Package 'PortfolioEffectHFT'

October 12, 2022

Type Package

Title High Frequency Portfolio Analytics by PortfolioEffect

Version 1.8

Date 2017-03-21

URL https://www.portfolioeffect.com/

Depends R (>= 2.13.2), ggplot2 (>= 2.2.0)

Imports methods, rJava, grid, zoo

Suggests testthat

SystemRequirements Java (>= 1.7)

LazyData yes

ByteCompile TRUE

Maintainer Andrey Kostin <andrey.kostin@portfolioeffect.com>

Description R interface to PortfolioEffect cloud service for backtesting high frequency trading (HFT) strategies, intraday portfolio analysis and optimization. Includes auto-calibrating model pipeline for market microstructure noise, risk factors, price jumps/outliers, tail risk (high-order moments) and price fractality (long memory). Constructed portfolios could use client-side market data or access HF intraday price history for all major US Equities. See <https://www.portfolioeffect.com/> for more information on the PortfolioEffect high frequency portfolio analytics platform.

License GPL-3

Copyright See file COPYRIGHTS

NeedsCompilation no

Repository CRAN

RoxygenNote 5.0.1

Author Andrey Kostin [aut, cre],
Aleksey Zemnitskiy [aut],
Oleg Nechaev [aut],
Craig Otis and others [ctb, cph] (OpenFAST library),

Daniel Lemire, Muraoka Taro and others [ctb, cph] (JavaFastPFOR library),
Joe Walnes, Jorg Schaible and others [ctb, cph] (XStream library),
Dain Sundstrom [ctb, cph] (Snappy library),
Extreme! Lab, Indiana University [ctb, cph] (XPP3 library),
The Apache Software Foundation [ctb, cph] (Apache Log4j and Commons Lang libraries),
Google, Inc. [ctb, cph] (GSON library),
Free Software Foundation [ctb, cph] (GNU Trove and GNU Crypto libraries)

Date/Publication 2017-03-24 19:54:25 UTC

R topics documented:

`aapl.data` *Sample Price Data*

Description

Sample historical prices of Google (goog.data), Apple (aapl.data) & S&P 500 ETF (spy.data)

Usage

```
aapl.data
goog.data
spy.data
```

Value

Void

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
data(aapl.data)
data(goog.data)
data(spy.data)
plot(aapl.data[,2])
```

alpha_exante *Alpha*

Description

Computes monetary value of a portfolio or position from the beginning of the holding period.

Usage

```
alpha_exante(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(alpha_exante(portfolio),alpha_exante(positionGOOG),alpha_exante(positionAAPL))
plot(alpha_exante(portfolio),alpha_exante(positionGOOG),alpha_exante(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Alpha')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(alpha_exante(positionC),alpha_exante(positionGOOG),alpha_exante(positionAAPL))
plot(alpha_exante(positionC),alpha_exante(positionGOOG),alpha_exante(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Alpha')
```

```
## End(Not run)
```

`alpha_jensens` *Jensen's Alpha*

Description

Computes portfolio Jensen's alpha (excess return) according to the Single Index Model.

Usage

```
alpha_jensens(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-risk-adjusted-measures/jensens-alpha

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

beta

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(alpha_jensens(portfolio),alpha_jensens(positionGOOG),alpha_jensens(positionAAPL))
plot(alpha_jensens(portfolio),alpha_jensens(positionGOOG),alpha_jensens(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title="Jensen's Alpha")
```

```
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(alpha_jensens(positionC),alpha_jensens(positionGOOG),alpha_jensens(positionAAPL))
plot(alpha_jensens(positionC),alpha_jensens(positionGOOG),alpha_jensens(positionAAPL),
legend=c('C','GOOG','AAPL'),title="Jensen's Alpha")

## End(Not run)
```

`beta` *Beta*

Description

Computes portfolio or position beta (market sensitivity) according to the Single Index Model.

Usage

```
## S4 method for signature 'portfolio'
beta(a)
## S4 method for signature 'position'
beta(a)
```

Arguments

a	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-risk-measures/beta.php

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(beta(portfolio),beta(positionGOOG),beta(positionAAPL))
plot(beta(portfolio),beta(positionGOOG),beta(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Beta')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(beta(positionC),beta(positionGOOG),beta(positionAAPL))
plot(beta(positionC),beta(positionGOOG),beta(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Beta')

## End(Not run)
```

`calmar_ratio` *Calmar Ratio*

Description

Computes Calmar ratio (cumulative return to maximum drawdown).

Usage

```
calmar_ratio(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-adjusted-measures/calmar-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

sharpe_ratio sortino_ratio omega_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(calmar_ratio(portfolio),calmar_ratio(positionGOOG),calmar_ratio(positionAAPL))
plot(calmar_ratio(portfolio),calmar_ratio(positionGOOG),calmar_ratio(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Calmar Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-19 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(calmar_ratio(positionC),calmar_ratio(positionGOOG),calmar_ratio(positionAAPL))
plot(calmar_ratio(positionC),calmar_ratio(positionGOOG),calmar_ratio(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Calmar Ratio')

## End(Not run)
```

`compute` *Compute Metrics*

Description

Metric object is not evaluated until compute() method is called on it. Method would display calculation progress and would use Metric object's disk cache to store any computational results obtained in the process.

Usage

```
compute(...)
```

Arguments

...	One or multiple objects of class Metric.

Value

One or multiple objects of class Metric.

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(variance(portfolio),variance(positionGOOG),variance(positionAAPL))
result[[1]][1:10,]

result=compute(variance(portfolio)-variance(positionGOOG))
result[[1]][1:10,]

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(variance(positionC),variance(positionGOOG),variance(positionAAPL))
result[[1]][1:10,]

## End(Not run)
```

---

correlation *Correlation*

Description

Computes correlation between positionA and positionB.

Usage

```
## S4 method for signature 'portfolio,missing'
correlation(positionA)
## S4 method for signature 'position,position'
correlation(positionA,positionB)
```

Arguments

positionA	Position object created using position_add( ) function
positionB	Position object created using position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(correlation(positionGOOG,positionAAPL))
plot(correlation(positionGOOG,positionAAPL),title='Correlation')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(correlation(positionC,positionAAPL))
plot(correlation(positionC,positionAAPL),title='Correlation')

## End(Not run)
```

covariance *Covariance*

Description

Computes covariance between positionA and positionB.

Usage

```
## S4 method for signature 'portfolio,missing'
covariance(positionA)
## S4 method for signature 'position,position'
covariance(positionA,positionB)
```

Arguments

positionA	Position object created using position_add( ) function
positionB	Position object created using position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(covariance(positionGOOG,positionAAPL))
plot(covariance(positionGOOG,positionAAPL),title='Covariance')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
```

```
result=compute(covariance(positionC,positionAAPL))
plot(covariance(positionC,positionAAPL),title='Covariance')

## End(Not run)
```

`create_metric` *Create metric class*

Description

Creates metric object using data provided in matrix or xts format.

Usage

```
## S4 method for signature 'matrix,character'
create_metric(metricData,symbol)
## S4 method for signature 'xts,character'
create_metric(metricData,symbol)
```

Arguments

`metricData`	Matrix of (time, value) rows or an xts object. Time should be in POSIX format expressed in milliseconds.
`symbol`	Unique identifier of the instrument

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
AAPL=create_metric(aapl.data,'AAPL')
GOOG=create_metric(goog.data,'GOOG')
plot(AAPL,GOOG,legend=c('AAPL','GOOG'))

## End(Not run)
```

cumulant *N-th Cumulant*

Description

Computes N-th cumulant of portfolio return distribution.

Usage

```
cumulant(asset,order)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
order	moment order (3 or 4).

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

moment

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(cumulant(portfolio, 3),cumulant(positionGOOG, 3),cumulant(positionAAPL, 3))
plot(cumulant(portfolio, 3),cumulant(positionGOOG, 3),cumulant(positionAAPL, 3),
legend=c('Portfolio','GOOG','AAPL'),title='3-th Cumulant')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
```

```
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(cumulant(positionC, 4),cumulant(positionGOOG, 4),cumulant(positionAAPL, 4))
plot(cumulant(positionC, 4),cumulant(positionGOOG, 4),cumulant(positionAAPL, 4),
legend=c('C','GOOG','AAPL'),title='4-th Cumulant')

## End(Not run)
```

`dist_density` *Probability Density Function of Portfolio Returns*

Description

Computes probability density of portfolio returns for a given interval (pValueLeft, pValueRight) at nPoints of approximation. Probability density is computed based on a "densityModel" specified in portfolio_settings( ) method.

Usage

```
dist_density(asset,pValueLeft,pValueRight,nPoints,addNormalDensity)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
`pValueLeft`	Left limit of probability density value in decimals.
`pValueRight`	Right limit of probability density value in decimals.
`nPoints`	Number of approximation points for the PDF function.
`addNormalDensity`	Flag used to add normal density to the final result. Defaults to FALSE.

Value

List of probability density values

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:

data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
```

```
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
util_plotDensity(dist_density(portfolio,pValueLeft=0.2,pValueRight=0.8,nPoints=100,
addNormalDensity=TRUE))

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
util_plotDensity(dist_density(portfolio,pValueLeft=0,pValueRight=1,nPoints=100,
addNormalDensity=TRUE))

## End(Not run)
```

`downside_variance` *Downside Variance*

Description

Computes downside variance of portfolio returns.

Usage

```
downside_variance(asset, thresholdReturn)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
`thresholdReturn`	Return value to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/downside-variance

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

upside_variance

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(downside_variance(portfolio,0.05),downside_variance(positionGOOG,0.05),
downside_variance(positionAAPL,0.05))
plot(downside_variance(portfolio,0.05),downside_variance(positionGOOG,0.05),
downside_variance(positionAAPL,0.05),legend=c('Portfolio','GOOG','AAPL'),
title='Downside Variance')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(downside_variance(positionC,0.05),downside_variance(positionGOOG,0.05),
downside_variance(positionAAPL,0.05))
plot(downside_variance(positionC,0.05),downside_variance(positionGOOG,0.05),
downside_variance(positionAAPL,0.05),legend=c('C','GOOG','AAPL'),
title='Downside Variance')

## End(Not run)
```

---

down_capture_ratio *Down Capture Ratio*

Description

Computes down capture ratio of a portfolio.

