Package: PCRA (via r-universe)

July 16, 2026

Type Package

Title Companion to Portfolio Construction and Risk Analysis

Version 1.3.1

Date 2026-07-05

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)

URL https://github.com/robustport/PCRA

Encoding UTF-8

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

Suggests R.rsp, CVXR, dplyr, ellipse, facmodCS, fit.models, foreach, ggplot2, hitandrun, lubridate, Matrix, reshape2, RPEIF, RPESE, sandwich, tensr

LazyLoad true

LazyData true

LazyDataCompression xz

Copyright (c) 2022-2024

VignetteBuilder R.rsp

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.3

Config/pak/sysreqs cmake coinor-symphony coinor-libsymphony-dev libfontconfig1-dev libfreetype6-dev libfribidi-dev git make libharfbuzz-dev libgit2-dev libicu-dev libjpeg-dev libpng-dev libtiff-dev libuv1-dev libwebp-dev libxml2-dev libssl-dev libx11-dev zlib1g-dev

Repository https://robustport.r-universe.dev

Date/Publication 2026-07-15 03:28:38 UTC

RemoteUrl https://github.com/robustport/pcra

RemoteRef HEAD

RemoteSha dea6a6f59f1af989b38157c8bb0e6e11c2081770

Contents

abbreviate_name *Abbreviate a vector of names*

Description

Abbreviate a vector of names

Usage

```
abbreviate_name(name, max_word_length = Inf, collapse = "")
```

Arguments

name	A character vector of long character strings.
max_word_length	Maximum word length before truncation.
collapse	A string used to join the abbreviation letters.

Value

A character vector of abbreviated names.

Examples

```
abbreviate_name(c("Large Cap Growth", "Small Value"), max_word_length = 3)
```

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)
```

---

buildPortfolios *Build a List of Portfolio Specifications (Default Example)*

Description

This function serves as the default example for the `buildPortfolios` argument in `runMultipleBacktests`. It demonstrates how to construct a list of portfolio specifications using `portfolio.spec`, `add.constraint`, and `add.objective` from the `PortfolioAnalytics` package.

Users are encouraged to write their own portfolio list following the same structure of this function: one input (`selected_stocks`), one output (a named list of `portfolio.spec` objects), and pass it to `runMultipleBacktests()` via the `buildPortfolios` argument.

Usage

```
buildPortfolios(selected_stocks)
```

Arguments

selected_stocks	Character vector. Tickers of the assets to include in the portfolio.

Value

A list of `portfolio.spec` objects, one per strategy. It is strongly recommended to name each element of the list, as the names are used as labels across all outputs.

See Also

`runMultipleBacktests`, `portfolio.spec`, `add.constraint`, `add.objective`

Examples

```
body(buildPortfolios)
```

BXMdata *CBOE S&P 500 BuyWrite Index*

Description

Monthly data for the S&P500 index, the S&P 500 BuyWrite Index, and the risk-free rate

Usage

```
data(BXMdata)
```

Format

A data.frame object

Source

TO BE ADDED.

Examples

```
library(PCRA)
names(BXMdata)
```

CboeOptionStrategies *CboeOptionStrategies*

Description

Monthly time series of total return indices from June 1986 to December 2021 for options based strategies created and maintained by Cboe Livevol, LLC along with data for the S&P 500®, as well as levels of the VIX and VXO volatility measures, and the 3-month T-Bill rate (GS3M) from the Federal Reserve Bank of St. Louis' FRED database.

Usage

