Package 'distributional'

June 4, 2020

Title Vectorised Probability Distributions

Version 0.1.0

Description Vectorised distribution objects with tools for manipulating, visualising, and using probability distributions. Designed to allow model prediction outputs to return distributions rather than their parameters, allowing users to directly interact with predictive distributions in a data-oriented workflow. In addition to providing generic replacements for p/d/q/r functions, other useful statistics can be computed including means, variances, intervals, and highest density regions.

License GPL-3

Imports vctrs (>= 0.3.0),
rlang (>= 0.4.5),
generics,
ellipsis,
stats,
numDeriv,
ggplot2,
scales,
farver,
digest,
utils,
lifecycle

Suggests testthat (>= 2.1.0),
covr,
mvtnorm,
actuar

RdMacros lifecycle

URL https://pkg.mitchelloharawild.com/distributional/, https://github.com/mitchelloharawild/distributional

BugReports https://github.com/mitchelloharawild/distributional/issues

Encoding UTF-8

Language en-GB

LazyData true

Roxygen list(markdown = TRUE, roclets=c('rd', 'collate', 'namespace'))

RoxygenNote 7.1.0

R topics documented:

autoplot.distribution . . . 3
cdf . . . 4
col2hex . . . 4
darken_fill . . . 5
density.distribution . . . 5
dist_bernoulli . . . 6
dist_beta . . . 6
dist_binomial . . . 7
dist_burr . . . 7
dist_cauchy . . . 8
dist_chisq . . . 8
dist_degenerate . . . 9
dist_exponential . . . 9
dist_f . . . 10
dist_gamma . . . 10
dist_geometric . . . 11
dist_gumbel . . . 11
dist_hypergeometric . . . 12
dist_inflated . . . 12
dist_inverse_exponential . . . 13
dist_inverse_gamma . . . 13
dist_inverse_gaussian . . . 14
dist_logarithmic . . . 14
dist_logistic . . . 15
dist_mixture . . . 15
dist_multinomial . . . 16
dist_multivariate_normal . . . 16
dist_negative_binomial . . . 17
dist_normal . . . 17
dist_pareto . . . 18
dist_percentile . . . 18
dist_poisson . . . 19
dist_poisson_inverse_gaussian . . . 19
dist_sample . . . 20
dist_studentized_range . . . 20
dist_student_t . . . 21
dist_transformed . . . 21
dist_truncated . . . 22
dist_uniform . . . 22
dist_weibull . . . 23
generate.distribution . . . 23
geom_hilo_linerange . . . 24
geom_hilo_ribbon . . . 25
guide_level . . . 26
hdr . . . 27
hdr.distribution . . . 27
hilo . . . 28
hilo.distribution . . . 28
is_hdr . . . 29
is_hilo . . . 29

mean.distribution 29
median.distribution 30
new_dist 30
new_hdr 31
new_hilo 31
quantile.distribution 32
scale_hilo_continuous 32
scale_level 34
variance 36
variance.distribution 36

Index **37**

`autoplot.distribution` *Plot a distribution*

Description

Experimental

Usage

```
## S3 method for class 'distribution'
autoplot(
  x,
  type = c("pdf", "cdf"),
  n = 100,
  quantile_range = c(0.001, 0.999),
  ...
)
```

Arguments

`x`	The distribution(s) to plot.
`type`	The type of plot to make (must be either `"pdf"` or `"cdf"`).
`n`	The resolution (number of points) used to display the distribution.
`quantile_range`	The range of the distribution (specified as quantiles).
`...`	Unused.

Details

Visualise distribution(s) by plotting its probability density function (`density()`) or cumulative distribution function (`cdf()`). Note: This function currently only works for continuous distributions.

Examples

```
library(ggplot2)
dist <- c(dist_normal(mu = 0, sigma = 1), dist_student_t(df = 3))
autoplot(dist, type = "pdf")
autoplot(dist, type = "cdf")
```

cdf *The cumulative distribution function*

Description

Stable

Usage

```
cdf(x, q, ...)

## S3 method for class 'distribution'
cdf(x, q, ...)
```

Arguments

x	The distribution(s).
q	The quantile at which the cdf is calculated.
...	Additional arguments used by methods.

col2hex *col2hex*

Description

converts colors to RGB

Usage

```
col2hex(col)
```

Arguments

col	colors

Value

RGB colors

darken_fill *darken_fill*

Description

darken fill colors for probability ranges

Usage

```
darken_fill(col, prob)
```

Arguments

col	colors
prob	probability values

density.distribution *The probability density/mass function*

Description

Stable

Usage

```
## S3 method for class 'distribution'
density(x, at, ...)
```

Arguments

x	The distribution(s).
at	The point at which to compute the density/mass.
...	Additional arguments passed to methods.

