Package 'PCRA'

July 21, 2025

Type Package

Title Companion to Portfolio Construction and Risk Analysis

Version 1.2

Date 2023-08-27

Description A collection of functions and data sets that support teaching a quantitative finance MS level course on Portfolio Construction and Risk Analysis, and the writing of a textbook for such a course. The package is unique in providing several real-world data sets that may be used for problem assignments and student projects. The data sets include cross-sections of stock data from the Center for Research on Security Prices, LLC (CRSP), corresponding factor exposures data from S&P Global, and several SP500 data sets.

License GPL-2

Depends R (>= 4.0.0),

Encoding UTF-8

Imports PerformanceAnalytics, PortfolioAnalytics, boot, methods, xts, zoo, lattice, corpcor, data.table, quadprog, RobStatTM, robustbase, R.cache

Suggests R.rsp

LazyLoad true

LazyData true

LazyDataCompression xz

Copyright (c) 2022-2023

RoxygenNote 7.2.3

VignetteBuilder R.rsp

NeedsCompilation no

Author Doug Martin [cre, aut],
Alexios Galanos [ctb],
Kirk Li [aut, ctb],
Jon Spinney [ctb],
Thomas Philips [ctb]

Maintainer Doug Martin <martinrd3d@gmail.com>

Repository CRAN

Date/Publication 2023-08-30 04:50:02 UTC

Contents

`barplotWts` *A Barplot of a Set of Portfolio Weights*

Description

Uses the R barplot function to make a barplot of efficient frontier weights. See the manual page for barplot()

Usage

```
barplotWts(
  wts.efront,
  legend.text = NULL,
  col = NULL,
  ylab = NULL,
  xlab = c("MU", "VOL"),
  bar.ylim = NULL,
  ...
)
```

Arguments

`wts.efront`	Matrix of weights along the efficient frontier
`legend.text`	Vector of text for the legend
`col`	Vector of colors for the bars
`ylab`	A label for the y axis
`xlab`	A label for the x axis
`bar.ylim`	Limits of the y axis for barplot
`...`	additional parameters from barplot

Value

No return value, just a barplot of efficient frontier weights

Examples

```
args(barplotWts)
```

`bootEfronts` *Bootstrapped Efficient Frontiers*

Description

Computes and plots bootstrapped portfolio efficient frontiers, with optional bullet points for GMV portfolios and tangency portfolios.

Usage

```
bootEfronts(
  returns,
  pspec,
  rf = 0.003,
  npoints = 20,
  B = 3,
  Seed = NULL,
  gmv = TRUE,
  maxSR = FALSE,
  xlim = NULL,
  ylim = NULL,
  k.sigma = 2,
  k.mu = 2,
  digits = 4,
  figTitle = NULL
)
```

Arguments

`returns`	A multivarite xts returns object
`pspec`	PortfolioAnalytics portfolio specification object
`rf`	Risk-free rate as a decimal, default 0.003
`npoints`	Number of points on efficient frontier, default 10
`B`	Number of bootstrap samples, default 3
`Seed`	Seed of bootstrap random number generator, default NULL
`gmv`	Logical variable, default TRUE
`maxSR`	Logical variable, default FALSE

xlim	Numeric x axis plot limits, default NULL
ylim	Numeric y axis plot limits, default NULL
k.sigma	Numeric value
k.mu	Numeric value
digits	Number of significant digits for numeric values
figTitle	Optional figure title, default NULL

Details

k.sigma controls horizontal axis plotting range if xlim = NULL, and k.mu controls vertical axis plotting range if ylim = NULL. Adjust k.mu and k.sigma to eliminate plot "Line out of bounds" Warnings. gmv = TRUE to display a bullet at global minimum variance portfolio maxSR = TRUE to display a bullet at tangency portfolio

Value

No value returned, instead a bootstrapped efficient frontiers plot with options described in the above details.

Examples

```
args(bootEfronts)
```

---

chart.Efront *Create Efficient Frontier*

Description

Utility function for creating initial efficient frontier, and for creating subsequent bootstrap efficient frontiers created, all of which are created by the PortfolioAnalytics function create.EfficientFrontier.

Usage

```
chart.Efront(
  returns,
  pspec,
  firstEfront = TRUE,
  gmv = TRUE,
  maxSR = TRUE,
  rf = 0.003,
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab = NULL,
  n.portfolios = 10
)
```

Arguments

returns	A multivarite xts returns object
pspec	PortfolioAnalytics portfolio specification object
firstEfront	Logical variable, default TRUE
gmv	Logical variable, default TRUE
maxSR	Logical variable, default TRUE
rf	Risk-free rate, default 0.003
xlim	Numeric value, default NULL
ylim	Numeric value, default NULL
xlab	Numeric value, default NULL
ylab	Numeric value, default NULL
n.portfolios	Number of efficient frontier portfolios, default 10

Details

The variable firstEfront is set to TRUE for the initial efficient frontier plot, but is set to FALSE for the bootstrap replicate efficient frontier plots. The choices gmv = TRUE and maxSR = TRUE result in bullet points at those locations on the initial efficient frontier plot

Value

No value returned, instead plots of efficient frontiers for use by bootEfronts()

Examples

```
args(chart.Efront)
```

---

cleanOutliers *Clean Returns Outliers*

Description

Outliers are "cleaned" by shrinking or rejecting data whose distance from the median (med) is larger in absolute value than a specified value k multiplied by the median absolute deviation from the median (mad). Outlier shrinkage results in the data value being set equal to the nearest of med-k*mad and med+k*mad. Rejected data is assigned an NA. Shrinkage is the default.

Usage

```
cleanOutliers(x, k = 3, shrink = TRUE)
```

Arguments

x	A numeric vector
k	A numeric value, which multiplies the mad. Smaller values of k result in greater fractions of data which is either shrunk of rejected, and larger values of k result in smaller fractions of the data that are shrunk or rejected.
shrink	A logical variable whose default is TRUE.

Value

an outlier cleaned numeric object

Examples

```
args(cleanOutliers)
```

---

crsp.returns8 *crsp.returns8*

Description

Monthly returns of 8 stocks with tickers GHI, PBCI, MODI, MGJ, MAT, EMN, AMAT, AMGN from 1997 to 2001

Usage

```
data(crsp.returns8)
```

Format

A multivariate xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business. NOTE: CRSP data is not covered by the GPL. Redistribution of the data in any form is not permitted, and use of the data in derivative works is not permitted without the written permission of CRSP.

Examples

```
library(PCRA)
library(zoo)
data(crsp.returns8)
names(crsp.returns8)
dim(crsp.returns8)
range(index(crsp.returns8))
```

datFF3W *Fama-French Weekly 3-Factor Model*

Description

Weekly values of the 3 factors MKT, SMB and HML

Usage

```
data(datFF3W)
```

Format

Multivariate time series xts object

Source

https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html

Examples

```
library(PCRA)
library(zoo)
data(datFF3W)
head(datFF3W)
range(index(datFF3W))
```

datFF4W *Fama-French-Carhart Weekly 4-Factor Model*

Description

Weekly values of the 4 factors MKT, SMB, HML and MOM

Usage

```
data(datFF4W)
```

Format

Multivariate time series xts object

Source

https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html

Examples

```
library(PCRA)
library(zoo)
data(datFF4W)
head(datFF4W)
range(index(datFF4W))
```

`divHHI` *HHI Based Diversification Index*

Description

divHHI calculates a portfolio diversification index DIV. The DIV is equal to 1 minus the Herfindahl-Hirschman Index (HHI), which is defined as the sum of the squared portfolio weights. The maximum HHI of a long-only portfolio is 1, which occurs when all of the portfolio's investment is in a single asset, and correspondingly HHI = 0.

Usage

```
divHHI(weights)
```

Arguments

`weights`	A numeric vector of portfolio weights

Value

a zoo time series object containing portfolio diversification values

Examples

```
args(divHHI)
```

`ellipsesPlotPCRA.covfm`
Overlaid Correlations Ellipses Plots

Description

When there are 3 or more variables in the data, this function produces a matrix with overlaid ellipses drawn in the upper triangle. The main use case is a sample covariance estimator and a robust covariance matrix estimator, so two overlaid ellipses. The ellipses in cell i,j of the plot is drawn to be a contour of a standard bivariate normal density with correlation 'rho(i,j)'. Two ellipses are drawn in each cell, one for the sample covariance matrix estimate and one for the robust covariance matrix estimate. When there are only 2 variables in the data, this function produces a scatter plot of the data with overlaid 95 robust covariance matrix estimates in the 'covfm' object. The lower triangle displays the sample correlation estimate value in red font, and robust correlation estimate in black font.