```
data(CboeOptionStrategies)
```

Format

A data frame with monthly time series of ten total return indices for options based strategies created and maintained by Cboe Livevol, LLC along with total return and price return indices for the S&P 500®, as well as the levels of the VIX and VXO volatility measures, and the 3-month T-Bill rate (GS3M) from the Federal Reserve Bank of St. Louis' FRED database. Links are provided to the relevant websites for each of the series. Many, but not all, of the total return series start with a value of 100.

- **Date:** type 'Date'. Last Day of Month. Many, but not all, of the time series have data from June 1986 to December 2021.
- **BXM:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 BuyWrite Index. Details of its construction can be found at https://www.cboe.com/us/indices/dashboard/BXM
- **BXMD:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 30-Delta BuyWrite Index series. Details of its construction can be found at https://www.cboe.com/us/indices/dashboard/BXMD/.
- **BXY:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 2 construction can be found at https://www.cboe.com/us/indices/dashboard/BXY/.
- **PUT:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 PutWrite Index. Details of its construction can be found at https://www.cboe.com/us/indices/dashboard/PUT/.
- **CLL:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 95-110 Collar Index. Details of its construction can be found at https://www.cboe.com/us/indices/dashboard/C
- **BFLY:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 Iron Butterfly Index. Details of its construction can be found at https://www.cboe.com/us/indices/dashboard/B
- **CLLZ:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 Zero-Cost Put Spread Collar. Details of its construction can be found at https://www.cboe.com/us/indices/dashboard/CLLZ/.
- **CMBO:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 Covered Combo Index. Details of its construction can be found at https://www.cboe.com/us/indices/dash
- **CNDR:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 Iron Condor Index. Details of its construction can be found at https://www.cboe.com/us/indices/dashboard/CN

- **PPUT:** type 'num'. Month-end closing level for the cumulative total return index for the Cboe S&P 500 5 construction can be found at https://www.cboe.com/us/indices/dashboard/PPUT/.
- **SPTR:** type 'num'. Month-end closing level for the cumulative total return index for the S&P 500® Index (SPX℠). Details of its construction can be found at https://www.spglobal.com/spdji/en/indices/equity/sp-500/.
- **SPX:** type 'num'. Month-end closing level for the cumulative price return index for the S&P 500® Index (SPX℠). Details of its construction can be found at https://www.spglobal.com/spdji/en/indices/equity/sp-500/.
- **VIX:** type 'num'. Month-end closing level for the Cboe Volatilty Index®, a measure of constant, 30-day expected volatility of the U.S. stock market derived from real-time, mid-quote prices of S&P 500® Index (SPX℠). Details of its construction can be found at https://www.cboe.com/tradable_products/
- **VXO:** type 'num'. Month-end closing level for the Cboe S&P 100 Volatility Index. The index was discontinued on 9/23/2021. Historical daily data can be downloaded from https://fred.stlouisfed.org/series/VXOCL
- **GS3M:** type 'num'. Average daily closing 3 month constant maturity T-Bill yield, averaged over all business days in a month. Details of its construction can be found at https://fred.stlouisfed.org/series/GS3M.

Details

This data set provides monthly time series of ten total return indices for options based strategies created and maintained by Cboe Livevol, LLC, along with total return and price return indices for the S&P 500®, as well as the levels of the VIX and VXO volatility measures, and the 3-month T-Bill rate (GS3M) from the Federal Reserve Bank of St. Louis' FRED database. Links are provided to the relevant websites for each of the series. Many, but not all, of the total return series start with a value of 100, so that their total return in any given month is the ratio of the value for that month to the value for the prior month -1.

Source

GS3M is obtained from the Federal Reserve Bank of St. Louis' FRED database at https://fred.stlouisfed.org/series/GS3M

References

Chapter 12 (Performance Analysis) of Martin, Philips, Scherer, Stoyanov and Li, Portfolio Construction and Risk Analysis, Springer, 2024.

Examples

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

---

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 Effectively*

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)
```

ConferenceBoardETI *ConferenceBoardETI*

Description

Monthly time series of the Conference Board Employment Trends Index from November 1973 to December 2023.

Usage

```
data(ConferenceBoardETI)
```

Format

A data frame with a time series of month-end observations of the Conference Board Employment Trends Index (ETI) from November 1973 to December 2023.

- **Date:** type 'Date'. Last Day of Month. The time series extends from November 1973 to December 2023.
- **ETI:** type 'num'. Level of the Conference Board Employment Trends Index on the last day of the month.

Details

This data set provides a monthly time series of the Conference Board Employment Trends Index (ETI) from November 1973 to December 2023. Details of the index's construction as well as its most recent value can be found at https://www.conference-board.org/data/eti.cfm. Technical details related to the construction of the index are available at https://www.conference-board.org/pdf_free/press/Employment while details of its revision in November 2020 can be found at https://www.conference-board.org/pdf_free/press/Employment The change was made following the COVID-19 driven recession in 2020. The Conference Board

ETI combines 8 measures of employment to create a composite index of employment trends. 1. Percentage of respondents who say they find "Jobs Hard to Get" (Source: The Conference Board Consumer Confidence Survey) 2. Initial Claims for Unemployment Insurance, State Programs (Source: U.S. Department of Labor). 3. Percentage of Firms With Positions Not Able to Fill Right Now (Source: © National Federation of Independent Business Research Foundation) 4. Number of Employees Hired by the Temporary-Help Industry (Source: U.S. Bureau of Labor Statistics) 5. Part-time Workers for Economic Reasons (Bureau of Labor Statistics) 6. Job Openings (Bureau of Labor Statistics, through Job Openings and Labor Turnover Survey): 7. Industrial Production (Federal Reserve Board) 8. Real Manufacturing and Trade Sales (U.S. Bureau of Economic Analysis).

Source

The Conference Board. The Conference Board ETI Data is provided by the Conference Board for educational purposes only. The Conference Board makes no representation or warranty of any kind, express or implied, including regarding the accuracy, adequacy, validity, reliability, availability or completeness of The Conference Board ETI Data. All rights reserved. Redistribution of the data is not permitted, and use use of the data in derivative works is not permitted without the written permission of the Conference Board.

References

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

Examples

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

---

crsp.returns8 *CRSP Returns for 8 stocks in 4 cap groups*

Description

Monthly returns of 8 stocks with tickers GHI, PBCI, MODI, MGJ, MAT, EMN, AMAT, AMGN, two in each of for capitalization groups 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))
```

`CRSPLiquidMktCapGrpsCnts`

CRSP Stocks Cap Groups Counts

Description

Biennial counts of bigcap, smallcap, microcap stocks among liquid CRSP database stocks from 1964 to 2018 using weekly returns. Bigcap stocks consist of midcap, largecap, and megacap stocks. For each contiguous two-year interval, liquid stocks are those with no missing returns and at most 4 returns with value 0.

Usage

```
data(CRSPLiquidMktCapGrpsCnts)
```

Format

A multivariate xts object

Source

The microcap, smallcap and bigcap groups were defined using the using the 20th and 50th percentiles of the NYSE capitalization data as breakpoints to separate these three market cap groups.

Examples

```
data(CRSPLiquidMktCapGrpsCnts)
names(CRSPLiquidMktCapGrpsCnts)
dim(CRSPLiquidMktCapGrpsCnts)
```

datFF3W *Fama-French 3-Factor Model Weekly Time Series*

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 4-Factor Model Weekly Time Series*

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)
```

---

ewmaMeanVol *EWMA Mean and Volatility*

Description

Computes EWMA joint mean and vol, with options for robust versus classic EWMA estimates.