Details

Computes the probability density function for a continuous distribution, or the probability mass function for a discrete distribution.

dist_bernoulli *The Bernoulli distribution*

Description

Stable

Usage

```
dist_bernoulli(prob)
```

Arguments

prob	The probability of success on each trial.

Examples

```
dist_bernoulli(prob = c(0.05, 0.5, 0.3, 0.9, 0.1))
```

dist_beta *The Beta distribution*

Description

Maturing

Usage

```
dist_beta(shape1, shape2)
```

Arguments

shape1, shape2	The non-negative shape parameters of the Beta distribution.

See Also

stats::Beta

Examples

```
dist_beta(shape1 = c(0.5, 5, 1, 2, 2), shape2 = c(0.5, 1, 3, 2, 5))
```

dist_binomial *The Binomial distribution*

Description

Stable

Usage

```
dist_binomial(size, prob)
```

Arguments

size	The number of trials.
prob	The probability of success on each trial.

Examples

```
dist_binomial(size = 1:5, prob = c(0.05, 0.5, 0.3, 0.9, 0.1))
```

dist_burr *The Burr distribution*

Description

Stable

Usage

```
dist_burr(shape1, shape2, rate = 1)
```

Arguments

shape1	parameters. Must be strictly positive.
shape2	parameters. Must be strictly positive.
rate	an alternative way to specify the scale.

See Also

actuar::Burr

Examples

```
dist_burr(shape1 = c(1,1,1,2,3,0.5), shape2 = c(1,2,3,1,1,2))
```

dist_cauchy *The Cauchy distribution*

Description

Maturing

Usage

```
dist_cauchy(location, scale)
```

Arguments

location	location and scale parameters.
scale	location and scale parameters.

See Also

stats::Cauchy

Examples

```
dist_cauchy(location = c(0, 0, 0, -2), scale = c(0.5, 1, 2, 1))
```

dist_chisq *The (non-central) Chi-Squared Distribution*

Description

Stable

Usage

```
dist_chisq(df, ncp = 0)
```

Arguments

df	degrees of freedom (non-negative, but can be non-integer).
ncp	non-centrality parameter (non-negative).

See Also

stats::Chisquare

Examples

```
dist_chisq(df = c(1,2,3,4,6,9))
```

dist_degenerate *The degenerate distribution*

Description

Stable

Usage

```
dist_degenerate(x)
```

Arguments

x	The value of the distribution.

Examples

```
dist_degenerate(x = 1:5)
```

dist_exponential *The Exponential Distribution*

Description

Stable

Usage

```
dist_exponential(rate)
```

Arguments

rate	vector of rates.

See Also

stats::Exponential

Examples

```
dist_exponential(rate = c(2, 1, 2/3))
```

dist_f *The (non-central) Chi-Squared Distribution*

Description

Stable

Usage

```
dist_f(df1, df2, ncp = NULL)
```

Arguments

df1	degrees of freedom. Inf is allowed.
df2	degrees of freedom. Inf is allowed.
ncp	non-centrality parameter. If omitted the central F is assumed.

See Also

stats::FDist

Examples

```
dist_f(df1 = c(1,2,5,10,100), df2 = c(1,1,2,1,100))
```

dist_gamma *The Gamma distribution*

Description

Stable

Usage

```
dist_gamma(shape, rate)
```

Arguments

shape	shape and scale parameters. Must be positive, scale strictly.
rate	an alternative way to specify the scale.

See Also

stats::GammaDist

Examples

```
dist_gamma(shape = c(1,2,3,5,9,7.5,0.5), rate = c(0.5,0.5,0.5,1,2,1,1))
```

dist_geometric *The Geometric Distribution*

Description

Stable

Usage

```
dist_geometric(prob)
```

Arguments

prob	probability of success in each trial. 0 < prob <= 1.

See Also

stats::Geometric

Examples

```
dist_geometric(prob = c(0.2, 0.5, 0.8))
```

dist_gumbel *The Gumbel distribution*

Description

Stable

Usage

```
dist_gumbel(alpha, scale)
```

Arguments

alpha	location parameter.
scale	parameter. Must be strictly positive.