Usage

```
ellipsesPlotPCRA.covfm(x, ...)
```

Arguments

x	a ‘covfm‘ object
...	additional arguments are ignored.

Value

x is invisibly returned

Author(s)

The original version ‘ellipsesPlot.covfm was wirtten by Kjell Konis for the ‘fit.models‘ package. This version, modified by Doug Martin, uses thicker lines for the ellipses, with red color for the sample correlation and black for the robust correlation, for a better overall visual display.

Examples

```
args(ellipsesPlotPCRA.covfm)
```

---

factorsSPGMI *factorsSPGMI*

Description

14 SPGMI monthly factor exposures for 294 CRSP stocks from 1993 to 2015

Usage

```
data(factorsSPGMI)
```

Format

A data.frame containing 14 SPGMI monthly factor exposures (alpha factors) for 294 stocks from 1993 to 2015 (276 months) of observations on 21 variables that include the 14 factor exposures, for each of 294 stocks

- **Date:** type ‘Date‘.
- **TickerLast:** type ‘chr‘. This is the ticker as of December 2015
- **Ticker:** type ‘chr‘. This is the monthly ticker
- **Company:** type ‘chr‘. The name of the company
- **CapGroupLast:** type ‘chr‘. Company market capitalization group as of December 2015, one of: MicroCap, SmallCap, MidCap or LargeCap
- **CapGroup:** type ‘chr‘. Monthly market capitalization group

- **GICS:** type 'chr'. An 8 digit S&P GICS code, the first two digits of which are codes for 11 GICS sectors
- **Sector:** type 'chr'. One of 8 of the 11 GICS sectors, with none of the 294 stocks in Financials, Real Estate or Utilities
- **AnnVol12M:** type 'num'. Annualized Volatility of Monthly Stock Returns (Last Twelve Months)
- **Beta60M:** type 'num'. 60 Month OLS Beta relative to the S&P 500 estimated using Monthly Total Returns
- **BP:** type 'num'. Most Recent Book Value of Common Equity divided by Market Value of Common Equity
- **EP:** type 'num'. Sum of trailing four quarters Earnings per Share divided by Current Price per share
- **LogMktCap:** type 'num'. Natural Logarithm of Current Market Capitalization in $
- **PM12M1M:** type 'num'. Price relative change from time t-12 to time t-1: PM12M1M(t) = (P(t-1)-P(t-12))/P(t-12) = P(t-1)/P(t-12) – 1
- **AccrualRatioCF:** type 'num'. Ratio of Accruals to Net Operating Assets, where Accruals = Income Before Extraordinary Items minus Net Operating Cash Flow minus Net Investing Cash Flow, and Net Operating Assets = Total Assets – Total Liabilities – Cash and Short-Term Investments + Short- and Long-Term Debt. Both numerator and denominator are computed over the trailing four quarters
- **AstAdjChg1YOCF:** type 'num'. One-Year Change in Trailing Four Quarter Operating Cash Flow divided by Trailing Four Quarter Average(Total Assets)
- **CFROIC:** type 'num'. Trailing Four Quarter Operating Cash Flow divided by Trailing Four Quarter Average of Invested Capital where Invested Capital = Sum of Long-Term Debt, Preferred Stock, Common Equity and Minority Interests – Treasury Stock
- **Chg1YAstTo:** type 'num'. Percentage change over 12 months in Trailing Four Quarter Revenues divided by Trailing Four Quarter Average of Total Assets
- **EBITDAEV:** type 'num'. Trailing Four Quarter EBITDA divided by Average of Trailing Four Quarter Enterprise Value where Enterprise Value = Book Value of Equity + Market Value of Debt
- **FCFP:** type 'num'. Trailing Four Quarter Free Cash Flow divided by Trailing Four Quarter Average of Market Value of Equity
- **PM1M:** type 'num'. Trailing 1-Month Price Return. Relative price change from time t-1 to t: PM1M(t) = (P(t) – P(t-1)) / P(t-1), commonly called one period return R(t)
- **SEV:** type 'num'. Trailing Four Quarter Sales divided by Average of Trailing Four Quarter Enterprise Value, where Enterprise Value = Market Value of Equity + Market Value of Debt

Details

The term "factor exposures" is often used for the values of the 14 factors, which SPGMI also refers to as "scores" or "alpha factors". Our names for the the 14 factors are identical to those used by SPGMI in their AFL library. For an introduction to the AFL library see: https://www.marketplace.spglobal.com/en/datasets/alpha-factor-library-(3)

The four CapGroupLast categorizations of the stocks were determined using the three capitalization breakpoints $xxxM, $yyyM, $zzzM. Details concerning the construction of the monthly CapGroup categorizations will eventually be provided in a Vignette.

The factorsSPGMI data contains stocks in 8 of the 11 GICS sectors, with no stocks in the Financials, Utilities and Real Estate sectors. On each of the next 11 lines we list all 11 of the two digit GICS code that defines the GICS Sector, followed by the GICS sector name:

10 Energy

15 Materials

20 Industrials

25 Consumer Discretionary

30 Consumer Staples

35 Health Care

40 Financials (none currently available)

45 Information Technology

50 Communication Services

55 Utilities (none currently available)

60 Real Estate (none currently available)

GICS is a joint product of SPGMI and MSCI. For details, see the GICS Global Industry Classification Standard document (The GICS MAP Book) available at https://www.spglobal.com/en/, and the MSCI GICS Methodology 2020 document available at https://www.msci.com/.

Source

Standard and Poors Global Market Intelligence (SPGMI). NOTE: SPGMI data is not covered by the GPL. Redistribution of this SPGMI data is not permitted, and use of the data in derivative works is not permitted without the written permission of SPGMI

References

A standard corporate finance textbook: Ross, Westerfield, Jaffe and Jordan (2019). Corporate Finance, McGraw-Hill Education. CFA: https://alphabetaprep.com/cfa-level-1/financial-ratio-analysis/

`FRBinterestRates` *Federal Reserve Board Interest Rates*

Description

Federal Reserve Board monthly interest rates of 90 day Bill from 1934 to 2014.

Usage

```
data(FRBinterestRates)
```

Format

A time series zoo object

Source

Federal Reserve Board

Examples

```
library(PCRA)
library(zoo)
data(FRBinterestRates)
class(FRBinterestRates)
range(index(FRBinterestRates))
```

---

getPCRAData *Download CRSP and SPGMI Data*

Description

Downloads stocksCRSPweekly, stocksCRSPdaily

Usage

```
getPCRAData(dataset = "stocksCRSPweekly", cache = TRUE, refresh = FALSE)
```

Arguments

dataset	a valid dataset name (see details)
cache	logical variable controlling whether or not to cache the data so that when calling the function for the same dataset it will be loaded from cache rather than re-downloading from the github site
refresh	logical variable controlling whether or not to re-download a cached dataset

Details

The following are valid names of datasets available:

- "stocksCRSPdaily""Details same as for stocksCRSP except now daily"
- "stocksCRSPweekly""Details same as for stocksCRSP except now weekly"

User must install R.cache package

Value

An object of class “data.table”.

Examples

```
stocksCRSPweekly <- getPCRAData(data = "stocksCRSPweekly")
class(stocksCRSPweekly)
names(stocksCRSPweekly)

stocksCRSPdaily <- getPCRAData(data = "stocksCRSPdaily")
class(stocksCRSPdaily)
names(stocksCRSPdaily)
```

---

gfunds5 *gfunds5*

Description

Monthly returns of 5 German investment funds November 1989 to July 2001: EM (emerging markets), PE (private equity), HY (high yield), ALT (alternatives), and BND (fixed income)

Usage

```
data(gfunds5)
```

Format

Multivariate xts object

Source

Unknown

Examples

```
library(PCRA)
library(zoo)
data(gfunds5)
class(gfunds5)
names(gfunds5)
range(index(gfunds5))
```

invensysEPS *Earnings per Share of Invensys*

Description

Yearly earnings-per-share of company Invensys for 17 years. The company's name was invensys prior to 2004.

Usage

```
data(invensysEPS)
```

Format

A numeric vector

Source

Corporate Finance Department of Dupont

Examples

```
library(PCRA)
data(invensysEPS)
invensysEPS
```

KRest *Kurtosis Estimator*

Description

Sample estimate of excess kurtosis, with option for ordinary kurtosis. This function will eventually have a robust estimator option.

Usage

```
KRest(x, excess = TRUE)
```

Arguments

x	A numeric vector
excess	A logical variable with default TRUE, which results in the computation of excess kurtosis, and FALSE results ordinary kurtosis.