Usage

```
ewmaMeanVol(
  x,
  nstart = 10,
  robMean = T,
  robVol = T,
  cc = 2.5,
  lambdaMean = 0.9,
  lambdaVol = 0.9
)
```

Arguments

x	An xts returns object
nstart	Integer number of returns for initial ewma estimates
robMean	Logical variable, if TRUE compute robust ewmaMean, if FALSE compute classic classic ewmaMean. Default is TRUE
robVol	Logical variable, if TRUE compute robust ewmaVol, if FALSE compute classic ewmaVol. Default is TRUE.
cc	Numeric value, robustness tuning constant. Default is 2.5
lambdaMean	Numeric value, decay rate constant for ewmaMean
lambdaVol	Numeric value, decay rate constant for ewmaVol

Value

A bivarate xts object containing the ewmaMean and ewmaVol

Examples

```
args(ewmaMeanVol)
```

---

factorsSPGMI *SPGMI Data 14 Factors for 294 Stocks*

Description

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

Usage

```
data(factorsSPGMI)
```

Format

A data.frame containing 14 SPGMI numeric factor exposures (alpha factors) for each of 294 stocks from 1993 to 2015 (276 months), among a total of 22 variables, that also include 4 categorical factor exposures.

- **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/spdji/en/landing/investment-themes/gics/, 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/

`factorsSPGMIr` *Rounded Version of SPGMI Data 14 Factors*

Description

Rounded version of factorsSPGMI to 4 decimal places.

Usage

```
data(factorsSPGMIr)
```

Format

A data.frame containing 14 SPGMI numeric factor exposures (alpha factors) for each of 294 stocks from 1993 to 2015 (276 months), among a total of 22 variables, that also include 4 categorical factor exposures.

- **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)

For details concerning the very small effects of rounding, see the Vignette "PCRA Package Overview.

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/spdji/en/landing/investment-themes/gics/, 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". If the download fails, e.g., because the internet resource is temporarily unavailable, the function returns NULL invisibly, with an informative message.

Examples

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

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

---

gfunds5 *Five German Investment Funds*

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))
```

HFstrategies *Hedge Fund Strategies Returns*

Description

Monthly returns of 9 hedge fund strategies from 1994 to 2004

Usage

```
data(HFstrategies)
```

Format

A multivariate xts object

Source

Unknown

Examples

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

invensysEPS *Invensys Earnings per Share*

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.

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)
```

`MarketData` *Global Baskets of Equity and Bonds*

Description

Daily price data for a global equity basket, a global bond basket, and cash

Usage

```
data(MarketData)
```

Format

An xts object

Source

TO BE ADDED.

Examples

```
library(PCRA)
range(range(MarketData$global.equity.basket))
```

`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)
```

---

minVarCashRisky *Minimum Variance Portfolio*

Description

Given a time series of asset returns and risk-free T-Bill returns, a target mean return, and a specification of whether the asset weights are constrained to be long-only, this function compputes the weights of a fully-invested minimum variance portfolio

Usage

```
minVarCashRisky(returns, mu0, LO = FALSE, bnd = 1000, bndRF = 100)
```

Arguments

returns	xts multivariate returns object that contains the returns of the risk-free T-Bill in the last column
mu0	Minimum variance portfolio mean return
LO	Logical variable with default FALSE
bnd	Bound on asset weights
bndRF	a bound on the risk-free T-Bill weight

Value

A list of the minimum variance portfolio numeric weights, mean value, volatility, and Sharpe Ratio.

Examples

```
args(minVarCashRisky)
```

minVarRiskyLO *Title Minimum Variance Long-Only Risky Assets Portfolio*

Description

Given a time series of risky asset returns and a target mean return, this function computes the mean and standard deviation of a fully-invested long-only minimum variance portfolio

Usage

```
minVarRiskyLO(returns, mu)
```

Arguments

returns	xts multivariate risky asset returns
mu	Portfolio mean return specification

Details

This function uses the PortfolioAnalytics function optimize.portfolio.R and the PCRA function opt.outputMvoPCRA. For details, see the man pages for those function.

Value

A list containing the weights, mean value, standard deviation and Sharpe ratio, with default names Wgts, Mean, StdDeve, SR

Author(s)

R. Douglas Martin

Examples

```
args(minVarRiskyLO)
```

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)
```

plotLSandHuberRobustSFM *Plot LS and Huber SFM Fits*

Description

Computes LS and Huber robust single factor model fits with standard errors and plots the results

Usage

```
plotLSandHuberRobustSFM(
  x,
  mainText = NULL,
  ylimits = NULL,
  legendPos = "topleft",
  goodOutlier = FALSE,
  makePct = FALSE
)
```

Arguments

x	xts time series vector
mainText	Character variable with NULL default
ylimits	Numeric vector of vertical axis limits with NULL default
legendPos	Character variable with default "topleft"
goodOutlier	Logical variable with default FALSE
makePct	Logical variable with default FALSE

Value

A plot of the LS and robust Huber SFM fits

Examples

```
args(plotLSandHuberRobustSFM)
```

`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)
```

psiHuber *Huber psi function*

Description

The Huber psi function psi(x) is the derivative of the Huber rho function, with a tuning constant cc that controls the trade-off between robustness toward outliers, and normal distribution estimator efficiency.

Usage