Usage

```
down_capture_ratio(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-return-measures/down-capture-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

up_capture_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(down_capture_ratio(portfolio),down_capture_ratio(positionGOOG),
down_capture_ratio(positionAAPL))
plot(down_capture_ratio(portfolio),down_capture_ratio(positionGOOG),
down_capture_ratio(positionAAPL),legend=c('Portfolio','GOOG','AAPL'),
title='Down Capture Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(down_capture_ratio(positionC),down_capture_ratio(positionGOOG),
down_capture_ratio(positionAAPL))
plot(down_capture_ratio(positionC),down_capture_ratio(positionGOOG),
down_capture_ratio(positionAAPL),legend=c('C','GOOG','AAPL'),
title='Down Capture Ratio')

## End(Not run)
```

down_number_ratio *Down Number Ratio*

Description

Computes down number ratio of a portfolio.

Usage

```
down_number_ratio(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-return-measures/down-number-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

up_number_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(down_number_ratio(portfolio),down_number_ratio(positionGOOG),
down_number_ratio(positionAAPL))
plot(down_number_ratio(portfolio),down_number_ratio(positionGOOG),down_number_ratio(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Down Number Ratio')

dateStart = "2014-11-17 09:30:00"
```

```
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(down_number_ratio(positionC),down_number_ratio(positionGOOG),
down_number_ratio(positionAAPL))
plot(down_number_ratio(positionC),down_number_ratio(positionGOOG),down_number_ratio(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Down Number Ratio')

## End(Not run)
```

`down_percentage_ratio` *Down Percentage Ratio*

Description

Computes down percentage ratio of portfolio returns.

Usage

```
down_percentage_ratio(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-return-measures/down-percentage-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

up_percentage_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(down_percentage_ratio(portfolio),down_percentage_ratio(positionGOOG),
down_percentage_ratio(positionAAPL))
plot(down_percentage_ratio(portfolio),down_percentage_ratio(positionGOOG),
down_percentage_ratio(positionAAPL),legend=c('Portfolio','GOOG','AAPL'),
title='Down Percentage Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(down_percentage_ratio(positionC),down_percentage_ratio(positionGOOG),
down_percentage_ratio(positionAAPL))
plot(down_percentage_ratio(positionC),down_percentage_ratio(positionGOOG),
down_percentage_ratio(positionAAPL),legend=c('C','GOOG','AAPL'),
title='Down Percentage Ratio')

## End(Not run)
```

`expected_downside_return` *Expected Downside Return*

Description

Computes portfolio cumulative expected return below a certain threshold.

Usage

```
expected_downside_return(asset, thresholdReturn)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
`thresholdReturn`	Return value to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-return-measures/expected-downside-return

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

expected_upside_return

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(expected_downside_return(portfolio,0.05),
expected_downside_return(positionGOOG,0.05),expected_downside_return(positionAAPL,0.05))
plot(expected_downside_return(portfolio,0.05),expected_downside_return(positionGOOG,0.05),
expected_downside_return(positionAAPL,0.05),legend=c('Portfolio','GOOG','AAPL'),
title='Expected Downside Return')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(expected_downside_return(positionC,0.05),
expected_downside_return(positionGOOG,0.05),expected_downside_return(positionAAPL,0.05))
plot(expected_downside_return(positionC,0.05),expected_downside_return(positionGOOG,0.05),
expected_downside_return(positionAAPL,0.05),legend=c('C','GOOG','AAPL'),
title='Expected Downside Return')

## End(Not run)
```

expected_return *Expected Return*

Description

Computes portfolio cumulative expected return.

Usage

```
expected_return(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-return-measures/expected-return

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(expected_return(portfolio),expected_return(positionGOOG),
expected_return(positionAAPL))
plot(expected_return(portfolio),expected_return(positionGOOG),expected_return(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Expected Return')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
```

```
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(expected_return(positionC),expected_return(positionGOOG),
expected_return(positionAAPL))
plot(expected_return(positionC),expected_return(positionGOOG),expected_return(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Expected Return')

## End(Not run)
```

`expected_shortfall` *Expected Shortfall*

Description

Computes portfolio conditional Value-at-Risk (Expected Tail Loss) at a given confidence interval. Computation employs distribution's skewness and kurtosis to account for non-normality.

Usage

```
expected_shortfall(asset, confidenceInterval)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
`confidenceInterval`	Confidence interval (in decimals) to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/tail-risk-measures/cvar

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

value_at_risk

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(expected_shortfall(portfolio,0.95),expected_shortfall(positionGOOG,0.95),
expected_shortfall(positionAAPL,0.95))
plot(expected_shortfall(portfolio,0.95),expected_shortfall(positionGOOG,0.95),
expected_shortfall(positionAAPL,0.95),legend=c('Portfolio','GOOG','AAPL'),
title='Expected Shortfall')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(expected_shortfall(positionC,0.95),expected_shortfall(positionGOOG,0.95),
expected_shortfall(positionAAPL,0.95))
plot(expected_shortfall(positionC,0.95),expected_shortfall(positionGOOG,0.95),
expected_shortfall(positionAAPL,0.95),legend=c('C','GOOG','AAPL'),
title='Expected Shortfall')

## End(Not run)
```

`expected_upside_return` *Expected Upside Return*

Description

Computes portfolio cumulative expected return above a certain threshold.

Usage

```
expected_upside_return(asset, thresholdReturn)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
`thresholdReturn`	Return value to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-return-measures/expected-upside-return

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

expected_downside_return

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(expected_upside_return(portfolio,0.05),
expected_upside_return(positionGOOG,0.05),expected_upside_return(positionAAPL,0.05))
plot(expected_upside_return(portfolio,0.05),expected_upside_return(positionGOOG,0.05),
expected_upside_return(positionAAPL,0.05),legend=c('Portfolio','GOOG','AAPL'),
title='Expected Upside Return')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(expected_upside_return(positionC,0.05),
expected_upside_return(positionGOOG,0.05),expected_upside_return(positionAAPL,0.05))
plot(expected_upside_return(positionC,0.05),expected_upside_return(positionGOOG,0.05),
expected_upside_return(positionAAPL,0.05),legend=c('C','GOOG','AAPL'),
title='Expected Upside Return')

## End(Not run)
```

forecast-class *Class "forecast"*

Description

Container class for storing forecast model and it's parameters

Slots

java: Object of class "jobjRef" ~~

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
showClass("forecast")
```

forecast_apply *Forecast Apply*

Description

Runs forecasting algorithm on a configured forecast object.

Usage

```
forecast_apply(forecast)
```

Arguments

forecast	Object of type forecast created using forecast_builder( ) method

Value

Forecast object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '360s',
                   resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)

forecastVariance_1=forecast_builder(variance(positionAAPL))
# plot(forecast_apply(forecastVariance),variance(positionAAPL),legend=c('Forecast','Simple'))

forecastVariance_2=forecast_builder(variance(positionAAPL),window="1d")
plot(forecast_apply(forecastVariance_1),forecast_apply(forecastVariance_2),
   variance(positionAAPL),legend=c('Forecast,window=20d','Forecast,window=1d','Simple'))

## End(Not run)
```

`forecast_builder` *Forecast builder*

Description

Create object of class forecast

Usage

```
forecast_builder(asset,model=c("EWMA", "HAR"), window="20d",
 step = "1d", transform = c("log", "none"),
seasonalityInterval="none",updateInterval="1m",valueType="forecast")
```

Arguments

`asset`	Object of class portfolio or position created using portfolio_create( ) or position_add( ) methods respectively
`model`	Forecast model to be used: • "EWMA" - exponentially-weighted moving average, • "HAR" - heterogeneous autoregresion
`window`	Rolling window length for forecast model. Observations outside of the forecast window are forgotten. Available interval values are: "Xs" - seconds, "Xm" - minutes, "Xh" - hours, "Xd" - trading days (6.5 hours in a trading day), "Xw" - weeks (5 trading days in 1 week), "Xmo" - month (21 trading day in 1 month), "Xy" - years (256 trading days in 1 year).

`step` Look-ahead forecast interval. Available interval values are: "Xs" - seconds, "Xm" - minutes, "Xh" - hours, "Xd" - trading days (6.5 hours in a trading day), "Xw" - weeks (5 trading days in 1 week), "Xmo" - month (21 trading day in 1 month), "Xy" - years (256 trading days in 1 year)

`transform` Transform applied to dependent and independent variables: "log" - logarithmic transform, "none" - no transform

`seasonalityInterval` Seasonality interval to be used in forecast model. Available interval values are: "Xs" - seconds, "Xm" - minutes, "Xh" - hours, "Xd" - trading days (6.5 hours in a trading day), "Xw" - weeks (5 trading days in 1 week), "Xmo" - month (21 trading day in 1 month), "Xy" - years (256 trading days in 1 year)

`updateInterval` Update interval for forecast estimates. Available interval values are: "Xs" - seconds, "Xm" - minutes, "Xh" - hours, "Xd" - trading days (6.5 hours in a trading day), "Xw" - weeks (5 trading days in 1 week), "Xmo" - month (21 trading day in 1 month), "Xy" - years (256 trading days in 1 year)

`valueType` Value returned from the forecast model:

- "forecast" - value of forecasted variable,
- "error" - residual error,
- "coef_n" - value of n-th coefficient (e.g. "coef_2")

Value

Object of class forecast

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '360s',
                   resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)

forecastVariance_1=forecast_builder(variance(positionAAPL))
# plot(forecast_apply(forecastVariance),variance(positionAAPL),legend=c('Forecast','Simple'))

forecastVariance_2=forecast_builder(variance(positionAAPL),window="1d")
plot(forecast_apply(forecastVariance_1),forecast_apply(forecastVariance_2),
   variance(positionAAPL),legend=c('Forecast,window=20d','Forecast,window=1d','Simple'))

## End(Not run)
```

forecast_input *Forecast Input*

Description

Adds given metric as an explanatory variable to forecast model

Usage

```
forecast_input(forecast,metric)
```

Arguments

forecast	Object of class forecast
metric	Object of class metric

Value

Object of class forecast

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '360s',
                   resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)

forecastVariance_1=forecast_builder(variance(positionAAPL),window="1d")
# plot(forecast_apply(forecastVariance),variance(positionAAPL),legend=c('Forecast','Simple'))

forecastVariance_2=forecast_builder(variance(positionAAPL),window="1d")
forecastVariance_2=forecast_input(forecastVariance_2,beta(positionAAPL))
plot(forecast_apply(forecastVariance_1),forecast_apply(forecastVariance_2),
   variance(positionAAPL),legend=c('Forecast,without input','Forecast,with input','Simple'))

## End(Not run)
```

`fractal_dimension` *Fractal Dimension*

Description

Computes portfolio fractal dimension as a weighted sum of fractal dimensions of its position returns.