Value

numeric value of excess kurtosis or ordinary kurtosis

Examples

```
args(KRest)
```

`levgLongShort` *Long Short Portfolio Leverage*

Description

This function computes a time series of portfolio leverages, defined as the sum of the absolute portfolio weights divided by the sum of the long position weights

Usage

```
levgLongShort(wts)
```

Arguments

`wts`	Multivariate xts portfolio weights object

Value

an xts time series of portfolio leverages

Author(s)

Doug Martin

Examples

```
args(levgLongShort)
```

`mathEfront` *Efficient Frontiers from Returns*

Description

Computes and plots the efficient frontier with and without risk-free asset, using a multivariate time series of returns to compute the mean vector and covariance matrix

Usage

```
mathEfront(
  returns,
  mu.max = NULL,
  sigma.max = NULL,
  rf = 0.003,
  rf.line = TRUE,
  stocks = TRUE,
  stock.names = TRUE,
  SRvalue = TRUE,
  npoints = 100,
  cexText = 0.8,
  cexPoints = 0.8,
  digits = NULL
)
```

Arguments

returns	Multivariate xts object of portfolio returns
mu.max	Numeric value, default NULL
sigma.max	Numeric value, default NULL
rf	Numeric value with default 0.003
rf.line	Logical variable with default TRUE
stocks	Logical variable with default TRUE
stock.names	Logical variable with default TRUE
SRvalue	Logical variable with default TRUE
npoints	Integer number of efficient frontier points, default 100
cexText	Character expansion factor for text
cexPoints	Expansion factor for points
digits	Integer variable number of significant digits, default NULL

Details

When rf.line = TRUE, the linear efficient frontier is displayed, and it is not displayed when rf.line = FALSE. When values = TRUE, the Sharpe ratio and risk-free rate values are displayed in the plot as SHARPE RATIO and RISK-FREE values.

Value

No value returned, instead a plot is displayed of the efficient frontier with cash and risky assets, with risky assets only efficient frontier overlaid

Examples

```
args(mathEfront)
```

`mathEfrontCashRisky` *Math Efficient Frontier: Cash and Risky Assets*

Description

This function computes and plots a linear efficient frontier that is a mix of a risk-free asset ("cash") and risky stocks (or other assets). It optionally returns the weights along the linear efficient frontier.

Usage

```
mathEfrontCashRisky(
  returns,
  npoints = 10,
  rf = 0.003,
  plot.efront = TRUE,
  stock.names = TRUE,
  values = FALSE,
  scalex = 1.1,
  scaley = 1.1,
  cexPoints = 0.8,
  cexText = 0.8
)
```

Arguments

`returns`	Risky asset returns multivariate xts object
`npoints`	Number of efficient frontier points with default 10
`rf`	A risk-free rate with default 0.003
`plot.efront`	Logical variable which if TRUE results in a plot of
`stock.names`	Logical variable with default TRUE
`values`	Logical variable for returning efront values with default FALSE
`scalex`	Horizontal axis scale parameter with default 1.1
`scaley`	Vertical axis scale parameter with default 1.1
`cexPoints`	Numerical size parameter for points with default 0.8
`cexText`	Numerical size parameter for text with default 0.8

Value

default is no value returned, and a plot is displayed of the linear efficient frontier. Optionally, a numeric object containing the weights along the linear efficient frontier are displayed. Optionally no plot is displayed.

Examples

```
args(mathEfrontCashRisky)
```

`mathEfrontRisky` *Efficient Frontier of Risky Stocks*

Description

Computes and plots the efficient frontier of risky assets only, using a multivariate time series of returns to compute the mean vector and covariance matrix

Usage

```
mathEfrontRisky(
  returns,
  npoints = 100,
  efront.only = TRUE,
  display = TRUE,
  cexGmv = 0.9,
  pchPoints = 20,
  cexPoints = 1,
  cexText = 0.7,
  values = FALSE,
  digits = NULL
)
```

Arguments

`returns`	Multivariate xts object of portfolio returns
`npoints`	Integer number of efficient frontier points, with default 100
`efront.only`	Logical variable with default TRUE
`display`	If TRUE the efficient frontier is plotted
`cexGmv`	A size parameter for the text "GMV"
`pchPoints`	A parameter of the type of points
`cexPoints`	A size parameter of points
`cexText`	A size parameter of text
`values`	Logical variable with default TRUE
`digits`	Integer variable number of significant digits, default NULL

Details

When efront.only = TRUE only the efficient frontier is computed, and if FALSE the entire frontier is computed. When value = TRUE the efficient frontier mean and volatility values are returned, and when value = FALSE these values are not returned.

Value

no values are returned by default, and a plot is displayed of the either the risky assets only efficient frontier, or the entire frontier. Optionally, the values of the mean and volatility along the efficient frontier are returned.

Examples

```
args(mathEfrontRisky)
```

`mathEfrontRiskyMuCov` *Efficient Frontier*

Description

Computes a frontier or efficient frontier based on user specified mean vector and covariance matrix. Default is to compute the efficient frontier and plot it. Optionally the mean and volatility values of the frontier or efficient frontier is returned at a user specified number of significant digits.

Usage

```
mathEfrontRiskyMuCov(
  muRet,
  volRet,
  corrRet,
  npoints = 100,
  display = TRUE,
  efront.only = TRUE,
  values = FALSE,
  digits = NULL
)
```

Arguments

`muRet`	Numeric vector of asset mean returns
`volRet`	Numeric vector of asset standard deviations/volatilities
`corrRet`	Correlation matrix of asset returns
`npoints`	Integer number of points on efficient frontier, default 100
`display`	Logical variable, default TRUE
`efront.only`	Logical variable, default TRUE
`values`	Logical variable, default = FALSE
`digits`	Integer number of significant

Details

When efront.only = TRUE only the efficient frontier is computed, and if FALSE the entire frontier is computed. When value = TRUE the efficient frontier mean and volatility values are returned, and when value = FALSE these values are not returned.

Value

Plot of efficient frontier

Examples

```
args(mathEfrontRiskyMuCov)
```

---

mathGmv *Global Minimum Variance Portfolio (GMV)*

Description

Computes the weights of a GMV portfolio, and its mean return and volatility based on portfolio asset returns

Usage

```
mathGmv(returns, digits = NULL)
```

Arguments

returns	Matrix or xts object of returns
digits	Integer value of number of significant digits, default NULL

Value

List of GMV portfolio weights, mean return and volatility

Examples

```
args(mathGmv)
```

mathGmvMuCov *Global Minimum Variance Portfolios From Mu and Cov*

Description

Compute the weights, mean return and volatility of a GMV portfolio based on user specified mean vector and covariance matrix

Usage

```
mathGmvMuCov(muRet, volRet, corrRet, digits = 3)
```

Arguments

muRet	Mean vector
volRet	Volatility vector
corrRet	matrix of correlations
digits	Integer value number of decimal places, default 3

Value

a list contains weights, mean return and volatility of a GMV portfolio

Examples

```
args(mathGmvMuCov)
```

mathTport *Tangency Portfolio Weights*

Description

Computes the portfolio weights of the tangency portfolio, and its mean return and volatility. The tangency portfolio is defined by the line connecting the zero volatility risk-free rate to its tangency point on the efficient frontier.

Usage

```
mathTport(returns, rf = 0.005, digits = NULL)
```

Arguments

returns	A vector or xts object
rf	The risk-free rate, default 0.005
digits	Number of significant digits default NULL

Value

Tangency portfoliow weights, mean and volatility

Examples

```
args(mathTport)
```

`mathWtsEfrontRisky` *Efficient Frontier Portfolio Weights Vectors*

Description

Uses time series of asset returns to compute the weights vectors for a set of points along the efficient frontier that are defined by their mean return values

Usage

```
mathWtsEfrontRisky(returns, mu.efront, digits = NULL)
```

Arguments

`returns`	A multivariate xts object of n asset returns
`mu.efront`	A vector of specified efficient frontier mean returns
`digits`	Integer number of significant digits with default NULL

Value

A matrix with first row containing the mean (MU) along the efficient frontier, the second row containing the standard deviation, and the following n rows contain the n weight vectors along the efficient fronier

Examples

```
args(mathWtsEfrontRisky)
```

mathWtsEfrontRiskyMuCov *Efficient Frontier Portfolio Weights Vectors*

Description

Same as function "mathWtsEfrontRisky" except that instead a user specified time series of portfolio asset returns, it is based on user specified returns mean vector and covariance matrix

Usage

```
mathWtsEfrontRiskyMuCov(muRet, volRet, corrRet, mu.efront, digits = NULL)
```

Arguments

muRet	Vector of asset mean returns
volRet	Vector of asset volatilities
corrRet	Asset correlation matrix
mu.efront	A vector of specified efficient frontier mean returns
digits	Integer number of significant digits with default NULL

Value

A matrix whose first row contains the mean returns along the efficient frontier, the second row contains the corresponding volatilities, and the remaining rows contain the components of the corresponding weight vectors.

