Package 'ffp'

July 14, 2021

Type Package

Title Stress Test Historical Scenarios with Fully Flexible Probabilities

Version 0.1.0

Description Implements numerical entropy-pooling for scenario analysis as described in Meucci, Attilio (2010) <doi:10.2139/ssrn.1696802>.

License MIT + file LICENSE

URL https://github.com/Reckziegel/FFP

BugReports https://github.com/Reckziegel/FFP/issues

Depends R (>= 2.10)

Imports assertthat (>= 0.2.1), dplyr (>= 1.0.6), forcats (>= 0.5.1), ggdist (>= 2.4.0), ggplot2 (>= 3.3.3), lubridate (>= 1.7.10), magrittr (>= 2.0.1), methods, mvtnorm (>= 1.1-1), NlcOptim (>= 0.6), pracma (>= 2.3.3), purrr (>= 0.3.4), rlang (>= 0.1.2), scales (>= 1.1.1), stats, tibble (>= 3.1.1), tidyr (>= 1.1.3), usethis (>= 0.2.1), vctrs (>= 0.3.7), xts (>= 0.12.1)

Suggests covr, knitr (>= 1.33), markdown, roxygen2, spelling, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Bernardo Reckziegel [aut, cre, cph]

Maintainer Bernardo Reckziegel <bernardo_cse@hotmail.com>

Repository CRAN

Date/Publication 2021-07-14 09:10:08 UTC

R topics documented:

`autoplot.ffp` *Inspection of a* `ffp` *object with ggplot2*

Description

Extends the `autoplot` method for the `ffp` class.

Usage

```
## S3 method for class 'ffp'
autoplot(object, color = TRUE, ...)

## S3 method for class 'ffp'
plot(object, ...)
```

Arguments

`object`	An objected of the `ffp` class.
`color`	A `logical` flag indicating whether (or not) the `color` argument should be added to the `ggplot2` aesthetics.
`...`	Additional arguments to be passed to `autoplot`.

Value

A `ggplot2` object.

Examples

```
library(ggplot2)

x <- exp_decay(EuStockMarkets, 0.001)
y <- exp_decay(EuStockMarkets, 0.01)

autoplot(x) +
  scale_color_viridis_c()
autoplot(y) +
  scale_color_viridis_c()
```

---

bind_probs *Stack Flexible Probabilities*

Description

This function mimics `dplyr` `bind`. It's useful if you have different `ffp` objects and want to stack them in the `tidy` (long) format.

Usage

```
bind_probs(...)
```

Arguments

`...`	`ffp` objects to combine.

Value

A tidy `tibble`.

The output adds two new columns:

- `rowid` (an `integer`) with the row number of each realization;
- `key` (a `factor`) that keeps track of the `ffp` inputs as separated objects.

See Also

`crisp` `exp_decay` `kernel_normal` `kernel_entropy` `double_decay`

Examples

```
library(ggplot2)
library(dplyr, warn.conflicts = FALSE)

x <- exp_decay(EuStockMarkets, 0.001)
y <- exp_decay(EuStockMarkets, 0.002)

bind_probs(x, y)
```

```
bind_probs(x, y) %>%
  ggplot(aes(x = rowid, y = probs, color = key)) +
  geom_line() +
  scale_color_viridis_d()
```

`bootstrap_scenarios` *Flexible Probabilities Driven Bootstrap*

Description

Resamples historical scenarios with flexible probabilities while keeping the empirical structure of the copulas intact.

Usage

```
bootstrap_scenarios(x, p, n)

## S3 method for class 'numeric'
bootstrap_scenarios(x, p, n)

## S3 method for class 'matrix'
bootstrap_scenarios(x, p, n)

## S3 method for class 'ts'
bootstrap_scenarios(x, p, n)

## S3 method for class 'xts'
bootstrap_scenarios(x, p, n)

## S3 method for class 'tbl'
bootstrap_scenarios(x, p, n)

## S3 method for class 'data.frame'
bootstrap_scenarios(x, p, n)
```

Arguments

`x`	A time series defining the scenario-probability distribution.
`p`	An object of the `ffp` class.
`n`	An `integer` scalar with the number of scenarios to be generated.

Details

The argument `x` is supposed to have the same size of `p`.

