# Package 'FRAPO'

January 20, 2025

**Version** 0.4-1

**Date** 2016-12-07

**Title** Financial Risk Modelling and Portfolio Optimisation with R

**Depends** R (>= 3.1.3), methods, cccp, Rglpk, timeSeries

**Description** Accompanying package of the book 'Financial Risk Modelling
and Portfolio Optimisation with R', second edition. The data sets used in the book are contained in this package.

**LazyData** TRUE

**License** GPL (>= 3)

**NeedsCompilation** no

**Author** Bernhard Pfaff [aut, cre]

**Maintainer** Bernhard Pfaff <bernhard@pfaffikus.de>

**Repository** CRAN

**Date/Publication** 2016-12-12 00:48:20

**Repository/R-Forge/Project** frapo

**Repository/R-Forge/Revision** 33

**Repository/R-Forge/DateTimeStamp** 2016-12-07 21:00:53

## Contents

## BookEx *Utility functions for handling book examples*

### Description

Utility functions for returning a list of the included examples and displaying, executing, saving and editing the example codes are provided.

## Usage

```
listEx()
showEx(Example)
saveEx(Example)
editEx(Example, ...)
runEx(Example, ...)
```

## Arguments

| | |
|---|---|
| `Example` | `Characater`, the name of the example as contained in `listEx()`. |
| `...` | Ellipsis argument. See details. |

## Details

The ellipsis arguments in the function `editEx()` are passed down to the function `file.edit()`. If the option `editor` is unset and/or a different editor shall be employed for openening the example code, then the ellipsis argument can be utilised by `editor = "foo"`, wherey `foo` is the name of the editor to be used.
The ellipsis arguments in the function `runEx()` are passed down to the function `source()`.

## Value

`listEx`
Returns a character vector of the examples' names.
`showEx`
Returns the example of of `Example` to the console.
`saveEx`
Returns a `logical` whether the saving of the R code example into the working directory was successful.
`editEx`
Opens a copy of the example code in an editor.
`runEx`
Executes the example code.

## Author(s)

Bernhard Pfaff

## See Also

`file.edit`, `source`

## Examples

```
## Not run:
listEx()
showEx(Example = "Part1Chapter3Ex2")
saveEx(Example = "Part1Chapter3Ex2")
runEx(Example = "Part1Chapter3Ex2", echo = TRUE)
```

```
editEx(Example = listEx()[1], editor = "emacs")

## End(Not run)
```

---

`capser` *Capping a series to bounds*

---

## Description

The values of a series that are absolute greater than `min` and/or `max` are capped to these specified values.

## Usage

```
capser(y, min, max)
```

## Arguments

| | |
|---|---|
| `y` | Objects of classes: numeric, matrix, data.frame, ts, mts, and timeSeries are supported. |
| `min` | Numeric, minimum value for the series. |
| `max` | Numeric, maximim value for the series. |

## Value

An object of the same class as `y`, containing the truncated series.

## Methods

**y = "data.frame"** The calculation is applied per column of the data.frame and only if all columns are numeric.

**y = "matrix"** The calculation is applied per column of the matrix.

**y = "mts"** The calculation is applied per column of the mts object. The attributes are preserved and an object of the same class is returned.

**y = "numeric"** Calculation of the es trend.

**y = "timeSeries"** The calculation is applied per column of the timeSeries object and an object of the same class is returned.

**y = "ts"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

**y = "xts"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

**y = "zoo"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

## Author(s)

Bernhard Pfaff

## See Also

trdbilson, trdbinary, trdes, trdhp, trdsma, trdwma

## Examples

```
data(StockIndex)
y <- StockIndex[, "SP500"]
cs <- capser(y, min = 100, max = 200)
head(cs)
```

---

`DivMeasures` *Diversification Measures*

---

## Description

These functions compute the diversification ratio, the volatility weighted average correlation and concentration ratio of a portfolio.

## Usage

```
dr(weights, Sigma)
cr(weights, Sigma)
rhow(weights, Sigma)
```

## Arguments

| | |
|---|---|
| `weights` | Vector: portfolio weights. |
| `Sigma` | Matrix: Variance-covariance matrix of portfolio assets. |

## Details

The diversification ratio of a portfolio is defined as:

$$DR(\omega) = \frac{\sum_{i=1}^N \omega_i \sigma_i}{\sqrt{\omega' \Sigma \omega}}$$

for a portfolio of $N$ assets and $\omega_i$ signify the weight of the i-th asset and $\sigma_i$ its standard deviation and $\Sigma$ the variance-covariance matrix of asset returns. The diversification ratio is therefore the weighted average of the assets' volatilities divided by the portfolio volatility.

The concentration ration is defined as:

$$CR = \frac{\sum_{i=1}^N (\omega_i \sigma_i)^2}{(\sum_{i=1}^N \omega_i \sigma_i)^2}$$

and the volatility-weighted average correlation of the assets as:

$$\rho(\omega) = \frac{\sum_{i>j}^{N}(\omega_i \sigma_i \omega_j \sigma_j)\rho_{ij}}{\sum_{i>j}^{N}(\omega_i \sigma_i \omega_j \sigma_j)}$$

The following equation between these measures does exist:

$$DR(\omega) = \frac{1}{\sqrt{\rho(\omega)(1 - CR(\omega)) + CR(\omega)}}$$

## Value

`numeric`, the value of the diversification measure.

## Author(s)

Bernhard Pfaff

## References

Choueifaty, Y. and Coignard, Y. (2008): Toward Maximum Diversification, *Journal of Portfolio Management*, Vol. 34, No. 4, 40–51.

Choueifaty, Y. and Coignard, Y. and Reynier, J. (2011): Properties of the Most Diversified Portfolio, Working Paper, `http://papers.ssrn.com`

## See Also

`PMD`

## Examples

```
data(MultiAsset)
Rets <- returnseries(MultiAsset, method = "discrete", trim = TRUE)
w <- Weights(PMD(Rets))
V <- cov(Rets)
DR <- dr(w, V)
CR <- cr(w, V)
RhoW <- rhow(w, V)
test <- 1 / sqrt(RhoW * (1 - CR) + CR)
all.equal(DR, test)
```

## `ESCBFX` *ESCB FX Reference Rates*

### Description

Daily spot rates of major currencies against the EUR.

### Usage

```
data(ESCBFX)
```

### Format

A data frame with 3,427 daily observations of the spot currency rates rates AUD, CAD, CHF, GBP, HKD, JPY and USD against EUR. The sample starts in 1999-01-04 and ends in 2012-04-04.

### Details

The data has been retrieved from the Statistical Data Warehouse (SDW) Internet-Site of the ECB. In case of missing data entries due to holidays, the last observed data point has been carried forward.

### Source

`http://sdw.ecb.europa.eu`

### Examples

```
data(ESCBFX)
```

## `EuroStoxx50` *EURO STOXX 50*

### Description

Weekly price data of 48 EURO STOXX 50 constituents.

### Usage

```
data(EuroStoxx50)
```

### Format

A data frame with 265 weekly observations of 48 members of the EURO STOXX 50 index. The sample starts at 2003-03-03 and ends in 2008-03-24.

## Details

The data set was used in the reference below. The authors adjusted the price data for dividends and have removed stocks if two or more consecutive missing values were found. In the remaining cases the NA entries have been replaced by interpolated values.