Usage

```
fractal_dimension(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(fractal_dimension(portfolio),fractal_dimension(positionGOOG),
fractal_dimension(positionAAPL))
plot(fractal_dimension(portfolio),fractal_dimension(positionGOOG),
fractal_dimension(positionAAPL),legend=c('Portfolio','GOOG','AAPL'),title='Fractal Dimension')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(fractal_dimension(positionC),fractal_dimension(positionGOOG),
fractal_dimension(positionAAPL))
plot(fractal_dimension(positionC),fractal_dimension(positionGOOG),
```

```
fractal_dimension(positionAAPL),legend=c('C','GOOG','AAPL'),title='Fractal Dimension')

## End(Not run)
```

`gain_loss_variance_ratio` *Gain Loss Variance Ratio*

Description

Computes gain to loss variance ratio of portfolio returns.

Usage

```
gain_loss_variance_ratio(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/gain-loss-variance-ra

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

loss_variance gain_variance

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(gain_loss_variance_ratio(portfolio),gain_loss_variance_ratio(positionGOOG),
gain_loss_variance_ratio(positionAAPL))
```

```
plot(gain_loss_variance_ratio(portfolio),gain_loss_variance_ratio(positionGOOG),
gain_loss_variance_ratio(positionAAPL),legend=c('Portfolio','GOOG','AAPL'),
title='Gain Loss Variance Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(gain_loss_variance_ratio(positionC),gain_loss_variance_ratio(positionGOOG),
gain_loss_variance_ratio(positionAAPL))
plot(gain_loss_variance_ratio(positionC),gain_loss_variance_ratio(positionGOOG),
gain_loss_variance_ratio(positionAAPL),legend=c('C','GOOG','AAPL'),
title='Gain Loss Variance Ratio')

## End(Not run)
```

`gain_variance` *Gain Variance*

Description

Computes gain variance of portfolio returns.

Usage

```
gain_variance(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/gain-variance

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

loss_variance

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(gain_variance(portfolio),gain_variance(positionGOOG),gain_variance(positionAAPL))
plot(gain_variance(portfolio),gain_variance(positionGOOG),gain_variance(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Gain Variance')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(gain_variance(positionC),gain_variance(positionGOOG),gain_variance(positionAAPL))
plot(gain_variance(positionC),gain_variance(positionGOOG),gain_variance(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Gain Variance')

## End(Not run)
```

---

hurst_exponent *Hurst Exponent*

Description

Computes portfolio Hurst exponent as a weighted sum of the Hurst exponents of its position returns.

Usage

```
hurst_exponent(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(hurst_exponent(portfolio),hurst_exponent(positionGOOG),hurst_exponent(positionAAPL))
plot(hurst_exponent(portfolio),hurst_exponent(positionGOOG),hurst_exponent(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Hurst Exponent')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(hurst_exponent(positionC),hurst_exponent(positionGOOG),hurst_exponent(positionAAPL))
plot(hurst_exponent(positionC),hurst_exponent(positionGOOG),hurst_exponent(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Hurst Exponent')

## End(Not run)
```

---

information_ratio *Information Ratio*

Description

Computes information ratio of a portfolio.

Usage

```
information_ratio(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-risk-adjusted-measures/information-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(information_ratio(positionC),information_ratio(positionGOOG),
information_ratio(positionAAPL))
plot(information_ratio(positionC),information_ratio(positionGOOG),
information_ratio(positionAAPL),legend=c('C','GOOG','AAPL'),title='Information Ratio')

## End(Not run)
```

---

kurtosis *Kurtosis*

Description

Computes kurtosis of portfolio returns.

Usage

```
kurtosis(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/kurtosis

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

skewness

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(kurtosis(portfolio),kurtosis(positionGOOG),kurtosis(positionAAPL))
plot(kurtosis(portfolio),kurtosis(positionGOOG),kurtosis(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Kurtosis')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(kurtosis(positionC),kurtosis(positionGOOG),kurtosis(positionAAPL))
plot(kurtosis(positionC),kurtosis(positionGOOG),kurtosis(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Kurtosis')

## End(Not run)
```

---

log_return *Returns*

Description

Computes portfolio log_return from the beginning of the holding period.

Usage

```
log_return(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-return-measures/return

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(log_return(portfolio),log_return(positionGOOG),log_return(positionAAPL))
plot(log_return(portfolio),log_return(positionGOOG),log_return(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Returns')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(log_return(positionC),log_return(positionGOOG),log_return(positionAAPL))
plot(log_return(positionC),log_return(positionGOOG),log_return(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Returns')

## End(Not run)
```

---

loss_variance *Loss Variance*

Description

Computes loss variance of portfolio returns.

Usage

```
loss_variance(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/loss-variance

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

gain_variance

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(loss_variance(portfolio),loss_variance(positionGOOG),loss_variance(positionAAPL))
plot(loss_variance(portfolio),loss_variance(positionGOOG),loss_variance(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Loss Variance')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(loss_variance(positionC),loss_variance(positionGOOG),loss_variance(positionAAPL))
plot(loss_variance(positionC),loss_variance(positionGOOG),loss_variance(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Loss Variance')

## End(Not run)
```

max_drawdown *Max Drawdown*

Description

Computes maximum drawdown of portfolio returns.

Usage

```
max_drawdown(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/max-drawdown

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(max_drawdown(portfolio),max_drawdown(positionGOOG),max_drawdown(positionAAPL))
plot(max_drawdown(portfolio),max_drawdown(positionGOOG),max_drawdown(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Max Drawdown')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(max_drawdown(positionC),max_drawdown(positionGOOG),max_drawdown(positionAAPL))
```

```
plot(max_drawdown(positionC),max_drawdown(positionGOOG),max_drawdown(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Max Drawdown')

## End(Not run)
```

`metric-class` *Class* `"metric"`

Description

Class that incorporates the notion of "lazy" portfolio or position metric.

Slots

`java`**:** Object of class `"jobjRef"` ~~

Methods

plot `signature(x = "metric", y = "missing")`: Displays summary plot of a metric

plot `signature(x = "metric", y = "ANY")`: Displays summary plot of a metric

show `signature(object = "metric")`: Displays summary plot of a metric

`+ signature("metric","numeric")`: Adds number to all metric elements

`+ signature("metric","metric")`: Adds one metric to another element-by-element

`- signature("metric","numeric")`: Subtracts number from all metric elements

`- signature("metric","metric")`: Subtracts one metric from another element-by-element

`* signature("metric","numeric")`: Multiplies all metric elements by a number

`* signature("metric","metric")`: Multiplies one metric by another element-by-element

`/ signature("metric","numeric")`: Divides all metric elements by a number

`/ signature("metric","metric")`: Divides one metric by another element-by-element

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
showClass("metric")
```

mod_sharpe_ratio *Modified Sharpe Ratio*

Description

Computes Modified Sharpe Ratio of a portfolio at a given confidence interval. Computation employs distribution's skewness and kurtosis to account for non-normality.

Usage

```
mod_sharpe_ratio(asset, confidenceInterval)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
confidenceInterval	Confidence interval (in decimals) to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/tail-risk-measures/modified-sharpe-ratio.php

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

sharpe_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(mod_sharpe_ratio(portfolio,0.95),mod_sharpe_ratio(positionGOOG,0.95),
mod_sharpe_ratio(positionAAPL,0.95))
plot(mod_sharpe_ratio(portfolio,0.95),mod_sharpe_ratio(positionGOOG,0.95),
```

```
mod_sharpe_ratio(positionAAPL,0.95),legend=c('Portfolio','GOOG','AAPL'),
title='Modified Sharpe Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(mod_sharpe_ratio(positionC,0.95),mod_sharpe_ratio(positionGOOG,0.95),
mod_sharpe_ratio(positionAAPL,0.95))
plot(mod_sharpe_ratio(positionC,0.95),mod_sharpe_ratio(positionGOOG,0.95),
mod_sharpe_ratio(positionAAPL,0.95),legend=c('C','GOOG','AAPL'),
title='Modified Sharpe Ratio')

## End(Not run)
```

`moment` *N-th Order Central Moment*

Description

Computes N-th order central moment of portfolio return distribution.

Usage

```
moment(asset, order)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
`order`	moment order (from 1 to 4)

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(moment(portfolio, 3),moment(positionGOOG, 3),moment(positionAAPL, 3))
plot(moment(portfolio, 3),moment(positionGOOG, 3),moment(positionAAPL, 3),
legend=c('Portfolio','GOOG','AAPL'),title='3-th Order Central Moment')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(moment(positionC,4),moment(positionGOOG,4),moment(positionAAPL,4))
plot(moment(positionC,4),moment(positionGOOG,4),moment(positionAAPL,4),
legend=c('C','GOOG','AAPL'),title='4-th Order Central Moment')

## End(Not run)
```

`omega_ratio` *Omega Ratio*

Description

Computes Omega Ratio of a portfolio. Computation employs distribution's skewness and kurtosis to account for non-normality.

Usage