```
psiHuber(x, cc = 2)
```

Arguments

x	Numeric argument of psi function
cc	numeric robustness tuning constant

Details

The choice cc = 1.345 results in a 95 distribution efficiency for the Huber location M-estimator. The default value cc = 2.0 is better for EWMA robust filters.

Value

Numeric value of psi(x)

Examples

```
psiHuber(0.5)
```

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 *CRSP Returns of 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 *CRSP Returns of 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 *CRSP Returns of 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 *CRSP Returns of 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 *CRSP Returns of 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 *CRSP Returns of 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 *CRSP Returns of 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 *CRSP Returns of 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 *CRSP Returns of 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))
```

runMultipleBacktests *Run Multiple Portfolio Backtests and Plot*

Description

Runs `n_simulations` independent portfolio backtests, each time drawing a random subset of stocks from `stock_list`. For every simulation a full backtest is executed via `runPortfolioBacktest()`, and an individual plot can optionally be saved. After all simulations finish, the function averages the cumulative return series across simulations and can save a summary plot of those averaged returns.

Usage

```
runMultipleBacktests(
  n_simulations,
  portfolio_size,
  seed = NULL,
  return_portfolio,
  stock_list,
  buildPortfolios = buildPortfolios,
  market_return = NULL,
  rebalance_on = NULL,
  rolling_window = NULL,
  optimize_method = "CVXR",
  moment_list = NULL,
  save_plot = TRUE,
  plot_path = "./",
  plot_name = "backtest",
  plot_main = NULL,
  plotType = "both",
  save_avg_plot = TRUE,
  avg_plot_path = "./",
  avg_plot_name = "avg_backtest",
  avg_plot_main = NULL,
  avgPlotType = "both",
  colorSet = NULL,
  ltySet = NULL,
  lwdSet = NULL
)
```

Arguments

n_simulations	Integer. Number of independent backtest simulations to run.
portfolio_size	Integer. Number of stocks to randomly sample from stock_list for each simulation.
seed	Integer. Random seed passed to set.seed() before the simulation loop for reproducibility.
return_portfolio	An xts matrix of asset returns. Column names must match the assets in stock_list.
stock_list	Character vector. Universe of stock tickers from which portfolio_size tickers are drawn at random in each simulation.
buildPortfolios	A function that accepts a character vector of selected stock tickers and returns a named list of portfolio.spec objects, one per strategy. The names of the list elements are used as strategy labels in plot legends and all other outputs. Typically built with buildPortfolios and can be customized.
market_return	An xts single-column object of benchmark returns.
rebalance_on	Character string passed to optimize.portfolio.rebalancing(). See endpoints for valid names.

`rolling_window` Positive integer. Length of the rolling estimation window in periods.

`optimize_method`
Character string specifying the solver. Default `"CVXR"`.

`moment_list` If different moment functions are passed into multiple GMV portfolios, please define each moment function via this parameter. For the portfolio that do not require moment function, please pass NULL. Example: `list('custom.covRob.Rocke', NULL, NULL)`.

`save_plot` Logical. Whether to save the plot to a PNG file. Default `TRUE`.

`plot_path` Character string. Full file path for the each simulation output. Required when `save_plot = TRUE`.

`plot_name` Plot name for each simulation output. Default `"backtest"`

`plot_main` Plot title for each simulation PNG output.

`plotType` "cumRet", "drawdown", or the default is "both"

`save_avg_plot` Logical. Whether to save the the average cumulative returns plot to a PNG file. Default `TRUE`.

`avg_plot_path` Character string. Full file path for the average simulation output. Required when `save_avg_plot = TRUE`.

`avg_plot_name` Plot name for average simulation output. Default `"avg_backtest"`.

`avg_plot_main` Plot title for the average simulation PNG output.

`avgPlotType` "cumRet", "drawdown", or the default is "both"

`colorSet` Optional character vector of colors passed to `backtest.plot()`.

`ltySet` Optional integer vector of line types passed to `backtest.plot()`.

`lwdSet` Optional integer vector of line width passed to `backtest.plot()`.

Value

A named list with four elements:

`results` A list of length `n_simulations`. Each element contains the `returns` component returned by `runPortfolioBacktest()` for that simulation.

`selected_stocks_all` A list of length `n_simulations`. Each element is a character vector of the tickers chosen for that simulation.

`avg_algorithm_returns` An `xts` object containing the element-wise average of all simulations' algorithm return series.

`avg_cumulative_return` An `xts` object containing the element-wise average of all simulations' cumulative return series.

Examples

```
args(runMultipleBacktests)
```

`runPortfolioBacktest` *Run Portfolio Backtest and Plot*

Description

Perform a backtest for a list of portfolio specifications. Portfolio objectives, constraints, rolling window, and rebalancing frequency can be customized using the same conventions as used in `PortfolioAnalytics`.