## Source

http://host.uniroma3.it/docenti/cesarone/DataSets.htm
http://finance.yahoo.com/

## References

Cesarone, F. and Scozzari, A. and Tardella, F.: Portfolio selection problems in practice: a comparison between linear and quadratic optimization models, Working Paper, Universita degli Studi Roma Tre, Universita Telematica delle Scienze Umane and Universita di Roma, July 2010.
http://arxiv.org/ftp/arxiv/papers/1105/1105.3594.pdf

## Examples

```
data(EuroStoxx50)
```

---

`FTSE100` *FTSE 100*

---

## Description

Weekly price data of 79 FTSE 100 constituents.

## Usage

```
data(FTSE100)
```

## Format

A data frame with 265 weekly observations of 79 members of the FTSE 100 index. The sample starts at 2003-03-03 and ends in 2008-03-24.

## Details

The data set was used in the reference below. The authors adjusted the price data for dividends and have removed stocks if two or more consecutive missing values were found. In the remaining cases the NA entries have been replaced by interpolated values.

## Source

http://host.uniroma3.it/docenti/cesarone/DataSets.htm
http://finance.yahoo.com/

## References

Cesarone, F. and Scozzari, A. and Tardella, F.: Portfolio selection problems in practice: a comparison between linear and quadratic optimization models, Working Paper, Universita degli Studi Roma Tre, Universita Telematica delle Scienze Umane and Universita di Roma, July 2010.
http://arxiv.org/ftp/arxiv/papers/1105/1105.3594.pdf

## Examples

```
data(FTSE100)
```

---

`INDTRACK1` *INDTRACK1: Hang Seng Index and Constituents*

---

## Description

Weekly price data of the Hang Seng index and 31 constituents.

## Usage

```
data(INDTRACK1)
```

## Format

A data frame with 291 weekly observations of the index and 31 members of the Hang Seng index. The sample starts in March 1991 and ends in September 1997.

## Details

The data set was used in the first two references below. Stocks with missing values during the sample period have been discarded. The data was downloaded from DATASTREAM and has been anonymized. The first columne refers to the index data itself. See the attached license file that is part of this package: ‘`BeasleyLicence`’.

## Source

http://people.brunel.ac.uk/~mastjjb/jeb/info.html
http://people.brunel.ac.uk/~mastjjb/jeb/orlib/legal.html

## References

Canakgoz, N.A. and J.E. Beasley (2008), Mixed-integer programming approaches for index tracking and enhanced indexation, *European Journal of Operational Research*, Vol. 196, 384–399.
Beasley, J.E. and N. Meade and T.-J. Chang (2003), An evolutionary heuristic for the index tracking problem, *European Journal of Operational Research*, Vol. 148, 621–643.
Beasley, J. E. (1990), OR-Library: Distributing Test Problems by Electronic Mail, *Journal of the Operational Research Society*, Vol. 41, No. 11, 1069–1072.

## Examples

```
data(INDTRACK1)
```

---

INDTRACK2 *INDTRACK2: DAX 100 Index and Constituents*

---

## Description

Weekly price data of the DAX 100 and 85 constituents.

## Usage

```
data(INDTRACK2)
```

## Format

A data frame with 291 weekly observations of the index and 85 members of the DAX 100 index. The sample starts in March 1991 and ends in September 1997.

## Details

The data set was used in the first two references below. Stocks with missing values during the sample period have been discarded. The data was downloaded from DATASTREAM and has been anonymized. The first columne refers to the index data itself. See the attached license file that is part of this package: 'BeasleyLicence'.

## Source

http://people.brunel.ac.uk/~mastjjb/jeb/info.html
http://people.brunel.ac.uk/~mastjjb/jeb/orlib/legal.html

## References

Canakgoz, N.A. and J.E. Beasley (2008), Mixed-integer programming approaches for index tracking and enhanced indexation, *European Journal of Operational Research*, Vol. 196, 384–399.
Beasley, J.E. and N. Meade and T.-J. Chang (2003), An evolutionary heuristic for the index tracking problem, *European Journal of Operational Research*, Vol. 148, 621–643.
Beasley, J. E. (1990), OR-Library: Distributing Test Problems by Electronic Mail, *Journal of the Operational Research Society*, Vol. 41, No. 11, 1069–1072.

## Examples

```
data(INDTRACK2)
```

INDTRACK3 *INDTRACK3: FTSE 100 Index and Constituents*

## Description

Weekly price data of of the FTSE 100 index and 89 constituents.

## Usage

```
data(INDTRACK3)
```

## Format

A data frame with 291 weekly observations of of the index and 89 members of the FTSE 100 index. The sample starts in March 1991 and ends in September 1997.

## Details

The data set was used in the first two references below. Stocks with missing values during the sample period have been discarded. The data was downloaded from DATASTREAM and has been anonymized. The first columne refers to the index data itself. See the attached license file that is part of this package: ‘`BeasleyLicence`’.

## Source

`http://people.brunel.ac.uk/~mastjjb/jeb/info.html`
`http://people.brunel.ac.uk/~mastjjb/jeb/orlib/legal.html`

## References

Canakgoz, N.A. and J.E. Beasley (2008), Mixed-integer programming approaches for index tracking and enhanced indexation, *European Journal of Operational Research*, Vol. 196, 384–399.
Beasley, J.E. and N. Meade and T.-J. Chang (2003), An evolutionary heuristic for the index tracking problem, *European Journal of Operational Research*, Vol. 148, 621–643.
Beasley, J. E. (1990), OR-Library: Distributing Test Problems by Electronic Mail, *Journal of the Operational Research Society*, Vol. 41, No. 11, 1069–1072.

## Examples

```
data(INDTRACK3)
```

## INDTRACK4 *INDTRACK4: S&P 100 Index and Constituents*

### Description

Weekly price data of S&P 100 index and 98 constituents.

### Usage

```
data(INDTRACK4)
```

### Format

A data frame with 291 weekly observations of the index 98 members of the S&P 100 index. The sample starts in March 1991 and ends in September 1997.

### Details

The data set was used in the first two references below. Stocks with missing values during the sample period have been discarded. The data was downloaded from DATASTREAM and has been anonymized. The first columne refers to the index data itself. See the attached license file that is part of this package: 'BeasleyLicence'.

### Source

http://people.brunel.ac.uk/~mastjjb/jeb/info.html
http://people.brunel.ac.uk/~mastjjb/jeb/orlib/legal.html

### References

Canakgoz, N.A. and J.E. Beasley (2008), Mixed-integer programming approaches for index tracking and enhanced indexation, *European Journal of Operational Research*, Vol. 196, 384–399.
Beasley, J.E. and N. Meade and T.-J. Chang (2003), An evolutionary heuristic for the index tracking problem, *European Journal of Operational Research*, Vol. 148, 621–643.
Beasley, J. E. (1990), OR-Library: Distributing Test Problems by Electronic Mail, *Journal of the Operational Research Society*, Vol. 41, No. 11, 1069–1072.

### Examples

```
data(INDTRACK4)
```

`INDTRACK5` *INDTRACK5: Nikkei 225 Index and Constituents*

## Description

Weekly price data of Nikkei 225 index and 225 constituents.

