Package ‘quarks’

November 21, 2021

Type Package

Title Simple Methods for Calculating Value at Risk and Expected Shortfall

Version 1.0.10

Description Enables the user to calculate Value at Risk (VaR) and Expected Shortfall (ES) by means of various types of historical simulation. Currently plain-, age-, volatility-weighted- and filtered historical simulation are implemented in this package. Volatility weighting can be carried out via an exponentially weighted moving average model (EWMA) or other GARCH-type models. The methods of the package are described in Gurrola-Perez, P. and Murphy, D. (2015) <https://EconPapers.repec.org/RePEc:boe:boeewp:0525>.

License GPL-3

Depends R (>= 2.10)

Imports rugarch, graphics, stats

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Sebastian Letmathe [aut, cre] (Paderborn University, Germany)

Maintainer Sebastian Letmathe <sebastian.letmathe@uni-paderborn.de>

Repository CRAN

Date/Publication 2021-11-21 17:00:02 UTC

R topics documented:

DAX30 ................................................................. 2
DJI ................................................................. 3
ewma ............................................................. 3
fhs ................................................................. 4
FTSE100 .......................................................... 5
German Stock Market Index (DAX) Financial Time Series Data

Description
A dataset that contains the daily financial data of the DAX from 2000 to December 2020 (currency in EUR).

Usage
DAX30

Format
A data frame with 5327 rows and 10 variables:

- **price.open** opening price (daily)
- **price.high** highest price (daily)
- **price.low** lowest price (daily)
- **price.close** closing price (daily)
- **volume** trading volume
- **price.adjusted** adjusted closing price (daily)
- **ref.date** date in format YY-MM-DD
- **ticker** ticker symbol
- **ret.adjusted.prices** returns obtained from the adj. closing prices
- **ret.closing.prices** returns obtained from the closing prices

Source
The data was obtained from Yahoo Finance.
DJI

Dow Jones Industrial Average (DJI) Financial Time Series Data

Description
A dataset that contains the daily financial data of the DJI from 2000 to December 2020 (currency in EUR).

Usage
DJI

Format
A data frame with 5283 rows and 10 variables:

- **price.open** opening price (daily)
- **price.high** highest price (daily)
- **price.low** lowest price (daily)
- **price.close** closing price (daily)
- **volume** trading volume
- **price.adjusted** adjusted closing price (daily)
- **ref.date** date in format YY-MM-DD
- **ticker** ticker symbol
- **ret.adjusted.prices** returns obtained from the adj. closing prices
- **ret.closing.prices** returns obtained from the closing prices

Source
The data was obtained from Yahoo Finance.

---

ewma

*Exponentially weighted moving average*

Description
Estimates volatility of a return series by means of an exponentially weighted moving average.

Usage
ewma(x, lambda = 0.94)
Arguments

- **x**: a numeric vector of asset returns
- **lambda**: decay factor for the calculation of weights; default is 0.94

Value

Returns a numerical vector `vol` that contains the computed volatility.

Examples

```r
prices <- DAX30$price.close
returns <- diff(log(prices))
date <- DAX30$ref.date[-1]
cvar <- ewma(x = returns, lambda = 0.94)
csig <- sqrt(cvar)
plot(date, csig, type = 'l',
     main = 'conditional standard deviations for the DAX30 return series')
```

Description

Calculates univariate Value at Risk and Expected Shortfall (Conditional Value at Risk) by means of filtered historical simulation. Volatility can be estimated with an exponentially weighted moving average or a GARCH-type model.

Usage

```r
fhs(x, p = 0.975, model = c("EWMA", "GARCH"), lambda = 0.94, nboot = NULL, ...)
```

Arguments

- **x**: a numeric vector of asset returns
- **p**: confidence level for VaR calculation; default is 0.975
- **model**: model for estimating conditional volatility; options are 'EWMA' and 'GARCH'; if model = 'GARCH', additional arguments can be adjusted via ...; default is 'EWMA'
- **lambda**: decay factor for the calculation of weights; default is 0.94
- **nboot**: size of bootstrap sample; must be a single non-NA integer value with nboot > 0; default is NULL
- **...**: additional arguments of the ugarchspec function from the rugarch-package; only applied if model = 'GARCH'; default settings for the arguments variance.model and mean.model are:
  - variance.model = list(model = 'sGARCH', garchOrder = c(1, 1))
  - mean.model = list(armaOrder = c(0, 0))
Value