Usage

```
runPortfolioBacktest(
  return_portfolio,
  portfolio_list,
  portfolio_names,
  market_return = NULL,
  rebalance_on = NULL,
  rolling_window = NULL,
  optimize_method = "CVXR",
  moment_list = NULL,
  save_plot = TRUE,
  plot_path = "./",
  plot_name = "backtest",
  plot_main = NULL,
  plotType = "both",
  colorSet = NULL,
  ltySet = NULL,
  lwdSet = NULL
)
```

Arguments

`return_portfolio`	An `xts` matrix of asset returns. Column names must match the assets in each `portfolio.spec` object.
`portfolio_list`	A list of `portfolio.spec` objects built with `PortfolioAnalytics`.
`portfolio_names`	Character vector of names corresponding to each portfolio in `portfolio_list`.
`market_return`	An `xts` single-column object of benchmark returns.
`rebalance_on`	Character string passed to `optimize.portfolio.rebalancing()`. See `endpoints` for valid names.
`rolling_window`	Positive integer. Length of the rolling estimation window in periods.
`optimize_method`	Character string specifying the solver. Default `"CVXR"`.

moment_list	If different moment functions are passed into multiple GMV portfolios, please define each moment function via this parameter. For the portfolio that do not require moment function, please pass NULL. Example: `list('custom.covRob.Rocke', NULL, NULL)`.
save_plot	Logical. Whether to save the plot to a PNG file. Default `TRUE`.
plot_path	Character string. Full file path for the PNG output. Required when `save_plot = TRUE`.
plot_name	Plot name for the PNG output. Default `"backtest"`.
plot_main	Plot title for the PNG output.
plotType	"cumRet", "drawdown", or the default is "both"
colorSet	Optional character vector of colors passed to `backtest.plot()`.
ltySet	Optional integer vector of line types passed to `backtest.plot()`.
lwdSet	Optional integer vector of line width passed to `backtest.plot()`.

Value

A list:

`returns` An `xts` matrix of period returns with one column per portfolio plus a `"Market"` column.

`cumRet` An `xts` matrix of cumulative returns.

`plot` The plot object returned by `backtest.plot()`.

Examples

```
args(runPortfolioBacktest)
```

`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(stocksCRSPmonthly)
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(stocksCRSPmonthly$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)
```

ShortDurationCredit *ShortDurationCredit*

Description

Monthly time series of total return, Weighted Average Maturity, Modified Duration, Yield to Maturity and Option Adjusted Spread from January 1988 to December 2021 for two short duration indices maintained by ICE Data Indices, LLC 1. ICE BofA 1-3 Year AAA-A US Corporate Index (Index C110), and 2. ICE BofA 1-3 Year US Treasury Index (Index G1O2) Returns are total returns (coupon income + price return) for the month, while characteristics are reported as of month-end.

Usage

