

Package ‘portfolio.optimization’

July 23, 2025

Type Package

Title Contemporary Portfolio Optimization

Version 1.0-0

Date 2018-08-20

Maintainer Ronald Hochreiter <ron@hochreiter.net>

Description Simplify your portfolio optimization process by applying a contemporary modeling way to model and solve your portfolio problems. While most approaches and packages are rather complicated this one tries to simplify things and is agnostic regarding risk measures as well as optimization solvers. Some of the methods implemented are described by Konno and Yamazaki (1991) <doi:10.1287/mnsc.37.5.519>, Rockafellar and Uryasev (2001) <doi:10.21314/JOR.2000.038> and Markowitz (1952) <doi:10.1111/j.1540-6261.1952.tb01525.x>.

Depends R (>= 3.5), xts, MASS, magrittr, modopt.matlab

License MIT + file LICENSE

URL <http://www.finance-r.com/>

RoxygenNote 6.1.0

NeedsCompilation no

Author Ronald Hochreiter [aut, cre]

Repository CRAN

Date/Publication 2018-08-24 16:10:18 UTC

Contents

portfolio.optimization-package	2
active.extension	3
alpha	3
aux_portfolio.default	4
aux_risk.alias	5
aux_simulate.scenarios	5
linear.constraint.eq	6
linear.constraint.iq	6

long.only	7
lower.bound	8
momentum	8
objective	9
optimal.portfolio	9
optimal.portfolio.lowerN	10
optimal.portfolio.expected.shortfall	11
optimal.portfolio.expected.shortfall.long.short	11
optimal.portfolio.mad	12
optimal.portfolio.mad.long.short	12
optimal.portfolio.markowitz	13
optimal.portfolio.momentum	13
optimal.portfolio.reward	14
po.tutorial	15
portfolio.loss	15
portfolio.model	16
portfolio.weights	17
print.portfolio.model	17
sp100w17	18
sp100w17av	19
sp100w17av30s	19
upper.bound	20

Index 21

portfolio.optimization-package

Contemporary Portfolio Optimization

Description

Simplify your portfolio optimization process by applying a contemporary modeling way to model and solve your portfolio problems. While most approaches and packages are rather complicated this one tries to simplify things and is agnostic regarding risk measures as well as optimization solvers. Some of the methods implemented are described by Konno and Yamazaki (1991) <doi:10.1287/mnsc.37.5.519>, Rockafellar and Uryasev (2001) <doi:10.21314/JOR.2000.038> and Markowitz (1952) <doi:10.1111/j.1540-6261.1952.tb01525.x>.

Author(s)

Ronald Hochreiter, <ronald@hochreiter.net>

References

<http://www.finance-r.com/>

See Also

Useful links:

- <http://www.finance-r.com/>

active.extension	<i>Enable active extension portfolios</i>
------------------	---

Description

active.extension adds corresponding long/short constraints for a diverse set of active extension portfolios (e.g. 130/30 portfolios)

Usage

```
active.extension(model, up = 130, down = 30)
```

Arguments

model	the portfolio.model to activate
up	percentage long (e.g. 130)
down	percentage short (e.g. 30)

Value

portfolio.model with active extension enabled

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

alpha	<i>Set new alpha of a portfolio.model</i>
-------	---

Description

alpha sets a new alpha for VaR and Expected Shortfall

Usage

```
alpha(model, alpha)
```

Arguments

model	the portfolio.model to be changed
alpha	the value alpha (between 0 and 1)

Value

the adapted portfolio.model

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

Examples

```
data(sp100w17av30s)
model <- optimal.portfolio(scenario.set)
cvar95 <- optimal.portfolio(objective(model, "expected.shortfall"))
cvar90 <- optimal.portfolio(alpha(cvar95, 0.1))
```

aux_portfolio.default *Set portfolio.model default values*

Description

aux_portfolio.default sets portfolio.model default values

Usage

```
aux_portfolio.default(model)
```

Arguments

model the portfolio.model to be reset

Value

a portfolio.model with all default values set

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

aux_risk.alias	<i>Convert risk alias names to internal names</i>
----------------	---

Description

aux_risk.alias converts risk alias names to internal names

Usage

```
aux_risk.alias(risk)
```

Arguments

risk	the risk name to be standardized
------	----------------------------------

Value

the standardized risk name (if any)

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

aux_simulate.scenarios	<i>Simulate a multivariate-normal scenario.set</i>
------------------------	--