Returns a list with the following elements:

- **VaR** Calculated Value at Risk
- **ES** Calculated Expected Shortfall (Conditional Value at Risk)
- **p** Confidence level for VaR calculation
- **garchmod** The model fit. Is the respective GARCH fit for model = "GARCH" (see rugarch documentation) and 'EWMA' for model = "EWMA"

Examples

```r
prices <- DAX30$price.close
returns <- diff(log(prices))
# volatility weighting via EWMA
ewma <- fhs(x = returns, p = 0.975, model = "EWMA", lambda = 0.94,
nboot = 10000)

# volatility weighting via GARCH
garch <- fhs(x = returns, p = 0.975, model = "GARCH", variance.model =
list(model = "sGARCH"), nboot = 10000)
print(garch)
```

**Description**

A dataset that contains the daily financial data of the FTSE from 2000 to December 2020 (currency in EUR).

**Usage**

FTSE100

**Format**

A data frame with 5291 rows and 10 variables:

- **price.open** opening price (daily)
- **price.high** highest price (daily)
- **price.low** lowest price (daily)
- **price.close** closing price (daily)
- **volume** trading volume
- **price.adjusted** adjusted closing price (daily)
- **ref.date** date in format YY-MM-DD
ticker  ticker symbol
ret.adjusted.prices  returns obtained from the adj. closing prices
ret.closing.prices  returns obtained from the closing prices

Source
The data was obtained from Yahoo Finance.

---

hs  Nonparametric calculation of univariate Value at Risk and Expected Shortfall

Description
Computes Value at Risk and Expected Shortfall (Conditional Value at Risk) by means of plain and age-weighted historical simulation.

Usage
hs(x, p = 0.975, method = c("age", "plain"), lambda = 0.98)

Arguments

x  a numeric vector of asset returns

p  confidence level for VaR calculation; default is 0.975

method  method to be used for calculation; default is 'plain'

lambda  decay factor for the calculation of weights; default is 0.98

Value
Returns a list with the following elements:

VaR  Calculated Value at Risk

ES  Calculated Expected Shortfall (Conditional Value at Risk)

p  Confidence level for VaR calculation

Examples
prices <- DAX30$price.close
returns <- diff(log(prices))
hs(x = returns, p = 0.975, method = 'plain')
hs(x = returns, p = 0.975, method = 'age', lambda = 0.98)
**HSI**

### Description

A dataset that contains the daily financial data of the HSI from 2000 to December 2020 (currency in EUR).

### Usage

HSI

### Format

A data frame with 5171 rows and 10 variables:

- **price.open**: opening price (daily)
- **price.high**: highest price (daily)
- **price.low**: lowest price (daily)
- **price.close**: closing price (daily)
- **volume**: trading volume
- **price.adjusted**: adjusted closing price (daily)
- **ref.date**: date in format YY-MM-DD
- **ticker**: ticker symbol
- **ret.adjusted.prices**: returns obtained from the adj. closing prices
- **ret.closing.prices**: returns obtained from the closing prices

### Source

The data was obtained from Yahoo Finance.

**NIK225**

### Description

A dataset that contains the daily financial data of the NIK from 2000 to December 2020 (currency in EUR).

### Usage

NIK225
Format

A data frame with 5145 rows and 10 variables:

- `price.open` opening price (daily)
- `price.high` highest price (daily)
- `price.low` lowest price (daily)
- `price.close` closing price (daily)
- `volume` trading volume
- `price.adjusted` adjusted closing price (daily)
- `ref.date` date in format YY-MM-DD
- `ticker` ticker symbol
- `ret.adjusted.prices` returns obtained from the adj. closing prices
- `ret.closing.prices` returns obtained from the closing prices

Source

The data was obtained from Yahoo Finance.

---

plot.quarks  
*Plot Method for the Package 'quarks'*

Description

This function regulates how objects created by the package quarks are plotted.

Usage

```r
## S3 method for class 'quarks'
plot(x, ...)
```

Arguments

- `x` an input object of class quarks.
- `...` additional arguments of the standard plot method.