```
omega_ratio(asset, thresholdReturn)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
`thresholdReturn`	Return value to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-adjusted-measures/omega-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

sharpe_ratio sortino_ratio calmar_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(omega_ratio(portfolio,0.05),omega_ratio(positionGOOG,0.05),
omega_ratio(positionAAPL,0.05))
plot(omega_ratio(portfolio,0.05),omega_ratio(positionGOOG,0.05),
omega_ratio(positionAAPL,0.05),legend=c('Portfolio','GOOG','AAPL'),title='Omega Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(omega_ratio(positionC,0.05),omega_ratio(positionGOOG,0.05),
omega_ratio(positionAAPL,0.05))
plot(omega_ratio(positionC,0.05),omega_ratio(positionGOOG,0.05),
omega_ratio(positionAAPL,0.05),legend=c('C','GOOG','AAPL'),title='Omega Ratio')

## End(Not run)
```

---

optimization_constraint

Portfolio Optimization Constraint

Description

Adds portfolio optimization constraint restricting optimal portfolio's beta to a certain range.

Usage

```
optimization_constraint(optimizer,
        constraintMertic,
constraintType,
constraintValue)
```

Arguments

optimizer	Object of class optimizer created using optimization_goal( ) method
constraintMertic	Object of class metric to be used for computing optimization constraint
constraintType	Optimization constraint type: "=" - an equality constraint, ">=" - an inclusive lower bound constraint, "<=" - an inclusive upper bound constraint
constraintValue	Value to be used as a constraint boundary

Value

Object of class optimizer

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
optimizer=optimization_goal(log_return(portfolio),"max")
optimizer=optimization_constraint(optimizer,beta(portfolio),"<=",0.5)
optimalPortfolio=optimization_run(optimizer)
print(optimalPortfolio)

## End(Not run)
```

optimization_forecast *Porfolio Optimization - Set Optimization Forecast*

Description

Sets user-defined forecasted values for a given metric and returns modified optimizer object. By default value of the metric at time "t" is used as a forecast for "t+1".

Usage

```
optimization_forecast(optimizer,
     metricType,
   forecast)
```

Arguments

optimizer	Optimizer object created using optimization_goal( ) function
metricType	Choose forecast metric type: "Beta" - position beta, "Variance" - position variance, "ExpReturn" - position expected return, "Cumulant3" - position 3-th cumulant, "Cumulant4" - position 4-th cumulant
forecast	Object of class metric-class( ) or forecast-class( )

Value

Optimizer object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-12-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)

# Add position AAPL and GOOG to portfolio
positionAAPL=position_add(portfolio,"AAPL",100)
positionGOOG=position_add(portfolio,"GOOG",200)
portfolio_settings(portfolio,inputSamplingInterval='30m',resultsSamplingInterval='1d')

forecastVarianceAAPL=forecast_builder(variance(positionAAPL),model="HAR",step ='1d')
forecastVarianceGOOG=forecast_builder(variance(positionGOOG),model="HAR",step ='1d')
```

```
optimizer=optimization_goal(variance(portfolio),"min")
optimizer=optimization_constraint(optimizer,log_return(portfolio),">=",0)
optimizer=optimization_forecast(optimizer, "Variance", forecastVarianceAAPL)
optimizer=optimization_forecast(optimizer, "Variance", forecastVarianceGOOG)
optimalPortfolioWithHAR=optimization_run(optimizer)

optimizer=optimization_goal(variance(portfolio),"min")
optimizer=optimization_constraint(optimizer,log_return(portfolio),">=",0)
optimalPortfolioWithoutHAR=optimization_run(optimizer)

plot(variance(optimalPortfolioWithHAR),variance(optimalPortfolioWithoutHAR),title="Variance",
legend=c("With HAR Forecast","Without HAR Forecast"))

## End(Not run)
```

`optimization_goal` *Porfolio Optimization - Set Optimization Goal*

Description

Initializes portfolio optimization goals and returns newly constructed optimizer object.

Usage

```
optimization_goal(goal,
direction=c("min","max"),
approxError=1e-12,
optimumProbability=0.99)
```

Arguments

`goal`	Object of class metric to be used as an optimization goal
`direction`	choose direction of optimization algorithm: "min" - maximization goal, "max" - minimization goal
`approxError`	Estimation error in decimal points for computing optimal weights. Smaller value slows down optimization algorithm, but increases precision.
`optimumProbability`	Required probability level of a global optimum. Higher value slows down optimization algorithm, but increases chance of finding globally optimal solution.

Value

Optimizer object.

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
optimizer=optimization_goal(log_return(portfolio),"max")
optimizer=optimization_constraint(optimizer,beta(portfolio),"<=",0.5)
optimalPortfolio=optimization_run(optimizer)
print(optimalPortfolio)

## End(Not run)
```

`optimization_info` *Porfolio Optimization - Print Optimization Details*

Description

Prints optimization details (constraint violations, local optima and etc.) for an optimal portfolio.

Usage

```
optimization_info(portfolio)
```

Arguments

`portfolio`	Portfolio object returned by optimization_run( ) function

Value

Prints summary table.

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
```

```
optimizer=optimization_goal(log_return(portfolio),"max")
optimizer=optimization_constraint(optimizer,beta(portfolio),"<=",0.5)
optimalPortfolio=optimization_run(optimizer)
optimization_info(optimalPortfolio)

## End(Not run)
```

`optimization_run` *Portfolio Optimization - Runs Optimization Algorithm*

Description

Runs portfolio optimization procedure and returns corresponding optimal portfolio.

Usage

```
optimization_run(optimizer)
```

Arguments

`optimizer`	Optimizer object created using optimization_goal( ) function

Value

Optimal portfolio object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
optimizer=optimization_goal(log_return(portfolio),"max")
optimizer=optimization_constraint(optimizer,beta(portfolio),"<=",0.5)
optimization_run(optimizer)

## End(Not run)
```

optimizer-class *Class* "optimizer"

Description

Class for storing optimization goals and constraints.

Slots

java: Object of class "jobjRef" ~~

portfolio: Object of class "jobjRef" ~~

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
showClass("optimizer")
```

portfolio-class *Class* "portfolio"

Description

Container class for storing portfolio parameters

Slots

java: Object of class "jobjRef" ~~

optimization_info: Object of class "ANY" ~~

Methods

plot signature(x = "portfolio", y = "missing"): ...

plot signature(x = "portfolio", y = "ANY"): arguments:

- "font_size" - Baseline font size.
- "line_size" -Line thickness.
- "bw" -Black and white color scheme flag.
- "axis.text.size" -Axis font size.
- "title.size" -Title font size.

position_add signature(portfolio = "portfolio", symbol = "character", quantity = "ANY", time = "ANY", priceData = "matrix"): ...

position_add `signature(portfolio = "portfolio", symbol = "character", quantity = "ANY", time = "ANY", priceData = "missing")`: ...

position_add `signature(portfolio = "portfolio", symbol = "character", quantity = "ANY", time = "missing", priceData = "matrix")`: ...

position_add `signature(portfolio = "portfolio", symbol = "character", quantity = "ANY", time = "missing", priceData = "missing")`: ...

show `signature(object = "portfolio")`: ...

expected_return `signature("portfolio")`: ...

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
showClass("portfolio")
## Not run:
portfolio=portfolio_create(fromTime="2014-10-02 09:30:00", toTime="2014-10-02 16:00:00")
portfolio_settings(portfolio,resultsSamplingInterval='60s')
positionSPY=position_add(portfolio,'SPY',500)
positionC=position_add(portfolio,'C',600)
plot(portfolio,font_size=7,bw=T)

## End(Not run)
```

---

portfolioPlot-class *Class* "portfolioPlot"

Description

Class for storing chart settings for portfolio metrics.

Objects from the Class

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

Slots

data: Object of class "data.frame" ~~

start.data: Object of class "data.frame" ~~

option: Object of class "list" ~~

bw: Object of class "logical" ~~

breaks: Object of class "numeric" ~~

labels: Object of class "character" ~~

Methods

+ signature(e1 = "portfolioPlot", e2 = "portfolioPlot"): ...

plot signature(x = "portfolioPlot", y = "missing"): ...

show signature(object = "portfolioPlot"): ...

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
showClass("portfolioPlot")
```

---

portfolio_availableSymbols

Get All Symbol List

Description

Returns a list of symbols .

Usage

```
portfolio_availableSymbols()
```

Value

List of symbols, exchanges and descriptions

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
portfolio=portfolio_create(fromTime="2014-10-02 09:30:00", toTime="2014-10-02 16:00:00")
positionSPY=position_add(portfolio,'SPY',500)
positionC=position_add(portfolio,'C',600)
list=portfolio_availableSymbols()
list[1:10,]

## End(Not run)
```

`portfolio_create` *Creates new portfolio*

Description

Creates new empty portfolio.
To add positions use position_add( ).
To remove positions use position_remove( ).

Usage

```
portfolio_create(index, fromTime, toTime, priceDataIx)
```

Arguments

`index`	Index symbol that should be used in the Single Index Model. Defaults to "SPY".
`fromTime`	Start of market data interval in "yyyy-MM-dd hh:mm:ss" format when internal market data is used. Offset from last available date/time by N days is denoted as "t-N" (e.g. "t-7" denotes offset by 7 days).
`toTime`	End of market data interval in "yyyy-MM-dd hh:mm:ss" format when internal market data is used. Offset from last available date/time by N days is denoted as "t-N" (e.g. "t-7" denotes offset by 7 days).
`priceDataIx`	Vector of (time, price) observations for market index asset when external market data is used.