## Usage

```
data(INDTRACK5)
```

## Format

A data frame with 291 weekly observations of the index and 225 members of the Nikkei 225 index. The sample starts in March 1991 and ends in September 1997.

## Details

The data set was used in the first two references below. Stocks with missing values during the sample period have been discarded. The data was downloaded from DATASTREAM and has been anonymized. The first columne refers to the index data itself. See the attached license file that is part of this package: '`BeasleyLicence`'.

## Source

http://people.brunel.ac.uk/~mastjjb/jeb/info.html
http://people.brunel.ac.uk/~mastjjb/jeb/orlib/legal.html

## References

Canakgoz, N.A. and J.E. Beasley (2008), Mixed-integer programming approaches for index tracking and enhanced indexation, *European Journal of Operational Research*, Vol. 196, 384–399.
Beasley, J. E. (1990), OR-Library: Distributing Test Problems by Electronic Mail, *Journal of the Operational Research Society*, Vol. 41, No. 11, 1069–1072.

## Examples

```
data(INDTRACK5)
```

## INDTRACK6 *INDTRACK6: S&P 500 Index and Constituents*

### Description

Weekly price data of S&P 500 index and 457 constituents.

### Usage

```
data(INDTRACK6)
```

### Format

A data frame with 291 weekly observations of the index and 457 members of the S&P 500 index. The sample starts in March 1991 and ends in September 1997.

### Details

The data set was used in the first two references below. Stocks with missing values during the sample period have been discarded. The data was downloaded from DATASTREAM and has been anonymized. The first columne refers to the index data itself. See the attached license file that is part of this package: 'BeasleyLicence'.

### Source

http://people.brunel.ac.uk/~mastjjb/jeb/info.html
http://people.brunel.ac.uk/~mastjjb/jeb/orlib/legal.html

### References

Canakgoz, N.A. and J.E. Beasley (2008), Mixed-integer programming approaches for index tracking and enhanced indexation, *European Journal of Operational Research*, Vol. 196, 384–399.
Beasley, J. E. (1990), OR-Library: Distributing Test Problems by Electronic Mail, *Journal of the Operational Research Society*, Vol. 41, No. 11, 1069–1072.

### Examples

```
data(INDTRACK6)
```

## `MIBTEL` *Milano Indice Borsa Telematica*

### Description

Weekly price data of 226 MIBTEL constituents.

### Usage

```
data(MIBTEL)
```

### Format

A data frame with 265 weekly observations of 226 members of the Milano Indice Borsa Telematica index. The sample starts at 2003-03-03 and ends in 2008-03-24.

### Details

The data set was used in the reference below. The authors adjusted the price data for dividends and have removed stocks if two or more consecutive missing values were found. In the remaining cases the NA entries have been replaced by interpolated values.

### Source

`http://host.uniroma3.it/docenti/cesarone/DataSets.htm`
`http://finance.yahoo.com/`

### References

Cesarone, F. and Scozzari, A. and Tardella, F.: Portfolio selection problems in practice: a comparison between linear and quadratic optimization models, Working Paper, Universita degli Studi Roma Tre, Universita Telematica delle Scienze Umane and Universita di Roma, July 2010.
`http://arxiv.org/ftp/arxiv/papers/1105/1105.3594.pdf`

### Examples

```
data(MIBTEL)
```

## mrc *Marginal Contribution to Risk*

### Description

This function returns the marginal contributions to portfolio risk, whereby the latter is defined in terms of the portfolio standard deviation.

### Usage

```
mrc(weights, Sigma, percentage = TRUE)
```

### Arguments

| | |
|---|---|
| `weights` | Vector: portfolio weights. |
| `Sigma` | Matrix: Variance-covariance matrix of portfolio assets. |
| `percentage` | `Logical`, whether the marginal risk contributions shall be returned as percentages that sum to 100 (default) or as decimal numbers. |

### Details

The marginal contributions to risk are computed for a given dispersion matrix and weight vector.

### Value

`numeric`, the marginal risk contributions of the portfolio's asset.

### Author(s)

Bernhard Pfaff

## MultiAsset *Multi Asset Index Data*

### Description

Month-end price data of stock and bond indices and gold.

### Usage

```
data(MultiAsset)
```

### Format

A data frame with 85 month-end observations of stock and bond indices and gold, represented by the ETF SPDR Gold. The sample starts at 2004-11-30 and ends in 2011-11-30.

## Details

The data set has been obtained from Yahoo Finance and hereby the unadjusted closing prices have been retrieved. If a month-end value was not reported, the value of the previous day has been used. The Yahoo mnemonics with the respective item description are listed below:

| | |
|---|---|
| **GSPC** | United States: S \& P 500 Index (Equity) |
| **RUA** | United States: Russell 3000 Index (Equity) |
| **GDAXI** | Germany: DAX (XETRA) Index (Equity) |
| **FTSE** | United Kingdom: FTSE 100 Index (Equity) |
| **N225** | Japan: Nikkei 225 Index (Equity) |
| **EEM** | iShares: MSCI Emerging Markets Index (Equity) |
| **DJCBTI** | United States: Dow Jones CBOT Treasury Index (Bonds) |
| **GREXP** | Germany: REX-Performance Index (Bonds) |
| **BG05.L** | United Kingdom: Gilt All Index (Bonds) |
| **GLD** | United States: SPDR Gold Shares (Commodities) |

## Source

http://finance.yahoo.com/

## Examples

```
data(MultiAsset)
```

---

`NASDAQ` *NASDAQ*

---

## Description

Weekly price data of 2,196 NASDAQ constituents.

## Usage

```
data(NASDAQ)
```

## Format

A data frame with 265 weekly observations of 2196 members of the NASDAQ index. The sample starts at 2003-03-03 and ends in 2008-03-24.

## Details

The data set was used in the reference below. The authors adjusted the price data for dividends and have removed stocks if two or more consecutive missing values were found. In the remaining cases the NA entries have been replaced by interpolated values.

## Source

http://host.uniroma3.it/docenti/cesarone/DataSets.htm
http://finance.yahoo.com/

## References

Cesarone, F. and Scozzari, A. and Tardella, F.: Portfolio selection problems in practice: a comparison between linear and quadratic optimization models, Working Paper, Universita degli Studi Roma Tre, Universita Telematica delle Scienze Umane and Universita di Roma, July 2010.
http://arxiv.org/ftp/arxiv/papers/1105/1105.3594.pdf

## Examples

```
data(NASDAQ)
```

---

`PAveDD` *Portfolio optimisation with average draw down constraint*

---

## Description

This function returns the result of a long-only portfolio optimization whereby the portfolio's (historic) average draw down is constrained to an upper limit.

## Usage

```
PAveDD(PriceData, AveDD = 0.1, softBudget = FALSE, ...)
```

## Arguments

| | |
|---|---|
| `PriceData` | A rectangular array of price data. |
| `AveDD` | Numeric, the upper bound of the average portfolio draw down. |
| `softBudget` | Logical, whether the budget constraint shall be implemented as a soft constraint, *i.e.* the sum of the weights can be less than one. The default is to use an equality constraint. |
| `...` | Arguments are passed down to `Rglpk_solve_LP` |

## Details

This function implements a long-only portfolio optimisation with an average draw down constraint (see references below). The problem can be stated in the form of a linear program and GLPK is used as solver.