See Also

actuar::Gumbel

Examples

```
dist_gumbel(alpha = c(0.5, 1, 1.5, 3), scale = c(2, 2, 3, 4))
```

dist_hypergeometric *The Hypergeometric distribution*

Description

Stable

Usage

```
dist_hypergeometric(m, n, k)
```

Arguments

m	The number of type I elements available.
n	The number of type II elements available.
k	The size of the sample taken.

See Also

stats::Hypergeometric

Examples

```
dist_hypergeometric(m = rep(500, 3), n = c(50, 60, 70), k = c(100, 200, 300))
```

dist_inflated *Inflate a value of a probability distribution*

Description

Maturing

Usage

```
dist_inflated(dist, prob, x = 0)
```

Arguments

dist	The distribution(s) to inflate.
prob	The added probability of observing x.
x	The value to inflate. The default of x = 0 is for zero-inflation.

dist_inverse_exponential *The Inverse Exponential distribution*

Description

Stable

Usage

```
dist_inverse_exponential(rate)
```

Arguments

rate	an alternative way to specify the scale.

See Also

actuar::InverseExponential

Examples

```
dist_inverse_exponential(rate = 1:5)
```

dist_inverse_gamma *The Inverse Gamma distribution*

Description

Stable

Usage

```
dist_inverse_gamma(shape, rate = 1/scale, scale)
```

Arguments

shape	parameters. Must be strictly positive.
rate	an alternative way to specify the scale.
scale	parameters. Must be strictly positive.

See Also

actuar::InverseGamma

Examples

```
dist_inverse_gamma(shape = c(1,2,3,3), rate = c(1,1,1,2))
```

dist_inverse_gaussian *The Inverse Gaussian distribution*

Description

Stable

Usage

```
dist_inverse_gaussian(mean, shape)
```

Arguments

mean	parameters. Must be strictly positive. Infinite values are supported.
shape	parameters. Must be strictly positive. Infinite values are supported.

See Also

actuar::InverseGaussian

Examples

```
dist_inverse_gaussian(mean = c(1,1,1,3,3), shape = c(0.2, 1, 3, 0.2, 1))
```

dist_logarithmic *The Logarithmic distribution*

Description

Stable

Usage

```
dist_logarithmic(prob)
```

Arguments

prob	parameter. 0 <= prob < 1.

See Also

actuar::Logarithmic

Examples

```
dist_logarithmic(prob = c(0.33, 0.66, 0.99))
```

dist_logistic *The Logistic distribution*

Description

Stable

Usage

```
dist_logistic(location, scale)
```

Arguments

location	location and scale parameters.
scale	location and scale parameters.

See Also

stats::Logistic

Examples

```
dist_logistic(location = c(5,9,9,6,2), scale = c(2,3,4,2,1))
```

dist_mixture *Create a mixture of distributions*

Description

Experimental

Usage

```
dist_mixture(..., weights = numeric())
```

Arguments

...	Distributions to be used in the mixture.
weights	The weight of each distribution passed to ....

Examples

```
dist_mixture(dist_normal(0, 1), dist_normal(5, 2), weights = c(0.3, 0.7))
```

dist_multinomial *The Multinomial distribution*

Description

Maturing

Usage

```
dist_multinomial(size, prob)
```

Arguments

size	integer, say N, specifying the total number of objects that are put into K boxes in the typical multinomial experiment. For dmultinom, it defaults to sum(x).
prob	numeric non-negative vector of length K, specifying the probability for the K classes; is internally normalized to sum 1. Infinite and missing values are not allowed.

See Also

stats::Multinom

Examples

```
dist_multinomial(size = c(4, 3), prob = list(c(0.3, 0.5, 0.2), c(0.1, 0.5, 0.4)))
```

dist_multivariate_normal *The multivariate normal distribution*

Description

Maturing

Usage

```
dist_multivariate_normal(mu = 0, sigma = diag(1))
```

Arguments

mu	A list of numeric vectors for the distribution's mean.
sigma	A list of matrices for the distribution's variance-covariance matrix.

Examples

```
dist_multivariate_normal(mu = list(c(1,2)), sigma = list(matrix(c(4,2,2,3), ncol=2)))
```

dist_negative_binomial *The Negative Binomial distribution*

Description

Stable

Usage

```
dist_negative_binomial(size, prob)
```

Arguments

size	target for number of successful trials, or dispersion parameter (the shape parameter of the gamma mixing distribution). Must be strictly positive, need not be integer.
prob	probability of success in each trial. `0 < prob <= 1`.