Value

portfolio object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

`position_add` `portfolio_settings` `position_remove`

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(alpha_exante(portfolio),alpha_exante(positionGOOG),alpha_exante(positionAAPL))
```

```
plot(alpha_exante(portfolio),alpha_exante(positionGOOG),alpha_exante(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Alpha')
print(portfolio)

portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',c(100,200),time=c(1412256601000,1412266600000),
priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',c(300,150),time=c(1412266600000,1412276600000),
priceData=aapl.data)
plot(expected_return(portfolio),title="Expected Return")

portfolio=portfolio_create(fromTime="2014-09-01 09:00:00", toTime="2014-09-14 16:00:00")
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionSPY=position_add(portfolio,'SPY',500)
positionC=position_add(portfolio,'C',600)
plot(expected_return(portfolio),title="Portfolio Expected Return")

portfolio=portfolio_create(fromTime="2014-10-02 09:30:00", toTime="2014-10-02 16:00:00")
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionSPY=position_add(portfolio,'SPY',500)
positionC=position_add(portfolio,'C',600)
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
position_add(portfolio,'AAPL',c(300,150),time=c(1412266600000,1412276600000),
priceData=aapl.data)
plot(expected_return(portfolio),title="Portfolio Expected Return")

portfolio=portfolio_create(fromTime="t-2", toTime="t")
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionSPY=position_add(portfolio,'SPY',500)
positionC=position_add(portfolio,'C',600)
plot(expected_return(portfolio),title="Portfolio Expected Return")

## End(Not run)
```

`portfolio_defaultSettings` *Portfolio Default Settings*

Description

Advanced settings that regulate how porfolio metrics are computed, returned and stored. Default:

- portfolioMetricsMode="portfolio",
- windowLength = "1d",
- holdingPeriodsOnly = FALSE,
- shortSalesMode = "lintner",
- synchronizationModel = TRUE,

- jumpsModel = "moments",
- noiseModel = TRUE,
- fractalPriceModel=TRUE,
- factorModel = "sim",
- densityModel="GLD",
- driftTerm=FALSE,
- resultsNAFilter= TRUE,
- resultsSamplingInterval = "1s",
- inputSamplingInterval="1s",
- timeScale="1d",
- txnCostPerShare=0,
- txnCostFixed=0

Usage

```
portfolio_defaultSettings(portfolio)
```

Arguments

portfolio	Portfolio object created using portfolio_create( ) function

Value

Void

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

portfolio_create

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
portfolio_defaultSettings(portfolio,
   inputSamplingInterval = '10s',
   resultsSamplingInterval = '10s')
portfolio_getSettings(portfolio)
portfolio_defaultSettings(portfolio)
```

```
portfolio_getSettings(portfolio)


## End(Not run)
```

`portfolio_getPosition` *Get position from portfolio*

Description

Returns position for a given symbol if this position is found inside a given portfolio

Usage

```
portfolio_getPosition(portfolio, symbol)
```

Arguments

`portfolio`	Object of class portfolio created using portfolio_create( ) function
`symbol`	Unique identifier of an instrument

Value

Object of class position for a given symbol, if found in the portfolio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

`portfolio_create`

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
position_add(portfolio,'GOOG',150)
positionGOOG=portfolio_getPosition(portfolio,'GOOG')

## End(Not run)
```

`portfolio_getSettings` *Get Portfolio Settings*

Description

Method returns active list of settings of a given portfolio.

Usage

```
portfolio_getSettings(portfolio)
```

Arguments

`portfolio`	Portfolio object created using portfolio_create( ) function

Value

List with portfolio settings.

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
portfolio_settings(portfolio,
   windowLength='600s',
   resultsSamplingInterval = '10s')
settings=portfolio_getSettings(portfolio)
settings

## End(Not run)
```

`portfolio_settings` *Portfolio Settings*

Description

Advanced settings that regulate how porfolio metrics are computed, returned and stored. Default:

- portfolioMetricsMode="portfolio",
- windowLength = "1d",
- holdingPeriodsOnly = FALSE,
- shortSalesMode = "lintner",
- synchronizationModel = TRUE,
- jumpsModel = "moments",
- noiseModel = TRUE,
- fractalPriceModel=TRUE,
- factorModel = "sim",
- densityModel="GLD",
- driftTerm=FALSE,
- resultsSamplingInterval = "1s",
- inputSamplingInterval="none",
- timeScale="1d",
- txnCostPerShare=0,
- txnCostFixed=0

Usage

```
portfolio_settings(portfolio,...)
```

Arguments

`portfolio`	Portfolio object created using portfolio_create( ) function
`...`	One of the following portfolio settings: • "portfolioMetricsMode" - Used to select method of computing portfolio metrics. Available modes are: "portfolio" - risk and performance metrics are computed based on the history of position rebalancing (see windowLength parameter) and should be used to backtest and compare trading strategies of different frequency and style, "price" - metrics are always computed without a history of previous rebalancing (classic interpretation). Defaults to "portfolio".

- "windowLength" - Rolling window length for metric estimations and position rebalancing history. Available interval values are: "Xs" - seconds, "Xm" - minutes, "Xh" - hours, "Xd" - trading days (6.5 hours in a trading day), "Xw" - weeks (5 trading days in 1 week), "Xmo" - month (21 trading day in 1 month), "Xy" - years (256 trading days in 1 year), "all" - all observations are used. Default value is "1d" - one trading day .
- "holdingPeriodsOnly - Used when portfolioMetricsMode = "portfolio". Defaults to FALSE, which means that trading strategy risk and performance metrics will be scaled to include intervals when trading strategy did not have market exposure. When TRUE, trading strategy metrics are scaled based on actual holding intervals when there was exposure to the market.
- "shortSalesMode" - Specifies how position weights are computed. Available modes are: "lintner" - the sum of absolute weights is equal to 1 (Lintner assumption), "markowitz" - the sum of weights must equal to 1 (Markowitz assumption). Defaults to "lintner", which implies that the sum of absolute weights is used to normalize investment weights.
- "synchronizationModel" - Synchronization mode for irregular spaced time series to be used when computing covariances. Defaults to TRUE, which implies that Hayashi-Yoshida algorithm would be used for synchronizing price observations.
- "jumpsModel" - Jump filtering mode when computing return statistics. Available modes are: "none" - price jumps are not filtered anywhere, "moments" - price jumps are filtered only when computing moments (variance, skewness, kurtosis) and derived metrics, "all" - price jumps are filtered everywhere. Defaults to "moments", which implies that only return moments and related metrics would be using jump-filtered returns in their calculations.
- "noiseModel" - Microstructure noise mode for intraday returns.Defaults to TRUE, which implies that microstructure effects are modeled and resulting HF noise is removed from metric calculations.
- "fractalPriceModel" - Fractal price model (fGBM) when time scaling return moments based on mono-fractality assumptions.Defaults to TRUE, which implies that computed Hurst exponent is used to scale return moments. When FALSE, price is assumed to follow regular GBM with Hurst exponent = 0.5.
- "factorModel" - Factors model for computing portfolio metrics. Available models are: "sim" - portfolio metrics are computed using the Single Index Model, "direct" - portfolio metrics are computed using portfolio value itself. Defaults to "sim", which implies that the Single Index Model is used to compute portfolio metrics.
- "densityModel" - Density approximation model of return distribution. Available models are: "GLD" - Generalized Lambda Distribution, "CORNER_FISHER" - Corner-Fisher approximation, "NORMAL" - Gaussian distribution. Defaults to "GLD", which would fit a broad range of distribution shapes.
- "driftTerm" - Drift term (expected return) presence when computing probability density approximation and related metrics (e.g. CVaR, Omega Ratio, etc.). Defaults to FALSE, which implies that distribution is centered around zero return.

- "resultsNAFilter" - Used to enable filtering of NA values in computed results. Defaults to TRUE, which implies that output results have all NA values removed.
- "resultsSamplingInterval" - Interval to be used for sampling computed results before returning them to the caller. Available interval values are: "Xs" - seconds, "Xm" - minutes, "Xh" - hours, "Xd" - trading days (6.5 hours in a trading day), "Xw" - weeks (5 trading days in 1 week), "Xmo" - month (21 trading day in 1 month), "Xy" - years (256 trading days in 1 year), "last" - last result in a series is returned, "none" - no sampling. Large sampling interval would produce smaller vector of results and would require less time spent on data transfer. Default value of "1s" indicates that data is returned for every second during trading hours.
- "inputSamplingInterval" - Interval to be used as a minimum step for sampling input prices. Available interval values are: "Xs" - seconds, "Xm" - minutes, "Xh" - hours, "Xd" - trading days (6.5 hours in a trading day), "Xw" - weeks (5 trading days in 1 week), "Xmo" - month (21 trading day in 1 month), "Xy" - years (256 trading days in 1 year), "none" - no sampling. Default value is "none", which indicates that no sampling is applied.
- "timeScale" - Interval to be used for scaling return distribution statistics and producing metrics forecasts at different horizons. Available interval values are: "Xs" - seconds, "Xm" - minutes, "Xh" - hours, "Xd" - trading days (6.5 hours in a trading day), "Xw" - weeks (5 trading days in 1 week), "Xmo" - month (21 trading day in 1 month), "Xy" - years (256 trading days in 1 year), "all" - actual interval specified in during portfolio creation. Default value is "1d" - one trading day.
- "txnCostPerShare" - Amount of transactional costs per share. Defaults to 0.
- "txnCostFixed" - Amount of fixed costs per transaction. Defaults to 0.

Value

Void

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

portfolio_create

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,
   windowLength='3600s',
```

```
   holdingPeriodsOnly=TRUE,
   resultsSamplingInterval = '10s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
print(portfolio)

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
plot(portfolio)

## End(Not run)
```

`position-class` *Class* `"position"`

Description

Container class for storing position parameters.

Slots

`java` Reference to the corresponding Java position object

`portfolio` Portfolio object that includes this position.

`symbol` Unique identifier of the instrument.

Methods

plot `signature(x = "position", y = "missing")`: ...

plot `signature(x = "position", y = "ANY")`: ...

show `signature(object = "position")`: ...

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
showClass("position")
```

position_add *Add position in portfolio*

Description

Adds position to an existing portfolio.

Usage

```
position_add(portfolio, symbol, quantity, time,priceData)
```

Arguments

portfolio	Portfolio object created using portfolio_create( ) function
symbol	Unique identifier of the instrument
quantity	One dimensional vector of position quantities or an integer number if quantity is constant
time	One dimensional vector of time values either as "yyyy-MM-dd hh:mm:ss" string or in milliseconds since the beginning of epoch.
priceData	Vector of (time, price) observations for market asset when external market data is used.

