from	starting period
to	ending period
length	length in periods

Author(s)

Peter Carl

`findDrawdowns` modified with permission from function by Sankalp Upadhyay

References

Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 88

See Also

`sortDrawdowns`
`maxDrawdown`
`sortDrawdowns`
`table.Drawdowns`
`table.DownsideRisk`
`chart.Drawdown`

Examples

```
data(edhec)
findDrawdowns(edhec[,"Funds of Funds", drop=FALSE])
sortDrawdowns(findDrawdowns(edhec[,"Funds of Funds", drop=FALSE]))
```

`InformationRatio` *InformationRatio = ActivePremium/TrackingError*

Description

The Active Premium divided by the Tracking Error.

InformationRatio = ActivePremium/TrackingError

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

Usage

```
InformationRatio(Ra, Rb, scale = NA)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)

Note

William Sharpe now recommends `InformationRatio` preferentially to the original `SharpeRatio`.

Author(s)

Peter Carl

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.

See Also

`TrackingError`
`ActivePremium`
`SharpeRatio`

Examples

```
data(managers)
InformationRatio(managers[,"HAM1",drop=FALSE], managers[, "SP500 TR", drop=FALSE])
InformationRatio(managers[,1:6], managers[,8,drop=FALSE])
InformationRatio(managers[,1:6], managers[,8:7])
```

`KellyRatio` *calculate Kelly criterion ratio (leverage or bet size) for a strategy*

Description

Kelly criterion ratio (leverage or bet size) for a strategy.

Usage

```
KellyRatio(R, Rf = 0, method="half")
```

Arguments

R	a vector of returns to perform a mean over
Rf	risk free rate, in same period as your returns
method	method=half will use the half-Kelly, this is the default

Details

The Kelly Criterion was identified by Bell Labs scientist John Kelly, and applied to blackjack and stock strategy sizing by Ed Thorpe.

The Kelly ratio can be simply stated as: "bet size is the ratio of edge over odds." Mathematically, you are maximizing log-utility. As such, the Kelly criterion is equal to the expected excess return of the strategy divided by the expected variance of the excess return, or

$$leverage = \frac{(\overline{R}_s - R_f)}{StdDev(R)^2}$$

As a performance metric, the Kelly Ratio is calculated retrospectively on a particular investment as a measure of the edge that investment has over the risk free rate. It may be use as a stack ranking method to compare investments in a manner similar to the various ratios related to the Sharpe ratio.

Author(s)

Brian G. Peterson

References

Thorp, Edward O. (1997; revised 1998). The Kelly Criterion in Blackjack, Sports Betting, and the Stock Market. http://www.bjmath.com/bjmath/thorp/paper.htm
http://en.wikipedia.org/wiki/Kelly_criterion

Examples

```
data(managers)
KellyRatio(managers[,1,drop=FALSE], Rf=.04/12)
KellyRatio(managers[,1,drop=FALSE], Rf=managers[,10,drop=FALSE])
KellyRatio(managers[,1:6], Rf=managers[,10,drop=FALSE])
```

---

kurtosis *Kurtosis*

Description

compute kurtosis of a univariate distribution

Usage

```
kurtosis(x, na.rm = FALSE, method = c("excess", "moment", "fisher"), ...)
```

Arguments

`na.rm`	a logical. Should missing values be removed?
`method`	a character string which specifies the method of computation. These are either `"moment"`, `"fisher"`, or `"excess"`. If `"excess"` is selected, then the value of the kurtosis is computed by the `"moment"` method and a value of 3 will be subtracted. The `"moment"` method is based on the definitions of kurtosis for distributions; these forms should be used when resampling (bootstrap or jackknife). The `"fisher"` method correspond to the usual "unbiased" definition of sample variance, although in the case of kurtosis exact unbiasedness is not possible.
`x`	a numeric vector or object.
`...`	arguments to be passed.

Details

This function was ported from the RMetrics package fUtilities to eliminate a dependency on fUtilities being loaded every time. This function is identical except for the addition of `checkData` and additional labeling.

Author(s)

Diethelm Wuertz

See Also

`skewness`.

Examples

```
## mean -
## var -
   # Mean, Variance:
   r = rnorm(100)
   mean(r)
   var(r)

## kurtosis -
   kurtosis(r)

data(managers)
kurtosis(managers[,1:8])
```

managers *Hypothetical Alternative Asset Manager and Benchmark Data*

Description

A xts object that contains columns of monthly returns for six hypothetical asset managers (HAM1 through HAM6), the EDHEC Long-Short Equity hedge fund index, the S&P 500 total returns, and total return series for the US Treasury 10-year bond and 3-month bill. Monthly returns for all series end in December 2006 and begin at different periods starting from January 1996.

Note that all the EDHEC indices are available in edhec.

Usage

```
managers
```

Format

CSV conformed into an xts object with monthly observations

Details

Please note that the 'managers' data set included with PerformanceAnalytics will be periodically updated with new managers and information. If you intend to use this data set in automated tests, please be sure to subset your data like `managers[1:120,1:6]` to use the first ten years of observations on HAM1-HAM6.

Examples

```
data(managers)

#preview the data
head(managers)

#summary period statistics
summary(managers)

#cumulative returns
tail(cumprod(1+managers),1)
```

`maxDrawdown` *caclulate the maximum drawdown from peak equity*

Description

To find the maximum drawdown in a return series, we need to first calculate the cumulative returns and the maximum cumulative return to that point. Any time the cumulative returns dips below the maximum cumulative returns, it's a drawdown. Drawdowns are measured as a percentage of that maximum cumulative return, in effect, measured from peak equity.

Usage

```
maxDrawdown(R)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns

Author(s)

Peter Carl

References

Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 88

See Also

`findDrawdowns`
`sortDrawdowns`
`table.Drawdowns`
`table.DownsideRisk`
`chart.Drawdown`

Examples

```
data(edhec)
maxDrawdown(edhec[,"Funds of Funds"])
data(managers)
maxDrawdown(managers)
```

`mean.utils` *calculate attributes relative to the mean of the observation series given, including geometric, stderr, LCL and UCL*

Description

mean.geometric	geometric mean
mean.stderr	standard error of the mean (S.E. mean)
mean.LCL	lower confidence level (LCL) of the mean
mean.UCL	upper confidence level (UCL) of the mean

Usage

```
## S3 method for class 'geometric':
mean(x, ...)
## S3 method for class 'stderr':
mean(x, ...)
## S3 method for class 'UCL':
mean(x,  ci = 0.95, ...)
## S3 method for class 'LCL':
mean(x,  ci = 0.95, ...)
```

Arguments

x	a vector, matrix, data frame, or time series to calculate the modified mean statistic over
ci	the confidence interval to use
...	any other passthru parameters

Author(s)

Peter Carl

See Also

sd
mean

Examples

```
data(edhec)
mean.geometric(edhec[,"Funds of Funds"])
mean.stderr(edhec[,"Funds of Funds"])
mean.UCL(edhec[,"Funds of Funds"])
mean.LCL(edhec[,"Funds of Funds"])
```

Omega *calculate Omega for a return series*

Description

Keating and Shadwick (2002) proposed Omega (referred to as Gamma in their original paper) as a way to capture all of the higher moments of the returns distribution.

Usage

```
Omega(R, L = 0, method = c("simple", "interp", "binomial", "blackscholes"), output
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
L	L is the loss threshold that can be specified as zero, return from a benchmark index, or an absolute rate of return - any specified level
method	one of: simple, interp, binomial, blackscholes
output	one of: point (in time), or full (distribution of Omega)
Rf	risk free rate, in same period as your returns
...	any other passthru parameters

Details

Mathematically, Omega is: integral[L to b](1 - F(r))dr / integral[a to L](F(r))dr

where the cumulative distribution F is defined on the interval (a,b). L is the loss threshold that can be specified as zero, return from a benchmark index, or an absolute rate of return - any specified level. When comparing alternatives using Omega, L should be common.

Input data can be transformed prior to calculation, which may be useful for introducing risk aversion.

This function returns a vector of Omega, useful for plotting. The steeper, the less risky. Above it's mean, a steeply sloped Omega also implies a very limited potential for further gain.

Omega has a value of 1 at the mean of the distribution.

Omega is sub-additive. The ratio is dimensionless.

Kazemi, Schneeweis, and Gupta (2003), in "Omega as a Performance Measure" show that Omega can be written as: Omega(L) = C(L)/P(L) where C(L) is essentially the price of a European call option written on the investment and P(L) is essentially the price of a European put option written on the investment. The maturity for both options is one period (e.g., one month) and L is the strike price of both options.

The numerator and the denominator can be expressed as: exp(-Rf=0) * E[max(x - L, 0)] exp(-Rf=0) * E[max(L - x, 0)] with exp(-Rf=0) calculating the present values of the two, where rf is the per-period riskless rate.

The first three methods implemented here focus on that observation. The first method takes the simplification described above. The second uses the Black-Scholes option pricing as implemented in fOptions. The third uses the binomial pricing model from fOptions. The second and third methods are not implemented here.

The fourth method, "interp", creates a linear interpolation of the cdf of returns, calculates Omega as a vector, and finally interpolates a function for Omega as a function of L. This method requires library `Hmisc`, which can be found on CRAN.

Author(s)

Peter Carl

References

Keating, J. and Shadwick, W.F. The Omega Function. working paper. Finance Development Center, London. 2002. Kazemi, Schneeweis, and Gupta. Omega as a Performance Measure. 2003.

See Also

`Ecdf`

Examples

```
data(edhec)
Omega(edhec)
Omega(edhec[,13],method="interp",output="point")
Omega(edhec[,13],method="interp",output="full")
```

---

prices *Selected Price Series Example Data*

Description

A object returned by get.hist.quote of price data for use in the example for `Return.calculate`

Usage

```
prices
```

Format

R variable 'prices'

Examples

```
data(prices)

#preview the data
head(prices)
```

`Return.annualized` *calculate an annualized return for comparing instruments with different length history*

Description

An average annualized return is convenient for comparing returns.

Usage

```
Return.annualized(R, scale = NA, geometric = TRUE)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`scale`	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
`geometric`	generate geometric (TRUE) or simple (FALSE) returns, default TRUE

Details

Annualized returns are useful for comparing two assets. To do so, you must scale your observations to an annual scale by raising the compound return to the number of periods in a year, and taking the root to the number of total observations:

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

where scale is the number of periods in a year, and n is the total number of periods for which you have observations.