Examples

```
args(mathWtsEfrontRiskyMuCov)
```

meanReturns4Types *Four Types of Mean Returns*

Description

Computation of arithmetic mean, logarithmic mean, geometric mean, and an approximate geometric mean.

Usage

```
meanReturns4Types(return, robust = FALSE, eff = 0.95)
```

Arguments

return	An xts object or a numeric vector of returns
robust	A logical value controlling whether a classical or robust sample mean and standard deviation is computed. Default is FALSE
eff	Normal distribution efficiency of RobStatTM function locScaleM() used for computing a robust location estimate

Value

fourMeans numeric values of the four means in the Description

Examples

```
args(meanReturns4Types)
```

---

opt.outputMvoPCRA *Optimal Portfolio Weights and Performance*

Description

Converts output of PortfolioAnalytics function optimize.portfolio, which computes a minimum variance portfolio, to a list containing the portfolio weights vector, mean, volatility and Sharpe Ratio.

Converts output of ‘optimize.portfolio‘ to a list of the portfolio weights, mean, volatility and Sharpe Ratio.

Usage

```
opt.outputMvoPCRA(
  opt,
  returns,
  digits = NULL,
  itemNames = NULL,
  annualize = TRUE,
  frequency = "monthly",
  rf = 0
)
```

Arguments

opt	List output of ‘optimize.portfolio‘
returns	Multivariate xts object of portfolio assets returns
digits	Integer number of significant digits with default NULL
itemNames	character vector of use-supplied names for portfolio weights, mean, standard deviation and Sharpe Ratio

annualize	Logical with default TRUE
frequency	Returns frequency: "monthly", "weekly" or "daily", with default "monthly"
rf	Numeric value of risk-free rate with default 0.0

Details

This function uses the weights returned by optimize.portfolio, along with the portfolio monthly, weekly or daily assets returns, and a risk-free rate, to compute the portfolio mean return, volatility, and Sharpe Ratio. By default the latter three are annualized, but the user may choose to return non-annualized performance values.

Value

A list containing the portfolio numeric weights, mean value, standard deviation and Sharpe Ratio, with default names Wgts, Mean, StdDev, and SR, or user-supplied names as a character vector value for the argument ‘itemNames‘.

Author(s)

R. Douglas Martin

Examples

```
args(opt.outputMvoPCRA)
```

---

plotLSandRobustSFM *Robust and Least Square Single Factor Model (SFM) Fits*

Description

Plot of Least squares and robust single factor model (SFM) fits, with outliers identified, and legend containing slope and intercept coefficient estimates with standard errors in parentheses.

Usage

```
plotLSandRobustSFM(
  x,
  family = "mopt",
  efficiency = 0.95,
  mainText = NULL,
  ylimits = NULL,
  legendPos = "topleft",
  makePct = FALSE
)
```

Arguments

x	A univariate xts object.
family	Robust loss function choice with default mopt
efficiency	Estimator Normal distribution efficiency, default 0.95
mainText	Main title, if any.
ylimits	Vertical axis limits.
legendPos	Legend position.
makePct	Logical variable with default FALSE

Details

The robust fit is computed using the lmrobdetMM() function in the R package RobStatTM. For other choices of efficiency and family see the RobStatTM package help(lmrobdetMM)

Value

No value returned, instead plot with straight line fits and legend is displayed

Examples

```
args(plotLSandRobustSFM)
```

---

qqnormDatWindat *qqnormDatWindat*

Description

Normal QQPlot of data and Winsorized data

Usage

```
qqnormDatWindat(
  dat,
  windat,
  fraction = 0.01,
  ylim = NULL,
  main = main,
  facName = NULL
)
```

Arguments

dat	Numeric data vector
windat	Numeric Winsorized data set
fraction	Fraction of data that is Winsorized
ylim	Numeric data with two values that control vertical plot range
main	Character main title of plot
facName	Character data for y axis label

Details

The result plot displays a normal QQPlot of the orginal data as solid points, along with the horizontal display of the Winsorized data as small circles.

Value

A normal QQPlot of data with overlaid Winsorized data

Examples

```
args(qqnormDatWindat)
```

---

retDD *Stock with Ticker DD*

Description

Weekly returns (RET) of stock with ticker DD for 1986 and 1987, along with market returns (MKT) and risk-free rate (RF).

Usage

```
data(retDD)
```

Format

Multivariate time series xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business.

Examples

```
library(PCRA)
library(zoo)
data(retDD)
head(retDD)
range(index(retDD))
```

retEDS *Stock with Ticker EDS*

Description

Weekly returns (RET) of stock with ticker EDS for 2002 and 2003, along with market returns (MKT) and risk-free rate (RF).

Usage

```
data(retEDS)
```

Format

Multivariate time series xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business.

Examples

```
library(PCRA)
library(zoo)
data(retEDS)
head(retEDS)
range(index(retEDS))
```

retFNB *Stock with Ticker FNB*

Description

Weekly returns (RET) of stock with ticker FNB for 2008

Usage

```
data(retFNB)
```

Format

Univariate time series xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business.

Examples

```
library(PCRA)
library(zoo)
data(retFNB)
head(retFNB)
range(index(retFNB))
```

---

retKBH *Stock with Ticker KBH*

Description

Weekly returns (RET) of stock with ticker KBH for 2007 and 2008, along with market returns (MKT) and risk-free rate (RF).

Usage

```
data(retKBH)
```

Format

Multivariate time series xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business.

Examples

```
library(PCRA)
library(zoo)
data(retKBH)
head(retKBH)
range(index(retKBH))
```

---

retMER *Stock with Ticker MER*

Description

Weekly returns (RET) of stock with ticker MER for 2002 and 2003, along with market returns (MKT) and risk-free rate (RF).

Usage

```
data(retMER)
```

Format

Multivariate time series xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business.

Examples

```
library(PCRA)
library(zoo)
data(retMER)
head(retMER)
range(index(retMER))
```

---

retOFG *Stock with Ticker OFG*

Description

Weekly returns (RET) of stock with ticker OFG for 2007 and 2008, along with market returns (MKT) and risk-free rate (RF).

Usage

```
data(retOFG)
```

Format

Multivariate time series xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business.

Examples

```
library(PCRA)
library(zoo)
data(retOFG)
head(retOFG)
range(index(retOFG))
```

retPSC *Stock with Ticker PSC*

Description

Weekly returns (RET) of stock with ticker PSC for 1987 and 1088, along with market returns (MKT) and risk-free rate (RF).

Usage

```
data(retPSC)
```

Format

Multivariate time series xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business.

Examples

```
library(PCRA)
library(zoo)
data(retPSC)
head(retPSC)
range(index(retPSC))
```

returnsCRSPxts *Select CRSP Stocks Returns*

Description

Uses selectCRSPandSPGMI to select a subset of the stocksCRSP data, and convert it to an xts object that contains the returns of a set of stocks, along with those of the MktIndexCRSP and the Ret13WkBill.

NOTE: For this function to work, the selectCRSPandSPGMI must include the the stockItems TickerLast, MktIndexCRSP and Ret13WkBill.

Usage

```
returnsCRSPxts(stocksData)
```

Arguments

stocksData	The data.table created by selectCRSPandSPGMI

Value

A multivariate xts object

Examples

```
data.table::setDTthreads(1)
library(PCRA)
library(xts)
library(data.table)
stockItems <- c("Date","TickerLast","CapGroupLast","Return","MktIndexCRSP",
                "Ret13WkBill")
dateRange <- c("1997-01-31","2002-12-31")
stocksDT <- selectCRSPandSPGMI("monthly",dateRange = dateRange, stockItems =
                                stockItems, factorItems = NULL,
                                outputType   = "data.table")
stocksDT <- stocksDT[CapGroupLast == "SmallCap"]
ret <- returnsCRSPxts(stocksDT)
tickers <- unique(stocksDT[,TickerLast])
tickers10 <- tickers[11:20]
colnames <- c(tickers10,"Market","RiskFree")
head(ret[,colnames],1)
```

---

retVHI *Stock with Ticker VHI*

Description

Weekly returns (RET) of stock with ticker VHI for 1990 and 1991, along with market returns (MKT) and risk-free rate (RF).

Usage

```
data(retVHI)
```

Format

Multivariate time series xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business.