## Value

An object of formal class `"PortAdd"`.

## Note

A warning is issued in case the solver had exit status not equal to zero.

## Author(s)

Bernhard Pfaff

## References

Chekhlov, A. and Uryasev, S. and Zabarankin, M., Portfolio Optimization with Drawdown Constraints, Department of Industrial and Systems Engineering, University of Florida, *Research Report 2000-5*, 2000, Gainesville, FL. Chekhlov, A. and Uryasev, S. and Zabarankin, M., Drawdown Measure in Portfolio Optimization, *International Journal of Theoretical and Applied Finance*, 2005, 8(1), 13–58.

## See Also

`"PortSol"`, `"PortAdd"`, `"PortDD"`, `PMaxDD`, `PCDaR`, `PMinCDaR`

## Examples

```
## Not run:
data(StockIndex)
popt <- PAveDD(PriceData = StockIndex, AveDD = 0.1, softBudget = TRUE)

## End(Not run)
```

---

`PCDaR` *Portfolio optimisation with conditional draw down at risk constraint*

---

## Description

This function returns the result of a long-only portfolio optimization whereby the portfolio's (historic) conditional draw down at risk is constrained to an upper limit.

## Usage

```
PCDaR(PriceData, alpha = 0.95, bound = 0.05, softBudget = FALSE, ...)
```

## Arguments

| | |
|---|---|
| `PriceData` | A rectangular array of price data. |
| `alpha` | Numeric, the confidence level for which the conditional draw down shall be computed. |
| `bound` | Numeric, the upper bound of the conditional draw down. |
| `softBudget` | Logical, whether the budget constraint shall be implemented as a soft constraint, *i.e.* the sum of the weights can be less than one. The default is to use an equality constraint. |
| `...` | Arguments are passed down to `Rglpk_solve_LP` |

## Details

This function implements a long-only portfolio optimisation with a CDaR constraint (see references below). The problem can be stated in the form of a linear program and GLPK is used as solver.

## Value

An object of formal class `"PortAdd"`.

## Note

A warning is issued in case the solver had exit status not equal to zero.

## Author(s)

Bernhard Pfaff

## References

Chekhlov, A. and Uryasev, S. and Zabarankin, M., Portfolio Optimization with Drawdown Constraints, Department of Industrial and Systems Engineering, University of Florida, *Research Report 2000-5*, 2000, Gainesville, FL. Chekhlov, A. and Uryasev, S. and Zabarankin, M., Drawdown Measure in Portfolio Optimization, *International Journal of Theoretical and Applied Finance*, 2005, 8(1), 13–58.

## See Also

`"PortSol"`, `"PortCdd"`, `"PortDD"`, `PMaxDD`, `PAveDD`, `PMinCDaR`

## Examples

```
## Not run:
data(StockIndex)
popt <- PCDaR(PriceData = StockIndex, alpha = 0.95,
              bound = 0.1, softBudget = TRUE)

## End(Not run)
```

---

`PERC` *Equal risk contributed portfolios*

---

## Description

This function solves for equal risk contributed portfolio weights.

## Usage

```
PERC(Sigma, par = NULL, percentage = TRUE, optctrl = ctrl(), ...)
```

## Arguments

| | |
|---|---|
| `Sigma` | Matrix, the variance-covariance matrix of asset returns |
| `par` | Vector, the initial values of the weights. |
| `percentage` | Logical, whether the weights shall be returned as decimals or percentages (default). |
| `optctrl` | Object of class `Rcpp_CTRL`. |
| `...` | Ellipsis argument is passed down to `nlminb()`. |

## Details

The objective function is the standard deviation of the marginal risk contributions, which is minimal, *i.e.* zero, if all contributions are equal. The weights are rescaled to sum to unity.

## Value

An object of formal class `"PortSol"`.

## Note

The optimisation is conducted by calling `nlminb()`. Hereby, the arguments `lower = 0` and `upper = 1` have been specified.

## Author(s)

Bernhard Pfaff

## References

Maillard, S. and Roncalli, T. and Teiletche, J.: The Properties of Equally Weighted Risk Contribution Portfolios, *Journal of Portfolio Management*, Vol. 36, No. 4, Summer 2010, 60–70.

## See Also

`"PortSol"`

## Examples

```
data(MultiAsset)
Rets <- returnseries(MultiAsset, method = "discrete", trim = TRUE,
                     percentage = TRUE)
V <- cov(Rets)
ERC <- PERC(V)
ERC
w <- Weights(ERC)
w * V
```

## PGMV — *Global Minimum Variance Portfolio*

### Description

This function returns the solution of the global minimum variance portfolio (long-only).

### Usage

```
PGMV(Returns, percentage = TRUE, optctrl = ctrl(), ...)
```

### Arguments

| | |
|---|---|
| `Returns` | A rectangular array of return data. |
| `percentage` | Logical, whether the weights shall be returned as decimals or percentages (default). |
| `optctrl` | Object of class `Rcpp_CTRL`. |
| `...` | Arguments are passed down to `cov`. |

### Value

An object of formal class `"PortSol"`.

### Note

The optimisation is conducted by calling `cccp()`.

### Author(s)

Bernhard Pfaff

### See Also

`"PortSol"`

### Examples

```
data(MultiAsset)
Rets <- returnseries(MultiAsset, method = "discrete", trim = TRUE)
PGMV(Rets)
```

## `plot-methods` *Methods for Function* `plot` *in Package* **graphics**

### Description

Additional arguments to the `plot`-method pertinent to the defined S4-classes in this package are detailed below.

### Usage

```
## S4 method for signature 'PortDD'
plot(x, main = NULL, xlab = NULL, ylab = NULL,
col = c("black", "red"), grid = TRUE, invert = TRUE, ...)
```

### Arguments

| | |
|---|---|
| `x` | `PortDD`: an object that belongs to this virtual class. |
| `main` | `character`: The title of the plot. |
| `xlab` | `character`: The description of the x-axis. |
| `ylab` | `character`: The description of the y-axis. |
| `col` | `character`: Two-element vector of the names of the colors for the portfolio's draw downs and the optimal level. |
| `grid` | `Logical`: Whether to superimpose a grid on the plot. |
| `invert` | `Logical`: Whether the draw downs shall be plotted as negative numbers; the default is `TRUE`. |
| `...` | Ellipsis argument is passed to the generic `plot` function. |

### Author(s)

Bernhard Pfaff

## `PMaxDD` *Portfolio optimisation with maximum draw down constraint*

### Description

This function returns the result of a long-only portfolio optimization whereby the portfolio's (historic) draw down is constrained to an upper limit.

### Usage

```
PMaxDD(PriceData, MaxDD = 0.1, softBudget = FALSE, ...)
```

## Arguments

| | |
|---|---|
| `PriceData` | A rectangular array of price data. |
| `MaxDD` | Numeric, the upper bound of the maximum draw down. |
| `softBudget` | Logical, whether the budget constraint shall be implemented as a soft constraint, *i.e.* the sum of the weights can be less than one. The default is to use an equality constraint. |
| `...` | Arguments are passed down to `Rglpk_solve_LP` |

## Details

This function implements a long-only portfolio optimisation with a maximum draw down constraint (see references below). The problem can be stated in the form of a linear program and GLPK is used as solver.