For simple returns (geometric=FALSE), the formula is:

$$\overline{R_a} \cdot scale$$

Author(s)

Peter Carl

References

Bacon, Carl. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 6

See Also

`Return.cumulative`,

Examples

```
data(managers)
Return.annualized(managers[,1,drop=FALSE])
Return.annualized(managers[,1:8])
Return.annualized(managers[,1:8],geometric=FALSE)
```

Return.calculate *calculate simple or compound returns from prices*

Description

calculate simple or compound returns from prices

Usage

```
Return.calculate(prices, method=c("compound","simple"))
CalculateReturns(prices, method=c("compound","simple"))
```

Arguments

prices	data object containing ordered price observations
method	calculate "simple" or "compound" returns, default compound

Details

Two requirements should be made clear. First, the function `Return.calculate` assumes regular price data. In this case, we downloaded monthly close prices. Prices can be for any time scale, such as daily, weekly, monthly or annual, as long as the data consists of regular observations. Irregular observations require time period scaling to be comparable. Fortunately, `to.period` in the `xts` package, or the `aggregate.zoo` in the `zoo` package supports supports management and conversion of irregular time series.

Second, if corporate actions, dividends, or other adjustments such as time- or money-weighting are to be taken into account, those calculations must be made separately. This is a simple function that assumes fully adjusted close prices as input. For the IBM timeseries in the example below, dividends and corporate actions are not contained in the "close" price series, so we end up with "price returns" instead of "total returns". This can lead to significant underestimation of the return series over longer time periods. To use adjusted returns, specify `quote="AdjClose"` in `get.hist.quote`, which is found in package `tseries`.

Author(s)

Peter Carl

References

Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. Chapter 2

See Also

Return.cumulative

Examples

```
  ## Not run:
    require(tseries)
    prices = get.hist.quote("IBM", start = "1999-01-01", end = "2007-01-01", quote = "AdjClc

## End(Not run)

R.IBM = Return.calculate(prices, method="simple")
R.IBM = as.xts(R.IBM)
colnames(R.IBM)="IBM"
chart.CumReturns(R.IBM,legend.loc="topleft", main="Cumulative Daily Returns for IBM")
round(R.IBM,2)
```

Return.centered *calculate centered Returns*

Description

the n-th centered moment is calculated as

$$\mu^{(n)}(R) = E[(R - E(R))^n]$$

These functions are used internally by PerformanceAnalytics to calculate centered moments for a multivariate distribution as well as the standardized moments of a portfolio distribution. They are exposed here for users who wish to use them directly, and we'll get more documentation written when we can.

Usage

```
centeredcomoment (Ra, Rb, p1, p2, normalize = FALSE)
centeredmoment (R, power)
Return.centered (R, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	an xts, vector, matrix, data frame, timeSeries or zoo object of index, benchmark, portfolio, or secondary asset returns to compare against
power	power or moment to calculate
p1	first power of the comoment
p2	second power of the comoment
normalize	whether to standardize the calculation to agree with common usage, or leave the default mathematical meaning
...	any other passthru parameters

Details

These functions were first utilized in Boudt, Peterson, and Croux (2008), and have been subsequently used in our other research.

~~ Additional Details will be added to documentation as soon as we have time to write them. ~~

Author(s)

Kris Boudt and Brian Peterson

References

Boudt, Kris, Brian G. Peterson, and Christophe Croux. 2008. Estimation and Decomposition of Downside Risk for Portfolios with Non-Normal Returns. Journal of Risk. Winter.

Martellini, Lionel, and Volker Ziemann. 2007. Improved Forecasts of Higher-Order Comoments and Implications for Portfolio Selection. EDHEC Risk and Asset Management Research Centre working paper.

Ranaldo, Angelo, and Laurent Favre Sr. 2005. How to Price Hedge Funds: From Two- to Four-Moment CAPM. SSRN eLibrary.

Scott, Robert C., and Philip A. Horvath. 1980. On the Direction of Preference for Moments of Higher Order than the Variance. Journal of Finance 35(4):915-919.

Examples

```
data(managers)
Return.centered(managers[,1:3,drop=FALSE])
```

`Return.clean` *clean returns in a time series to to provide more robust risk estimates*

Description

A function that provides access to multiple methods for cleaning outliers from return data.

Usage

```
Return.clean(R, method = c("none", "boudt", "geltner"), alpha = .01, ...)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`method`	one of "none", "boudt", which applies the function `clean.boudt` or "geltner" which applies the function `Return.Geltner`to R
`alpha`	the percentage of outliers you want to clean
`...`	additional parameters passed into the underlying cleaning function

Details

This is a wrapper for offering multiple data cleaning methods for data objects containing returns.

The primary value of data cleaning lies in creating a more robust and stable estimation of the distribution generating the large majority of the return data. The increased robustness and stability of the estimated moments using cleaned data should be used for portfolio construction. If an investor wishes to have a more conservative risk estimate, cleaning may not be indicated for risk monitoring.

In actual practice, it is probably best to back-test the results of both cleaned and uncleaned series to see what works best when forecasting risk with the particular combination of assets under consideration.

In this version, only one method is supported. See `clean.boudt` for more details.

Author(s)

Peter Carl

See Also

`clean.boudt`
`Return.Geltner`

Examples

```
data(managers)
head(Return.clean(managers[,1:4]),n=20)
chart.BarVaR(managers[,1,drop=FALSE], show.clean=TRUE, clean="boudt", lwd=2, methods="Modifi
```

`Return.cumulative` *calculate a compounded (geometric) cumulative return*

Description

This is a useful function for calculating cumulative return over a period of time, say a calendar year. Can produce simple or geometric return.

Usage

```
Return.cumulative(R, geometric = TRUE)
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`geometric`	generate geometric (TRUE) or simple (FALSE) returns, default TRUE

Details

product of all the individual period returns

$$(1 + r_1)(1 + r_2)(1 + r_3)\dots(1 + r_n) - 1 = prod(1 + R) - 1$$

Author(s)

Peter Carl

References

Bacon, Carl. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 6

See Also

Return.annualized

Examples

```
data(managers)
Return.cumulative(managers[,1,drop=FALSE])
Return.cumulative(managers[,1:8])
Return.cumulative(managers[,1:8],geometric=FALSE)
```

Return.excess *Calculates 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.

Ideally, your risk free rate will be for each period you have returns observations, but a single average return for the period will work too.

Usage

```
Return.excess(R, Rf=0)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rf	risk free rate, in same period as your returns, or as a sinlge digit average

Details

Mean of the period return minus the period risk free rate

$$\overline{(R_a - R_f)}$$

OR

mean of the period returns minus a single numeric risk free rate

$$\overline{R_a} - R_f$$

Note that while we have, in keeping with common academic usage, assumed that the second parameter will be a risk free rate, you may also use any other timeseries as the second argument. A common alteration would be to use a benchmark to produce excess returns over a specific benchmark, as demonstrated in the examples below.

Author(s)

Peter Carl

References

Bacon, Carl. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 47-52

Examples

```
data(managers)
head(Return.excess(managers[,1,drop=FALSE], managers[,10,drop=FALSE]))
head(Return.excess(managers[,1,drop=FALSE], .04/12))
head(Return.excess(managers[,1:6], managers[,10,drop=FALSE]))
head(Return.excess(managers[,1,drop=FALSE], managers[,8,drop=FALSE]))
```

Return.Geltner *calculate Geltner liquidity-adjusted return series*

Description

David Geltner developed a method to remove estimating or liquidity bias in real estate index returns. It has since been applied with success to other return series that show autocorrelation or illiquidity effects.

The theory is that by correcting for autocorrelation, you are uncovering a "true" return from a series of observed returns that contain illiquidity or manual pricing effects.