Value

A `tibble` with the number of rows equal to `n`.

Examples

```
set.seed(123)
ret <- diff(log(EuStockMarkets))
ew  <- rep(1 / nrow(ret), nrow(ret))

bootstrap_scenarios(x = ret, p = as_ffp(ew), n = 10)
```

`crisp` *Full Information by Market Conditioning*

Description

Give full weight to occurrences when a macroeconomic statement satisfies a logical condition.

Usage

```
crisp(x, lgl)

## Default S3 method:
crisp(x, lgl)

## S3 method for class 'numeric'
crisp(x, lgl)

## S3 method for class 'matrix'
crisp(x, lgl)

## S3 method for class 'ts'
crisp(x, lgl)

## S3 method for class 'xts'
crisp(x, lgl)

## S3 method for class 'data.frame'
crisp(x, lgl)

## S3 method for class 'tbl_df'
crisp(x, lgl)
```

Arguments

`x`	An univariate or a multivariate distribution.
`lgl`	A `logical` vector with TRUE's and FALSE's indicating which scenarios should considered.

Value

A numerical vector of class `ffp` with the new probabilities distribution.

See Also

`exp_decay` `kernel_normal`

Examples

```
library(ggplot2)
# invariance (stationarity)
ret <- diff(log(EuStockMarkets))

# full weight on scenarios where CAC operated above 2%
market_condition <- crisp(x = ret, ret[ , 3] > 0.02)
market_condition

autoplot(market_condition) +
  scale_color_viridis_c()
```

---

db *Dataset used in Historical Scenarios with Fully Flexible Probabilities (`matrix` format).*

Description

Dataset used in Historical Scenarios with Fully Flexible Probabilities (`matrix` format).

Usage

```
db
```

Format

An object of class `matrix` (inherits from `array`) with 1083 rows and 9 columns.

See Also

`db_tbl`

db_tbl — *Dataset used in Historical Scenarios with Fully Flexible Probabilities (tibble format).*

Description

Dataset used in Historical Scenarios with Fully Flexible Probabilities (tibble format).

Usage

```
db_tbl
```

Format

An object of class tbl_df (inherits from tbl, data.frame) with 1083 rows and 9 columns.

See Also

db

double_decay — *Flexible Probabilities using Partial Information*

Description

This function uses entropy-pooling to match different decay-factors on the covariance matrix.

Usage

```
double_decay(x, slow, fast)

## Default S3 method:
double_decay(x, slow, fast)

## S3 method for class 'numeric'
double_decay(x, slow, fast)

## S3 method for class 'matrix'
double_decay(x, slow, fast)

## S3 method for class 'ts'
double_decay(x, slow, fast)

## S3 method for class 'xts'
double_decay(x, slow, fast)
```

```
## S3 method for class 'tbl'
double_decay(x, slow, fast)

## S3 method for class 'data.frame'
double_decay(x, slow, fast)
```

Arguments

x	An univariate or a multivariate distribution.
slow	A number with the long half-life (slow decay) for the correlation matrix.
fast	A number with the short-life (high decay) for the volatility.

Value

A numerical vector of class ffp with the new probabilities distribution.

References

De Santis, G., R. Litterman, A. Vesval, and K. Winkelmann, 2003, Covariance matrix estimation, Modern investment management: an equilibrium approach, Wiley.

See Also

kernel_entropy half_life

Examples

```
library(ggplot2)

slow <- 0.0055
fast <- 0.0166
ret <- diff(log(EuStockMarkets))

dd <- double_decay(ret, slow, fast)
dd

autoplot(dd) +
  scale_color_viridis_c()
```

---

empirical_stats *Summary Statistics for Empirical Distributions*

Description

Computes the mean, standard deviation, skewness, kurtosis, Value-at-Risk (VaR) and Conditional Value-at-Risk CVaR) under flexible probabilities.