Description

aux_simulate.scenarios simulates a scenario.set given a mean vector and a covariance matrix using mvnrm of the MASS package

Usage

```
aux_simulate.scenarios(mu, Sigma, n = 1000, seed = 280277)
```

Arguments

mu	mean vector of asset returns
Sigma	covariance matrix of asset returns
n	number of scenarios to simulate (default 1000)
seed	random number seed (default 280277)

Value

A scenario set 'simulation' with mean 'mu' and covariance 'Sigma'

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

linear.constraint.eq *Create or update a vector-based linear equality constraint set*

Description

linear.constraint.eq creates a vector-based linear equality constraint: $Aeq(range) * factors == beq$

Usage

```
linear.constraint.eq(constraints.linear, range, beq, factors = NULL)
```

Arguments

constraints.linear	the current set of equality constraints
range	the range of the variables to set (default 1 if factors is NULL)
beq	right-hand side scalar
factors	values to set for each variable in the given range

Value

the new (updated) set of equality constraints

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

linear.constraint.iq *Create or update a vector-based linear inequality constraint set*

Description

linear.constraint.iq creates a vector-based linear inequality constraint: $Aeq(range) * factors \leq beq$

Usage

```
linear.constraint.iq(constraints.linear, range, b, factors = NULL,
  leq = TRUE)
```

Arguments

constraints.linear	the current set of inequality constraints
range	the range of the variables to set (default 1 if factors is NULL)
b	right-hand side scalar
factors	values to set for each variable in the given range
leq	if false then the sign of the constraint will be inversed

Value

the new (updated) set of inequality constraints

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

long.only

Disable active extension portfolios

Description

long.only switches a portfolio.model back to long-only by disabling the active extension

Usage

```
long.only(model)
```

Arguments

model	the portfolio.model to deactivate active extensions
-------	---

Value

portfolio.model with active extension disabled

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

<code>lower.bound</code>	<i>Set lower bounds on assets</i>
--------------------------	-----------------------------------

Description

`lower.bound` sets lower bounds on assets within a `portfolio.model`

Usage

```
lower.bound(model, v1 = NULL, v2 = NULL)
```

Arguments

<code>model</code>	the <code>portfolio.model</code> to adapt the lower bounds
<code>v1</code>	either one lower bound or lower bounds for all assets
<code>v2</code>	if not empty then <code>v1</code> contains the positions (or names) and <code>v2</code> the bounds

Value

`portfolio.model` with new lower bounds

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

<code>momentum</code>	<i>Set momentum parameters for a portfolio.model</i>
-----------------------	--

Description

`momentum` sets a new alpha for VaR and Expected Shortfall

Usage

```
momentum(model, n_momentum, n_momentum.short = NULL)
```

Arguments

<code>model</code>	the <code>portfolio.model</code> to be changed
<code>n_momentum</code>	amount of momentum assets long
<code>n_momentum.short</code>	amount of momentum assets short

Value

the adapted `portfolio.model`

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

objective	<i>Set new objective of a portfolio.model</i>
-----------	---

Description

objective sets a new objective for VaR and Expected Shortfall

Usage

```
objective(model, objective = "markowitz")
```

Arguments

model	the portfolio.model to be changed
objective	the new objective

Value

the adapted portfolio.model

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

Examples

```
data(sp100w17av30s)
model <- portfolio.model(scenario.set)
mad <- optimal.portfolio(objective(model, "mad"))
```

optimal.portfolio	<i>Meta-function to optimize portfolios given a portfolio.model instance</i>
-------------------	--

Description

optimal.portfolio optimizes the portfolio of a model given the current specification

Usage

```
optimal.portfolio(input = NULL, ...)

p.opt(input = NULL, ...)

opt.p(input = NULL, ...)
```

Arguments

input either a portfolio.model or something to convert to a new model
... other parameters to be passed on to the optimization sub-functions.

Value

an S3 object of class portfolio.model with the optimized portfolio.