Usage

```
Return.Geltner(Ra, ...)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
...	any other passthru parameters

Details

The Geltner autocorrelation adjusted return series may be calculated via:

$$R_G = \frac{R_t - (R_{t-1} \cdot \rho_1)}{1 - \rho_1}$$

where ρ_1 is the first-order autocorrelation of the return series R_a and R_t is the return of R_a at time t and R_{t-1} is the one-period lagged return.

Author(s)

Brian Peterson

References

"Edhec Funds of Hedge Funds Reporting Survey : A Return-Based Approach to Funds of Hedge Funds Reporting",Edhec Risk and Asset Management Research Centre, January 2005,p. 27

Geltner, David, 1991, Smoothing in Appraisal-Based Returns, Journal of Real Estate Finance and Economics, Vol.4, p.327-345.

Geltner, David, 1993, Estimating Market Values from Appraised Values without Assuming an Efficient Market, Journal of Real Estate Research, Vol.8, p.325-345.

Examples

```
data(managers)
head(Return.Geltner(managers[,1:3]),n=20)
```

Return.portfolio *Calculates weighted returns for a portfolio of assets*

Description

Calculates weighted returns for a portfolio of assets. If you have a single weighting vector, or want the equal weighted portfolio, use `Return.portfolio`. If you have a portfolio that is periodically rebalanced, and multiple time periods with different weights, use `Return.rebalancing`. Both functions will subset the return series to only include returns for assets for which `weight` is provided.

Usage

```
Return.portfolio(R, weights = NULL, wealth.index = FALSE, contribution = FALSE, met
Return.rebalancing(R, weights, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
weights	a time series or sinlge-row matrix/vector containing asset weights, as percentages
wealth.index	TRUE/FALSE whether to return a wealth index, default FALSE
contribution	if contribution is TRUE, add the weighted return contributed by the asset in this period
method	"simple" or "compound"
...	any other passthru parameters

Value

returns a time series of returns weighted by the weights parameter, possibly including contribution for each period

Author(s)

Brian G. Peterson

References

Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. Chapter 2

See Also

Return.calculate

Examples

```
data(edhec)
data(weights)

# calculate an equal weighted portfolio return
Return.portfolio(edhec)

# now return the contribution too
Return.portfolio(edhec,contribution=TRUE)

# calculate a portfolio return with rebalancing
Return.rebalancing(edhec,weights)
```

`Return.read` *Read returns data with different date formats*

Description

A simple wrapper of read.zoo with some defaults for different date formats and xts conversion

Usage

```
Return.read(filename = stop("Please specify a filename"),
                frequency = c("d", "m", "q", "i", "o"), format.in =
                c("ISO8601", "excel", "oo", "gnumeric"), sep = ",",
                header = TRUE, check.names = FALSE, ...)
```

Arguments

`filename`	the name of the file to be read
`frequency`	• "d" sets as a daily timeseries using `as.Date`, • "m" sets as monthly timeseries using `as.yearmon`, • "q" sets as a quarterly timeseries using `as.yearqtr`, and • "i" sets as irregular timeseries using `as.POSIXct`
`format.in`	says how the data being read is formatted. Although the default is set to the ISO 8601 standard (which can also be set as "%F"), most spreadsheets have less sensible date formats as defaults. See below.
`sep`	separator, default is ","
`header`	a logical value indicating whether the file contains the names of the variables as its first line.
`check.names`	logical. If TRUE then the names of the variables in the data frame are checked to ensure that they are syntactically valid variable names. If necessary they are adjusted (by make.names) so that they are, and also to ensure that there are no duplicates. See `read.table`
`...`	passes through any other parameters to `read.zoo`

Details

The parameter 'format.in' takes several values, including:

- excel: default date format for MS Excel spreadsheet csv format, which is "%m/%d/%Y",
- oo: default date format for OpenOffice spreadsheet csv format, "%m/%d/%y", although this may be operating system dependent,
- gnumeric: default date format for Gnumeric spreadsheet, which is "%d-%b-%Y",
- ...: alternatively, any specific format may be passed in, such as "%M/%y"

Author(s)

Peter Carl

See Also

read.zoo, read.table

Examples

```
## Not run:
Return.read("managers.cvs", frequency="d")

## End(Not run)
```

Return.relative *calculate the relative return of one asset to another*

Description

Calculates the ratio of the cumulative performance for two assets through time.

Usage

```
Return.relative(Ra, Rb, ...)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return object for the benchmark asset
...	ignored

Value

xts or other time series of relative return

Author(s)

Peter Carl

See Also

chart.RelativePerformance

Examples

```
data(managers)
head(Return.relative(managers[,1:3], managers[,8,drop=FALSE]),n=20)
```

`sd.multiperiod` *calculate a multiperiod or annualized Standard Deviation*

Description

Standard Deviation of a set of observations R_a is given by:

$$\sigma = variance(R_a), std = \sqrt{\sigma}$$

It should follow that the variance is not a linear function of the number of observations. To determine possible variance over multiple periods, it wouldn't make sense to multiply the single-period variance by the total number of periods: this could quickly lead to an absurd result where total variance (or risk) was greater than 100%. It follows then that the total variance needs to demonstrate a decreasing period-to-period increase as the number of periods increases. Put another way, the increase in incremental variance per additional period needs to decrease with some relationship to the number of periods. The standard accepted practice for doing this is to apply the inverse square law. 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.

$$\sqrt{\sigma} \cdot \sqrt{periods}$$

Note that any multiperiod or annualized number should be viewed with suspicion if the number of observations is small.

Usage

```
sd.multiperiod(x, scale = NA)
sd.annualized(x, scale = NA)
StdDev.annualized(R, scale = NA)
```

Arguments

`x, R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`scale`	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)

Author(s)

Brian G. Peterson

References

Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 27

See Also

sd
http://wikipedia.org/wiki/inverse-square_law

Examples

```
data(edhec)
sd.annualized(edhec)
sd.annualized(edhec[,6,drop=FALSE])
# now for three periods:
sd.multiperiod(edhec[,6,drop=FALSE],scale=3)
```

SharpeRatio *Sharpe Ratio*

Description

The Sharpe Ratio is a risk-adjusted measure of return that uses standard deviation to represent risk.

Usage

```
SharpeRatio(Ra, Rf = 0)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rf	risk free rate, in same period as your returns

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.

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

William Sharpe now recommends InformationRatio preferentially to the original Sharpe Ratio.

Author(s)

Peter Carl

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.

See Also

SharpeRatio.annualized
InformationRatio
TrackingError
ActivePremium
SortinoRatio

Examples

```
data(managers)
SharpeRatio(managers[,1,drop=FALSE], Rf=.035/12)
SharpeRatio(managers[,1,drop=FALSE], Rf = managers[,10,drop=FALSE])
SharpeRatio(managers[,1:6], Rf=.035/12)
SharpeRatio(managers[,1:6], Rf = managers[,10,drop=FALSE])
```

SharpeRatio.annualized

calculate annualized Sharpe Ratio

Description

The Sharpe Ratio is a risk-adjusted measure of return that uses standard deviation to represent risk.

Usage

```
SharpeRatio.annualized(R, Rf = 0, scale = NA, geometric=TRUE)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rf	risk free rate, in same period as your returns
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
geometric	generate geometric (TRUE) or simple (FALSE) returns, default TRUE

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 function annualizes the number based on the scale parameter.

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

Using an annualized Sharpe Ratio is useful for comparison of multiple return streams. The annualized Sharpe ratio is computed by dividing the annualized mean monthly excess return by the annualized monthly standard deviation of excess return.

William Sharpe now recommends Information Ratio preferentially to the original Sharpe Ratio.

Author(s)

Peter Carl

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.

See Also

SharpeRatio
InformationRatio
TrackingError
ActivePremium
SortinoRatio

Examples

```
data(managers)
SharpeRatio.annualized(managers[,1,drop=FALSE], Rf=.035/12)
SharpeRatio.annualized(managers[,1,drop=FALSE], Rf = managers[,10,drop=FALSE])
SharpeRatio.annualized(managers[,1:6], Rf=.035/12)
SharpeRatio.annualized(managers[,1:6], Rf = managers[,10,drop=FALSE])
SharpeRatio.annualized(managers[,1:6], Rf = managers[,10,drop=FALSE],geometric=FALSE)
```

SharpeRatio.modified

calculate a modified Sharpe Ratio of Return over VaR or ES

Description

The Sharpe ratio is simply the return per unit of risk (represented by variability). The higher the Sharpe ratio, the better the combined performance of "risk" and return.

The Sharpe Ratio is a risk-adjusted measure of return that uses standard deviation to represent risk.

A number of papers now recommend using a "modified Sharpe" ratio using a Modified Cornish-Fisher VaR as the measure of Risk.

We have recently extended this concept to create multivariate modified Sharpe-like Ratios for standard deviation, Gaussian VaR, modified VaR, Gaussian Expected Shortfall, and modified Expected Shortfall. See VaR and ES.

This function returns a modified Sharpe ratio for the same periodicity of the data being input (e.g., monthly data -> monthly SR)

Usage

```
SharpeRatio.modified(R, Rf = 0, p = 0.95, FUNCT = c("VaR","ES"), ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rf	risk free rate, in same period as your returns
p	confidence level for calculation, default p=.95
FUNCT	one of "VaR" or "ES" to use as the denominator
...	any other passthru parameters to the VaR or ES functions

Author(s)

Brian G. Peterson

References

Laurent Favre and Jose-Antonio Galeano. Mean-Modified Value-at-Risk Optimization with Hedge Funds. Journal of Alternative Investment, Fall 2002, v 5.