```
data(ShortDurationCredit)
```

Format

A data frame with monthly time series of returns, Weighted Average Maturity (WAM), Modified Duration (Dmod), Yield to Maturity (YTM) and Option Adjusted Spread (OAS) from January 1988 to December 2021 for two short duration indices maintained by ICE Data Indices, LLC 1. ICE BofA 1-3 Year AAA-A US Corporate Index (Index C110), and 2. ICE BofA 1-3 Year US Treasury Index (Index G1O2) Returns are total returns (i.e. coupon income + price change divided by initial price) for the month, and are expressed in percentage points (e.g. 2.05 corresponds to a total return of 2.05 WAM and Dmod are expressed in years and are reported as of month end. YTM, like total return, is reported in percentage points (e.g. 5.05 corresponds to a Yield to Maturity of 5.05 in basis points or 1/100th of 1 Spread of 8 basis points or 0.08 as of month-end. Returns are reported in every month, but not all characteristics are reported at the end of every month.

- **Date:** type 'Date'. Last Day of Month. Many, but not all, of the time series have data from January 1988 to December 2021.
- **C110_Return:** type 'num'. Total return including coupon income and change in price for the month of the ICE BofA 1-3 Year AAA-A US Corporate Index (Index C110). Return is expressed in percentage points (e.g. 2.05 corresponds to a total return of 2.05
- **C110_WAM:** type 'num'. Month-end Weighted Average Maturity (WAM) of the ICE BofA 1-3 Year AAA-A US Corporate Index (Index C110). WAM is expressed in years.
- **C110_Dmod:** type 'num'. Month-end Modified Duration (Dmod) of the ICE BofA 1-3 Year AAA-A US Corporate Index (Index C110). Dmod is expressed in years.
- **C110_YTM:** type 'num'. Month-end Yield to Maturity (YTM) of the ICE BofA 1-3 Year AAA-A US Corporate Index (Index C110). YTM is reported in percentage points (e.g. 5.05 corresponds to a Yield to Maturity of 5.05
- **C110_OAS:** type 'num'. Month-end Option Adjusted Spread (OAS) of the ICE BofA 1-3 Year AAA-A US Corporate Index (Index C110). OAS is expressed in basis points or 1/100th of 1 Adjusted Spread of 8 basis points or 0.08
- **G1O2_Return:** type 'num'. Total return including coupon income and change in price for the month of the ICE BofA 1-3 Year US Treasury Index (Index G1O2). Return is expressed in percentage points (e.g. 2.05 corresponds to a total return of 2.05
- **G1O2_WAM:** type 'num'. Month-end Weighted Average Maturity (WAM) of the ICE BofA 1-3 Year US Treasury Index (Index G1O2). WAM is expressed in years.
- **G1O2_Dmod:** type 'num'. Month-end Modified Duration (Dmod) of the ICE BofA 1-3 Year US Treasury Index (Index G1O2). Dmod is expressed in years.
- **G1O2_YTM:** type 'num'. Month-end Yield to Maturity (YTM) of the ICE BofA 1-3 Year US Treasury Index (Index G1O2). YTM is reported in percentage points (e.g. 5.05 corresponds to a Yield to Maturity of 5.05
- **G1O2_OAS:** type 'num'. Month-end Option Adjusted Spread (OAS) of the ICE BofA 1-3 Year US Treasury Index (Index G1O2). OAS is expressed in basis points or 1/100th of 1 Adjusted Spread of 8 basis points or 0.08

Details

This data set provides monthly time series of total return, Weighted Average Maturity, Modified Duration, Yield to Maturity and Option Adjusted Spread from January 1988 to December 2021 for two short duration indices maintained by ICE Data Indices, LLC 1. ICE BofA 1-3 Year AAA-A US Corporate Index (Index C110), and 2. ICE BofA 1-3 Year US Treasury Index (Index G1O2) Returns are total returns (coupon income + price return) for the month, while characteristics are reported as of month-end.

Source

ICE DATA INDICES, LLC. ICE DATA IS PROVIDED BY ICE DATA INDICES, LLC ("ICE DATA") FOR EDUCATIONAL PURPOSES. ICE® IS A REGISTERED TRADEMARK OF ICE DATA OR ITS AFFILIATES. ICE DATA,ITS AFFILIATES AND THEIR RESPECTIVE THIRD-PARTY SUPPLIERS DISCLAIM ANY AND ALL WARRANTIES AND REPRESENTATIONS, EXPRESS AND/OR IMPLIED, INCLUDING ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE OR USE, INCLUDING THE INDICES, INDEX DATA AND ANY DATA INCLUDED IN, RELATED TO, OR DERIVED THEREFROM. NEITHER ICE DATA, ITS AFFILIATES NOR THEIR RESPECTIVE THIRD.PARTY SUPPLIERS SHALL BE SUBJECT TO ANY DAMAGES OR LIABILITY WITH RESPECT TO THE ADEQUACY, ACCURACY, TIMELINESS OR COMPLETENESS OF THE INDICES OR THE INDEX DATA OR ANY COMPONENT THEREOF, AND THE INDICES AND INDEX DATA AND ALL COMPONENTS THEREOF ARE PROVIDED ON AN "AS IS" BASIS AND YOUR USE IS AT YOUR OWN RISK. ICE DATA, ITS AFFILIATES AND THEIR RESPECTIVE THIRD-PARTY SUPPLIERS DO NOT SPONSOR, ENDORSE, OR RECOMMEND SPRINGER, PCRA, OR ANY OF ITS PRODUCTS OR SERVICES. 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 ICE DATA.

References

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

Examples

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

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

- **Date:** type 'Date'. End of year date formatted as YYYY-12-31. Useful when creating a time series object for exploratory time series plots. Convert the data frame to an xts object using xts::as.xts(SP500).
- **Year:** type 'num'. Year corresponding to Date.
- **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.

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

- **Date:** type 'Date'. End of year date formatted as YYYY-12-31. Useful when creating a time series object for exploratory time series plots. Convert the data frame to an xts object using xts::as.xts(SP500).
- **Year:** type 'num'. Year corresponding to Date.
- **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.

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:

- **Date:** type 'Date'. End of year date formatted as YYYY-12-31. Useful when creating a time series object for exploratory time series plots. Convert the data frame to an xts object using xts::as.xts(SP500).
- **Year:** type 'num'. Year corresponding to Date.
- **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. S&P's definition of Operating Earnings is different from the industry standard of Earnings Before Interest and Taxes (EBIT). It is closer to As-Reported Earnings + non-recurring items, but with judgement about what is added back.
- **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 12 (Expected Returns) of Martin, Philips, Scherer, Stoyanov and Li, Portfolio Construction and Risk Analysis, Springer, 2026.

Examples

```
data(SP500)
names(SP500)
```

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

SP500data *SP500data*

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(SP500data)
```

Format

A data.frame with observations on the S&P500 from 1925 to the most recent year end for which final data is available:

- **Year:** type 'num'.
- **SP500Price:** 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.
- **SP500Revenue:** type 'num'. Annual Revenues per share for the S&P 500 during the calendar year.
- **SP500DPS:** type 'num'. Annual 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.
- **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.
- **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

CPIAUCNS is obtained from the Federal Reserve's FRED database at https://fred.stlouisfed.org/series/CPIAUCNS
GDPA is obtained from the Federal Reserve's FRED database at https://fred.stlouisfed.org/series/CPIAUCNS
Data for the S&P 500 is updated using the QUARTERLY DATA tab of https://www.spglobal.com/spdji/en/documents/additio
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® 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. © 2022 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(SP500data)
names(SP500data)
head(SP500data, 5)
tail(SP500data, 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

- **Date:** type 'Date'. End of year date formatted as YYYY-12-31. Useful when creating a time series object for exploratory time series plots. Convert the data frame to an xts object using xts::as.xts(SP500).
- **Year:** type 'num'. Year corresponding to Date.
- **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

- **Date:** type 'Date'. End of year date formatted as YYYY-12-31. Useful when creating a time series object for exploratory time series plots. Convert the data frame to an xts object using xts::as.xts(SP500).
- **Year:** type 'num'. Year corresponding to Date.
- **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)
```

---

stocksCRSPdaily *CRSP daily stocks data for 294 stocks*

Description

CRSP daily stocks data for 294 stocks for 1993 to 2015. This dataset is not shipped with the package due to CRAN package size limits. Download it with `stocksCRSPdaily <-getPCRAData(dataset = "stocksCRSPdaily")`.

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

Source

Unknown

Examples