Value

Void

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

portfolio_create

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(alpha_exante(portfolio),alpha_exante(positionGOOG),alpha_exante(positionAAPL))
plot(alpha_exante(portfolio),alpha_exante(positionGOOG),alpha_exante(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Alpha')
```

```
print(portfolio)

portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',c(100,200),time=c(1412256601000,1412266600000),
priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',c(300,150),time=c(1412266600000,1412276600000),
priceData=aapl.data)
plot(expected_return(portfolio),title="Expected Return")

portfolio=portfolio_create(fromTime="2014-09-01 09:00:00", toTime="2014-09-14 16:00:00")
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionSPY=position_add(portfolio,'SPY',500)
positionC=position_add(portfolio,'C',600)
plot(expected_return(portfolio),title="Portfolio Expected Return")

portfolio=portfolio_create(fromTime="2014-10-02 09:30:00", toTime="2014-10-02 16:00:00")
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionSPY=position_add(portfolio,'SPY',500)
positionC=position_add(portfolio,'C',600)
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
position_add(portfolio,'AAPL',c(300,150),time=c(1412266600000,1412276600000),
priceData=aapl.data)
plot(expected_return(portfolio),title="Portfolio Expected Return")

portfolio=portfolio_create(fromTime="t-2", toTime="t")
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionSPY=position_add(portfolio,'SPY',500)
positionC=position_add(portfolio,'C',600)
plot(expected_return(portfolio),title="Portfolio Expected Return")

## End(Not run)
```

`position_list` *Portfolio Symbols*

Description

Returns a list of portfolio symbols with non-zero weights.

Usage

```
position_list(portfolio)
```

Arguments

`portfolio`	Portfolio object created using portfolio_create( ) function

Value

List of position symbols with non-zero weights

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
position_list(portfolio)

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
position_list(portfolio)

## End(Not run)
```

---

position_remove *Remove position from portfolio*

Description

Removes position from an existing portfolio.

Usage

```
## S4 method for signature 'portfolio,character'
position_remove(asset, symbol)
## S4 method for signature 'position,missing'
position_remove(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
symbol	Unique identifier of an instrument

Value

Void

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

position_add portfolio_create

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
position_remove(portfolio,'GOOG')
position_list(portfolio)

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
position_remove(portfolio,'C')
position_list(portfolio)

## End(Not run)
```

---

price *Price*

Description

Returns position price.

Usage

```
price(asset)
```

Arguments

asset	Position object created using position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(price(positionGOOG),price(positionAAPL))
plot(price(positionGOOG),price(positionAAPL),legend=c('GOOG','AAPL'),title='Price')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(price(positionC),price(positionGOOG),price(positionAAPL))
plot(price(positionC),price(positionGOOG),price(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Price')

## End(Not run)
```

---

profit *Profit*

Description

Computes portfolio monetary profit from the beginning of the holding period.

Usage

```
profit(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(profit(portfolio),profit(positionGOOG),profit(positionAAPL))
plot(profit(portfolio),profit(positionGOOG),profit(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Profit')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(profit(positionC),profit(positionGOOG),profit(positionAAPL))
plot(profit(positionC),profit(positionGOOG),profit(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Profit')

## End(Not run)
```

---

quantity *Quantity*

Description

Returns total number of shares associated with the given symbol in this portfolio.

Usage

```
quantity(asset)
```

Arguments

asset	Position object created using position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(quantity(positionGOOG),quantity(positionAAPL))
plot(quantity(positionGOOG),quantity(positionAAPL),legend=c('GOOG','AAPL'),title='Quantity')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(quantity(positionC),quantity(positionGOOG),quantity(positionAAPL))
plot(quantity(positionC),quantity(positionGOOG),quantity(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Quantity')

## End(Not run)
```

---

rachev_ratio *Rachev Ratio*

Description

Computes Rachev ratio of a portfolio at given confidence intervals. Computation employs distribution skewness and kurtosis to account for non-normality.

Usage

```
rachev_ratio(asset, confidenceIntervalA = 0.95,
  confidenceIntervalB = 0.95)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
confidenceIntervalA	Confidence interval (in decimals) to be used as a cut-off point in the numerator
confidenceIntervalB	Confidence interval (in decimals) to be used as a cut-off point in the denominator

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/tail-risk-measures/rachev-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(rachev_ratio(portfolio,0.95,0.95),rachev_ratio(positionGOOG,0.95,0.95),
rachev_ratio(positionAAPL,0.95,0.95))
plot(rachev_ratio(portfolio,0.95,0.95),rachev_ratio(positionGOOG,0.95,0.95),
rachev_ratio(positionAAPL,0.95,0.95),legend=c('Portfolio','GOOG','AAPL'),title='Rachev Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(rachev_ratio(positionC,0.95,0.95),rachev_ratio(positionGOOG,0.95,0.95),
rachev_ratio(positionAAPL,0.95,0.95))
plot(rachev_ratio(positionC,0.95,0.95),rachev_ratio(positionGOOG,0.95,0.95),
rachev_ratio(positionAAPL,0.95,0.95),legend=c('C','GOOG','AAPL'),title='Rachev Ratio')

## End(Not run)
```

`return_autocovariance` *Return Autocovariance*

Description

Computes autocovariance of position returns for a certain time lag.

Usage

```
return_autocovariance(asset, lag)
```

Arguments

`asset`	Position object created using position_add( ) function
`lag`	Time lag (in seconds) between observations in question.

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(return_autocovariance(positionGOOG,10),return_autocovariance(positionAAPL,10))
plot(return_autocovariance(positionGOOG,10),return_autocovariance(positionAAPL,10),
legend=c('GOOG','AAPL'),title='Return Autocovariance')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(return_autocovariance(positionC,10),return_autocovariance(positionGOOG,10),
return_autocovariance(positionAAPL,10))
plot(return_autocovariance(positionC,10),return_autocovariance(positionGOOG,10),
```

```
return_autocovariance(positionAAPL,10),legend=c('C','GOOG','AAPL'),title='Return Autocovariance')

## End(Not run)
```

`return_jump_size` *Return Jump Size*

Description

Computes relative magnitude of jumps in position returns.

Usage

```
return_jump_size(asset)
```

Arguments

`asset`	Position object created using position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-30 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=list(return_jump_size(positionC),return_jump_size(positionGOOG),
return_jump_size(positionAAPL))

## End(Not run)
```

set_quantity *Set Position Quantity*

Description

Sets new position quantity.

Usage

```
set_quantity(asset,quantity)
```

Arguments

asset	Position object created using position_add( ) function
quantity	One dimensional vector of position quantities or an integer number if quantity is constant

Value

Void

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
print(portfolio)

set_quantity(positionGOOG,400)
print(portfolio)

## End(Not run)
```

sharpe_ratio *Sharpe Ratio*

Description

Computes Sharpe Ratio of a portfolio.

Usage

```
sharpe_ratio(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-adjusted-measures/sharpe-ratio.php

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

mod_sharpe_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(sharpe_ratio(portfolio),sharpe_ratio(positionGOOG),sharpe_ratio(positionAAPL))
plot(sharpe_ratio(portfolio),sharpe_ratio(positionGOOG),sharpe_ratio(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Sharpe Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
```

```
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(sharpe_ratio(positionC),sharpe_ratio(positionGOOG),sharpe_ratio(positionAAPL))
plot(sharpe_ratio(positionC),sharpe_ratio(positionGOOG),sharpe_ratio(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Sharpe Ratio')

## End(Not run)
```

---

skewness *Skewness*

Description

Computes skewness of portfolio returns.

Usage

```
skewness(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/skewness

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

kurtosis

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(skewness(portfolio),skewness(positionGOOG),skewness(positionAAPL))
plot(skewness(portfolio),skewness(positionGOOG),skewness(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Skewness')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(skewness(positionC),skewness(positionGOOG),skewness(positionAAPL))
plot(skewness(positionC),skewness(positionGOOG),skewness(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Skewness')

## End(Not run)
```

---

sortino_ratio *Sortina Ratio*

Description

Computes Sortino ratio of a portfolio.

Usage

```
sortino_ratio(asset, thresholdReturn)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
thresholdReturn	Return value to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-adjusted-measures/sortino-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

sharpe_ratio calmar_ratio omega_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(sortino_ratio(portfolio,0.05),sortino_ratio(positionGOOG,0.05),
sortino_ratio(positionAAPL,0.05))
plot(sortino_ratio(portfolio,0.05),sortino_ratio(positionGOOG,0.05),
sortino_ratio(positionAAPL,0.05),legend=c('Portfolio','GOOG','AAPL'),title='Sortina Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(sortino_ratio(positionC,0.05),sortino_ratio(positionGOOG,0.05),
sortino_ratio(positionAAPL,0.05))
plot(sortino_ratio(positionC,0.05),sortino_ratio(positionGOOG,0.05),
sortino_ratio(positionAAPL,0.05),legend=c('C','GOOG','AAPL'),title='Sortina Ratio')

## End(Not run)
```

---

starr_ratio *STARR Ratio*

Description

Computes Stable Tail Adjusted Return Ratio (STARR) of a portfolio at a given confidence interval. Computation employs distribution's skewness and kurtosis to account for non-normality.

Usage

```
starr_ratio(asset, confidenceInterval)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
confidenceInterval	Confidence interval (in decimals) to be used as a cut-off point.