See Also

VaR
ES
SharpeRatio

Examples

```
data(edhec)
SharpeRatio.modified(edhec[, 6, drop = FALSE])
SharpeRatio.modified(edhec[, 6, drop = FALSE], Rf = .04/12)
SharpeRatio.modified(edhec[, 6, drop = FALSE], Rf = .04/12, method="gaussian")
SharpeRatio.modified(edhec[, 6, drop = FALSE], FUNCT="ES")
```

---

skewness *Skewness*

Description

compute skewness of a univariate distribution.

Usage

```
skewness(x, na.rm = FALSE, method = c("moment", "fisher"), ...)
```

Arguments

na.rm	a logical. Should missing values be removed?
method	a character string which specifies the method of computation. These are either "moment" or "fisher" The "moment" method is based on the definitions of skewnessfor distributions; these forms should be used when resampling (bootstrap or jackknife). The "fisher" method correspond to the usual "unbiased" definition of sample variance, although in the case of skewness exact unbiasedness is not possible.
x	a numeric vector or object.
...	arguments to be passed.

Details

This function was ported from the RMetrics package fUtilities to eliminate a dependency on fUtilties being loaded every time. The function is identical except for the addition of checkData and column support.

Author(s)

Diethelm Wuertz

See Also

kurtosis

Examples

```
## mean -
## var -
   # Mean, Variance:
   r = rnorm(100)
   mean(r)
   var(r)

## skewness -
   skewness(r)
data(managers)
skewness(managers)
```

SmoothingIndex *calculate Normalized Getmansky Smoothing Index*

Description

Proposed by Getmansky et al to provide a normalized measure of "liquidity risk."

Usage

```
SmoothingIndex(R, neg.thetas = FALSE, MAorder = 2, verbose=FALSE, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
neg.thetas	if FALSE, function removes negative coefficients (thetas) when calculating the index
MAorder	specify the number of periods used to calculate the moving average, defaults to 2
verbose	if TRUE, return a list containing the Thetas in addition to the smoothing index/
...	any other passthru parameters

Details

To measure the effects of smoothing, Getmansky, Lo, et al (2004) define a "smoothing profile" as a vector of coefficients for an MLE fit on returns using a two-period moving-average process.

The moving-average process of order $k = 2$ (specified using `MAorder`) gives $R_t = \theta_0 R_t + \theta_1 R_{t-1} + \theta_2 R_{t-2}$, under the constraint that the sum of the coefficients is equal to 1. In R, the `arima` function allows us to create an MA(2) model using an "ARIMA(p,d,q)" model, where p is the number of autoregressive terms (AR), d is the degree of differencing, and q is the number of lagged forecast errors (MA) in the prediction equation. The `order` parameter allows us to specify the three components (p, d, q) as an argument, e.g., `order = c(0, 0, 2)`. The `method` specifies how to fit the model, in this case using maximum likelihood estimation (MLE) in a fashion similar to the estimation of standard moving-average time series models, using:

```
arima(ra, order=c(0,0,2), method="ML", transform.pars=TRUE, include.mean=FALSE)
```

`include.mean`: Getmansky, et al. (2004) p 555 "By applying the above procedure to observed de-meaned returns...", so we set that parameter to 'FALSE'.

`transform.pars`: ibid, "we impose the additional restriction that the estimated MA(k) process be invertible," so we set the parameter to 'TRUE'.

The coefficients, θ_j, are then normalized to sum to interpreted as a "weighted average of the fund's true returns over the most recent $k + 1$ periods, including the current period."

If these weights are disproportionately centered on a small number of lags, relatively little serial correlation will be induced. However, if the weights are evenly distributed among many lags, this would show higher serial correlation.

The paper notes that because $\theta_j \in [0, 1]$, ξ is also confined to the unit interval, and is minimized when all the θ_j's are identical. That implies a value of $1/(k + 1)$ for ξ, and a maximum value of $\xi = 1$ when one coefficient is 1 and the rest are 0. In the context of smoothed returns, a lower value of ξ implies more smoothing, and the upper bound of 1 implies no smoothing.

The "smoothing index," represented as ξ, is calculated the same way the Herfindahl index. The Herfindal measure is well known in the industrial organization literature as a measure of the concentration of firms in a given industry where y_j represents the market share of firm j.

This method (as well as the implementation described in the paper), does not enforce $\theta_j \in [0, 1]$, so ξ is not limited to that range either. All we can say is that lower values are "less liquid" and higher

values are "more liquid" or mis-specified. In this function, setting the parameter neg.thetas = FALSE does enforce the limitation, eliminating negative autocorrelation coefficients from the calculation (the papers below do not make an economic case for eliminating negative autocorrelation, however).

Interpretation of the resulting value is difficult. All we can say is that lower values appear to have autocorrelation structure like we might expect of "less liquid" instruments. Higher values appear "more liquid" or are poorly fit or mis-specified.

Acknowledgments

Thanks to Dr. Stefan Albrecht, CFA, for invaluable input.

Author(s)

Peter Carl

References

Chan, Nicholas, Mila Getmansky, Shane M. Haas, and Andrew W. Lo. 2005. Systemic Risk and Hedge Funds. NBER Working Paper Series (11200). Getmansky, Mila, Andrew W. Lo, and Igor Makarov. 2004. An Econometric Model of Serial Correlation and Illiquidity in Hedge Fund Returns. Journal of Financial Economics (74): 529-609.

Examples

```
data(managers)
data(edhec)
SmoothingIndex(managers[,1,drop=FALSE])
SmoothingIndex(managers[,1:8])
SmoothingIndex(edhec)
```

---

sortDrawdowns *order list of drawdowns from worst to best*

Description

sortDrawdowns(findDrawdowns(R)) Gives the drawdowns in order of worst to best

Usage

```
sortDrawdowns(runs)
```

Arguments

runs	pass in runs array from findDrawdowns to be sorted

Details

Returns a sorted list:

return	depth of drawdown
from	starting period
to	ending period
length	length in periods

Author(s)

Peter Carl
modified with permission from prototype function by Sankalp Upadhyay

References

Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 88

See Also

DownsideDeviation
maxDrawdown
findDrawdowns
sortDrawdowns
chart.Drawdown
table.Drawdowns
table.DownsideRisk

Examples

```
data(edhec)
findDrawdowns(edhec[,"Funds of Funds", drop=FALSE])
sortDrawdowns(findDrawdowns(edhec[,"Funds of Funds", drop=FALSE]))
```

SortinoRatio *calculate Sortino Ratio of performance over downside risk*

Description

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

Usage

```
SortinoRatio(R, MAR = 0)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
MAR	Minimum Acceptable Return, in the same periodicity as your returns

Details

Sortino contends that risk should be measured in terms of not meeting the investment goal. This gives rise to the notion of "Minimum Acceptable Return" or MAR. All of Sortino's proposed measures include the MAR, and are more sensitive to downside or extreme risks than measures that use volatility(standard deviation of returns) as the measure of risk.

Choosing the MAR carefully is very important, especially when comparing disparate investment choices. If the MAR is too low, it will not adequately capture the risks that concern the investor, and if the MAR is too high, it will unfavorably portray what may otherwise be a sound investment. When comparing multiple investments, some papers recommend using the risk free rate as the MAR. Practitioners may wish to choose one MAR for consistency, several standardized MAR values for reporting a range of scenarios, or a MAR customized to the objective of the investor.

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

where δ_{MAR} is the DownsideDeviation.

Author(s)

Brian G. Peterson

References

Sortino, F. and Price, L. Performance Measurement in a Downside Risk Framework. *Journal of Investing*. Fall 1994, 59-65.

See Also

SharpeRatio
DownsideDeviation
SemiVariance
SemiDeviation
InformationRatio

Examples

```
data(managers)
SortinoRatio(managers[, 1])
SortinoRatio(managers[, 1:8])
```

table.AnnualizedReturns *Annualized Returns Summary: Statistics and Stylized Facts*

Description

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

Usage

```
table.AnnualizedReturns(R, scale = NA, Rf = 0, geometric = TRUE, digits = 4)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
geometric	generate geometric (TRUE) or simple (FALSE) returns, default TRUE
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
Rf	risk free rate, in same period as your returns
digits	number of digits to round results to

Author(s)

Peter Carl

See Also

Return.annualized
StdDev.annualized
SharpeRatio.annualized

Examples

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

require("Hmisc")
result = t(table.AnnualizedReturns(managers[,1:8], Rf=.04/12))

textplot(format.df(result, na.blank=TRUE, numeric.dollar=FALSE, cdec=c(3,3,1)), rmar = 0.8,
title(main="Annualized Performance")
```

`table.Arbitrary` *wrapper function for combining arbitrary function list into a table*

Description

This function creates a table of statistics from vectors of functions and labels passed in. The resulting table is formatted such that metrics are calculated separately for each column of returns in the data object.

Assumes an input of period returns. Scale arguements can be used to specify the number of observations during a year (e.g., 12 = monthly returns).