See Also

stats::NegBinomial

Examples

```
dist_negative_binomial(size = 10, prob = 0.5)
```

dist_normal *The Normal distribution*

Description

Stable

Usage

```
dist_normal(mu = 0, sigma = 1)
```

Arguments

mu	The mean (location parameter) of the distribution.
sigma	The standard deviation (scale parameter) of the distribution.

See Also

stats::Normal

Examples

```
dist_normal(mu = 1:5, sigma = 3)
```

dist_pareto *The Pareto distribution*

Description

Questioning

Usage

```
dist_pareto(shape, scale)
```

Arguments

shape	parameters. Must be strictly positive.
scale	parameters. Must be strictly positive.

See Also

actuar::Pareto

Examples

```
dist_pareto(shape = c(10, 3, 2, 1), scale = rep(1, 4))
```

dist_percentile *Percentile distribution*

Description

Maturing

Usage

```
dist_percentile(x, percentile)
```

Arguments

x	A list of values
percentile	A list of percentiles

Examples

```
dist <- dist_normal()
percentiles <- seq(0.01, 0.99, by = 0.01)
x <- vapply(percentiles, quantile, double(1L), x = dist)
dist_percentile(list(x), list(percentiles*100))
```

dist_poisson *The Poisson Distribution*

Description

Stable

Usage

```
dist_poisson(lambda)
```

Arguments

lambda	vector of (non-negative) means.

See Also

stats::Poisson

Examples

```
dist_poisson(lambda = c(1, 4, 10))
```

dist_poisson_inverse_gaussian *The Poisson-Inverse Gaussian distribution*

Description

Stable

Usage

```
dist_poisson_inverse_gaussian(mean, shape)
```

Arguments

mean	parameters. Must be strictly positive. Infinite values are supported.
shape	parameters. Must be strictly positive. Infinite values are supported.

See Also

actuar::PoissonInverseGaussian

Examples

```
dist_poisson_inverse_gaussian(mean = rep(0.1, 3), shape = c(0.4, 0.8, 1))
```

dist_sample *Sampling distribution*

Description

Stable

Usage

```
dist_sample(x)
```

Arguments

x	A list of sampled values.

Examples

```
dist_sample(x = list(rnorm(100), rnorm(100, 10)))
```

dist_studentized_range *The Studentized Range distribution*

Description

Stable

Usage

```
dist_studentized_range(nmeans, df, nranges)
```

Arguments

nmeans	sample size for range (same for each group).
df	degrees of freedom for s (see below).
nranges	number of *groups* whose **maximum** range is considered.

See Also

stats::Tukey

Examples

```
dist_studentized_range(nmeans = c(6, 2), df = c(5, 4), nranges = c(1, 1))
```

dist_student_t *The (non-central) Student t Distribution*

Description

Stable

Usage

```
dist_student_t(df, ncp = NULL)
```

Arguments

df	degrees of freedom (> 0, maybe non-integer). df = Inf is allowed.
ncp	non-centrality parameter δ; currently except for rt(), only for abs(ncp) <= 37.62. If omitted, use the central t distribution.

See Also

stats::TDist

Examples

```
dist_student_t(df = c(1,2,5))
```

dist_transformed *Modify a distribution with a transformation*

Description

Experimental

Usage

```
dist_transformed(dist, transform, inverse)
```

Arguments

dist	A univariate distribution vector.
transform	A function used to transform the distribution. This transformation should be monotonic over appropriate domain.
inverse	The inverse of the transform function.

Details

The density(), mean(), and variance() methods are approximate as they are based on numerical derivatives.

Examples

```
# Create a log normal distribution
dist <- dist_transformed(dist_normal(0, 0.5), exp, log)
density(dist, 1) # dlnorm(1, 0, 0.5)
cdf(dist, 4) # plnorm(4, 0, 0.5)
quantile(dist, 0.1) # qlnorm(0.1, 0, 0.5)
generate(dist, 10) # rlnorm(10, 0, 0.5)
```

---

dist_truncated *Truncate a distribution*

Description

Experimental

Usage

```
dist_truncated(dist, lower = -Inf, upper = Inf)
```

Arguments

dist	The distribution(s) to truncate.
lower, upper	The range of values to keep from a distribution.