## Value

An object of formal class `"PortMdd"`.

## Note

A warning is issued in case the solver had exit status not equal to zero.

## Author(s)

Bernhard Pfaff

## References

Chekhlov, A. and Uryasev, S. and Zabarankin, M., Portfolio Optimization with Drawdown Constraints, Department of Industrial and Systems Engineering, University of Florida, *Research Report 2000-5*, 2000, Gainesville, FL. Chekhlov, A. and Uryasev, S. and Zabarankin, M., Drawdown Measure in Portfolio Optimization, *International Journal of Theoretical and Applied Finance*, 2005, 8(1), 13–58.

## See Also

`"PortSol"`, `"PortMdd"`, `"PortDD"`, `PCDaR`, `PAveDD`, `PMinCDaR`

## Examples

```
## Not run:
data(StockIndex)
popt <- PMaxDD(PriceData = StockIndex, MaxDD = 0.1, softBudget = TRUE)

## End(Not run)
```

## PMD *Most Diversified Portfolio*

### Description

This function returns the solution of the most diversified portfolio (long-only).

### Usage

```
PMD(Returns, percentage = TRUE, optctrl = ctrl(),...)
```

### Arguments

| | |
|---|---|
| `Returns` | A rectangular array of return data. |
| `percentage` | Logical, whether the weights shall be returned as decimals or percentages (default). |
| `optctrl` | Object of class `Rcpp_CTRL`. |
| `...` | Arguments are passed down to `cov()`. |

### Details

The optimisation problem is akin to that of a global minimum-variance portfolio, but instead of using the variance-covariance matrix of the asset returns, the correlation matrix is utilised as dispersion measure. The weights are then recovered by rescaling the optimal solution with the assets' standard deviations and normalizing, such that the weights sum to one.

### Value

An object of formal class `"PortSol"`.

### Note

The optimisation is conducted by calling `cccp()`.

### Author(s)

Bernhard Pfaff

### References

Choueifaty, Y. and Coignard, Y. (2008): Toward Maximum Diversification, *Journal of Portfolio Management*, Vol. 34, No. 4, 40–51.

Choueifaty, Y. and Coignard, Y. and Reynier, J. (2011): Properties of the Most Diversified Portfolio, Working Paper, `http://papers.ssrn.com`

### See Also

`"PortSol"`

## Examples

```
data(MultiAsset)
Rets <- returnseries(MultiAsset, method = "discrete", trim = TRUE)
PMD(Rets)
```

---

PMinCDaR *Portfolio optimisation for minimum conditional draw down at risk*

---

## Description

This function returns the result of a long-only portfolio optimization whereby the portfolio's (historic) conditional draw down at risk is minimized.

## Usage

```
PMinCDaR(PriceData, alpha = 0.95, softBudget = FALSE, ...)
```

## Arguments

| | |
|---|---|
| `PriceData` | A rectangular array of price data. |
| `alpha` | Numeric, the confidence level for which the conditional draw down shall be computed. |
| `softBudget` | Logical, whether the budget constraint shall be implemented as a soft constraint, *i.e.* the sum of the weights can be less than one. The default is to use an equality constraint. |
| `...` | Arguments are passed down to `Rglpk_solve_LP` |

## Details

This function implements a long-only portfolio optimisation for a minimum conditional draw down at risk (see references below). The problem can be stated in the form of a linear program and GLPK is used as solver.

## Value

An object of formal class `"PortAdd"`.

## Note

A warning is issued in case the solver had exit status not equal to zero.

## Author(s)

Bernhard Pfaff

## References

Chekhlov, A. and Uryasev, S. and Zabarankin, M., Portfolio Optimization with Drawdown Constraints, Department of Industrial and Systems Engineering, University of Florida, *Research Report 2000-5*, 2000, Gainesville, FL. Chekhlov, A. and Uryasev, S. and Zabarankin, M., Drawdown Measure in Portfolio Optimization, *International Journal of Theoretical and Applied Finance*, 2005, 8(1), 13–58.

## See Also

"PortSol", "PortCdd", "PortDD", PMaxDD, PAveDD, PCDaR

## Examples

```
## Not run:
data(StockIndex)
popt <- PMinCDaR(PriceData = StockIndex, alpha = 0.95, softBudget = FALSE)

## End(Not run)
```

---

PMTD *Minimum Tail Dependent Portfolio*

---

## Description

This function computes the solution of a minimum tail dependent portfolio (long-only).

## Usage

```
PMTD(Returns, method = c("EmpTC", "EVT"), k = NULL, percentage = TRUE,
     optctrl = ctrl(),...)
```

## Arguments

| | |
|---|---|
| Returns | A rectangular array of return data. |
| method | Character, the type of non-parametric estimation. |
| k | Integer, the threshold value for the order statistic. If left NULL, then k = sqrt(nrow(x)) is used. |
| percentage | Logical, whether the weights shall be returned as decimals or percentages (default). |
| optctrl | Object of class Rcpp_CTRL. |
| ... | Arguments are passed down to rank. |

## Details

Akin to the optimisation of a global minimum-variance portfolio, the minimum tail dependennt portfolio is determined by replacing the variance-covariance matrix with the matrix of the lower tail dependence coefficients as returned by tdc.

## Value

An object of formal class "PortSol".

## Note

The optimisation is conducted by calling cccp().

## Author(s)

Bernhard Pfaff

## See Also

tdc, "PortSol"

## Examples

```
data(StockIndex)
Rets <- returnseries(StockIndex, method = "discrete", trim = TRUE,
                     percentage = TRUE)
PMTD(Rets)
```

---

PortAdd-class          *Class* "PortAdd"

---

## Description

This class is intended to hold the results from a portfolio optimisation with a constraint on its average draw down.

## Objects from the Class

Objects can be created by calls of the form new("PortAdd", ...). This class extends the "PortSol" class.

## Slots

**AveDD:** Numeric, the average draw down.

**DrawDown:** timeSeries, the hsitoric portfolio's draw downs.

**weights:** Numeric, vector of optimal weights.

**opt:** List, the result of the call to GLPK.

**type:** Character, the type of the optimized portfolio.

**call:** The call to the function that created the object.

## Extends

Class "PortSol", directly.

## Methods

No methods defined with class "PortAdd" in the signature.

## Author(s)

Bernhard Pfaff

## See Also

`"PortSol"`, `"PortMdd"`, `"PortCdd"`

## Examples

```
showClass("PortAdd")
```

---

`PortCdd-class` *Class* `"PortCdd"`

---

## Description

This class is intended to hold the results from a portfolio optimisation with a constraint on its average draw down.

## Objects from the Class

Objects can be created by calls of the form `new("PortCdd", ...)`. This class extends the `"PortSol"` class.

## Slots

`CDaR`**:** Numeric, the conditional draw down at risk.

`thresh`**:** Numeric, threshold value for draw downs at the $\alpha$ level.

`DrawDown`**:** timeSeries, the hsitoric portfolios draw downs.

`weights`**:** Numeric, vector of optimal weights.

`opt`**:** List, the result of the call to GLPK.

`type`**:** Character, the type of the optimized portfolio.

`call`**:** The call to the function that created the object.

## Extends

Class `"PortSol"`, directly.