Examples

```
library(PCRA)
library(zoo)
data(retVHI)
head(retVHI)
range(index(retVHI))
```

retWTS *Stock with Ticker WTS*

Description

Weekly returns (RET) of stock with ticker WTS for 2009 and 2010, along with market returns (MKT) and risk-free rate (RF).

Usage

```
data(retWTS)
```

Format

Multivariate time series xts object

Source

Center for Research in Security Prices, LLC (CRSP), an Affiliate of the University of Chicago Booth School of Business.

Examples

```
library(PCRA)
library(zoo)
data(retWTS)
head(retWTS)
range(index(retWTS))
```

selectCRSPandSPGMI *Select and merge data from the stocksCRSP and factorsSPGMI data sets*

Description

Select data from stocksCRSP and merge with factorsSPGMI for use in risk model estimation or returns analysis. This version of selectCRSPandSPGMI allows various options for subsetting. Users may specify a dateRange for the data as well as specifying specific lists of tickers, market capitalization groups, or sectors via the subsetType and subsetValues parameters. Additionally, for data.table output, users may select specific columns for each of stocksCRSP and factorsSPGMI to be included in the final output via the stockItems and factorItems parameters.

Usage

```
selectCRSPandSPGMI(
  periodicity = "monthly",
  dateRange = c("1993-01-31", "2015-12-31"),
 stockItems = c("Date", "TickerLast", "CapGroupLast", "Sector", "Return", "Ret13WkBill",
    "MktIndexCRSP"),
  factorItems = c("BP", "LogMktCap", "SEV"),
  subsetType = NULL,
  subsetValues = NULL,
  outputType = "xts"
)
```

Arguments

periodicity	Character "monthly","weekly","daily". Currently only "monthly" is supported.
dateRange	A character vector providing a start data and an end date, having the same form as c("2006-01-31", "2010-12-31").
stockItems	A character vector that is a subset of the names of columns in the stocksCRSP data.table. Set to "NULL" when no data from this data set is desired in the final output.
factorItems	A character vector that is a subset of the names of columns in the factorsSPGMI data.table. Set to "NULL" when no data from this data set is desired in the final output.
subsetType	Character "TickerLast", "sector" or "CapGroupLast". Default NULL for no subsetting.
subsetValues	Character vector containing either a list of TickerLast values, Sector values, or CapGroup values.
outputType	Character "xts" for a wide multivariate xts returns object, or a long format "data.table" object foranalysis and risk model estimation. Set to "xts" by default.

Details

Users select a periodicity for the data (stocksCRSP is available in daily, weekly, and monthly variants). When weekly or daily data are selected, the function re-samples the lower frequency factorsSPGMI data up to the chosen stocksCRSP frequency.

IMPORTANT: When using selectCRSPandSPGMI with periodicity = "weekly", you must first use the code line stocksCRSPweekly <- getPCRAData(data = "stocksCRSPweekly"), and for "daily" data use stocksCRSPdaily <- getPCRAData(data = "stocksCRSPdaily").

Users may select all columns from both data sets, a specified set of columns, or by setting either stockItems or factorItems to "NULL", may select only items from the other data set (that is, if only the stocksCRSP data is desired, set factorItems to NULL).

Users may select a specific range of dates ("dateRange") for the data.

Smaller sub-samples of the data (fewer rows) can be returned by selecting a specific Sectors, CapGroupLast (MicroCap, SmallCap, MidCap, LargeCap) of interest, or by specifying a list of TickerLast values for which data can be returned. This is accomplished via the subsetType and subsetValues parameters.

Value

Either a multivariate xts object of returns, plus the risk-free rate ("Ret13WkBill") and market return ("MktIndexCRSP") values, or a data.table consisting of selected stocks and/or factor exposures data.

Examples

```
data.table::setDTthreads(1)
data(stocksCRSP)
return_data <- selectCRSPandSPGMI(periodicity  = "monthly",
                                  dateRange  = c("2006-01-31", "2006-07-31"),
                                  stockItems = c("Date", "TickerLast",
                                  "CapGroupLast", "Sector", "Return",
                                  "Ret13WkBill", "MktIndexCRSP"),
                                  factorItems  = NULL,
                                  subsetType   = NULL,
                                  subsetValues = NULL,
                                  outputType   = "xts")

length(unique(stocksCRSP$TickerLast))
dim(return_data) #includes all tickers plus risk free rate & market return columns

stocks_factors <- selectCRSPandSPGMI(periodicity = "monthly",
                                  dateRange    = c("2006-01-31", "2006-07-31"),
                                  stockItems   = c("Date", "TickerLast",
                                  "CapGroupLast", "Sector", "Return",
                                  "Ret13WkBill", "MktIndexCRSP"),
                                  factorItems  = c("BP", "LogMktCap", "SEV"),
                                  subsetType   = NULL,
                                  subsetValues = NULL,
                                  outputType   = "data.table")
names(stocks_factors)
str(stocks_factors)
```

SKest *Skewness estimator*

Description

Sample estimate of skewness This function will eventually have a robust estimation option

Usage

```
SKest(x)
```

Arguments

x	A numeric vector

Value

numeric value of estimate of skewness

Examples

```
args(SKest)
```

SP400Industrials *SP400Industrials*

Description

Year-end data on the S&P 400 Industrials® Index from 1957 to 1987 extracted from a paper copy of the S&P Analysts' Handbook.

Usage

```
data(SP400Industrials)
```

Format

A data frame with observations on the S&P 400 Industrials® index from 1957 to 1987

- **Year:** type 'num'.
- **Sales:** type 'num'. Revenues per share for the S&P 400 Industrials for the calendar year.
- **Operating_Profit:** type 'num'. Operating Income per share for the S&P 400 Industrials for the calendar year.
- **Profit_Margin_Pct:** type 'num'. Ratio of Operating_Profit to Sales for the S&P 400 Industrials expressed as a percentage.
- **Depreciation:** type 'num'. Depreciation expense per share for the S&P 400 Industrials for the calendar year.
- **Income_Taxes:** type 'num'. Tax expense per share for the S&P 400 Industrials for the calendar year.
- **Earnings_Per_Share:** type 'num'. Fully Diluted As-Reported Earnings per share for the S&P 400 Industrials for the calendar year.
- **Earnings_Pct_of_Sales:** type 'num'. Ratio of Diluted_EPS to Sales for the S&P 400 Industrials from 1993 to 2007 expressed as a percentage. Definition currently unknown for earlier years.
- **Dividends_Per_Share:** type 'num'. Dividends per share for the S&P 400 Industrials for the calendar year.

- **Dividends_Pct_of_Earnings:** type 'num'. Ratio of Dividends_Per_Share to Diluted_EPS for the S&P 400 Industrials, expressed as a percentage.
- **Price_High:** type 'num'. Highest price level achieved by the S&P 400 Industrials Index during the calendar year.
- **Price_Low:** type 'num'. Lowest price level achieved by the S&P 400 Industrials Index during the calendar year.
- **PE_Ratio_High:** type 'num'. Ratio of Price_High to Diluted_EPS for the S&P 400 Industrials Index.
- **PE_Ratio_Low:** type 'num'. Ratio of Price_Low to Diluted_EPS for the S&P 400 Industrials Index.
- **Dividend_Yld_High:** type 'num'. Ratio of Dividends_Per_Share to Price_High for the S&P 400 Industrials Index.
- **Dividend_Yld_Low:** type 'num'. Ratio of Dividends_Per_Share to Price_Low for the S&P 400 Industrials Index.
- **Book_Value_Per_Share:** type 'num'. Year-end (12/31) Book Value (or Shareholders' Equity) per share for the S&P 400 Industrials Index.
- **Book_Value_Pct_Return:** type 'num'. Definition Unknown.
- **Working_Capital:** type 'num'. Definition Unknown.
- **Capital_Expenditures:** type 'num'. Capital Expenditures per share for the S&P 400 Industrials for the calendar year.

Details

Data for the S&P® 400 Industrials is taken from a paper copy of the S&P® Analysts' Handbook published in 1988. The average price level of the index in 1941-1943 was set to 10. The index is based on 70 individual groups, and price information on it was backfilled to 1918, though we do not have access to it. The original S&P® 500 index was created in late February 1957 and included 425 industrial stocks, 15 rail stocks and 60 utility stocks. In July 1976, financial stocks were added to the index, which now included 400 industrials, 40 utilities, 40 finance and 20 transport stocks. It is possible that the pre-1967 history was recreated by S&P® for the Analysts' Handbook. See https://globalfinancialdata.com/the-sp-composite-before-1957 for a useful history of the various S&P® indices.