Details

Note that the samples are generated using inverse transform sampling, and the means and variances are estimated from samples.

---

dist_uniform *The Uniform distribution*

Description

Stable

Usage

```
dist_uniform(min, max)
```

Arguments

min	lower and upper limits of the distribution. Must be finite.
max	lower and upper limits of the distribution. Must be finite.

See Also

stats::Uniform

Examples

```
dist_uniform(min = c(3, -2), max = c(5, 4))
```

---

dist_weibull *The Weibull distribution*

Description

Stable

Usage

```
dist_weibull(shape, scale)
```

Arguments

shape	shape and scale parameters, the latter defaulting to 1.
scale	shape and scale parameters, the latter defaulting to 1.

See Also

stats::Weibull

Examples

```
dist_weibull(shape = c(0.5, 1, 1.5, 5), scale = rep(1, 4))
```

---

generate.distribution *Randomly sample values from a distribution*

Description

Stable

Usage

```
## S3 method for class 'distribution'
generate(x, times, ...)
```

Arguments

x	The distribution(s).
times	The number of samples.
...	Additional arguments used by methods.

Details

Generate random samples from probability distributions.

`geom_hilo_linerange` *Line ranges for hilo intervals*

Description

Experimental

Usage

```
geom_hilo_linerange(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()` or `aes_()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.
`data`	The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x,10)`).
`stat`	The statistical transformation to use on the data for this layer, as a string.
`position`	Position adjustment, either as a string, or the result of a call to a position adjustment function.
`na.rm`	If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.
`show.legend`	logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.
`inherit.aes`	If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.
`...`	Other arguments passed on to `layer()`. These are often aesthetics, used to set an aesthetic to a fixed value, like `colour = "red"` or `size = 3`. They may also be parameters to the paired geom/stat.

Details

geom_hilo_linerange() displays the interval defined by a hilo object. The luminance of the shaded area indicates its confidence level. The shade colour can be controlled by the fill aesthetic, however the luminance will be overwritten to represent the confidence level.

See Also

geom_hilo_ribbon() for continuous hilo intervals (ribbons)

Examples

```
dist <- dist_normal(1:3, 1:3)
library(ggplot2)
ggplot(
  data.frame(x = rep(1:3, 2), interval = c(hilo(dist, 80), hilo(dist, 95)))
) +
  geom_hilo_linerange(aes(x = x, hilo = interval))
```

---

geom_hilo_ribbon *Ribbon plots for hilo intervals*

Description

Maturing

Usage

```
geom_hilo_ribbon(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot(). A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created.

	A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x,10)).
stat	The statistical transformation to use on the data for this layer, as a string.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. borders().
...	Other arguments passed on to layer(). These are often aesthetics, used to set an aesthetic to a fixed value, like colour = "red" or size = 3. They may also be parameters to the paired geom/stat.

Details

geom_hilo_ribbon() displays the interval defined by a hilo object. The luminance of the shaded area indicates its confidence level. The shade colour can be controlled by the fill aesthetic, however the luminance will be overwritten to represent the confidence level.

See Also

geom_hilo_linerange() for discrete hilo intervals (vertical lines)

Examples

```
dist <- dist_normal(1:3, 1:3)
library(ggplot2)
ggplot(
  data.frame(x = rep(1:3, 2), interval = c(hilo(dist, 80), hilo(dist, 95)))
) +
  geom_hilo_ribbon(aes(x = x, hilo = interval))
```

---

guide_level *Level shade bar guide*

Description

The level guide shows the colour from the forecast intervals which is blended with the series colour.

Usage

```
guide_level(title = waiver(), max_discrete = 5, ...)
```

Arguments

title	A character string or expression indicating a title of guide. If NULL, the title is not shown. By default (waiver()), the name of the scale object or the name specified in labs() is used for the title.
max_discrete	The maximum number of levels to be shown using guide_legend. If the number of levels exceeds this value, level shades are shown with guide_colourbar.
...	Further arguments passed onto either guide_colourbar or guide_legend

---

hdr *Compute highest density regions*

Description

Used to extract a specified prediction interval at a particular confidence level from a distribution.

Usage

```
hdr(x, ...)
```

Arguments

x	Object to create hilo from.
...	Additional arguments used by methods.