Usage

```
empirical_stats(x, p, level = 0.01)

## Default S3 method:
empirical_stats(x, p, level = 0.01)

## S3 method for class 'numeric'
empirical_stats(x, p, level = 0.01)

## S3 method for class 'matrix'
empirical_stats(x, p, level = 0.01)

## S3 method for class 'xts'
empirical_stats(x, p, level = 0.01)

## S3 method for class 'ts'
empirical_stats(x, p, level = 0.01)

## S3 method for class 'data.frame'
empirical_stats(x, p, level = 0.01)

## S3 method for class 'tbl_df'
empirical_stats(x, p, level = 0.01)
```

Arguments

x	A time series defining the scenario-probability distribution.
p	An object of the ffp class.
level	A number with the desired probability level. The default is level = 0.01.

Details

The data in x and p are expected to have the same number of rows (size).

Value

A tidy tibble with 3 columns:

- stat: a column with Mu, Std, Skew, Kurt, VaR and CVaR.
- name: the asset names.
- value: the computed value for each statistic.

Examples

```
library(dplyr, warn.conflicts = FALSE)
library(ggplot2)

ret <- diff(log(EuStockMarkets))
```

```
# with equal weights (standard scenario)
ew  <- rep(1 / nrow(ret), nrow(ret))
empirical_stats(x = ret, p = as_ffp(ew)) %>%
  ggplot(aes(x = name, y = value)) +
  geom_col() +
  facet_wrap(~stat, scales = "free") +
  labs(x = NULL, y = NULL)

# with ffp
exp_smooth <- exp_decay(ret, 0.015)
empirical_stats(ret, exp_smooth) %>%
  ggplot(aes(x = name, y = value)) +
  geom_col() +
  facet_wrap(~stat, scales = "free") +
  labs(x = NULL, y = NULL)
```

---

ens *Effective Number of Scenarios*

Description

Shows how many scenarios are effectively been considered when using flexible probabilities.

Usage

```
ens(p)
```

Arguments

p	An object of the ffp class.

Value

A single double.

Examples

```
set.seed(123)
p <- exp_decay(stats::rnorm(100), 0.01)

# ens is smaller than 100
ens(p)
```

exp_decay *Full Information by Exponential Decay*

Description

Exponential smoothing twists probabilities by giving relatively more weight to recent observations at an exponential rate.

Usage

```
exp_decay(x, lambda)

## Default S3 method:
exp_decay(x, lambda)

## S3 method for class 'numeric'
exp_decay(x, lambda)

## S3 method for class 'matrix'
exp_decay(x, lambda)

## S3 method for class 'ts'
exp_decay(x, lambda)

## S3 method for class 'xts'
exp_decay(x, lambda)

## S3 method for class 'data.frame'
exp_decay(x, lambda)

## S3 method for class 'tbl'
exp_decay(x, lambda)
```

Arguments

x	An univariate or a multivariate distribution.
lambda	A number for the decay parameter.

Details

The half-life is linked with the lambda parameter as follows:

- HL = log(2) / lambda.

For example: log(2) / 0.0166 is approximately 42. So, a parameter lambda of 0.0166 can be associated with a half-life of two-months.

Value

A numerical vector of class ffp with the new probabilities distribution.

See Also

crisp kernel_normal half_life

Examples

```
library(ggplot2)

# long half_life
long_hl <- exp_decay(EuStockMarkets, 0.001)
long_hl
autoplot(long_hl) +
  scale_color_viridis_c()

# short half_life
short_hl <- exp_decay(EuStockMarkets, 0.015)
short_hl
autoplot(short_hl) +
  scale_color_viridis_c()
```

---

ffp *Manipulate the* ffp *Class*

Description

Helpers and Constructors from ffp.

Usage

```
ffp(x = double(), ...)

is_ffp(x)

as_ffp(x)

## Default S3 method:
as_ffp(x)

## S3 method for class 'integer'
as_ffp(x)
```

Arguments

x	• For `ffp()`: A numeric vector. • For `is_ffp()`: An object to be tested. • For `as_ffp()`: An object to convert to `ffp`.
...	Additional attributes to be passed to `ffp`.

Details

The `ffp` class is designed to interact with doubles, but the output of `c(ffp,double)` or `c(double,ffp)` will always return a `double` (not an `ffp` object), since there is no way to guarantee the interaction between a numeric vector and a probability will also be a probability.

Value

- `ffp()` and `as_ffp()` return an S3 vector of class `ffp` (built upon `double`'s);
- `is_ffp()` returns a `logical` object.