## Methods

No methods defined with class "PortCdd" in the signature.

## Author(s)

Bernhard Pfaff

## See Also

"PortSol", "PortMdd", "PortAdd"

## Examples

```
showClass("PortCdd")
```

---

PortDD-class *Class* "PortDD"

---

## Description

Class union of "PortAdd", "PortCdd" and "PortMdd"

## Objects from the Class

A virtual Class: No objects may be created from it.

## Methods

**DrawDowns** `signature(object = "PortDD")`: Returns the portfolio draw downs.

**plot** `signature(object = "PortDD")`: Time series plot of draw downs.

## Note

This virtual class is intended for specifying methods that are common to all type of draw down portfolios.

## Author(s)

Bernhard Pfaff

## See Also

"PortAdd", "PortMdd", "PortCdd", PMinCDaR, PCDaR, PAveDD, PMaxDD

## Examples

```
showClass("PortDD")
```

## `PortMdd-class` *Class* `"PortMdd"`

### Description

This class is intended to hold the results from a portfolio optimisation with a constraint on its maximum draw down.

### Objects from the Class

Objects can be created by calls of the form `new("PortMdd", ...)`. This class extends the `"PortSol"` class.

### Slots

`MaxDD`**:** Numeric, the maximum draw down.

`DrawDown`**:** timeSeries, the hsitoric portfolio's draw downs.

`weights`**:** Numeric, vector of optimal weights.

`opt`**:** List, the result of the call to GLPK.

`type`**:** Character, the type of the optimized portfolio.

`call`**:** The call to the function that created the object.

### Extends

Class `"PortSol"`, directly.

### Methods

No methods defined with class "PortMdd" in the signature.

### Author(s)

Bernhard Pfaff

### See Also

`"PortSol"`, `"PortAdd"`, `"PortCdd"`

### Examples

```
showClass("PortMdd")
```

`PortSol-class` *Class* `"PortSol"`

## Description

This class is intended to hold the results for the weights of an optimal portfolio. Currently, this class is used for minimum-variance and equal-risk-contributed portfolios. It can further be used to store the results of optimal factor weights according to one of the aforementioned portfolio types.

## Objects from the Class

Objects can be created by calls of the form `new("PortSol", ...)`.

## Slots

`weights`**:** Numeric, vector of optimal weights.

`opt`**:** List, the result of the call to the optimizing function.

`type`**:** Character, the type of the optimized portfolio.

`call`**:** The call to the function that created the object.

## Methods

**show** `signature(object = "PortSol")`: Returns the portfolio type as text with the optimal weights from the object.

**Solution** `signature(object = "PortSol")`: Returns the list object of the optimizer, *i.e.* the slot `opt` from the object.

**Weights** `signature(object = "PortSol")`: Returns the list object of the optimizer, *i.e.* the slot `weights` from the object.

**update** `signature(object = "PortSol")`: updates object by calling the issuing function with altered arguments.

## Author(s)

Bernhard Pfaff

## Examples

```
showClass("PortSol")
```

## returnconvert *Convert Returns from continuous to discrete and vice versa*

### Description

Either continuous returns or discrete returns can be converted into the other type.

### Usage

```
returnconvert(y, convdir = c("cont2disc", "disc2cont"), percentage = TRUE)
```

### Arguments

| | |
|---|---|
| `y` | Objects of classes: numeric, matrix, data.frame, ts, mts, and timeSeries are supported. |
| `convdir` | Character, the type of return conversion. |
| `percentage` | Logical, if `TRUE` (the default) the returns, `y`, are expressed as percentages. |

### Value

An object of the same class as `y`, containing the converted returns.

### Methods

**y = "data.frame"** The calculation is applied per column of the data.frame and only if all columns are numeric.

**y = "matrix"** The calculation is applied per column of the matrix.

**y = "mts"** The calculation is applied per column of the mts object. The attributes are preserved and an object of the same class is returned.

**y = "numeric"** Calculation of the returns.

**y = "timeSeries"** The calculation is applied per column of the timeSeries object and an object of the same class is returned.

**y = "ts"** Calculation of the returns. The attributes are preserved and an object of the same class is returned.

**y = "xts"** Calculation of the returns. The attributes are preserved and an object of the same class is returned.

**y = "zoo"** Calculation of the returns. The attributes are preserved and an object of the same class is returned.

### Author(s)

Bernhard Pfaff

## Examples

```
data(StockIndex)
yc <- diff(log(StockIndex[, "SP500"])) * 100
yd <- returnseries(StockIndex[, "SP500"], method = "discrete",
                   percentage = TRUE, trim = TRUE)
yconv <- returnconvert(yd, convdir = "disc2cont",
                       percentage = TRUE)
all.equal(yc, yconv)
```

---

`returnseries` *Continuous and discrete returns*

---

## Description

Either continuous returns or discrete returns are computed for an object. The returns can be expressed as percenatges and the first `NA` value can be trimmed.

## Usage

```
returnseries(y, method = c("continuous", "discrete"), percentage = TRUE,
             trim = FALSE, compound = FALSE)
```

## Arguments

| | |
|---|---|
| `y` | Objects of classes: numeric, matrix, data.frame, ts, mts, and timeSeries are supported. |
| `method` | Character, the type of return to be computed. |
| `percentage` | Logical, if `TRUE` (the default) the returns are expressed as percenatges. |
| `trim` | Logical, if `FALSE` (the default) the first value is set to `NA` such that the length of the return series coincides with the length of the series in levels. |
| `compound` | Logical, if `FALSE` (the default), then simple returns are computed and otherwise compounded returns. |

## Value

An object of the same class as `y`, containing the truncated series.

## Methods

**y = "data.frame"** The calculation is applied per column of the data.frame and only if all columns are numeric.

**y = "matrix"** The calculation is applied per column of the matrix.

**y = "mts"** The calculation is applied per column of the mts object. The attributes are preserved and an object of the same class is returned.

**y = "numeric"** Calculation of the es trend.

**y = "timeSeries"** The calculation is applied per column of the timeSeries object and an object of the same class is returned.

**y = "ts"** Calculation of the returns. The attributes are preserved and an object of the same class is returned.

**y = "xts"** Calculation of the returns. The attributes are preserved and an object of the same class is returned.

**y = "zoo"** Calculation of the returns. The attributes are preserved and an object of the same class is returned.

### Author(s)

Bernhard Pfaff

### Examples

```
data(StockIndex)
y <- StockIndex[, "SP500"]
ret <- returnseries(y)
head(ret)
```

---

## SP500 *Standard & Poor's 500*

---

### Description

Weekly price data of 476 S&P 500 constituents.

### Usage

```
data(SP500)
```

### Format

A data frame with 265 weekly observations of 476 members of the S&P 500 index. The sample starts at 2003-03-03 and ends in 2008-03-24.

### Details

The data set was used in the reference below. The authors adjusted the price data for dividends and have removed stocks if two or more consecutive missing values were found. In the remaining cases the NA entries have been replaced by interpolated values.