Source

S&P Dow Jones Indices. S&P®, S&P 400 Industrials®, S&P 425 Industrials®, S&P Industrials®and S&P 500® are registered trademarks of Standard & Poor's Financial Services LLC, and Dow Jones® is a registered trademark of Dow Jones Trademark Holdings LLC. © 2023 S&P Dow Jones Indices LLC, #' its affiliates and/or its licensors. All rights reserved. Redistribution of the data is not permitted, and use of the data in derivative works is not permitted without the written permission of S&P Dow Jones Indices LLC.

References

Chapter 13 (Expected Returns) of Martin, Philips, Scherer, Stoyanov and Li, Portfolio Construction and Risk Analysis, Springer, 2024.

Examples

```
data(SP400Industrials)
names(SP400Industrials)
head(SP400Industrials, 5)
tail(SP400Industrials, 5)
```

SP425Industrials *SP425Industrials*

Description

Year-end data on the S&P 425 Industrials® Index from 1946 to 1966 extracted from a paper copy of the S&P Analysts' Handbook.

Usage

```
data(SP425Industrials)
```

Format

A data frame with observations on the S&P 425 Industrials® index from 1946 to 1966

- **Year:** type 'num'.
- **Sales:** type 'num'. Revenues per share for the S&P 425 Industrials for the calendar year.
- **Operating_Profit:** type 'num'. Operating Income per share for the S&P 425 Industrials for the calendar year.
- **Profit_Margin_Pct:** type 'num'. Ratio of Operating_Profit to Sales for the S&P 425 Industrials expressed as a percentage.
- **Depreciation:** type 'num'. Depreciation expense per share for the S&P 425 Industrials for the calendar year.
- **Federal_Income_Taxes:** type 'num'. Federl Tax expense per share for the S&P 425 Industrials for the calendar year.
- **Earnings_Per_Share:** type 'num'. Fully Diluted As-Reported Earnings per share for the S&P 425 Industrials for the calendar year.
- **Earnings_Pct_of_Sales:** type 'num'. Ratio of Diluted_EPS to Sales for the S&P 425 Industrials from 1993 to 2007 expressed as a percentage. Definition currently unknown for earlier years.
- **Dividends_Per_Share:** type 'num'. Dividends per share for the S&P 425 Industrials for the calendar year.
- **Dividends_Pct_of_Earnings:** type 'num'. Ratio of Dividends_Per_Share to Diluted_EPS for the S&P 425 Industrials, expressed as a percentage.
- **Price_High:** type 'num'. Highest price level achieved by the S&P 425 Industrials Index during the calendar year.

- **Price_Low:** type 'num'. Lowest price level achieved by the S&P 425 Industrials Index during the calendar year.
- **PE_Ratio_High:** type 'num'. Ratio of Price_High to Diluted_EPS for the S&P 425 Industrials Index.
- **PE_Ratio_Low:** type 'num'. Ratio of Price_Low to Diluted_EPS for the S&P 425 Industrials Index.
- **Dividend_Yld_High:** type 'num'. Ratio of Dividends_Per_Share to Price_High for the S&P 425 Industrials Index.
- **Dividend_Yld_Low:** type 'num'. Ratio of Dividends_Per_Share to Price_Low for the S&P 425 Industrials Index.
- **Book_Value_Per_Share:** type 'num'. Year-end (12/31) Book Value (or Shareholders' Equity) per share for the S&P 425 Industrials Index.
- **Book_Value_Pct_Return:** type 'num'. Definition Unknown.
- **Working_Capital:** type 'num'. Definition Unknown.
- **Capital_Expenditures:** type 'num'. Capital Expenditures per share for the S&P 425 Industrials for the calendar year.

Details

Data for the S&P® 425 Industrials is taken from a paper copy of the S&P® Analysts' Handbook published in 1967. The average price level of the index in 1941-1943 was set to 10. The index is based on 70 individual groups, and price information on it was backfilled to 1918, though we do not have access to it. The original S&P® 500 index was created in late February 1957 and included 425 industrial stocks, 15 rail stocks and 60 utility stocks. It maintained this composition until July 1976 when finance stocks were added to the index. See https://globalfinancialdata.com/the-sp-composite-before-1957 for a useful history of the various S&P® indices.

Source

S&P Dow Jones Indices. S&P®, S&P 400 Industrials®, S&P 425 Industrials®, S&P Industrials®and S&P 500® are registered trademarks of Standard & Poor's Financial Services LLC, and Dow Jones® is a registered trademark of Dow Jones Trademark Holdings LLC. © 2023 S&P Dow Jones Indices LLC, #' its affiliates and/or its licensors. All rights reserved. Redistribution of the data is not permitted, and use of the data in derivative works is not permitted without the written permission of S&P Dow Jones Indices LLC.

References

Chapter 13 (Expected Returns) of Martin, Philips, Scherer, Stoyanov and Li, Portfolio Construction and Risk Analysis, Springer, 2024.

Examples

```
data(SP425Industrials)
names(SP425Industrials)
head(SP425Industrials, 5)
tail(SP425Industrials, 5)
```

SP500 *SP500*

Description

Year-end data on the S&P 500®, Nominal GDP and Consumer Prices from 1925 to the most recent year-end for which final data is available.

Usage

```
data(SP500)
```

Format

A data frame with observations on the S&P 500® from 1925 to the most recent year end for which final data is available:

- **Year:** type 'num'.
- **SP500PriceHigh:** type 'num'. Highest price level achieved by the S&P 500 during the calendar year.
- **SP500PriceLow:** type 'num'. Lowest price level achieved by the S&P 500 during the calendar year.
- **SP500PriceClose:** type 'num'. Year-end (12/31) price of the S&P 500.
- **SP500EpsAll4Q:** type 'num'. As-Reported Earnings per share for the S&P 500 for the entire calendar year.
- **SP500EpsBest3Q:** type 'num'. 4/3 x Sum of the three highest quarterly earnings per share for the S&P 500 during the calendar year.
- **SP500EpsBest2Q:** type 'num'. 2 x Sum of the two highest quarterly earnings per share for the S&P 500 during the calendar year.
- **SP500EpsBest1Q:** type 'num'. 4 x the highest earnings per share in a quarter for the S&P 500 during the calendar year.
- **SP500RevenuePS:** type 'num'. Annual Revenues per share for the S&P 500 during the calendar year.
- **SP500BookValuePS:** type 'num'. Year-end (12/31) Book Value (or Shareholders' Equity) per share for the S&P 500.
- **SP500DPS:** type 'num'. Dividends per share for the S&P 500 during the calendar year.
- **SP500OperatingEPS:** type 'num'. Operating Earnings per share for the S&P 500 for the calendar year.
- **SP500NomRet:** type 'num'. Nominal total return including both change in price and dividends and not adjusted for inflation for the S&P 500 for the current calendar year.
- **SP500Nom1YrFwdRet:** type 'num'. Nominal total return including both change in price and dividends and not adjusted for inflation for the S&P 500 for the FOLLOWING calendar year. This is the same as SP500NomRet with a one year lag. It is included primarily to make it easy to build forecasting models without any need to apply a lag operator to SP500NomRet.

- **CPIAUCNS:** type 'num'. Consumer Price Index for All Urban Consumers: All Items in U.S. City Average, as of year end.
- **GDPA:** type 'num'. Nominal GDP at an annual frequency.

Details

This dataset was constructed by combining information in various datasets, and is updated annually using data published in https://www.spglobal.com/spdji/en/documents/additional-material/sp-500-eps-est.xlsx. Final year-end numbers Revenues per share from 1992 to 199 are taken from SP500from1967to2007, and prior to this are estimated from the per-share revenues of the S&P 425 Industrials® and S&P Industrials® indices, using the procedure described in Philips, Thomas and Ural, Cenk, "Uncloaking Campbell and Shiller's CAPE: A Comprehensive Guide to its Construction and Use", Journal of Portfolio Management, Vol 43, No. 1, Fall 2016, pp. 109-125.

Source

S&P Dow Jones Indices and Federal Reserve Bank of St. Louis. S&P®, S&P 400 Industrials®, S&P 425 Industrials®, S&P Industrials®and S&P 500® are registered trademarks of Standard & Poor's Financial Services LLC, and Dow Jones® is a registered trademark of Dow Jones Trademark Holdings LLC. © 2023 S&P Dow Jones Indices LLC, #' its affiliates and/or its licensors. All rights reserved. Redistribution of the data is not permitted, and use of the data in derivative works is not permitted without the written permission of S&P Dow Jones Indices LLC. Data for the S&P 500® is updated using the QUARTERLY DATA tab of https://www.spglobal.com/spdji/en/documents/additional-material/sp-500-eps-est.xlsx. Final year-end numbers are typically reported in April or May of the following year. CPIAUCNS is obtained from the Federal Reserve Bank of St. Louis' FRED database at https://fred.stlouisfed.org/series/CPIAUCNS. GDPA is obtained from the Federal Reserve Bank of St. Louis' FRED database at https://fred.stlouisfed.org/series/GDPA.