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

Examples

```
data(sp100w17av30s)
model <- optimal.portfolio(scenario.set)
```

optimal.portfolio.1overN
1 over N portfolio

Description

optimal.portfolio.1overN adds a 1 over N portfolio to the portfolio.model

Usage

```
optimal.portfolio.1overN(model)
```

Arguments

model the portfolio.model to compute the portfolio of

Value

the portfolio.model including the newly computed optimal portfolio

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

`optimal.portfolio.expected.shortfall`*Portfolio Optimization minimizing Conditional Value at Risk (CVaR)*

Description

`optimal.portfolio.expected.shortfall` conducts a Portfolio Optimization minimizing Conditional Value at Risk (CVaR) based on Rockafellar and Uryasev (2001)

Usage

```
optimal.portfolio.expected.shortfall(model)
```

Arguments

`model` the portfolio.model to compute the portfolio of

Value

the portfolio.model including the newly computed optimal portfolio

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

`optimal.portfolio.expected.shortfall.long.short`*Portfolio Optimization minimizing Conditional Value at Risk (CVaR)
with active extensions*

Description

`optimal.portfolio.expected.shortfall.long.short` conducts a Portfolio Optimization minimizing Conditional Value at Risk (CVaR) based on Rockafellar and Uryasev (2001) with active extensions

Usage

```
optimal.portfolio.expected.shortfall.long.short(model)
```

Arguments

`model` the portfolio.model to compute the portfolio of

Value

the portfolio.model including the newly computed optimal portfolio

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

optimal.portfolio.mad *Portfolio Optimization minimizing MAD*

Description

optimal.portfolio.mad conducts a Portfolio Optimization minimizing Mean Absolute Deviation (MAD) based on Konno and Yamazaki (1991)

Usage

```
optimal.portfolio.mad(model)
```

Arguments

model the portfolio.model to compute the portfolio of

Value

the portfolio.model including the newly computed optimal portfolio

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

optimal.portfolio.mad.long.short
Portfolio Optimization minimizing MAD (Active Extension)

Description

optimal.portfolio.mad.long.short conducts a Portfolio Optimization minimizing Mean Absolute Deviation (MAD) based on Konno and Yamazaki (1991) including an active extension

Usage

```
optimal.portfolio.mad.long.short(model)
```

Arguments

model the portfolio.model to compute the portfolio of

Value

the portfolio.model including the newly computed optimal portfolio

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

optimal.portfolio.markowitz

Portfolio Optimization minimizing Standard Deviation

Description

portfolio.weights conducts a Portfolio Optimization minimizing Standard Deviation based on Markowitz (1952).

Usage

optimal.portfolio.markowitz(model)

Arguments

model the portfolio.model to compute the portfolio of

Value

the portfolio.model including the newly computed optimal portfolio

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

optimal.portfolio.momentum

Momentum portfolio including momentum for active extensions

Description

optimal.portfolio.momentum adds a momentum portfolio to the portfolio.model

Usage

optimal.portfolio.momentum(model)

Arguments

model the portfolio.model to compute the portfolio of

Value

the portfolio.model including the newly computed optimal portfolio

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

optimal.portfolio.reward

Compute maximum/minimum return portfolio given the constraints

Description

optimal.portfolio.reward computes a maximum/minimum return portfolio given the constraints

Usage

optimal.portfolio.reward(model)

Arguments

model the portfolio.model to compute the portfolio of

Value

the portfolio.model including the newly computed optimal portfolio

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

po.tutorial	<i>Open a specific portfolio.optimization package tutorial</i>
-------------	--

Description

po.tutorial returns the filename of a specific portfolio.optimization package tutorial. If no tutorial is given or the tutorial is misspelled, a list of available tutorials is printed.

Usage

```
po.tutorial(tutorial = "")
```

Arguments

tutorial	name of the tutorial to open
----------	------------------------------

Value

Nothing if no tutorial specified, otherwise the path to the tutorial.

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

Examples

```
## Not run:  
file.edit(po.tutorial("101"))  
file.edit(po.tutorial("compare"))  
  
## End(Not run)
```

portfolio.loss	<i>Return the loss distribution of the portfolio.model</i>
----------------	--

Description

portfolio.loss return the loss distribution of the portfolio.model

Usage

```
portfolio.loss(model)  
  
l(model)
```

Arguments

model the portfolio.model to display

Value

nothing

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

portfolio.model *Create a portfolio.model instance (or fix an existing one)*

Description

portfolio.model creates a new S3 portfolio.model instance or fixes an existing one.