Usage

```
table.Arbitrary(R, metrics = c("mean", "sd"), metricsNames = c("Average Return", "S
statsTable(R, metrics = c("mean", "sd"), metricsNames = c("Average Return", "Standa
```

Arguments

`R`	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
`metrics`	lisdt of functions to apply
`metricsNames`	column names for each function
`...`	any other passthru parameters

Details

The idea here is to be able to pass in sets of metrics and values, like:

metrics = c(DownsideDeviation(x,MAR=mean(x)), sd(subset(x,x>0)), sd(subset(x,x<0)), DownsideDeviation(x,MAR=MAR), DownsideDeviation(x,MAR=Rf=0), DownsideDeviation(x,MAR=0),maxDrawdown(x))

metricsNames = c("Semi Deviation", "Gain Deviation", "Loss Deviation", paste("Downside Deviation (MAR=",MAR*scale*100,"%)", sep=""), paste("Downside Deviation (rf=",rf*scale*100,"%)", sep=""), paste("Downside Deviation (0%)", sep=""), "Maximum Drawdown")

Here's how it's working right now: > table.Arbitrary(monthlyReturns.ts,metrics=c("VaR","mean"), metricsNames=c("modVaR","mean"),p=.95)

```
           Actual   S&P500TR
modVaR 0.04186461 0.06261451
mean   0.00945000 0.01013684
```

Passing in two different sets of attributes to the same function doesn't quite work currently. The issue is apparent in: > table.Arbitrary(edhec,metrics=c("VaR", "VaR"), metricsNames=c("Modified VaR","Traditional VaR"), modified=c(TRUE,FALSE))

```
                Convertible.Arbitrage CTA.Global Distressed.Securities
Modified VaR               0.04081599  0.0456767             0.1074683
Traditional VaR            0.04081599  0.0456767             0.1074683
```

```
                Emerging.Markets Equity.Market.Neutral Event.Driven
Modified VaR          0.1858624            0.01680917    0.1162714
Traditional VaR       0.1858624            0.01680917    0.1162714
                Fixed.Income.Arbitrage Global.Macro Long.Short.Equity
Modified VaR                 0.2380379   0.03700478        0.04661244
Traditional VaR              0.2380379   0.03700478        0.04661244
                Merger.Arbitrage Relative.Value Short.Selling Funds.of.Funds
Modified VaR          0.07510643     0.04123920     0.1071894     0.04525633
Traditional VaR       0.07510643     0.04123920     0.1071894     0.04525633
```

In the case of this example, you would simply call VaR as the second function, like so: > table.Arbitrary(edhec,metrics=c("VaR", "VaR"),metricsNames=c("Modified VaR","Traditional VaR"))

```
                Convertible.Arbitrage CTA.Global Distressed.Securities
Modified VaR               0.04081599 0.04567670            0.10746831
Traditional VaR            0.02635371 0.04913361            0.03517855
                Emerging.Markets Equity.Market.Neutral Event.Driven
Modified VaR          0.18586240            0.01680917   0.11627142
Traditional VaR       0.07057278            0.01746554   0.03563019
                Fixed.Income.Arbitrage Global.Macro Long.Short.Equity
Modified VaR                 0.23803787   0.03700478        0.04661244
Traditional VaR              0.02231236   0.03692096        0.04318713
                Merger.Arbitrage Relative.Value Short.Selling Funds.of.Funds
Modified VaR          0.07510643     0.04123920     0.1071894     0.04525633
Traditional VaR       0.02510709     0.02354012     0.0994635     0.03502065
```

but we don't know of a way to compare the same function side by side with different parameters for each. Suggestions Welcome.

Author(s)

Peter Carl

Examples

```
data(edhec)
table.Arbitrary(edhec,metrics=c("VaR", "ES"),metricsNames=c("Modified VaR","Modified Expecte
```

`table.Autocorrelation` *table for calculating the first six autocorrelation coefficients and significance*

Description

Produces data table of autocorrelation coefficients ρ and corresponding Q(6)-statistic for each column in R.

Usage

```
table.Autocorrelation(R, digits = 4)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
digits	number of digits to round results to for display

Note

To test returns for autocorrelation, Lo (2001) suggests the use of the Ljung-Box test, a significance test for the auto-correlation coefficients. Ljung and Box (1978) provide a refinement of the Q-statistic proposed by Box and Pierce (1970) that offers a better fit for the χ^2 test for small sample sizes. `Box.test` provides both.

Author(s)

Peter Carl

References

Lo, Andrew W. 2001. Risk Management for Hedge Funds: Introduction and Overview. SSRN eLibrary.

See Also

`Box.test`, `acf`

Examples

```
data(managers)
t(table.Autocorrelation(managers))

result = t(table.Autocorrelation(managers[,1:8]))
textplot(result, rmar = 0.8, cmar = 2,  max.cex=.9, halign = "center", valign = "top", row.v
title(main="Autocorrelation")
```

---

table.CalendarReturns

Monthly and Calendar year Return table

Description

Returns a table of returns formatted with years in rows, months in columns, and a total column in the last column. For additional columns in `R`, annual returns will be appended as columns.

Usage

```
table.CalendarReturns(R, digits = 1, as.perc = TRUE)
table.Returns(R, digits = 1, as.perc = TRUE)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
digits	number of digits to round results to for presentation
as.perc	TRUE/FALSE if TRUE, multiply simple returns by 100 to get %

Note

This function assumes monthly returns and does not currently have handling for other scales.

This function defaults to the first column as the monthly returns to be formatted.

Author(s)

Peter Carl

Examples

```
data(managers)
t(table.CalendarReturns(managers[,c(1,7,8)]))

# prettify with format.df in hmisc package
require("Hmisc")
result = t(table.CalendarReturns(managers[,c(1,8)]))
textplot(format.df(result, na.blank=TRUE, numeric.dollar=FALSE, cdec=rep(1,dim(result)[2])),
title(main="Calendar Returns")
```

---

table.CAPM *Asset-Pricing Model Summary: Statistics and Stylized Facts*

Description

Takes a set of returns and relates them to a market benchmark. Provides a set of measures related to the excess return single index model, or CAPM.

Usage

```
table.CAPM(Ra, Rb, scale = NA, Rf = 0, digits = 4)
```

Arguments

Ra	a vector of returns to test, e.g., the asset to be examined
Rb	a matrix, data.frame, or timeSeries of benchmark(s) to test the asset against.
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
Rf	risk free rate, in same period as your returns
digits	number of digits to round results to

Details

This table will show statistics pertaining to an asset against a set of benchmarks, or statistics for a set of assets against a benchmark.

Author(s)

Peter Carl

See Also

CAPM.alpha
CAPM.beta
TrackingError
ActivePremium
InformationRatio
TreynorRatio

Examples

```
data(managers)
table.CAPM(managers[,1:3,drop=FALSE], managers[,8,drop=FALSE], Rf = managers[,10,drop=FALSE]

result = table.CAPM(managers[,1:3,drop=FALSE], managers[,8,drop=FALSE], Rf = managers[,10,dr
textplot(result, rmar = 0.8, cmar = 1.5,  max.cex=.9, halign = "center", valign = "top", row
title(main="CAPM-Related Statistics")
```

---

table.CaptureRatios

Calculate and display a table of capture ratio and related statistics

---

Description

Creates a table of capture ratios and similar metrics for a set of returns against a benchmark.

Usage

```
table.CaptureRatios(Ra, Rb, digits = 4)
table.UpDownRatios(Ra, Rb, digits = 4)
```

Arguments

Ra	a vector of returns to test, e.g., the asset to be examined
Rb	a matrix, data.frame, or timeSeries of benchmark(s) to test the asset against.
digits	number of digits to round results to for presentation

Details

This table will show statistics pertaining to an asset against a set of benchmarks, or statistics for a set of assets against a benchmark. `table.CaptureRatios` shows only the capture ratio; `table.UpDownRatios` shows three: the capture ratio, the number ratio, and the percentage ratio.

Author(s)

Peter Carl

See Also

`UpDownRatios`, `chart.CaptureRatios`

Examples

```
data(managers)
table.CaptureRatios(managers[,1:6], managers[,7,drop=FALSE])
table.UpDownRatios(managers[,1:6], managers[,7,drop=FALSE])

result = t(table.UpDownRatios(managers[,1:6], managers[,7,drop=FALSE]))
colnames(result)=colnames(managers[,1:6])
textplot(result, rmar = 0.8, cmar = 1.5,  max.cex=.9, halign = "center", valign = "top", row
title(main="Capture Ratios for EDHEC LS EQ")
```

---

table.Correlation *calculate correlalations of multicolumn data*

Description

This is a wrapper for calculating correlation and significance against each column of the data provided.

Usage

```
table.Correlation(Ra, Rb, ...)
```

Arguments

Ra	a vector of returns to test, e.g., the asset to be examined
Rb	a matrix, data.frame, or timeSeries of benchmark(s) to test the asset against.
...	any other passthru parameters to cor.test

Author(s)

Peter Carl

See Also

cor.test

Examples

```
# First we load the data
data(managers)
table.Correlation(managers[,1:6],managers[,7:8])

result=table.Correlation(managers[,1:6],managers[,8])
rownames(result)=colnames(managers[,1:6])
require("Hmisc")
textplot(format.df(result, na.blank=TRUE, numeric.dollar=FALSE, cdec=rep(3,dim(result)[2])),
title(main="Correlations to SP500 TR")

ctable = table.Correlation(managers[,1:6],managers[,8,drop=FALSE], conf.level=.99)
dotchart(ctable[,1],labels=rownames(ctable),xlim=c(-1,1))
```

---

table.DownsideRisk *Downside Risk Summary: Statistics and Stylized Facts*

Description

Creates a table of estimates of downside risk measures for comparison across multiple instruments or funds.