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/tail-risk-measures/starr-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(starr_ratio(portfolio,0.95),starr_ratio(positionGOOG,0.95),
starr_ratio(positionAAPL,0.95))
plot(starr_ratio(portfolio,0.95),starr_ratio(positionGOOG,0.95),
starr_ratio(positionAAPL,0.95),legend=c('Portfolio','GOOG','AAPL'),title='STARR Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(starr_ratio(positionC,0.95),starr_ratio(positionGOOG,0.95),
starr_ratio(positionAAPL,0.95))
plot(starr_ratio(positionC,0.95),starr_ratio(positionGOOG,0.95),
starr_ratio(positionAAPL,0.95),legend=c('C','GOOG','AAPL'),title='STARR Ratio')

## End(Not run)
```

`treynor_ratio` *Treynor Ratio*

Description

Computes Treynor Ratio of a portfolio.

Usage

```
treynor_ratio(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-risk-adjusted-measures/treynor-ratio.php

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(treynor_ratio(portfolio),treynor_ratio(positionGOOG),treynor_ratio(positionAAPL))
plot(treynor_ratio(portfolio),treynor_ratio(positionGOOG),treynor_ratio(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Treynor Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
```

```
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(treynor_ratio(positionC),treynor_ratio(positionGOOG),treynor_ratio(positionAAPL))
plot(treynor_ratio(positionC),treynor_ratio(positionGOOG),treynor_ratio(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Treynor Ratio')

## End(Not run)
```

`txn_costs` *Transactional Costs*

Description

Computes monetary value of accumulated portfolio transactional costs.

Usage

```
txn_costs(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:

data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s',txnCostFixed=100)
positionGOOG=position_add(portfolio,'GOOG',c(100,200),time=c(1412256601000,1412266600000),
priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',c(300,150),time=c(1412266600000,1412276600000),
priceData=aapl.data)
result=compute(txn_costs(portfolio),txn_costs(positionGOOG),txn_costs(positionAAPL))
plot(txn_costs(portfolio),txn_costs(positionGOOG),txn_costs(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Transactional Costs')

## End(Not run)
```

upside_downside_variance_ratio *Upside/Downside Variance Ratio*

Description

Computes upside to downside variance ratio of a portfolio.

Usage

```
upside_downside_variance_ratio(asset, thresholdReturn)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
thresholdReturn	Return value to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/upside-downside-varia

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

upside_variance downside_variance

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(upside_downside_variance_ratio(positionC,0.05),
upside_downside_variance_ratio(positionGOOG,0.05),
upside_downside_variance_ratio(positionAAPL,0.05))
plot(upside_downside_variance_ratio(positionC,0.05),
upside_downside_variance_ratio(positionGOOG,0.05),
```

```
upside_downside_variance_ratio(positionAAPL,0.05),
legend=c('C','GOOG','AAPL'),title='Upside/Downside Variance Ratio')

## End(Not run)
```

---

upside_variance *Upside Variance*

Description

Computes upside variance of a portfolio.

Usage

```
upside_variance(asset, thresholdReturn)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
thresholdReturn	Return value to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/upside-variance

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

downside_variance

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(upside_variance(positionC,0.05),upside_variance(positionGOOG,0.05),
```

```
upside_variance(positionAAPL,0.05))
plot(upside_variance(positionC,0.05),upside_variance(positionGOOG,0.05),
upside_variance(positionAAPL,0.05),legend=c('C','GOOG','AAPL'),title='Upside Variance')

## End(Not run)
```

`up_capture_ratio` *Up Capture Ratio*

Description

Computes up capture ratio of a portfolio.

Usage

```
up_capture_ratio(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-return-measures/up-capture-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

down_capture_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
```

```
result=compute(up_capture_ratio(portfolio),up_capture_ratio(positionGOOG),
up_capture_ratio(positionAAPL))
plot(up_capture_ratio(portfolio),up_capture_ratio(positionGOOG),
up_capture_ratio(positionAAPL),legend=c('Portfolio','GOOG','AAPL'),title='Up Capture Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(up_capture_ratio(positionC),up_capture_ratio(positionGOOG),
up_capture_ratio(positionAAPL))
plot(up_capture_ratio(positionC),up_capture_ratio(positionGOOG),
up_capture_ratio(positionAAPL),legend=c('C','GOOG','AAPL'),title='Up Capture Ratio')

## End(Not run)
```

`up_number_ratio` *Up Number Ratio*

Description

Computes up number ratio of a portfolio.

Usage

```
up_number_ratio(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-return-measures/up-number-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

down_number_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(up_number_ratio(portfolio),up_number_ratio(positionGOOG),
up_number_ratio(positionAAPL))
plot(up_number_ratio(portfolio),up_number_ratio(positionGOOG),
up_number_ratio(positionAAPL),legend=c('Portfolio','GOOG','AAPL'),title='Up Number Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(up_number_ratio(positionC),up_number_ratio(positionGOOG),
up_number_ratio(positionAAPL))
plot(up_number_ratio(positionC),up_number_ratio(positionGOOG),
up_number_ratio(positionAAPL),legend=c('C','GOOG','AAPL'),title='Up Number Ratio')

## End(Not run)
```

---

up_percentage_ratio *Up Percentage Ratio*

Description

Computes up percentage ratio of a portfolio.

Usage

```
up_percentage_ratio(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/relative-return-measures/up-percentage-ratio

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

down_percentage_ratio

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(up_percentage_ratio(portfolio),up_percentage_ratio(positionGOOG),
up_percentage_ratio(positionAAPL))
plot(up_percentage_ratio(portfolio),up_percentage_ratio(positionGOOG),
up_percentage_ratio(positionAAPL),legend=c('Portfolio','GOOG','AAPL'),title='Up Percentage Ratio')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(up_percentage_ratio(positionC),up_percentage_ratio(positionGOOG),
up_percentage_ratio(positionAAPL))
plot(up_percentage_ratio(positionC),up_percentage_ratio(positionGOOG),
up_percentage_ratio(positionAAPL),legend=c('C','GOOG','AAPL'),title='Up Percentage Ratio')

## End(Not run)
```

---

util_cleanCredentials *Clean API Credentials*

Description

Removes existing records of client API credentials from the system.

Usage

```
util_cleanCredentials()
```

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

portfolio_create

Examples

```
## Not run:
util_cleanCredentials()

## End(Not run)
```

---

util_colorScheme *Color scheme for charts*

Description

Custom color scheme for charts based on ggplot2 discrete scale.

Usage

```
util_colorScheme()
```

Value

Constructed ggplot2 discrete color scheme.

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
aapl.frame=data.frame(Data=aapl.data[,2],Time=as.POSIXct(aapl.data[,1]/1000,
origin = "1970-01-01", tz = "America/New_York"),legend='AAPL')
ggplot() + geom_line(data=aapl.frame, aes(x=Time,y=Data,col=legend))+util_colorScheme()

## End(Not run)
```

util_dateToPOSIXTime *Date To POSIX Time*

Description

Converts date strings to corresponding timestamps in milliseconds.

Usage

```
util_dateToPOSIXTime(time)
```

Arguments

time	One dimensional vector of time values in "yyyy-MM-dd hh:mm:ss" string format.

Value

Numerical vector of milliseconds since the beginning of the epoch.

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
time = "2014-11-17 09:30:00"
util_dateToPOSIXTime(time)

## End(Not run)
```

util_fillScheme *Fill scheme for charts*

Description

Custom fill scheme for charts based on ggplot2 discrete scale.

Usage

```
util_fillScheme()
```

Value

Constructed ggplot2 discrete color scheme.

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:

util_fillScheme()

## End(Not run)
```

---

util_getComputeTime *Remaining compute time*

Description

Returns remaining/maximum compute time in seconds. Maximum time is limited by client's current subscription plan until. Remaining time is reset to maximum time every day at 12am ET.

Usage

```
util_getComputeTime(time=c("timeMax","timeLeft"))
```

Arguments

time	One of the following option: "timeMax" - maximum available compute time, "timeLeft" - remaining compute time.

Value

Time value in seconds.

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
util_getComputeTime("timeMax")
util_getComputeTime("timeLeft")

## End(Not run)
```

util_ggplot *Converter of portfolioPlot to ggplot2*

Description

Converts a given portfolioPlot object to ggplot2 object.

Usage

```
util_ggplot(portfolioPlot)
```

Arguments

portfolioPlot	portfolioPlot object.

Value

ggplot2 object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
plot1=util_ggplot(plot(weight(positionAAPL),title="AAPL Weight"))
plot2=util_ggplot(plot(weight(positionC),title="C Weight"))
util_multiplot(plot1,plot2,cols=1)

## End(Not run)
```

util_line2d *Adds line chart to existing plot*

Description

Adds another line chart on the existing plot using a time series of metric values.

Usage

```
util_line2d(metric,legend="")
```

Arguments

metric	Time series of (time, value) returned by metric functions.
legend	Legend of the line

Value

plot

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
positionSPY=position_add(portfolio,'SPY',500,priceData=spy.data)
plot(log_return(positionGOOG),title="Positions returns",legend="GOOG")+
util_line2d(compute(log_return(positionAAPL))[[1]],legend="AAPL")+
util_line2d(compute(log_return(positionSPY))[[1]],legend="SPY")

## End(Not run)
```

util_multiplot *Multiple ggplot2 charts on one page*

Description

Plots several ggplot2 charts on one page.

Usage

```
util_multiplot(..., cols=1)
```

Arguments

...	Any number of ggplot2 objects.
cols	Number of plot columns.

Value

plot

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
plot1=util_ggplot(plot(weight(positionAAPL),title="AAPL Weight"))
plot2=util_ggplot(plot(weight(positionC),title="C Weight"))
util_multiplot(plot1,plot2,cols=1)

## End(Not run)
```

`util_plot2d` *Line plot of portfolio metric (for a time series)*

Description

Draws a new line plot using a time series of metric values.

Usage

```
util_plot2d(metric,
title=NULL,
subtitle=NULL,
font_size=10,
line_size=1.2,
bw=FALSE,
legend="",
axis.text.size=1.5,
title.size=2)
```

Arguments

`metric`	Time series of (time, value) returned by metric functions.
`title`	Plot title.
`subtitle`	Plot subtitle.
`font_size`	Baseline font size.
`line_size`	Line thickness.
`bw`	Black and white color scheme flag.
`legend`	Plot legend.
`axis.text.size`	Axis font size.
`title.size`	Title font size.