Usage

```
portfolio.model(input = NULL)
```

```
p.mo(input = NULL)
```

Arguments

input model, scenario.set or mean.covariance list

Value

an S3 object of class portfolio.model

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

portfolio.weights	<i>Return the portfolio weights of a portfolio.model</i>
-------------------	--

Description

portfolio.weights return the portfolio weights of a portfolio.model

Usage

```
portfolio.weights(model)
```

```
portfolio(model)
```

```
w(model)
```

```
weights(model)
```

```
x(model)
```

Arguments

model the portfolio.model to return the portfolio weights from

Value

a vector of portfolio weights or NULL if no weights are available yet.

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

Examples

```
data(sp100w17av30s)
portfolio.weights(optimal.portfolio(scenario.set))
```

print.portfolio.model	<i>Overload print() for portfolio.model</i>
-----------------------	---

Description

print.portfolio.model overloads print() and outputs some information about the portfolio.model object

Usage

```
## S3 method for class 'portfolio.model'  
print(x, ...)
```

Arguments

x	the portfolio.model to display
...	further parameters

Value

nothing

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

sp100w17

S&P 100 weekly stock returns 2017

Description

A dataset sp100w17 containing the (crude) weekly returns of (almost) all S&P 100 stocks of 2017, daily basis (101 stocks, 251 returns).

Usage

```
data(sp100w17)
```

Format

One xts time series object with 251 rows and 101 columns.

Details

Furthermore contains a vector sp100w17av with the average trading volume of all stocks in 2017 - to be used for a subselection.

`sp100w17av`*S&P 100 average trading volume over the whole year 2017*

Description

A vector `sp100w17av` with the average trading volume of all stocks in 2017 - to be used e.g. for a subselection.

Usage

```
data(sp100w17)
```

Format

One named numeric vector of length 101.

`sp100w17av30s`*S&P 100 weekly stock returns 2017 of 30 stocks with the highest average trading volume over the whole year*

Description

A scenario `sp100w17` containing the (crude) weekly returns of (almost) all S&P 100 stocks of 2017, daily basis (101 stocks, 251 returns).

Usage

```
data(sp100w17av30s)
```

Format

A named matrix `scenario.set` with 251 rows and 30 columns.

upper.bound	<i>Set upper bounds on assets</i>
-------------	-----------------------------------

Description

upper.bound sets lower bounds on assets within a portfolio.model

Usage

```
upper.bound(model, v1 = NULL, v2 = NULL)
```

Arguments

model	the portfolio.model to adapt the upper bounds
v1	either one upper bound or lower bounds for all assets
v2	if not empty then v1 contains the positions (or names) and v2 the bounds

Value

portfolio.model with new upper bounds

Author(s)

Ronald Hochreiter, <ronald@algorithmic.finance>

Index

- * **datasets**
 - sp100w17, [18](#)
 - sp100w17av, [19](#)
 - sp100w17av30s, [19](#)
- active.extension, [3](#)
- alpha, [3](#)
- aux_portfolio.default, [4](#)
- aux_risk.alias, [5](#)
- aux_simulate.scenarios, [5](#)
- l (portfolio.loss), [15](#)
- linear.constraint.eq, [6](#)
- linear.constraint.iq, [6](#)
- long.only, [7](#)
- lower.bound, [8](#)
- momentum, [8](#)
- objective, [9](#)
- opt.p (optimal.portfolio), [9](#)
- optimal.portfolio, [9](#)
- optimal.portfolio.1overN, [10](#)
- optimal.portfolio.expected.shortfall,
[11](#)
- optimal.portfolio.expected.shortfall.long.short,
[11](#)
- optimal.portfolio.mad, [12](#)
- optimal.portfolio.mad.long.short, [12](#)
- optimal.portfolio.markowitz, [13](#)
- optimal.portfolio.momentum, [13](#)
- optimal.portfolio.reward, [14](#)
- p.mo (portfolio.model), [16](#)
- p.opt (optimal.portfolio), [9](#)
- po.tutorial, [15](#)
- portfolio (portfolio.weights), [17](#)
- portfolio.loss, [15](#)
- portfolio.model, [16](#)
- portfolio.optimization
(portfolio.optimization-package),
[2](#)
- portfolio.optimization-package, [2](#)
- portfolio.weights, [17](#)
- print.portfolio.model, [17](#)
- scenario.set (sp100w17av30s), [19](#)
- sp100w17, [18](#)
- sp100w17av, [19](#)
- sp100w17av30s, [19](#)
- upper.bound, [20](#)
- w (portfolio.weights), [17](#)
- weights (portfolio.weights), [17](#)
- x (portfolio.weights), [17](#)