Value

None
print.quarks  

Print Method for the Package 'quarks'

Description
This function regulates how objects created by the package quarks are printed.

Usage

```r
## S3 method for class 'quarks'
print(x, ...)
```

Arguments

- **x**: an input object of class quarks.
- **...**: included for compatibility; additional arguments will however not affect the output.

Value

None

rollcast  

Rolling one-step forecasts of Value at Risk and Expected Shortfall

Description
Computes rolling one-step forecasts of Value at Risk and Expected Shortfall (Conditional Value at Risk) by means of plain historical simulation age- and volatility-weighted historical simulation as well as filtered historical simulation.

Usage

```r
rollcast(
  x,
  p = 0.975,
  model = c("EWMA", "GARCH"),
  method = c("plain", "age", "vwhs", "fhs"),
  lambda = c(0.94, 0.98),
  nout = NULL,
  nwin = NULL,
  nboot = NULL,
...
)
```
Arguments

- **x**: a numeric vector of asset returns
- **p**: confidence level for VaR calculation; default is 0.975
- **model**: model for estimating conditional volatility; options are 'EWMA' and 'GARCH'; if model = 'GARCH', additional arguments can be adjusted via ...; default is 'EWMA'
- **method**: method to be used for calculation; default is 'plain'
- **lambda**: decay factor for the calculation of weights; default is 0.98 for method = 'age' and 0.94 for method = 'vwhs' or method = 'fhs'
- **nout**: number of out-of-sample observations; default is NULL
- **nwin**: window size for rolling one-step forecasting; default is NULL
- **nboot**: size of bootstrap sample; must be a single non-NA integer value with nboot > 0; default is NULL
- **...**: additional arguments of the ugarchspec function from the rugarch-package; only applied if model = 'GARCH'; default settings for the arguments variance.model and mean.model are:
  - **variance.model**: list(model = 'sGARCH', garchOrder = c(1, 1))
  - **mean.model**: list(armaOrder = c(0, 0))

Value

Returns a list with the following elements:

- **VaR**: Numerical vector containing out-of-sample forecasts of Value at Risk
- **ES**: Numerical vector containing out-of-sample forecasts of Expected Shortfall (Conditional Value at Risk)
- **xout**: Numerical vector containing out-of-sample returns
- **p**: Confidence level for VaR calculation
- **model**: Model for estimating conditional volatility
- **method**: Method to be used for calculation
- **nout**: Number of out-of-sample observations
- **nwin**: Window size for rolling one-step forecasting
- **nboot**: Size of bootstrap sample

Examples

```r
prices <- DAX30$price.close
returns <- diff(log(prices))
n <- length(returns)
nout <- 250 # number of obs. for out-of-sample forecasting
nwin <- 500 # window size for rolling forecasts
```
### Example 1 - plain historical simulation

```r
code1 <- rollcast(x = returns, p = 0.975, method = 'plain', nout = nout, nwin = nwin)
matplot(1:nout, cbind(-code1$xout, code1$VaR, code1$ES),
        type = 'h',
        xlab = 'number of out-of-sample obs.', ylab = 'losses, VaR and ES',
        main = 'Plain HS - 97.5% VaR and ES for the DAX30 return series')
```

### Example 2 - age weighted historical simulation

```r
code2 <- rollcast(x = returns, p = 0.975, method = 'age', nout = nout, nwin = nwin)
matplot(1:nout, cbind(-code2$xout, code2$VaR, code2$ES),
        type = 'h',
        xlab = 'number of out-of-sample obs.', ylab = 'losses, VaR and ES',
        main = 'Age weighted HS - 97.5% VaR and ES for the DAX30 return series')
```

### Example 3 - volatility weighted historical simulation - EWMA

```r
code3 <- rollcast(x = returns, p = 0.975, model = 'EWMA', method = 'vwhs', nout = nout, nwin = nwin)
matplot(1:nout, cbind(-code3$xout, code3$VaR, code3$ES),
        type = 'h',
        xlab = 'number of out-of-sample obs.', ylab = 'losses, VaR and ES',
        main = 'Vol. weighted HS (EWMA) - 97.5% VaR and ES for the DAX30 return series')
```