Value

plot

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
util_plot2d(compute(log_return(portfolio))[[1]],title="Portfolio return")

## End(Not run)
```

---

util_plot2df *Line plot of portfolio metric (for a dataframe)*

Description

Draws a new line plot using a a dataframe with one or many metric values.

Usage

```
util_plot2df(formula,
data,
title=NULL,
subtitle=NULL,
font_size=10,
line_size=1.2,
bw=FALSE,
axis.text.size=1.5,
title.size=2)
```

Arguments

formula	Formula that describes data titles to be plotted.
data	Dataframe with metric data. Dataframe must have a variable 'legend'. If you want to correctly display the time in POSIX format, name the variable as 'time'.
title	Plot title.
subtitle	Plot subtitle.
font_size	Baseline font size.
line_size	Line thickness.
bw	Black and white color scheme flag.
axis.text.size	Axis font size.
title.size	Title font size.

Value

plot

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
portfolio=portfolio_create(fromTime="2014-10-02 09:30:00", toTime="2014-10-02 16:00:00")
portfolio_settings(portfolio,resultsSamplingInterval='60s')
positionSPY=position_add(portfolio,'SPY',500)
positionC=position_add(portfolio,'C',600)
metricSPY=compute(log_return(positionSPY))[[1]]
metricSPY=data.frame(metricSPY,legend="SPY return")
util_plot2df(value~time,metricSPY,title="Return, SPY")

metricC=compute(log_return(positionC))[[1]]
metricC=data.frame(metricC,legend="C return")
metric=rbind(metricSPY,metricC)
util_plot2df(value~time,metric,title="Return")

data(aapl.data)
data(goog.data)
data(spy.data)
AAPLprice=data.frame(Price=aapl.data[,'Value'],time=aapl.data[,'Time'],legend='AAPL price')
GOOGprice=data.frame(Price=goog.data[,'Value'],time=goog.data[,'Time'],legend='GOOG price')
SPYprice=data.frame(Price=spy.data[,'Value'],time=spy.data[,'Time'],legend='SPY price')
price=rbind(AAPLprice,GOOGprice,SPYprice)
util_plot2df(Price~time,price,title="Stock prices")

## End(Not run)
```

---

util_plotDensity *Line plot of probability density*

Description

Plot probability density by produced by the dist_density( ) functions.

Usage

```
util_plotDensity(PDF, bw=FALSE)
```

Arguments

PDF	List of density values produced by portfolio or position pdf methods.
bw	Black and white color scheme flag.

Value

plot

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
util_plotDensity(dist_density(portfolio,pValueLeft=0.2,pValueRight=0.8,nPoints=100,
addNormalDensity=TRUE))

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
util_plotDensity(dist_density(portfolio,pValueLeft=0,pValueRight=1,nPoints=100,
addNormalDensity=TRUE))

## End(Not run)
```

---

util_plotTheme *Plot style settings for PortfolioEffect theme*

Description

Customizable plot style for PortfolioEffect color theme.

Usage

```
util_plotTheme(base_size = 10,
base_family = "sans",
horizontal = TRUE,
dkpanel = FALSE,
bw = FALSE,
axis.text.size=1.5,
title.size=2,
has.subtitle = FALSE)
```

Arguments

base_size	Base font size.
base_family	Base font family.
horizontal	Horizontal alignment flag.
dkpanel	dkpanel flag.
bw	Black and white color scheme flag.
axis.text.size	Axis font size.
title.size	Title font size.
has.subtitle	Subtitle flag.

Value

Void

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
aapl.frame=data.frame(Data=aapl.data[,2],Time=as.POSIXct(aapl.data[,1]/1000,
origin = "1970-01-01", tz = "America/New_York"),legend='AAPL')
ggplot() + geom_line(data=aapl.frame, aes(x=Time,y=Data,col=legend))+
util_plotTheme()+util_colorScheme()+util_fillScheme()

## End(Not run)
```

---

util_POSIXTimeToDate *POSIX Time To Date*

Description

Converts timestamps in milliseconds to corresponding date strings.

Usage

```
util_POSIXTimeToDate(time)
```

Arguments

time	One dimensional vector of milliseconds since the beginning of epoch.

Value

One dimensional vector of time values in "yyyy-MM-dd hh:mm:ss" string format.

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
util_POSIXTimeToDate(compute(kurtosis(portfolio))[[1]][,1])

## End(Not run)
```

---

util_setCredentials *Set API Credentials*

Description

Saves platform client log-in credentials. To retrieve your account credentials, please log in to your account or register for a free account at https://www.portfolioeffect.com/registration. This function should be called before any other requests to the server are made.

Usage

```
util_setCredentials(username,password,apiKey,hostname="quant07.portfolioeffect.com")
```

Arguments

username	User name
password	User password
apiKey	User api key
hostname	Server hostname

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

portfolio_create

Examples

```
#util_setCredentials("User Name", "User Password ", "User apiKey")
```

value *Value*

Description

Computes monetary value of a portfolio from the beginning of the holding period.

Usage

```
value(asset)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(value(portfolio),value(positionGOOG),value(positionAAPL))
plot(value(portfolio),value(positionGOOG),value(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Value')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(value(positionC),value(positionGOOG),value(positionAAPL))
plot(value(positionC),value(positionGOOG),value(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Value')
```

```
## End(Not run)
```

---

value_at_risk *Value-at-Risk*

Description

Computes portfolio Value-at-Risk at a given confidence interval. Computation employs distribution's skewness and kurtosis to account for non-normality.

Usage

```
value_at_risk(asset, confidenceInterval)
```

Arguments

asset	Portfolio or Position object created using portfolio_create( ) or position_add( ) function
confidenceInterval	Confidence interval (in decimals) to be used as a cut-off point

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/tail-risk-measures/var

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

expected_shortfall

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(value_at_risk(portfolio,0.95),value_at_risk(positionGOOG,0.95),
value_at_risk(positionAAPL,0.95))
```

```
plot(value_at_risk(portfolio,0.95),value_at_risk(positionGOOG,0.95),
value_at_risk(positionAAPL,0.95),legend=c('Portfolio','GOOG','AAPL'),title='Value-at-Risk')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(value_at_risk(positionC,0.95),value_at_risk(positionGOOG,0.95),
value_at_risk(positionAAPL,0.95))
plot(value_at_risk(positionC,0.95),value_at_risk(positionGOOG,0.95),
value_at_risk(positionAAPL,0.95),legend=c('C','GOOG','AAPL'),title='Value-at-Risk')

## End(Not run)
```

`variance` *Variance*

Description

Computes variance of portfolio returns.

Usage

```
variance(asset)
```

Arguments

`asset`	Portfolio or Position object created using portfolio_create( ) or position_add( ) function

Value

Metric object

Note

https://www.portfolioeffect.com/docs/glossary/measures/absolute-risk-measures/variance

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(variance(portfolio),variance(positionGOOG),variance(positionAAPL))
plot(variance(portfolio),variance(positionGOOG),variance(positionAAPL),
legend=c('Portfolio','GOOG','AAPL'),title='Variance')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(variance(positionC),variance(positionGOOG),variance(positionAAPL))
plot(variance(positionC),variance(positionGOOG),variance(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Variance')

## End(Not run)
```

`weight` *Weight*

Description

Computes ratio of a monetary position value to the monetary value of the whole portfolio. Expressed in decimal points of portfolio value.

Usage

```
weight(asset)
```

Arguments

`asset`	Position object created using position_add( ) function

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

Examples

```
## Not run:
data(aapl.data)
data(goog.data)
data(spy.data)
portfolio=portfolio_create(priceDataIx=spy.data)
portfolio_settings(portfolio,windowLength = '3600s',resultsSamplingInterval='60s')
positionGOOG=position_add(portfolio,'GOOG',100,priceData=goog.data)
positionAAPL=position_add(portfolio,'AAPL',300,priceData=aapl.data)
result=compute(weight(positionGOOG),weight(positionAAPL))
plot(weight(positionGOOG),weight(positionAAPL),legend=c('GOOG','AAPL'),title='Weight')

dateStart = "2014-11-17 09:30:00"
dateEnd = "2014-11-17 16:00:00"
portfolio=portfolio_create(dateStart,dateEnd)
portfolio_settings(portfolio,portfolioMetricsMode="price",windowLength = '3600s',
resultsSamplingInterval='60s')
positionAAPL=position_add(portfolio,'AAPL',100)
positionC=position_add(portfolio,'C',300)
positionGOOG=position_add(portfolio,'GOOG',150)
result=compute(weight(positionC),weight(positionGOOG),weight(positionAAPL))
plot(weight(positionC),weight(positionGOOG),weight(positionAAPL),
legend=c('C','GOOG','AAPL'),title='Weight')

## End(Not run)
```

---

weight_transform *Weight Transform*

Description

Returns transformed position weights for a given list of symbols. This metric is commonly used for creating position constraints during portfolio optimization.

Usage

```
weight_transform(portfolio,transformType=c('sum_abs_weight','equiweight'),symbols=NULL)
```

Arguments

portfolio	Portfolio object created using portfolio_create( ) function
transformType	Transform applied to position weights: "sum_abs_weights" - sum of absolute position weights, "equiweight" - equal position weights
symbols	List of symbols

Value

Metric object

Author(s)

Kostin Andrey <andrey.kostin@portfolioeffect.com>

See Also

portfolio_create

Examples

```
## Not run:
portfolio<-portfolio_create("SPY", "2014-11-19 09:30:00", "2014-11-19 16:00:00")
portfolio_settings(portfolio,portfolioMetricsMode="price",resultsSamplingInterval='1m')
position_AAPL=position_add(portfolio,"AAPL",1000)
position_GOOG=position_add(portfolio,"GOOG",1000)
position_SPY=position_add(portfolio,"SPY",1000)

optimizer=optimization_goal(variance(portfolio),direction="min")
optimizer=optimization_constraint(optimizer,value(portfolio),'=',10^9)
optimizer=optimization_constraint(optimizer,weight_transform(portfolio,
"sum_abs_weight",c(position_AAPL,position_GOOG)),">=",0.5)
plot(optimization_run(optimizer))

optimizer=optimization_goal(weight_transform(portfolio,"equiweight"))
plot(optimization_run(optimizer))

## End(Not run)
```

Index