Usage

```
table.DownsideRisk(R, ci = 0.95, scale = NA, Rf = 0, MAR = 0.1/12, p = 0.95, digits
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
ci	confidence interval, defaults to 95%
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
Rf	risk free rate, in same period as your returns

MAR	Minimum Acceptable Return, in the same periodicity as your returns
p	confidence level for calculation, default p=.99
digits	number of digits to round results to

Author(s)

Peter Carl

See Also

DownsideDeviation
maxDrawdown
VaR
ES

Examples

```
data(edhec)
table.DownsideRisk(edhec, Rf=.04/12, MAR =.05/12, p=.95)

result=t(table.DownsideRisk(edhec, Rf=.04/12, MAR =.05/12, p=.95))
require("Hmisc")
textplot(format.df(result, na.blank=TRUE, numeric.dollar=FALSE, cdec=rep(3,dim(result)[2])),
title(main="Downside Risk Statistics")
```

---

table.Drawdowns *Worst Drawdowns Summary: Statistics and Stylized Facts*

Description

Creates table showing statistics for the worst drawdowns.

Usage

```
table.Drawdowns(R, top= 5, digits = 4)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
top	the number of drawdowns to include
digits	number of digits to round results to

Author(s)

Peter Carl

References

Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 88

See Also

DownsideDeviation
maxDrawdown
findDrawdowns
sortDrawdowns
chart.Drawdown
table.DownsideRisk

Examples

```
data(edhec)
table.Drawdowns(edhec[,1,drop=FALSE])
table.Drawdowns(edhec[,12,drop=FALSE])
data(managers)
table.Drawdowns(managers[,8,drop=FALSE])

result=table.Drawdowns(managers[,1,drop=FALSE])
require("Hmisc")
textplot(format.df(result, na.blank=TRUE, numeric.dollar=FALSE, cdec=c(rep(3,4), rep(0,3))),
title(main="Largest Drawdowns for HAM1")
```

---

table.HigherMoments

Higher Moments Summary: Statistics and Stylized Facts

Description

Summary of the higher moements and Co-Moments of the return distribution. Used to determine diversification potential. Also called "systematic" moments by several papers.

Usage

```
table.HigherMoments(Ra, Rb, scale = NA, Rf = 0, digits = 4, method = "moment")
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)
Rf	risk free rate, in same period as your returns
digits	number of digits to round results to
method	method to use when computing kurtosis one of: excess, moment, fisher

Author(s)

Peter Carl

References

Martellini L., Vaissie M., Ziemann V. Investing in Hedge Funds: Adding Value through Active Style Allocation Decisions. October 2005. Edhec Risk and Asset Management Research Centre.

See Also

CoSkewness
CoKurtosis
BetaCoVariance
BetaCoSkewness
BetaCoKurtosis
skewness
kurtosis

Examples

```
data(managers)
table.HigherMoments(managers[,1:3],managers[,8,drop=FALSE])
result=t(table.HigherMoments(managers[,1:6],managers[,8,drop=FALSE]))
rownames(result)=colnames(managers[,1:6])
require("Hmisc")
textplot(format.df(result, na.blank=TRUE, numeric.dollar=FALSE, cdec=rep(3,dim(result)[2])),
title(main="Higher Co-Moments with SP500 TR")
```

---

table.Stats *Returns Summary: Statistics and Stylized Facts*

Description

Returns a basic set of statistics that match the period of the data passed in (e.g., monthly returns will get monthly statistics, daily will be daily stats, and so on)

Usage

```
table.Stats(R, ci = 0.95, digits = 4)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
ci	confidence interval, defaults to 95%
digits	number of digits to round results to

Details

This was created as a way to display a set of related statistics together for comparison across a set of instruments or funds. Careful consideration to missing data or unequal time series should be given when intepreting the results.

Author(s)

Peter Carl

Examples

```
data(edhec)
table.Stats(edhec[,1:3])
t(table.Stats(edhec))

result=t(table.Stats(edhec))
require("Hmisc")
textplot(format.df(result, na.blank=TRUE, numeric.dollar=FALSE, cdec=c(rep(1,2),rep(3,14))),
title(main="Statistics for EDHEC Indexes")
```

---

table.TrailingPeriods

Rolling Periods Summary: Statistics and Stylized Facts

Description

A table of estimates of rolling period return measures

Usage

```
table.TrailingPeriods(R,  periods = subset(c(12,36,60), c(12,36,60)
< length(as.matrix(R[,1]))), FUNCS=c("mean","sd"), funcs.names = c("Average", "Std
table.TrailingPeriodsRel(R, Rb, periods = subset(c(12,36,60), c(12,36,60)
< length(as.matrix(R[,1]))), FUNCS=c("cor","CAPM.beta"), funcs.names = c("Correlati
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	an xts, vector, matrix, data frame, timeSeries or zoo object of index, benchmark, portfolio, or secondary asset returns to compare against
periods	number of periods to use as rolling window(s), subset of c(3, 6, 9, 12, 18, 24, 36, 48)
funcs.names	vector of function names used for labeling table rows
FUNCS	list of functions to apply the rolling period to
digits	number of digits to round results to
...	any other passthru parameters for functions specified in FUNCS

Author(s)

Peter Carl

See Also

rollapply

Examples

```
data(edhec)
table.TrailingPeriods(edhec[,10:13], FUNCS=c("SharpeRatio","VaR"), funcs.names = c("Sharpe R

result=table.TrailingPeriods(edhec[,10:13], FUNCS=c("SharpeRatio","VaR"), funcs.names = c("S
require("Hmisc")
textplot(format.df(result, na.blank=TRUE, numeric.dollar=FALSE, cdec=rep(3,dim(result)[2])),
title(main="Trailing Period Statistics")
```

---

textplot *Display text information in a graphics plot.*

Description

This function displays text output in a graphics window. It is the equivalent of 'print' except that the output is displayed as a plot.

Usage

```
textplot(object, halign="center", valign="center", cex, max.cex = 1, cmar = 2, rmar

replaceTabs(text, width=8)
```

Arguments

object	Object to be displayed.
halign	Alignment in the x direction, one of "center", "left", or "right".
valign	Alignment in the y direction, one of "center", "top" , or "bottom"
cex	Character size, see par for details. If unset, the code will attempt to use the largest value which allows the entire object to be displayed.
mar	Figure margins, see the documentation for par.
rmar, cmar	Space between rows or columns, in fractions of the size of the letter 'M'.
show.rownames, show.colnames	Logical value indicating whether row or column names will be displayed.
hadj,vadj	Vertical and horizontal location of elements within matrix cells. These have the same meaning as the adj graphics paramter (see par).

`col.data` Colors for data elements. If a single value is provided, all data elements will be the same color. If a matrix matching the dimensions of the data is provided, each data element will receive the specified color.

`col.rownames, col.colnames` Colors for row names and column names, respectively. Either may be specified as a scalar or a vector of appropriate length.

`max.cex` Sets the largest text size as a ceiling

`row.valign` Sets the vertical alignment of the row as "top", "bottom", or (default) "center".

`heading.valign` Sets the vertical alignment of the heading as "top", (default) "bottom", or "center".

`wrap` If TRUE (default), will wrap column names and rownames

`wrap.colnames` The number of characters after which column labels will be wrapped. Default is 10.

`wrap.rownames` The number of characters after which row headings will be wrapped. Default is 10.

`text` in the function 'replaceTabs', the text string to be processed

`width` in the function 'replaceTabs', the number of spaces to replace tabs with

`...` Optional arguments passed to the text plotting command or specialized object methods

Details

A new plot is created and the object is displayed using the largest font that will fit on in the plotting region. The `halign` and `valign` parameters can be used to control the location of the string within the plotting region.

For matrixes and vectors a specialized textplot function is available, which plots each of the cells individually, with column widths set according to the sizes of the column elements. If present, row and column labels will be displayed in a bold font.

textplot also uses replaceTabs, a function to replace all tabs in a string with an appropriate number of spaces. That function was also written by Gregory R. Warnes and included in the 'gplots' package.

Author(s)

Originally written by Gregory R. Warnes ⟨warnes@bst.rochester.edu⟩ for the package 'gplots', modified by Peter Carl

See Also

`plot`,
`text`,
`capture.output`,
`textplot`

Examples

```
# Also see the examples in the original gplots textplot function
data(managers)
textplot(table.AnnualizedReturns(managers[,1:6]))

# prettify with format.df in hmisc package
require("Hmisc")
  result = t(table.CalendarReturns(managers[,1:8]))[-1:-12,]

  textplot(format.df(result, na.blank=TRUE, numeric.dollar=FALSE, cdec=rep(1,dim(result)[2]))

  title(main="Calendar Returns")
```

TrackingError *Calculate Tracking Error of returns against a benchmark*

Description

A measure of the unexplained portion of performance relative to a benchmark.

Usage

```
TrackingError(Ra, Rb, scale = NA)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)

Details

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

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

Author(s)

Peter Carl

References

Sharpe, W.F. The Sharpe Ratio,*Journal of Portfolio Management*,Fall 1994, 49-58.