References

Chapter 13 (Expected Returns) of Martin, Philips, Scherer, Stoyanov and Li, Portfolio Construction and Risk Analysis, Springer, 2024.

Examples

```
data(SP500)
names(SP500)
head(SP500, 5)
tail(SP500, 5)
```

SP500from1967to2007 *SP500from1967to2007*

Description

Year-end data on the S&P 500® Index from 1967 to 2007 extracted from a paper copy of the S&P Analysts' Handbook. The title of the page from which this data was extracted says "Historical Index - S&P 500 Composite - 500 stocks". It includes some information (e.g. Cash Flow) that is no longer provided. An extensive dataset for the S&P 500® and various other S&P® indices can be downloaded from https://www.spglobal.com/spdji/en/documents/additional-material/sp-500-eps-est.xlsx. Final year-end numbers are typically reported in April or May of the following year.

Usage

```
data(SP500from1967to2007)
```

Format

A data frame with observations on the S&P 500® index from 1967 to 2007

- **Year:** type 'num'.
- **Sales:** type 'num'. Revenues per share for the S&P 500® for the calendar year.
- **Cash_Flow:** type 'num'. Cash Flow per share for the S&P 500® for the calendar year.
- **Diluted_EPS:** type 'num'. Fully Diluted As-Reported Earnings per share for the S&P 500® for the calendar year.
- **Dividends_Per_Share:** type 'num'. Dividends per share for the S&P 500® for the calendar year.
- **Dividends_Pct_of_Earnings:** type 'num'. Ratio of Dividends per share to Fully Diluted As-Reported Earnings per share for the S&P 500® for the calendar year, expressed as a percentage.
- **Price_High:** type 'num'. Highest price level achieved by the S&P 500® during the calendar year.
- **Price_Low:** type 'num'. Lowest price level achieved by the S&P 500® during the calendar year.
- **Price_Close:** type 'num'. Year-end (12/31) price of the S&P® Index.
- **PE_Ratio_High:** type 'num'. Ratio of Price_High to Diluted_EPS for the S&P 500®.
- **PE_Ratio_Low:** type 'num'. Ratio of Price_Low to Diluted_EPS for the S&P 500®.
- **PE_Ratio_Close:** type 'num'. Ratio of Price_Close to Diluted_EPS for the S&P 500®.
- **Dividend_Yld_High:** type 'num'. Ratio of Dividends_Per_Share to Price_High for the S&P 500®.
- **Dividend_Yld_Low:** type 'num'. Ratio of Dividends_Per_Share to Price_Low for the S&P 500®.
- **Dividend_Yld_Close:** type 'num'. Ratio of Dividends_Per_Share to Price_Close for the S&P 500®.
- **Total_Return_Index:** type 'num'. Cumulative total return of the S&P 500® including both dividends and price return. Start date for the series (when it was likely normalized to 100) is not known.

- **Book_Value_Per_Share:** type 'num'. Year-end (12/31) Book Value (or Shareholders' Equity) per share for the S&P 500®.
- **Book_Value_Pct_Return:** type 'num'. Definition currently unknown.
- **Price_to_Book_Ratio:** type 'num'. Ratio of Price_Close to Book_Value_Per_Share for the S&P 500®.

Details

Data for the S&P® Industrials is taken from a paper copy of the S&P® Analysts' Handbook published in 2008. It includes one variable (Cash Flow) that is no longer provided, and excludes many others (Operating Earnings, Capital Expenditures, Earnings Estimates, Index Divisor, beaten estimates, sector breakdowns, projected growth rates by sector, effective tax rate etc.) that are now provided by S&P® in the spreadsheet https://www.spglobal.com/spdji/en/documents/additional-material/sp-500-eps-est.xlsx. Final year-end numbers are typically reported in April or May of the following year.

Source

S&P Dow Jones Indices. S&P®, S&P 400 Industrials®, S&P 425 Industrials®, S&P Industrials®and S&P 500® are registered trademarks of Standard & Poor's Financial Services LLC, and Dow Jones® is a registered trademark of Dow Jones Trademark Holdings LLC. © 2023 S&P Dow Jones Indices LLC, its affiliates and/or its licensors. All rights reserved. ' Redistribution of the data is not permitted, and use of the data in derivative works is not permitted without the written permission of S&P Dow Jones Indices LLC.

References

Chapter 13 (Expected Returns) of Martin, Philips, Scherer, Stoyanov and Li, Portfolio Construction and Risk Analysis, Springer, 2024.

Examples

```
data(SP500from1967to2007)
names(SP500from1967to2007)
head(SP500from1967to2007, 5)
tail(SP500from1967to2007, 5)
```

`SPIndustrials` *SPIndustrials*

Description

Year-end data on the S&P Industrials® Index from 1967 to 2007 extracted from a paper copy of the S&P Analysts' Handbook.

Usage

```
data(SPIndustrials)
```

Format

A data frame with observations on the S&P Industrials® index from 1967 to 2007

- **Year:** type 'num'.
- **Sales:** type 'num'. Revenues per share for the S&P Industrials for the calendar year.
- **Operating_Profit:** type 'num'. Operating Income per share for the S&P Industrials for the calendar year.
- **Profit_Margin_Pct:** type 'num'. Ratio of Operating_Profit to Sales for the S&P Industrials from 1993 to 2007 expressed as a percentage. Definition currently unknown for earlier years.
- **Depreciation:** type 'num'. Depreciation expense per share for the S&P Industrials for the calendar year.
- **Income_Tax:** type 'num'. Tax expense per share for the S&P Industrials for the calendar year.
- **Cash_Flow:** type 'num'. Cash Flow per share for the S&P Industrials for the calendar year.
- **Diluted_EPS:** type 'num'. Fully Diluted As-Reported Earnings per share for the S&P Industrials for the calendar year.
- **Earnings_Pct_of_Sales:** type 'num'. Ratio of Diluted_EPS to Sales for the S&P Industrials from 1993 to 2007 expressed as a percentage. Definition currently unknown for earlier years.
- **Dividends_Per_Share:** type 'num'. Dividends per share for the S&P Industrials for the calendar year.
- **Dividends_Pct_of_Earnings:** type 'num'. Ratio of Dividends_Per_Share to Diluted_EPS for the S&P Industrials, expressed as a percentage.
- **Price_High:** type 'num'. Highest price level achieved by the S&P Industrials Index during the calendar year.
- **Price_Low:** type 'num'. Lowest price level achieved by the S&P Industrials Index during the calendar year.
- **Price_Close:** type 'num'. Year-end (12/31) price of the S&P Industrials Index.
- **PE_Ratio_High:** type 'num'. Ratio of Price_High to Diluted_EPS for the S&P Industrials Index.
- **PE_Ratio_Low:** type 'num'. Ratio of Price_Low to Diluted_EPS for the S&P Industrials Index.
- **PE_Ratio_Close:** type 'num'. Ratio of Price_Close to Diluted_EPS for the S&P Industrials Index.
- **Dividend_Yld_High:** type 'num'. Ratio of Dividends_Per_Share to Price_High for the S&P Industrials Index.
- **Dividend_Yld_Low:** type 'num'. Ratio of Dividends_Per_Share to Price_Low for the S&P Industrials Index.
- **Dividend_Yld_Close:** type 'num'. Ratio of Dividends_Per_Share to Price_Close for the S&P Industrials Index.
- **Total_Return_Index:** type 'num'. Cumulative total return of the S&P Industrials Index including both dividends and price return. Start date for the series is not known.

- **Book_Value_Per_Share:** type 'num'. Year-end (12/31) Book Value (or Shareholders' Equity) per share for the S&P Industrials Index.
- **Book_Value_Pct_Return:** type 'num'. Definition Unknown.
- **Price_to_Book_Ratio:** type 'num'. Ratio of Price_Close to Book_Value_Per_Share for the S&P Industrials Index.

Details

Data for the S&P® Industrials is taken from a paper copy of the S&P® Analysts' Handbook published in 2008. The average price level of the index in 1941-1943 was set to 100. See https://globalfinancialdata.com/the-sp-composite-before-1957 for a useful history of the various S&P® indices.