```
stocksCRSPdaily <- getPCRAData(dataset = "stocksCRSPdaily")
```

---

stocksCRSPmonthly *stocksCRSPmonthly*

Description

CRSP monthly stocks data for 294 stocks 1993 to 2015

Usage

```
data(stocksCRSPmonthly)
```

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(stocksCRSPmonthly)
names(stocksCRSPmonthly)
unique(stocksCRSPmonthly$Sector)
unique(stocksCRSPmonthly$CapGroup)
head(stocksCRSPmonthly,2)
```

---

stocksCRSPweekly *CRSP weekly stocks data for 294 stocks*

Description

CRSP Weekly stocks data for 294 stocks for 1993 to 2015. This dataset is not shipped with the package due to CRAN package size limits. Download it with `stocksCRSPweekly <-getPCRAData(dataset = "stocksCRSPweekly")`.

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

Source

Unknown

Examples

```
stocksCRSPweekly <- getPCRAData(dataset = "stocksCRSPweekly")
```

`stocksCRSPxts` *Select CRSP Stocks Returns*

Description

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

Value

A multivariate xts object of stock returns

Examples

```
data.table::setDTthreads(1)
library(PCRA)
library(xts)
library(data.table)
class(stocksCRSPmonthly)
args(stocksCRSPxts)
tickers4 <- c("DHR","CSL","AVP","AMWD")
dateRange <- c("2011-01-31","2015-12-31")
returns4 <- stocksCRSPxts(stocksCRSPmonthly, dateRange = dateRange,
                          tickerSet = tickers4)
```

```
class(returns4)
dim(returns4)
names(returns4)
range(index(returns4))
```

`to_monthly` *Function to convert from daily to weekly returns.*

Description

to_monthly will convert daily returns to monthly returns.

Usage

```
to_monthly(daily, index_last = TRUE)
```

Arguments

`daily`	An xts object of daily returns.
`index_last`	Controls whether the return date label will fall on the month end date provided in the dataset (usually the final trading date of the month in most financial data sets) or the calendar month end date (potentially a non-trading date). Regardless, the returns are identical under two scenarios, but the date may differ for the "month end".

Details

These will be added

Value

monthly

Examples

```
args(to_monthly)
```

to_weekly *Function to convert from daily to weekly returns.*

Description

to_weekly will convert daily returns to weekly returns, while allowing the user flexibility in the choice of which day to end the week.

Usage

```
to_weekly(
  daily,
  days_in_week = 5,
  week_ending_day_str = c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday",
    "Saturday", "Sunday")
)
```

Arguments

daily	An xts object of daily returns.
days_in_week	Default 5.
week_ending_day_str	controls what is the week end day. If the week_ending_day is "Wednesday", then Wednesday is the end of a week. It means the return from previous Thursday to current Wednesday will be included in current Wednesday week's cumulative return. If the beginning of the dataset is before week_ending_day (in this case, Wednesday) , the first week's return would be from the begining of the dataset to the first week_ending day. If the last week is short of the full week,i.e.if week_ending_day is "Friday" and 2015-12-31 is a Thursday, then the last four day return will be labeled as 2016-01-01.

Details

These will be added.

Value

returns

Examples

```
args(to_weekly)
```

transferCoef *Transfer Coefficent*

Description

Computes the transfer coefficient (TF), which measures the reduction in mean excess return of an MV portfolio with weights constraint relative to the mean excess return of an unconstrained MV portfolio. This description also holds with portfolios mean returns replaced by Sharpe ratios.

Usage

```
transferCoef(returns, wtVec)
```

Arguments

returns	An xts multivariate returns object that contains the returns of the risk-free T-Bill in the last column
wtVec	The weight vector of a constrained MV portfolio

Value

Numeric value of the TC

Examples

```
args(transferCoef)
```

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("stocksCRSPmonthly")
dat = stocksCRSPmonthly
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)
```

`update_dev_pkg` *Update to Developer version on Github that have access to additional functions and data*

Description

Update to Developer version on Github that have access to additional functions and data

Usage

```
update_dev_pkg(
  pkg = "PCRA",
  repo = "https://github.com/robustport/PCRA",
  field = "Version",
  lib = NULL
)
```

Arguments

pkg	Default to "PCRA" package name.
repo	Default to "https://github.com/robustport/PCRA"
field	Default to "Version" field under Package Description file.
lib	library path where the package would be installed.

USTreasuryTradeweb *USTreasuryTradeweb*

Description

A data frame with end-of-day quotes from Tradeweb Markets LLC of bid and ask prices and yields on the On-the-Run and First-Off-the-Run US Treasury Notes and Bonds at 4 maturities: 2 Years, 5 Years, 10 Years and 30 Years.

Usage