See Also

InformationRatio TrackingError

Examples

```
data(managers)
TrackingError(managers[,1,drop=FALSE], managers[,8,drop=FALSE])
TrackingError(managers[,1:6], managers[,8,drop=FALSE])
TrackingError(managers[,1:6], managers[,8:7,drop=FALSE])
```

TreynorRatio *calculate Treynor Ratio of excess return over CAPM beta*

Description

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

Usage

```
TreynorRatio(Ra, Rb, Rf = 0, scale = NA)
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
Rf	risk free rate, in same period as your returns
scale	number of periods in a year (daily scale = 252, monthly scale = 12, quarterly scale = 4)

Details

Equation:

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

Author(s)

Peter Carl

References

http://en.wikipedia.org/wiki/Treynor_ratio

See Also

SharpeRatio SortinoRatio CAPM.beta

Examples

```
data(managers)
TreynorRatio(managers[,1,drop=FALSE], managers[,8,drop=FALSE], Rf=.035/12)
TreynorRatio(managers[,1,drop=FALSE], managers[,8,drop=FALSE], Rf = managers[,10,drop=FALSE]
TreynorRatio(managers[,1:6], managers[,8,drop=FALSE], Rf=.035/12)
TreynorRatio(managers[,1:6], managers[,8,drop=FALSE], Rf = managers[,10,drop=FALSE])
TreynorRatio(managers[,1:6], managers[,8:7,drop=FALSE], Rf=.035/12)
TreynorRatio(managers[,1:6], managers[,8:7,drop=FALSE], Rf = managers[,10,drop=FALSE])
```

UpDownRatios *calculate metrics on up and down markets for the benchmark asset*

Description

Calculate metrics on how the asset in R performed in up and down markets, measured by periods when the benchmark asset was up or down.

Usage

```
UpDownRatios(Ra, Rb, method = c("Capture", "Number", "Percent"), side = c("Up", "Do
```

Arguments

Ra	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
Rb	return vector of the benchmark asset
method	"Capture", "Number", or "Percent" to indicate which measure to return
side	"Up" or "Down" market statistics

Details

This is a function designed to calculate several related metrics:

Up (Down) Capture Ratio: this is a measure of an investment's compound return when the benchmark was up (down) divided by the benchmark's compound return when the benchmark was up (down). The greater (lower) the value, the better.

Up (Down) Number Ratio: similarly, this is a measure of the number of periods that the investment was up (down) when the benchmark was up (down), divided by the number of periods that the Benchmark was up (down).

Up (Down) Percentage Ratio: this is a measure of the number of periods that the investment outperformed the benchmark when the benchmark was up (down), divided by the number of periods that the benchmark was up (down). Unlike the prior two metrics, in both cases a higher value is better.

Author(s)

Peter Carl

References

Bacon, C. *Practical Portfolio Performance Measurement and Attribution*. Wiley. 2004. p. 47

Examples

```
data(managers)
UpDownRatios(managers[,1, drop=FALSE], managers[,8, drop=FALSE])
UpDownRatios(managers[,1:6, drop=FALSE], managers[,8, drop=FALSE])
UpDownRatios(managers[,1, drop=FALSE], managers[,8, drop=FALSE], method="Capture")
# Up Capture:
UpDownRatios(managers[,1, drop=FALSE], managers[,8, drop=FALSE], side="Up", method="Capture"
# Down Capture:
UpDownRatios(managers[,1, drop=FALSE], managers[,8, drop=FALSE], side="Down", method="Captur
```

UpsidePotentialRatio

calculate Upside Potential Ratio of upside performance over downside risk

---

Description

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

Usage

```
UpsidePotentialRatio(R, MAR = 0, method=c("subset","full"))
UPR(R, MAR = 0, method=c("subset","full"))
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
MAR	Minimum Acceptable Return, in the same periodicity as your returns
method	one of "full" or "subset", indicating whether to use the length of the full series or the length of the subset of the series above(below) the MAR as the denominator, defaults to "subset"

Details

Sortino contends that risk should be measured in terms of not meeting the investment goal. This gives rise to the notion of "Minimum Acceptable Return" or MAR. All of Sortino's proposed measures include the MAR, and are more sensitive to downside or extreme risks than measures that use volatility(standard deviation of returns) as the measure of risk.

Choosing the MAR carefully is very important, especially when comparing disparate investment choices. If the MAR is too low, it will not adequately capture the risks that concern the investor, and if the MAR is too high, it will unfavorably portray what may otherwise be a sound investment. When comparing multiple investments, some papers recommend using the risk free rate as the MAR. Practitioners may wish to choose one MAR for consistency, several standardized MAR values for reporting a range of scenarios, or a MAR customized to the objective of the investor.

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

where δ_{MAR} is the DownsideDeviation.

The numerator in UpsidePotentialRatio only uses returns that exceed the MAR, and the denominator (in DownsideDeviation) only uses returns that fall short of the MAR by default. Sortino contends that this is a more accurate and balanced protrayal of return potential, wherase SortinoRatio can reward managers most at the peak of a cycle, without adequately penalizing them for past mediocre performance. Others have used the full series, and this is provided as an option by the method argument.

Author(s)

Brian G. Peterson

References

Sortino, F. and Price, L. Performance Measurement in a Downside Risk Framework. *Journal of Investing*. Fall 1994, 59-65.

Plantinga, A., van der Meer, R. and Sortino, F. The Impact of Downside Risk on Risk-Adjusted Performance of Mutual Funds in the Euronext Markets. July 19, 2001. Available at SSRN: http://ssrn.com/abstract=277352

See Also

SharpeRatio
SortinoRatio
DownsideDeviation
SemiVariance
SemiDeviation
InformationRatio

Examples

```
data(edhec)
UpsidePotentialRatio(edhec[, 6], MAR=.05/12) #5 percent/yr MAR
UpsidePotentialRatio(edhec[, 1:6], MAR=0)
```

VaR *calculate various Value at Risk (VaR) measures*

Description

Calculates Value-at-Risk(VaR) for univariate, component, and marginal cases using a variety of analytical methods.

Usage

```
VaR(R, p = 0.95, method = c("modified", "gaussian","historical", "kernel"),
           clean = c("none", "boudt", "geltner"),
                 portfolio_method = c("single", "component","marginal"),
                 weights = NULL, mu = NULL, sigma = NULL,
                 m3 = NULL, m4 = NULL, invert = TRUE, ...)
```

Arguments

R	an xts, vector, matrix, data frame, timeSeries or zoo object of asset returns
p	confidence level for calculation, default p=.99
method	one of "modified","gaussian","historical", "kernel", see Details.
clean	method for data cleaning through Return.clean. Current options are "none", "boudt", or "geltner".
portfolio_method	one of "single","component","marginal" defining whether to do univariate, component, or marginal calc, see Details.
weights	portfolio weighting vector, default NULL, see Details
mu	If univariate, mu is the mean of the series. Otherwise mu is the vector of means of the return series , default NULL, , see Details
sigma	If univariate, sigma is the variance of the series. Otherwise sigma is the covariance matrix of the return series , default NULL, see Details
m3	If univariate, m3 is the skewness of the series. Otherwise m3 is the coskewness matrix of the returns series, default NULL, see Details
m4	If univariate, m4 is the excess kurtosis of the series. Otherwise m4 is the cokurtosis matrix of the return series, default NULL, see Details
invert	TRUE/FALSE whether to invert the VaR measure. see Details.
...	any other passthru parameters

Background

This function provides several estimation methods for the Value at Risk (typically written as VaR) of a return series and the Component VaR of a portfolio. Take care to capitalize VaR in the commonly accepted manner, to avoid confusion with var (variance) and VAR (vector auto-regression). VaR is an industry standard for measuring downside risk. For a return series, VaR is defined as the high

quantile (e.g. ~a 95% or 99% quantile) of the negative value of the returns. This quantile needs to be estimated. With a sufficiently large data set, you may choose to utilize the empirical quantile calculated using `quantile`. More efficient estimates of VaR are obtained if a (correct) assumption is made on the return distribution, such as the normal distribution. If your return series is skewed and/or has excess kurtosis, Cornish-Fisher estimates of VaR can be more appropriate. For the VaR of a portfolio, it is also of interest to decompose total portfolio VaR into the risk contributions of each of the portfolio components. For the above mentioned VaR estimators, such a decomposition is possible in a financially meaningful way.

Univariate VaR estimation methods

The VaR at a probability level p (e.g. 95%) is the p-quantile of the negative returns, or equivalently, is the negative value of the $c = 1 - p$ quantile of the returns. In a set of returns for which sufficently long history exists, the per-period Value at Risk is simply the quantile of the period negative returns :

$$VaR = q_{.99}$$

where $q_{.99}$ is the 99% empirical quantile of the negative return series.

This method is also sometimes called "historical VaR", as it is by definition *ex post* analysis of the return distribution, and may be accessed with `method="historical"`.

When you don't have a sufficiently long set of returns to use non-parametric or historical VaR, or wish to more closely model an ideal distribution, it is common to us a parmetric estimate based on the distribution. J.P. Morgan's RiskMetrics parametric mean-VaR was published in 1994 and this methodology for estimating parametric mean-VaR has become what most literature generally refers to as "VaR" and what we have implemented as `VaR`. See *Return to RiskMetrics: Evolution of a Standard*`http://www.riskmetrics.com/publications/techdocs/r2rovv.html`.

Parametric mean-VaR does a better job of accounting for the tails of the distribution by more precisely estimating shape of the distribution tails of the risk quantile. The most common estimate is a normal (or Gaussian) distribution $R \sim N(\mu, \sigma)$ for the return series. In this case, estimation of VaR requires the mean return $\bar{R}$, the return distribution and the variance of the returns σ. In the most common case, parametric VaR is thus calculated by

$$\sigma = variance(R)$$

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

where z_c is the c-quantile of the standard normal distribution. Represented in R by `qnorm(c)`, and may be accessed with `method="gaussian"`.