Source

S&P Dow Jones Indices. S&P®, S&P 400 Industrials®, S&P 425 Industrials®, S&P Industrials®and S&P 500® are registered trademarks of Standard & Poor's Financial Services LLC, and Dow Jones® is a registered trademark of Dow Jones Trademark Holdings LLC. © 2023 S&P Dow Jones Indices LLC, #' its affiliates and/or its licensors. All rights reserved. Redistribution of the data is not permitted, and use of the data in derivative works is not permitted without the written permission of S&P Dow Jones Indices LLC.

References

Chapter 13 (Expected Returns) of Martin, Philips, Scherer, Stoyanov and Li, Portfolio Construction and Risk Analysis, Springer, 2024.

Examples

```
data(SPIndustrials)
names(SPIndustrials)
head(SPIndustrials, 5)
tail(SPIndustrials, 5)
```

`stocksCRSP` *stocksCRSP*

Description

CRSP monthly stocks data for 294 stocks 1993 to 2015

Usage

```
data(stocksCRSP)
```

Format

A data.table object with 82000 observations on 15 variables:

- **Date:** type ‘Date‘.
- **TickerLast:** type ‘chr‘. The ticker as of December 2015
- **Ticker:** type ‘chr‘. Monthly ticker period
- **Company:** type ‘chr‘. The name of company with TickerLast
- **CapGroupLast:** type ‘chr‘. Company market capitalization group as of December 2015, one of: MicroCap, SmallCap, MidCap or LargeCap
- **CapGroup:** type ‘chr‘. Monthly market capitalization group
- **GICS:** type ‘chr‘. 6 digit S&P GICS code
- **Sector:** type ‘chr‘. One of 10 sectors specified by the first two digits of the GICS code
- **Return:** type ‘num‘. Arithmetic stock return from one period to the next in decimal form
- **RetExDiv:** type ‘num‘.
- **Price:** type ‘num‘. Stock price at each time period in decimal form
- **PrcSplitAdj:** type ‘num‘.
- **Ret4WkBill:** type ‘num‘. Return of 4 week Treasury bill
- **Ret13WkBill:** type ‘num‘. Return of 13 week Treasury bill
- **Ret1YrBill:** type ‘num‘. Return of 1 year Treasury bill
- **mktIndexCRSP:** type ‘num‘. CRSP value weighted market return

Details

The four CapGroupLast categorizations of the stocks were determined using the three capitalization breakpoints $15.6B, $5.4B, $600M. Details concerning the construction of the monthly CapGroup categorizations will eventually be provided in a Vignette.

Weekly and daily versions stocksCRSPweekly and stocksCRSPdaily may be obtained using the function getPCRAData() - see PCRAData.R.

Source

Center for Research in Security Prices (CRSP) at the University of Chicago’s Booth School of Business (CRSP). NOTE: CRSP data is not covered by the GPL. Redistribution of the data is not permitted, and use of the data in derivative works is not permitted without the written permission of CRSP.

References

A standard corporate finance textbook: Ross, Westerfield, Jaffe and Jordan (2019). Corporate Finance, McGraw-Hill Education.

Examples

```
data.table::setDTthreads(1)
data(stocksCRSP)
names(stocksCRSP)
unique(stocksCRSP$Sector)
unique(stocksCRSP$CapGroup)
head(stocksCRSP,2)
```

---

stocksCRSPxts *Select CRSP Stocks Returns*

Description

A function to extract a subset of the stocksCRSP data.table specified by a date range and a set of tickers, with convenient defaults, and convert it to an xts object

Usage

```
stocksCRSPxts(
  data,
  dateRange = c("1993-01-31", "2015-12-31"),
  tickerSet = NULL
)
```

Arguments

data	One of the data.table objects stocksCRSP, stocksCRSPweekly, stocksCRSP-daily
dateRange	Character vector with two components a start date and an end date using format "yyyy-mm-dd". Default is the entire stocksCRSP data dates range c("1993-01-31","2015-12-32")
tickerSet	A subset of tickers of the stocks in stocksCRSP. The default is tickerSet = NULL, which results in selection of all stocks in stocksCRSP.

Value

A multivariate xts object of stock returns

Examples

```
data.table::setDTthreads(1)
library(PCRA)
library(xts)
library(data.table)
class(stocksCRSP)
args(stocksCRSPxts)
tickers4 <- c("DHR","CSL","AVP","AMWD")
dateRange <- c("2011-01-31","2015-12-31")
```

```
returns4 <- stocksCRSPxts(stocksCRSP, dateRange = dateRange,
                          tickerSet = tickers4)
class(returns4)
dim(returns4)
names(returns4)
range(index(returns4))
```

`strategies` *Hedge Fund Strategies Returns*

Description

Monthly returns of 9 hedge fund strategies from 1994 to 2004

Usage

```
data(strategies)
```

Format

A multivariate xts object

Source

Unknown

Examples

```
library(PCRA)
library(zoo)
data(strategies)
names(strategies)
dim(strategies)
range(index(strategies))
```

`tsPlotMP` *Lattice Multi-Panel Time Series Plots*

Description

Lattice multi-panel time series plot with several plotting style control parameters

Usage

```
tsPlotMP(
  ret,
  add.grid = FALSE,
  layout = NULL,
  type = "l",
  yname = "RETURNS (%)",
  Pct = FALSE,
  scaleType = "free",
  stripLeft = TRUE,
  main = NULL,
  lwd = 1,
  stripText.cex = 1,
  axis.cex = 1,
  color = "black",
  zeroLine = TRUE
)
```

Arguments

ret	A multivariate xts object
add.grid	Logical variable, if 'TRUE', type = c('l', 'g'), and if 'FALSE', type = c('l')
layout	Numeric vector of length 2 or 3 giving the number of columns, rows, and pages (optional) for a multipanel lattice display
type	Character variable type of plot: 'l' for a line, 'p' for a point, and 'b' and 'o' both denote both together, deafault 'l'
yname	Character or expression giving label(s) for the y-axis
Pct	Logical variable with default TRUE
scaleType	Character variable that controls scale of y-axis, choose from c('same', 'free')
stripLeft	Logical variable to choose the position of Lattice strip, TRUE for drawing strips at the left of each panel, FALSE for drawing strips at the top of each panel
main	A character string, or possibly an expression, for main title
lwd	The line width, a positive number, defaulting to 1
stripText.cex	Numeric factor by which strip text in the plot(s) are scaled relative to the default 1, 1.5 is 50 percent larger
axis.cex	Numeric factor by which axis in the plot(s) are scaled relative to default of 1, 1.5 is 50 larger larger, 0.5 is 50 percent smaller
color	Specification of plotting color, with default black
zeroLine	Logical variable specifying whether or not a dotted horizontal line is location at the zero vertical distance, default TRUE

Value

No value returned, instead a time series multi-panel Lattice plot

Author(s)

Kirk Li and Doug Martin

Examples

```
#Load the data
library(xts)
data("stocksCRSP")
dat = stocksCRSP
returns = tapply(dat$Return,list(dat$Date,dat$TickerLast),I)
ret = xts(returns[,1:5],as.yearmon(rownames(returns)))

#generate return time series plot
tsPlotMP(ret, color = 'Blue')
tsPlotMP(ret, scaleType = "same", zeroLine = FALSE)
tsPlotMP(ret, stripLeft = FALSE, main = 'Time Series Plot')
```

---

turnOver *Portfolio Turnover*

Description

Calculates T-1 turn-over values for a times of portfolio weight vectors from time t = 1 to time t = T, where the turnover from time t-1 to time t is the sum of the absolute differences between the portfolio weights at time t-1 and time t.

Usage

```
turnOver(weights)
```

Arguments

weights	A multivariate xts object of portfolio weights

Value

A zoo time series object containing T-1 turnover values

Examples

```
args(turnOver)
```

winsorize *Winsorize Data*

Description

This function Winsorizes a fraction gamma of a numeric data set.

Usage

```
winsorize(x, fraction = 0.1)
```

Arguments

x	A numeric data set
fraction	A fraction greater than 0 and less than 0.5

Details

The Winsorized data is obtained by by setting the gamma smallest data values equal to the next smallest value, and setting the gamma largest data values equal to the next largest data value.

Value

The Winsorized numeric data

Examples

```
x <- rt(10,8)
winsorize(x,0.2)
```

winsorMean *Winsorized Mean*

Description

Winsorized Mean

Usage

```
winsorMean(x, winFrac = 0, na.rm = FALSE, ...)
```

Arguments

x	Numeric vector
winFrac	Fraction of data to be Winsorized
na.rm	Logical variable with default FALSE
...	Pass-through parameters

Value

Numeric value of Winsorized mean

Examples

```
args(winsorMean)
```

Index