Examples

```
set.seed(123)
p <- runif(5)
p <- p / sum(p)

is_ffp(p)
ffp(p)
```

`half_life` *Half-Life Calculation*

Description

Compute the implied half-life of a decay parameter.

Usage

```
half_life(lambda)
```

Arguments

`lambda`	A number.

Value

A single number with the half-life in days.

See Also

`exp_decay` `double_decay`

Examples

```
half_life(0.0166)
half_life(0.01)
```

`kernel_entropy` *Partial Information Kernel-Damping*

Description

This function uses entropy-pooling to find the probability distribution that can constrain the first two moments while imposing the minimal structure in the data.

Usage

```
kernel_entropy(x, mean, sigma = NULL)

## Default S3 method:
kernel_entropy(x, mean, sigma = NULL)

## S3 method for class 'numeric'
kernel_entropy(x, mean, sigma = NULL)

## S3 method for class 'matrix'
kernel_entropy(x, mean, sigma = NULL)

## S3 method for class 'ts'
kernel_entropy(x, mean, sigma = NULL)

## S3 method for class 'xts'
kernel_entropy(x, mean, sigma = NULL)

## S3 method for class 'tbl_df'
kernel_entropy(x, mean, sigma = NULL)

## S3 method for class 'data.frame'
kernel_entropy(x, mean, sigma = NULL)
```

Arguments

`x`	An univariate or a multivariate distribution.
`mean`	A numeric vector in which the kernel should be centered.
`sigma`	The uncertainty (volatility) around the mean. When `NULL`, only the mean is constrained.

Value

A numerical vector of class `ffp` with the new probabilities distribution.

See Also

double_decay

Examples

```
library(ggplot2)

ret <- diff(log(EuStockMarkets[ , 1]))
mean <- -0.01 # scenarios around -1%
sigma <- var(diff(ret))

ke <- kernel_entropy(ret, mean, sigma)
ke

autoplot(ke) +
  scale_color_viridis_c()
```

---

kernel_normal *Full Information by Kernel-Damping*

Description

In this framework, historical realizations receive a weight proportional to its distance from a target mean that is surrounded by normal kernel.

Usage

```
kernel_normal(x, mean, sigma)

## Default S3 method:
kernel_normal(x, mean, sigma)

## S3 method for class 'numeric'
kernel_normal(x, mean, sigma)

## S3 method for class 'matrix'
kernel_normal(x, mean, sigma)

## S3 method for class 'ts'
kernel_normal(x, mean, sigma)

## S3 method for class 'xts'
kernel_normal(x, mean, sigma)

## S3 method for class 'tbl_df'
kernel_normal(x, mean, sigma)

## S3 method for class 'data.frame'
kernel_normal(x, mean, sigma)
```

Arguments

x	An univariate or a multivariate distribution.
mean	A numeric vector in which the kernel should be centered.
sigma	The uncertainty (volatility) around the mean.

Value

A numerical vector of class ffp with the new probabilities distribution.

See Also

crisp exp_decay

Examples

```
library(ggplot2)

ret <- diff(log(EuStockMarkets[ , 1]))
mean <- -0.01 # scenarios around -1%
sigma <- var(diff(ret))

kn <- kernel_normal(ret, mean, sigma)
kn

autoplot(kn) +
  scale_color_viridis_c()

# A larger sigma spreads out the distribution
sigma <- var(diff(ret)) / 0.05
kn <- kernel_normal(ret, mean, sigma)

autoplot(kn) +
  scale_color_viridis_c()
```

---

scenario_density *Plot Scenarios*

Description

This functions are designed to make it easier to visualize the impact of a *View* in the P&L distribution.

Usage

```
scenario_density(x, p, n = 10000)

scenario_histogram(x, p, n = 10000)
```

Arguments

x	An univariate marginal distribution.
p	A probability from the ffp class.
n	An integer scalar with the number of scenarios to be generated.

Details

To generate a scenario-distribution the margins are bootstrapped using bootstrap_scenarios. The number of resamples can be controlled with the n argument (default is n = 10000).

Value

A ggplot2 object.

Examples

```
pnl <- diff(log(EuStockMarkets))[, 1]
p <- exp_decay(pnl, 0.005)

scenario_density(pnl, p, 500)
scenario_histogram(pnl, p, 500)
```

Index