---

hdr.distribution *Highest density regions of probability distributions*

Description

Experimental

Usage

```
## S3 method for class 'distribution'
hdr(x, size = 95, n = 512, ...)
```

Arguments

x	The distribution(s).
size	The size of the interval (between 0 and 100).
n	The resolution used to estimate the distribution's density.
...	Additional arguments used by methods.

Details

This function is highly experimental and will change in the future. In particular, improved functionality for object classes and visualisation tools will be added in a future release.

Computes minimally sized probability intervals highest density regions.

hilo *Compute intervals*

Description

Used to extract a specified prediction interval at a particular confidence level from a distribution.

Usage

```
hilo(x, ...)
```

Arguments

x	Object to create hilo from.
...	Additional arguments used by methods.

hilo.distribution *Probability intervals of a probability distribution*

Description

Maturing

Usage

```
## S3 method for class 'distribution'
hilo(x, size = 95, ...)
```

Arguments

x	The distribution(s).
size	The size of the interval (between 0 and 100).
...	Additional arguments used by methods.

Details

Returns a hilo central probability interval with probability coverage of size. By default, the distribution's quantile() will be used to compute the lower and upper bound for a centered interval

See Also

hdr.distribution()

is_hdr *Is the object a hdr*

Description

Is the object a hdr

Usage

```
is_hdr(x)
```

Arguments

x	An object.

is_hilo *Is the object a hilo*

Description

Is the object a hilo

Usage

```
is_hilo(x)
```

Arguments

x	An object.

mean.distribution *Mean of a probability distribution*

Description

Stable

Usage

```
## S3 method for class 'distribution'
mean(x, ...)
```

Arguments

x	The distribution(s).
...	Additional arguments used by methods.

Details

Returns the empirical mean of the probability distribution. If the method does not exist, the mean of a random sample will be returned.

median.distribution *Median of a probability distribution*

Description

Stable

Usage

```
## S3 method for class 'distribution'
median(x, na.rm = FALSE, ...)
```

Arguments

x	The distribution(s).
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.
...	Additional arguments used by methods.

Details

Returns the median (50th percentile) of a probability distribution. This is equivalent to `quantile(x,p=0.5)`.

new_dist *Create a new distribution*

Description

Create a new distribution

Usage

```
new_dist(..., class = NULL, dimnames = NULL)
```

Arguments

...	Parameters of the distribution (named).
class	The class of the distribution for S3 dispatch.
dimnames	The names of the variables in the distribution (optional).

new_hdr *Construct hdr intervals*

Description

Construct hdr intervals

Usage

```
new_hdr(x = list())
```

Arguments

x	A list of hilo() objects.

Value

A "hdr" vector

Author(s)

Mitchell O'Hara-Wild

new_hilo *Construct hilo intervals*

Description

Construct hilo intervals

Usage

```
new_hilo(lower = double(), upper = double(), size = double())
```

Arguments

lower, upper	A numeric vector of values for lower and upper limits.
size	Size of the interval between [0, 100].

Value

A "hilo" vector

Author(s)

Earo Wang & Mitchell O'Hara-Wild

Examples

```
new_hilo(lower = rnorm(10), upper = rnorm(10) + 5, size = 95)
```

quantile.distribution *Distribution Quantiles*

Description

Stable

Usage

```
## S3 method for class 'distribution'
quantile(x, p, ...)
```

Arguments

x	The distribution(s).
p	The probability of the quantile.
...	Additional arguments passed to methods.

Details

Computes the quantiles of a distribution.

scale_hilo_continuous *Hilo interval scales*

Description

Hilo interval scales

Usage

```
scale_hilo_continuous(
  name = waiver(),
  breaks = waiver(),
  minor_breaks = waiver(),
  n.breaks = NULL,
  labels = waiver(),
  limits = NULL,
  expand = waiver(),
  oob = identity,
  na.value = NA,
  trans = "identity",
  guide = waiver(),
  position = "left",
  sec.axis = waiver()
)
```