Other forms of parametric mean-VaR estimation utilize a different distribution for the distribution of losses to better account for the possible fat-tailed nature of downside risk. The package `VaR` contains methods for simulating and estimating lognormal `VaR.norm` and generalized Pareto `VaR.gpd` distributions to overcome some of the problems with nonparametric or parametric mean-VaR calculations on a limited sample size or on potentially fat-tailed distributions. There is also a `VaR.backtest` function to apply simulation methods to create a more robust estimate of the potential distribution of losses. Less commonly a covariance matrix of multiple risk factors may be applied.

The limitations of mean Value-at-Risk are well covered in the literature. The limitations of traditional mean-VaR are all related to the use of a symetrical distribution function. Use of simulations, resampling, or Pareto distributions all help in making a more accurate prediction, but they are still flawed for assets with significantly non-normal (skewed or kurtotic) distributions. Zangari (1996) and Favre and Galeano(2002) provide a modified VaR calculation that takes the higher moments of non-normal distributions (skewness, kurtosis) into account through the use of a Cornish Fisher expansion, and collapses to standard (traditional) mean-VaR if the return stream follows a standard distribution. This measure is now widely cited and used in the literature, and is usually referred to as "Modified VaR" or "Modified Cornish-Fisher VaR". They arrive at their modified VaR calculation in the following manner:

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

$$Cornish - FisherVaR = -\bar{R} - \sqrt{(}\sigma) \cdot z_{cf}$$

where S is the skewness of R and K is the excess kurtosis of R.

Cornish-Fisher VaR collapses to traditional mean-VaR when returns are normally distributed. As such, the `VaR` and `VaR` functions are wrappers for the `VaR` function. The Cornish-Fisher expansion also naturally encompasses much of the variability in returns that could be uncovered by more computationally intensive techniques such as resampling or Monte-Carlo simulation. This is the default method for the `VaR` function, and may be accessed by setting `method="modified"`.

Favre and Galeano also utilize modified VaR in a modified Sharpe Ratio as the return/risk measure for their portfolio optimization analysis, see `SharpeRatio.modified` for more information.

Component VaR

By setting `portfolio_method="component"` you may calculate the risk contribution of each element of the portfolio. The return from the function in this case will be a list with three components: the univariate portfolio VaR, the scalar contribution of each component to the portfolio VaR (these will sum to the portfolio VaR), and a percentage risk contribution (which will sum to 100%).

Both the numerical and percentage component contributions to VaR may contain both positive and negative contributions. A negative contribution to Component VaR indicates a portfolio risk diversifier. Increasing the position weight will reduce overall portoflio VaR.

If a weighting vector is not passed in via `weights`, the function will assume an equal weighted (neutral) portfolio.

Multiple risk decomposition approaches have been suggested in the literature. A naive approach is to set the risk contribution equal to the stand-alone risk. This approach is overly simplistic and neglects important diversification effects of the units being exposed differently to the underlying risk factors. An alternative approach is to measure the VaR contribution as the weight of the position in the portfolio times the partial derivative of the portfolio VaR with respect to the component weight.

$$C_i \text{VaR} = w_i \frac{\partial \text{VaR}}{\partial w_i}.$$

Because the portfolio VaR is linear in position size, we have that by Euler's theorem the portfolio VaR is the sum of these risk contributions. Gourieroux (2000) shows that for VaR, this mathematical

decomposition of portfolio risk has a financial meaning. It equals the negative value of the asset's expected contribution to the portfolio return when the portfolio return equals the negative portfolio VaR:

$$C_i\text{VaR} == -E\left[w_i r_i | r_p = -\text{VaR}\right]$$

For the decomposition of Gaussian VaR, the estimated mean and covariance matrix are needed. For the decomposition of modified VaR, also estimates of the coskewness and cokurtosis matrices are needed. If r denotes the $Nx1$ return vector and mu is the mean vector, then the $N \times N^2$ co-skewness matrix is

$$m3 = E\left[(r-\mu)(r-\mu)' \otimes (r-\mu)'\right]$$

The $N \times N^3$ co-kurtosis matrix is

$$m_4 = E\left[(r-\mu)(r-\mu)' \otimes (r-\mu)' \otimes (r-\mu)'\right]$$

where $\otimes$ stands for the Kronecker product. The matrices can be estimated through the functions `skewness.MM` and `kurtosis.MM`. More efficient estimators have been proposed by Martellini and Ziemann (2007) and will be implemented in the future.

As discussed among others in Cont, Deguest and Scandolo (2007), it is important that the estimation of the VaR measure is robust to single outliers. This is especially the case for modified VaR and its decomposition, since they use higher order moments. By default, the portfolio moments are estimated by their sample counterparts. If `clean="boudt"` then the $1-p$ most extreme observations are winsorized if they are detected as being outliers. For more information, see Boudt, Peterson and Croux (2008) and `Return.clean`. If your data consist of returns for highly illiquid assets, then `clean="geltner"` may be more appropriate to reduce distortion caused by autocorrelation, see `Return.Geltner` for details.

Epperlein and Smillie (2006) introduced a non-parametric kernel estimator for component risk contributions, which is available via `method="kernel"` and `portfolio_method="component"`.

Marginal VaR

Different papers call this different things. In the Denton and Jayaraman paper referenced here, this calculation is called Incremental VaR. We have chosen the more common usage of calling this difference in VaR's in portfolios without the instrument and with the instrument as the "difference at the Margin", thus the name Marginal VaR. This is incredibly confusing, and hasn't been resolved in the literature at this time. Simon Keel and David Ardia (2009) attempt to reconcile some of the definitional issues and address some of the shortcomings of this measure in their working paper titled "Generalized Marginal Risk". Hopefully their improved Marginal Risk measures may be included here in the future.

Note

The option to `invert` the VaR measure should appease both academics and practitioners. The mathematical definition of VaR as the negative value of a quantile will (usually) produce a positive number. Practitioners will argue that VaR denotes a loss, and should be internally consistent with the quantile (a negative number). For tables and charts, different preferences may apply for clarity and compactness. As such, we provide the option, and set the default to TRUE to keep the return consistent with prior versions of PerformanceAnalytics, but make no value judgement on which approach is preferable.

The prototype of the univariate Cornish Fisher VaR function was completed by Prof. Diethelm Wuertz. All corrections to the calculation and error handling are the fault of Brian Peterson.

Author(s)

Brian G. Peterson and Kris Boudt

References

Boudt, Kris, Peterson, Brian, and Christophe Croux. 2008. Estimation and decomposition of downside risk for portfolios with non-normal returns. 2008. The Journal of Risk, vol. 11, 79-103.

Cont, Rama, Deguest, Romain and Giacomo Scandolo. Robustness and sensitivity analysis of risk measurement procedures. Financial Engineering Report No. 2007-06, Columbia University Center for Financial Engineering.

Denton M. and Jayaraman, J.D. Incremental, Marginal, and Component VaR. Sunguard. 2004.

Epperlein, E., Smillie, A. Cracking VaR with kernels. RISK, 2006, vol. 19, 70-74.

Gourieroux, Christian, Laurent, Jean-Paul and Olivier Scaillet. Sensitivity analysis of value at risk. Journal of Empirical Finance, 2000, Vol. 7, 225-245.

Keel, Simon and Ardia, David. Generalized marginal risk. Aeris CAPITAL discussion paper.

Laurent Favre and Jose-Antonio Galeano. Mean-Modified Value-at-Risk Optimization with Hedge Funds. Journal of Alternative Investment, Fall 2002, v 5.

Martellini, Lionel, and Volker Ziemann. Improved Forecasts of Higher-Order Comoments and Implications for Portfolio Selection. 2007. EDHEC Risk and Asset Management Research Centre working paper.

Return to RiskMetrics: Evolution of a Standard http://www.riskmetrics.com/publications/techdocs/r2rovv.html

Zangari, Peter. A VaR Methodology for Portfolios that include Options. 1996. RiskMetrics Monitor, First Quarter, 4-12.

See Also

SharpeRatio.modified
chart.VaRSensitivity
VaR.gpd
VaR.norm
VaR.backtest
Return.clean

Examples

```
data(edhec)

# first do normal VaR calc
VaR(edhec, p=.95, method="historical")

# now use Gaussian
```

```
VaR(edhec, p=.95, method="gaussian")

# now use modified Cornish Fisher calc to take non-normal distribution into account
VaR(edhec, p=.95, method="modified")

# now use p=.99
VaR(edhec, p=.99)
# or the equivalent alpha=.01
VaR(edhec, p=.01)

# now with outliers squished
VaR(edhec, clean="boudt")

# add Component VaR for the equal weighted portfolio
VaR(edhec, clean="boudt", portfolio_method="component")
```

`weights` *Selected Portfolio Weights Data*

Description

An xts object that contains columns of monthly weights for a subset of the EDHEC hedge fund indexes that demonstrate rebalancing portfolios through time.

Note that all the EDHEC indices are available in `edhec`.

Usage

```
managers
```

Format

CSV conformed into an xts object with monthly observations

Details

A relatively random weights file used for charting examples.

Examples

```
data(weights)

#preview the data
head(weights)
```

Index