### Source

http://host.uniroma3.it/docenti/cesarone/DataSets.htm
http://finance.yahoo.com/

## References

Cesarone, F. and Scozzari, A. and Tardella, F.: Portfolio selection problems in practice: a comparison between linear and quadratic optimization models, Working Paper, Universita degli Studi Roma Tre, Universita Telematica delle Scienze Umane and Universita di Roma, July 2010.
http://arxiv.org/ftp/arxiv/papers/1105/1105.3594.pdf

## Examples

```
data(SP500)
```

---

`sqrm` *Square root of a quadratic matrix*

---

## Description

This function returns the square root of a quadratic and diagonalisable matrix.

## Usage

```
sqrm(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | `matrix`, must be quadratic. |
| `...` | The ellipsis argument is passed down to `eigen()`. |

## Details

The computation of the square root of a matrix is based upon its eigen values and corresponding eigen vectors. The square matrix $A$ is diagonisable if there is a matrix $V$ such that $D = V^{-1}AV$, whereby $D$ is a diagonal matrix. This is only achieved if the eigen vectors of the $(n \times n)$ matrix $A$ constitute a basis of dimension $n$. The square root of $A$ is then $A^{1/2} = VD^{1/2}V'$.

## Value

A `matrix` object and a scalar in case a $(1 \times 1)$ matrix has been provided.

## Author(s)

Bernhard Pfaff

## See Also

`eigen`

### Examples

```
data(StockIndex)
S <- cov(StockIndex)
SR <- sqrm(S)
all.equal(crossprod(SR), S)
```

---

## `StockIndex` *Stock Index Data*

---

### Description

Month-end price data of six stock indices.

### Usage

```
data(StockIndex)
```

### Format

A data frame with 240 month-end observations of six stock indices: SP 500, Nikkei 225, FTSE 100, CAC40, GDAX and Hang Seng index. The sample starts at 1991-07-31 and ends in 2011-06-30.

### Details

The data set has been obtained from Yahoo Finance and hereby the unadjusted closing prices have been retrieved.

### Source

`http://finance.yahoo.com/`

### Examples

```
data(StockIndex)
```

## `StockIndexAdj` *Stock Index Data*

### Description

Adjusted month-end price data of six stock indices.

### Usage

```
data(StockIndexAdj)
```

### Format

A data frame with 240 month-end observations of six stock indices: SP 500, Nikkei 225, FTSE 100, CAC40, GDAX and Hang Seng index. The sample starts at 1991-07-31 and ends in 2011-06-30.

### Details

The data set has been obtained from Yahoo Finance and hereby the adjusted closing prices have been retrieved.

### Source

http://finance.yahoo.com/

### Examples

```
data(StockIndexAdj)
```

## `StockIndexAdjD` *Stock Index Data*

### Description

Adjusted daily price data of six stock indices.

### Usage

```
data(StockIndexAdj)
```

### Format

A data frame with 5,202 daily observations of six stock indices: SP 500, Nikkei 225, FTSE 100, CAC40, GDAX and Hang Seng index. The sample starts at 1991-07-01 and ends in 2011-06-30.

## Details

The data set has been obtained from Yahoo Finance and hereby the adjusted closing prices have been retrieved.

## Source

http://finance.yahoo.com/

## Examples

```
data(StockIndexAdjD)
```

---

## tdc *Tail Dependence Coefficient*

---

### Description

This function returns the pairwise tail dependence coefficients between $N$ series. The TDCs are estimated non-parametrically by either the empirical tail copula or based on the stable tail-dependence function.

### Usage

```
tdc(x, method = c("EmpTC", "EVT"), lower = TRUE, k = NULL, ...)
```

### Arguments

| | |
|---|---|
| `x` | Matrix, or an object that can be coerced to it. |
| `method` | Character, the type of non-parametric estimation. |
| `lower` | Logical, if `TRUE` (default), lower TDC are computed and upper TDC, else. |
| `k` | Integer, the threshold value for the order statistic. If left `NULL`, then `k = sqrt(nrow(x))` is used. |
| `...` | Ellipsis, arguments are passed down to `rank`. |

### Details

For a matrix or an object that can be coerced to it with `ncol(x) >= 2`, the pair wise tail dependencies are estimated non-parametrically and returned as a symmetric matrix. The threshold value `k` is the upper/lower bound for the order statistics to be considered. The diagonal elements are always equal to one, because a series has a dependence of one with itself, of course.

### Value

A matrix with the tail dependent coefficients.

### Author(s)

Bernhard Pfaff

### References

Schmidt, R. and Stadtm\"uller, U., Nonparametric estimation of tail dependence, *The Scandinavian Journal of Statistics*, 33, 307–335.

### See Also

PMTD

### Examples

```
data(StockIndex)
Rets <- returnseries(StockIndex, method = "discrete", trim = TRUE,
                     percentage = TRUE)
tdc(Rets, method = "EmpTC")
tdc(Rets, method = "EVT")
```

---

## trdbilson *Bilson Trend*

---

### Description

Calculation of the Bilson Trend as a technical trading indicator.

### Usage

```
trdbilson(y, exponent)
```

### Arguments

| | |
|---|---|
| y | Objects of classes: numeric, matrix, data.frame, ts, mts, and timeSeries are supported. |
| exponent | Numeric, the value for $\alpha$ in the equation below. |

### Details

The Bilson trend is calculated according to the formula:

$$z = sign(y) \times |y|^{(1-|y|^{\alpha})}$$

### Value

An object of the same class as y, containing the computed Bilson trend values.

## Methods

**y = "data.frame"** The calculation is applied per column of the data.frame and only if all columns are numeric.

**y = "matrix"** The calculation is applied per column of the matrix.

**y = "mts"** The calculation is applied per column of the mts object. The attributes are preserved and an object of the same class is returned.

**y = "numeric"** Calculation of the bilson trend.

**y = "timeSeries"** The calculation is applied per column of the timeSeries object and an object of the same class is returned.

**y = "ts"** Calculation of the bilson trend. The attributes are preserved and an object of the same class is returned.

**y = "xts"** Calculation of the bilson trend. The attributes are preserved and an object of the same class is returned.

**y = "zoo"** Calculation of the bilson trend. The attributes are preserved and an object of the same class is returned.

## Author(s)

Bernhard Pfaff

## See Also

trdbinary, trdes, trdhp, trdsma, trdwma, capser

## Examples

```
data(StockIndex)
y <- StockIndex[, "SP500"]
yret <- diff(log(y))
bilson <- trdbilson(yret, exponent = 2)
head(bilson)
```

---

## trdbinary *Binary Trend*

---

## Description

Calculation of the Binary Trend as a technical trading indicator.

## Usage

```
trdbinary(y)
```

## Arguments

| | |
|---|---|
| y | Objects of classes: numeric, matrix, data.frame, ts, mts, and timeSeries are supported. |

## Details

The Binary trend is calculated according to the formula:

$$z = sign(y) \times \min(|4/\pi \arctan(y)|, 1)$$

## Value

An object of the same class as y, containing the computed Binary trend values.

## Methods

**y = "data.frame"** The calculation is applied per column of the data.frame and only if all columns are numeric.

**y = "matrix"** The calculation is applied per column of the matrix.

**y = "mts"** The calculation is applied per column of the mts object. The attributes are preserved and an object of the same class is returned.

**y = "numeric"** Calculation of the binary trend.

**y = "timeSeries"** The calculation is applied per column of the timeSeries object and an object of the same class is returned.

**y = "ts"** Calculation of the binary trend. The attributes are preserved and an object of the same class is returned.

**y = "xts"** Calculation of the binary trend. The attributes are preserved and an object of the same class is returned.

**y = "zoo"** Calculation of the binary trend. The attributes are preserved and an object of the same class is returned.

## Author(s)

Bernhard Pfaff

## See Also

trdbilson, trdes, trdhp, trdsma, trdwma, capser

## Examples