Arguments

name	The name of the scale. Used as the axis or legend title. If `waiver()`, the default, the name of the scale is taken from the first mapping used for that aesthetic. If `NULL`, the legend title will be omitted.
breaks	One of: • `NULL` for no breaks • `waiver()` for the default breaks computed by the transformation object • A numeric vector of positions • A function that takes the limits as input and returns breaks as output (e.g., a function returned by `scales::extended_breaks()`)
minor_breaks	One of: • `NULL` for no minor breaks • `waiver()` for the default breaks (one minor break between each major break) • A numeric vector of positions • A function that given the limits returns a vector of minor breaks.
n.breaks	An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if `breaks = waiver()`. Use `NULL` to use the default number of breaks given by the transformation.
labels	One of: • `NULL` for no labels • `waiver()` for the default labels computed by the transformation object • A character vector giving labels (must be same length as `breaks`) • A function that takes the breaks as input and returns labels as output
limits	One of: • `NULL` to use the default scale range • A numeric vector of length two providing limits of the scale. Use `NA` to refer to the existing minimum or maximum • A function that accepts the existing (automatic) limits and returns new limits Note that setting limits on positional scales will **remove** data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see `coord_cartesian()`).
expand	For position scales, a vector of range expansion constants used to add some padding around the data to ensure that they are placed some distance away from the axes. Use the convenience function `expansion()` to generate the values for the expand argument. The defaults are to expand the scale by 5% on each side for continuous variables, and by 0.6 units on each side for discrete variables.
oob	One of: • Function that handles limits outside of the scale limits (out of bounds). • The default (`scales::censor()`) replaces out of bounds values with `NA`. • `scales::squish()` for squishing out of bounds values into range. • `scales::squish_infinite()` for squishing infitite values into range.
na.value	Missing values will be replaced with this value.

trans	For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time". A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called <name>_trans (e.g., scales::boxcox_trans()). You can create your own transformation with scales::trans_new().
guide	A function used to create a guide or its name. See guides() for more information.
position	For position scales, The position of the axis. left or right for y axes, top or bottom for x axes.
sec.axis	sec_axis() is used to specify a secondary axis.

---

scale_level *level luminance scales*

Description

This set of scales defines new scales for prob geoms equivalent to the ones already defined by ggplot2. This allows the shade of confidence intervals to work with the legend output.

Usage

```
scale_level_continuous(..., guide = "level")
```

Arguments

...	Arguments passed on to continuous_scale scale_name The name of the scale that should be used for error messages associated with this scale. palette A palette function that when called with a numeric vector with values between 0 and 1 returns the corresponding output values (e.g., scales::area_pal()). name The name of the scale. Used as the axis or legend title. If waiver(), the default, the name of the scale is taken from the first mapping used for that aesthetic. If NULL, the legend title will be omitted. breaks One of: • NULL for no breaks • waiver() for the default breaks computed by the transformation object • A numeric vector of positions • A function that takes the limits as input and returns breaks as output (e.g., a function returned by scales::extended_breaks()) minor_breaks One of: • NULL for no minor breaks • waiver() for the default breaks (one minor break between each major break) • A numeric vector of positions • A function that given the limits returns a vector of minor breaks.

`n.breaks` An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if `breaks = waiver()`. Use `NULL` to use the default number of breaks given by the transformation.

`labels` One of:

- `NULL` for no labels
- `waiver()` for the default labels computed by the transformation object
- A character vector giving labels (must be same length as `breaks`)
- A function that takes the breaks as input and returns labels as output

`limits` One of:

- `NULL` to use the default scale range
- A numeric vector of length two providing limits of the scale. Use `NA` to refer to the existing minimum or maximum
- A function that accepts the existing (automatic) limits and returns new limits Note that setting limits on positional scales will **remove** data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see `coord_cartesian()`).

`rescaler` A function used to scale the input values to the range [0, 1]. This is always `scales::rescale()`, except for diverging and n colour gradients (i.e., `scale_colour_gradient2()`, `scale_colour_gradientn()`). The `rescaler` is ignored by position scales, which always use `scales::rescale()`.

`oob` One of:

- Function that handles limits outside of the scale limits (out of bounds).
- The default (`scales::censor()`) replaces out of bounds values with `NA`.
- `scales::squish()` for squishing out of bounds values into range.
- `scales::squish_infinite()` for squishing infitite values into range.

`trans` For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time".

A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called <name>_trans (e.g., `scales::boxcox_trans()`). You can create your own transformation with `scales::trans_new()`.

`expand` For position scales, a vector of range expansion constants used to add some padding around the data to ensure that they are placed some distance away from the axes. Use the convenience function `expansion()` to generate the values for the `expand` argument. The defaults are to expand the scale by 5% on each side for continuous variables, and by 0.6 units on each side for discrete variables.

`position` For position scales, The position of the axis. `left` or `right` for y axes, `top` or `bottom` for x axes.