### Example 4 - volatility weighted historical simulation - GARCH

```r
code4 <- rollcast(x = returns, p = 0.975, model = 'GARCH', method = 'vwhs', nout = nout, nwin = nwin)
matplot(1:nout, cbind(-code4$xout, code4$VaR, code4$ES),
        type = 'h',
        xlab = 'number of out-of-sample obs.', ylab = 'losses, VaR and ES',
        main = 'Vol. weighted HS (GARCH) - 97.5% VaR and ES for the DAX30 return series')
```

### Example 5 - filtered historical simulation - EWMA

```r
code5 <- rollcast(x = returns, p = 0.975, model = 'EWMA', method = 'fhs', nout = nout, nwin = nwin, nboot = 10000)
matplot(1:nout, cbind(-code5$xout, code5$VaR, code5$ES),
        type = 'h',
        xlab = 'number of out-of-sample obs.', ylab = 'losses, VaR and ES',
        main = 'Filtered HS (EWMA) - 97.5% VaR and ES for the DAX30 return series')
```

### Example 6 - filtered historical simulation - GARCH

```r
code6 <- rollcast(x = returns, p = 0.975, model = 'GARCH', method = 'fhs', nout = nout, nwin = nwin, nboot = 10000)
matplot(1:nout, cbind(-code6$xout, code6$VaR, code6$ES),
        type = 'h',
        xlab = 'number of out-of-sample obs.', ylab = 'losses, VaR and ES',
        main = 'Filtered HS (GARCH) - 97.5% VaR and ES for the DAX30 return series')
```
**SP500**

*Standard and Poor's (SP500) Financial Time Series Data*

**Description**

A dataset that contains the daily financial data of the SP500 from 2000 to December 2020 (currency in EUR).

**Usage**

SP500

**Format**

A data frame with 5284 rows and 10 variables:

- **price.open** opening price (daily)
- **price.high** highest price (daily)
- **price.low** lowest price (daily)
- **price.close** closing price (daily)
- **volume** trading volume
- **price.adjusted** adjusted closing price (daily)
- **ref.date** date in format YY-MM-DD
- **ticker** ticker symbol
- **ret.adjusted.prices** returns obtained from the adj. closing prices
- **ret.closing.prices** returns obtained from the closing prices

**Source**

The data was obtained from Yahoo Finance.

---

**vwhs**

*Volatility weighted historical simulation*

**Description**

Calculates univariate Value at Risk and Expected Shortfall (Conditional Value at Risk) by means of volatility weighted historical simulation. Volatility can be estimated with an exponentially weighted moving average or a GARCH-type model.

**Usage**

vwhs(x, p = 0.975, model = c("EWMA", "GARCH"), lambda = 0.94, ...)

---
vwhs

Arguments

- **x**: a numeric vector of asset returns
- **p**: confidence level for VaR calculation; default is 0.975
- **model**: model for estimating conditional volatility; default is 'EWMA'
- **lambda**: decay factor for the calculation of weights; default is 0.94
- **...**: additional arguments of the ugarchspec function from the rugarch-package; the default settings for the arguments variance.model and mean.model are `list(model = 'sGARCH', garchOrder = c(1,1))` and `list(armaOrder = c(0,0))`, respectively

Value

Returns a list with the following elements:

- **VaR**: Calculated Value at Risk
- **ES**: Calculated Expected Shortfall (Conditional Value at Risk)
- **p**: Confidence level for VaR calculation
- **garchmod**: The model fit. Is the respective GARCH fit for `model = 'GARCH'` (see rugarch documentation) and 'EWMA' for `model = 'EWMA'

Examples

```r
prices <- DAX30$price.close
returns <- diff(log(prices))
# volatility weighting via EWMA
ewma <- vwhs(x = returns, p = 0.975, model = "EWMA", lambda = 0.94)
ewma
# volatility weighting via GARCH
garch <- vwhs(x = returns, p = 0.975, model = "GARCH", variance.model = list(model = "sGARCH"))
garch
```
Index

* datasets
  DAX30, 2
  DJI, 3
  FTSE100, 5
  HSI, 7
  NIK225, 7
  SP500, 12

DAX30, 2
DJI, 3

ewma, 3

fhs, 4
FTSE100, 5

hs, 6
HSI, 7

NIK225, 7

plot.quarks, 8
print.quarks, 9

rollcast, 9

SP500, 12

vwhs, 12