```
data(StockIndex)
y <- StockIndex[, "SP500"]
yret <- diff(log(y))
binary <- trdbinary(yret)
head(binary)
```

## trdes *Exponentially Smoothed Trend*

### Description

Calculation of the exponentially smoothed trend as a technical trading indicator.

### Usage

```
trdes(y, lambda, init = NULL)
```

### Arguments

| | |
|---|---|
| `y` | Objects of classes: numeric, matrix, data.frame, ts, mts, and timeSeries are supported. |
| `lambda` | Numeric, the smoothing parameter for $\lambda$ in the equation below. The value for the parameter must be in the interval $0 < \lambda < 1$. |
| `init` | The initial value in the recursive calculation of the filter. Specifies the initial values of the time series just prior to the start value, in reverse time order. The default, *i.e.* `NULL`, is a set of zeros. |

### Details

The exponetially smoothed trend is calculated according to the formula:

$$z_t = \lambda y_t + (1 - \lambda) * z_{t-1}$$

### Value

An object of the same class as `y`, containing the computed exponetially smoothed values.

### Methods

**y = "data.frame"** The calculation is applied per column of the data.frame and only if all columns are numeric.

**y = "matrix"** The calculation is applied per column of the matrix.

**y = "mts"** The calculation is applied per column of the mts object. The attributes are preserved and an object of the same class is returned.

**y = "numeric"** Calculation of the es trend.

**y = "timeSeries"** The calculation is applied per column of the timeSeries object and an object of the same class is returned.

**y = "ts"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

**y = "xts"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

**y = "zoo"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

### Author(s)

Bernhard Pfaff

### See Also

filter, trdbilson, trdbinary, trdhp, trdsma, trdwma, capser

### Examples

```
data(StockIndex)
y <- StockIndex[, "SP500"]
yret <- diff(log(y))
es <- trdes(yret, lambda = 0.95)
head(es)
```

## trdhp *Hodrick-Prescott Filter*

### Description

Calculation of the Hodrick-Prescott filter as a technical trading indicator.

### Usage

```
trdhp(y, lambda)
```

### Arguments

| | |
|---|---|
| y | Objects of classes: numeric, matrix, data.frame, ts, mts, and timeSeries are supported. |
| lambda | Numeric, the value for $\lambda$ in the equation below. |

### Details

The Hodrick-Prescott filter is calculated according to the formula:

$$\min(\tau_t) = \sum_{t=1}^{T}(y_t - \tau_t)^2 + \lambda \sum_{t=2}^{T-1}(\Delta^2 \tau_{t+1})^2$$

### Value

An object of the same class as y, containing the computed Hodrick-Prescott values.

## Methods

**y = "data.frame"** The calculation is applied per column of the data.frame and only if all columns are numeric.

**y = "matrix"** The calculation is applied per column of the matrix.

**y = "mts"** The calculation is applied per column of the mts object. The attributes are preserved and an object of the same class is returned.

**y = "numeric"** Calculation of the bilson trend.

**y = "timeSeries"** The calculation is applied per column of the timeSeries object and an object of the same class is returned.

**y = "ts"** Calculation of the bilson trend. The attributes are preserved and an object of the same class is returned.

**y = "xts"** Calculation of the bilson trend. The attributes are preserved and an object of the same class is returned.

**y = "zoo"** Calculation of the bilson trend. The attributes are preserved and an object of the same class is returned.

## Author(s)

Bernhard Pfaff

## References

Hodrick, R. and E.C. Prescott (1997), Postwar U.S. Business Cycles: An Empirical Investigation, *Journal of Money, Credit and Banking* 29(1).

## See Also

`trdbinary`, `trdes`, `trdbilson`, `trdsma`, `trdwma`, `capser`

## Examples

```
data(StockIndex)
y <- StockIndex[, "SP500"]
hp <- trdhp(y, lambda = 1600)
head(hp)
```

---

## `trdsma` *Simple Moving Average*

---

## Description

Calculation of a right ended simple moving average with equal weights determined by `n.periods`.

## Usage

```
trdsma(y, n.periods, trim = FALSE)
```

## Arguments

| | |
|---|---|
| `y` | Objects of classes: numeric, matrix, data.frame, ts, mts, and timeSeries are supported. |
| `n.periods` | Integer, the number of periods to be included in the calculation of the simple moving average. |
| `trim` | Logical, if `FALSE` (the default) the first value is set to `NA`, otherwise the object is trimmed by the first obeservation. |

## Value

An object of the same class as `y`, containing the computed simple moving averages.

## Methods

**y = "data.frame"** The calculation is applied per column of the data.frame and only if all columns are numeric.

**y = "matrix"** The calculation is applied per column of the matrix.

**y = "mts"** The calculation is applied per column of the mts object. The attributes are preserved and an object of the same class is returned.

**y = "numeric"** Calculation of the es trend.

**y = "timeSeries"** The calculation is applied per column of the timeSeries object and an object of the same class is returned.

**y = "ts"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

**y = "xts"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

**y = "zoo"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

## Author(s)

Bernhard Pfaff

## See Also

`filter`, `trdbilson`, `trdbinary`, `trdhp`, `trdwma`, `capser`, `trdes`

## Examples

```
data(StockIndex)
y <- StockIndex[, "SP500"]
sma <- trdsma(y, n.periods = 24)
head(sma, 30)
```

## trdwma *Weighted Moving Average*

### Description

Calculation of a right ended weighted moving average with weights according to `weights`.

### Usage

```
trdwma(y, weights, trim = FALSE)
```

### Arguments

| | |
|---|---|
| `y` | Objects of classes: numeric, matrix, data.frame, ts, mts, and timeSeries are supported. |
| `weights` | Numeric, a vector containing the weights. |
| `trim` | Logical, if `FALSE` (the default) the first value is set to `NA`, otherwise the object is trimmed by the first obeservation. |

### Details

If the sum of the weights is greater than unity, a warning is issued.

### Value

An object of the same class as `y`, containing the computed weighted moving averages.

### Methods

**y = "data.frame"** The calculation is applied per column of the data.frame and only if all columns are numeric.

**y = "matrix"** The calculation is applied per column of the matrix.

**y = "mts"** The calculation is applied per column of the mts object. The attributes are preserved and an object of the same class is returned.

**y = "numeric"** Calculation of the es trend.

**y = "timeSeries"** The calculation is applied per column of the timeSeries object and an object of the same class is returned.

**y = "ts"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

**y = "xts"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

**y = "zoo"** Calculation of the es trend. The attributes are preserved and an object of the same class is returned.

## Author(s)

Bernhard Pfaff

## See Also

`filter`, `trdbilson`, `trdbinary`, `trdhp`, `trdes`, `trdsma`, `capser`

## Examples

```
data(StockIndex)
y <- StockIndex[, "SP500"]
wma <- trdwma(y, weights = c(0.4, 0.3, 0.2, 0.1))
head(wma, 30)
```

# Index