`super` The super class to use for the constructed scale

`guide` Type of legend. Use `"colourbar"` for continuous colour bar, or `"legend"` for discrete colour legend.

Value

A ggproto object inheriting from `Scale`

`variance` *Variance*

Description

A generic function for computing the variance of an object. The default method will use `stats::var()` to compute the variance.

Usage

```
variance(x, ...)
```

Arguments

`x`	An object.
`...`	Additional arguments used by methods.

See Also

`variance.distribution()`

`variance.distribution` *Variance of a probability distribution*

Description

Stable

Usage

```
## S3 method for class 'distribution'
variance(x, ...)
```

Arguments

`x`	The distribution(s).
`...`	Additional arguments used by methods.

Details

Returns the empirical mean of the probability distribution. If the method does not exist, the mean of a random sample will be returned.

Index

actuar::Burr, *7*
actuar::Gumbel, *11*
actuar::InverseExponential, *13*
actuar::InverseGamma, *13*
actuar::InverseGaussian, *14*
actuar::Logarithmic, *14*
actuar::Pareto, *18*
actuar::PoissonInverseGaussian, *19*
aes(), *24, 25*
aes_(), *24, 25*
autoplot.distribution, 3

borders(), *24*, *26*

cdf, 4
cdf(), *3*
col2hex, 4
continuous_scale, *34*
coord_cartesian(), *33*, *35*

darken_fill, 5
density(), *3*, *21*
density.distribution, 5
dist_bernoulli, 6
dist_beta, 6
dist_binomial, 7
dist_burr, 7
dist_cauchy, 8
dist_chisq, 8
dist_degenerate, 9
dist_exponential, 9
dist_f, 10
dist_gamma, 10
dist_geometric, 11
dist_gumbel, 11
dist_hypergeometric, 12
dist_inflated, 12
dist_inverse_exponential, 13
dist_inverse_gamma, 13
dist_inverse_gaussian, 14
dist_logarithmic, 14
dist_logistic, 15
dist_mixture, 15
dist_multinomial, 16
dist_multivariate_normal, 16
dist_negative_binomial, 17
dist_normal, 17
dist_pareto, 18
dist_percentile, 18
dist_poisson, 19
dist_poisson_inverse_gaussian, 19
dist_sample, 20
dist_student_t, 21
dist_studentized_range, 20
dist_transformed, 21
dist_truncated, 22
dist_uniform, 22
dist_weibull, 23

expansion(), *33*, *35*

fortify(), *24, 25*

generate.distribution, 23
geom_hilo_linerange, 24
geom_hilo_linerange(), *26*
geom_hilo_ribbon, 25
geom_hilo_ribbon(), *25*
ggplot(), *24, 25*
guide_colourbar, *27*
guide_legend, *27*
guide_level, 26
guides(), *34*

hdr, 27
hdr.distribution, 27
hdr.distribution(), *28*
hilo, 28
hilo(), *31*
hilo.distribution, 28

is_hdr, 29
is_hilo, 29

labs(), *27*
layer(), *24*, *26*

mean(), *21*
mean.distribution, 29

median.distribution, 30

new_dist, 30
new_hdr, 31
new_hilo, 31

quantile(), *28*
quantile.distribution, 32

scale_colour_gradient2(), *35*
scale_colour_gradientn(), *35*
scale_hilo_continuous, 32
scale_level, 34
scale_level_continuous *(*scale_level*)*, 34
scales::area_pal(), *34*
scales::boxcox_trans(), *34, 35*
scales::censor(), *33*, *35*
scales::extended_breaks(), *33, 34*
scales::rescale(), *35*
scales::squish(), *33*, *35*
scales::squish_infinite(), *33*, *35*
scales::trans_new(), *34, 35*
sec_axis(), *34*
stats::Beta, *6*
stats::Cauchy, *8*
stats::Chisquare, *8*
stats::Exponential, *9*
stats::FDist, *10*
stats::GammaDist, *10*
stats::Geometric, *11*
stats::Hypergeometric, *12*
stats::Logistic, *15*
stats::Multinom, *16*
stats::NegBinomial, *17*
stats::Normal, *17*
stats::Poisson, *19*
stats::TDist, *21*
stats::Tukey, *20*
stats::Uniform, *22*
stats::var(), *36*
stats::Weibull, *23*

transformation object, *33, 34*

variance, 36
variance(), *21*
variance.distribution, 36
variance.distribution(), *36*

waiver(), *27*