```
data(USTreasuryTradeweb)
```

Format

A data frame with end-of-day quotes from Tradeweb Markets LLC of bid and ask prices and yields on the On-the-Run and First-Off-the-Run US Treasury Notes and Bonds at 4 maturities: 2 Years, 5 Years, 10 Years and 30 Years. Prices are reported to the nearest 1/256 (0.00390625), while yields are reported to 3 decimal places. Yields are percentages, i.e. a yield of 2.125 should be interpreted as 2.125

- **Date:** type ‘Date‘. Trade Date. Not all bonds have data on all days.
- **US2Y_OTR_BidPrice:** type ‘num‘. Closing Bid Price for the 2 Year On-the-Run bond.
- **US2Y_OTR_AskPrice:** type ‘num‘. Closing Ask Price for the 2 Year On-the-Run bond.
- **US2Y_OTR_BidYield:** type ‘num‘. Closing Bid Yield to Maturity for the 2 Year On-the-Run bond.
- **US2Y_OTR_AskYield:** type ‘num‘. Closing Ask Yield to Maturity for the 2 Year On-the-Run bond.
- **US2Y_OFTR_1_BidPrice:** type ‘num‘. Closing Bid Price for the 2 Year First-Off-the-Run bond.
- **US2Y_OFTR_1_AskPrice:** type ‘num‘. Closing Ask Price for the 2 Year First-Off-the-Run bond.
- **US2Y_OFTR_1_BidYield:** type ‘num‘. Closing Bid Yield to Maturity for the 2 Year First-Off-the-Run bond.
- **US2Y_OFTR_1_AskYield:** type ‘num‘. Closing Ask Yield to Maturity for the 2 Year First-Off-the-Run bond.
- **US5Y_OTR_BidPrice:** type ‘num‘. Closing Bid Price for the 5 Year On-the-Run bond.

- **US5Y_OTR_AskPrice:** type ‘num‘. Closing Ask Price for the 5 Year On-the-Run bond.
- **US5Y_OTR_BidYield:** type ‘num‘. Closing Bid Yield to Maturity for the 5 Year On-the-Run bond.
- **US5Y_OTR_AskYield:** type ‘num‘. Closing Ask Yield to Maturity for the 5 Year On-the-Run bond.
- **US5Y_OFTR_1_BidPrice:** type ‘num‘. Closing Bid Price for the 5 Year First-Off-the-Run bond.
- **US5Y_OFTR_1_AskPrice:** type ‘num‘. Closing Ask Price for the 5 Year First-Off-the-Run bond.
- **US5Y_OFTR_1_BidYield:** type ‘num‘. Closing Bid Yield to Maturity for the 5 Year First-Off-the-Run bond.
- **US5Y_OFTR_1_AskYield:** type ‘num‘. Closing Ask Yield to Maturity for the 5 Year First-Off-the-Run bond.
- **US10Y_OTR_BidPrice:** type ‘num‘. Closing Bid Price for the 10 Year On-the-Run bond.
- **US10Y_OTR_AskPrice:** type ‘num‘. Closing Ask Price for the 10 Year On-the-Run bond.
- **US10Y_OTR_BidYield:** type ‘num‘. Closing Bid Yield to Maturity for the 10 Year On-the-Run bond.
- **US10Y_OTR_AskYield:** type ‘num‘. Closing Ask Yield to Maturity for the 10 Year On-the-Run bond.
- **US10Y_OFTR_1_BidPrice:** type ‘num‘. Closing Bid Price for the 10 Year First-Off-the-Run bond.
- **US10Y_OFTR_1_AskPrice:** type ‘num‘. Closing Ask Price for the 10 Year First-Off-the-Run bond.
- **US10Y_OFTR_1_BidYield:** type ‘num‘. Closing Bid Yield to Maturity for the 10 Year First-Off-the-Run bond.
- **US10Y_OFTR_1_AskYield:** type ‘num‘. Closing Ask Yield to Maturity for the 10 Year First-Off-the-Run bond.
- **US30Y_OTR_BidPrice:** type ‘num‘. Closing Bid Price for the 30 Year On-the-Run bond.
- **US30Y_OTR_AskPrice:** type ‘num‘. Closing Ask Price for the 30 Year On-the-Run bond.
- **US30Y_OTR_BidYield:** type ‘num‘. Closing Bid Yield to Maturity for the 30 Year On-the-Run bond.
- **US30Y_OTR_AskYield:** type ‘num‘. Closing Ask Yield to Maturity for the 30 Year On-the-Run bond.
- **US30Y_OFTR_1_BidPrice:** type ‘num‘. Closing Bid Price for the 30 Year First-Off-the-Run bond.
- **US30Y_OFTR_1_AskPrice:** type ‘num‘. Closing Ask Price for the 30 Year First-Off-the-Run bond.
- **US30Y_OFTR_1_BidYield:** type ‘num‘. Closing Bid Yield to Maturity for the 30 Year First-Off-the-Run bond.
- **US30Y_OFTR_1_AskYield:** type ‘num‘. Closing Ask Yield to Maturity for the 30 Year First-Off-the-Run bond.

Details

This data set with the bid and ask prices and yields of 2, 5, 10 and 30-year U.S. Treasuries is supplied by Tradeweb Markets LLC. All prices and yields are reported at the end of the trading day. Prices are reported to the nearest 1/256 (0.00390625), while yields are reported to 3 decimal places.

Source

Tradeweb OTR-OFTR Spread Data is provided by Tradeweb Markets LLC for educational purposes only. Tradeweb Markets LLC makes no representation or warranty of any kind, express or implied, including regarding the accuracy, adequacy, validity, reliability, availability or completeness of the Tradeweb OTR-OFTR Spread Data. All rights reserved.

References

Chapter 14 (Fixed Income Portfolio Management) of Martin, Philips, Scherer, Stoyanov and Li, Portfolio Construction and Risk Analysis, Springer, 2024.

Examples

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

`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