Package ‘distributional’

February 2, 2021

Title Vectorised Probability Distributions

Version 0.2.2

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,
        ggdist

RdMacros lifecycle


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 7.1.1
R topics documented:

- autoplot.distribution
- cdf
- density.distribution
- dist_binomial
- dist_beta
- dist_burr
- dist_cauchy
- dist_chisq
- dist_degenerate
- dist_exponential
- dist_f
- dist_gamma
- dist_geometric
- dist_gumbel
- dist_hypergeometric
- dist_inflated
- dist_inverse_exponential
- dist_inverse_gamma
- dist_inverse_gaussian
- dist_logarithmic
- dist_logistic
- dist_missing
- dist_mixture
- dist_multinomial
- dist_multivariate_normal
- dist_negative_binomial
- dist_normal
- dist_pareto
- dist_percentile
- dist_poisson
- dist_poisson_inverse_gaussian
- dist_sample
- dist_studentized_range
- dist_student_t
- dist_transformed
- dist_truncated
- dist_uniform
- dist_weibull
- dist_wrap
- generate.distribution
- geom_hilo_linerange
- geom_hilo_ribbon
- guide_level
- hdr
- hdr.distribution
- hilo
- hilo.distribution
- is_distribution
- is_hdr
Description

Deprecated

Usage

```r
## 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

```r
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

```r
cdf(x, q, ..., log = FALSE)
```

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

Arguments

- `x` The distribution(s).
- `q` The quantile at which the cdf is calculated.
- `...` Additional arguments passed to methods.
- `log` If TRUE, probabilities will be given as log probabilities.

density.distribution

The probability density/mass function

Description

Stable

Usage

```r
density(x, at, ..., log = FALSE)
```

## S3 method for class 'distribution'
density(x, at, ..., log = FALSE)

Arguments

- `x` The distribution(s).
- `at` The point at which to compute the density/mass.
- `...` Additional arguments passed to methods.
- `log` If TRUE, probabilities will be given as log probabilities.

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**

```r
dist_bernoulli(prob)
```

**Arguments**

- `prob` The probability of success on each trial, `prob` can be any value in [0, 1].

**Details**

Bernoulli distributions are used to represent events like coin flips when there is a single trial that is either successful or unsuccessful. The Bernoulli distribution is a special case of the `Binomial()` distribution with `n = 1`.

We recommend reading this documentation on [https://pkg.mitchelloharawild.com/distributional/](https://pkg.mitchelloharawild.com/distributional/), where the math will render nicely.

In the following, let $X$ be a Bernoulli random variable with parameter $p = p$. Some textbooks also define $q = 1 - p$, or use $\pi$ instead of $p$.

The Bernoulli probability distribution is widely used to model binary variables, such as 'failure' and 'success'. The most typical example is the flip of a coin, when $p$ is thought as the probability of flipping a head, and $q = 1 - p$ is the probability of flipping a tail.

**Support**: {0, 1}

**Mean**: $p$

**Variance**: $p \cdot (1 - p) = p \cdot q$

**Probability mass function (p.m.f):**

$$P(X = x) = p^x (1 - p)^{1-x} = p^x q^{1-x}$$

**Cumulative distribution function (c.d.f):**

$$P(X \leq x) = \begin{cases} 
0 & x < 0 \\
1 - p & 0 \leq x < 1 \\
1 & x \geq 1 
\end{cases}$$

**Moment generating function (m.g.f):**

$$E(e^{tX}) = (1 - p) + pe^t$$
dist_beta

Examples

```r
dist <- dist_bernoulli(prob = c(0.05, 0.5, 0.3, 0.9, 0.1))

dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)

generate(dist, 10)

density(dist, 2)
density(dist, 2, log = TRUE)

cdf(dist, 4)

quintile(dist, 0.7)
```

---

dist_beta  
*The Beta distribution*

Description

Maturing

Usage

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

Arguments

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

See Also

- `stats::Beta`

Examples

```r
dist <- dist_beta(shape1 = c(0.5, 5, 1, 2, 2), shape2 = c(0.5, 1, 3, 2, 5))

dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)

generate(dist, 10)

density(dist, 2)
density(dist, 2, log = TRUE)

cdf(dist, 4)
```
quantile(dist, 0.7)

The Binomial distribution

Description

Stable

Usage

dist_binomial(size, prob)

Arguments

size The number of trials. Must be an integer greater than or equal to one. When size = 1L, the Binomial distribution reduces to the Bernoulli distribution. Often called n in textbooks.

prob The probability of success on each trial, prob can be any value in [0, 1].

Details

Binomial distributions are used to represent situations can that can be thought as the result of n Bernoulli experiments (here the n is defined as the size of the experiment). The classical example is n independent coin flips, where each coin flip has probability p of success. In this case, the individual probability of flipping heads or tails is given by the Bernoulli(p) distribution, and the probability of having x equal results (x heads, for example), in n trials is given by the Binomial(n, p) distribution. The equation of the Binomial distribution is directly derived from the equation of the Bernoulli distribution.

We recommend reading this documentation on https://pkg.mitchelloharawild.com/distributional/, where the math will render nicely.

The Binomial distribution comes up when you are interested in the portion of people who do a thing. The Binomial distribution also comes up in the sign test, sometimes called the Binomial test (see stats::binom.test()), where you may need the Binomial C.D.F. to compute p-values.

In the following, let X be a Binomial random variable with parameter size = n and p = p. Some textbooks define q = 1 − p, or called π instead of p.

Support: \{0, 1, 2, ..., n\}

Mean: np

Variance: np \cdot (1 - p) = np \cdot q

Probability mass function (p.m.f):

\[ P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k} \]

Cumulative distribution function (c.d.f):
\[ P(X \leq k) = \sum_{i=0}^{[k]} \binom{n}{i} p^i (1 - p)^{n-i} \]

Moment generating function (m.g.f):
\[ E(e^{tX}) = (1 - p + p e^t)^n \]

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

dist
dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)

generate(dist, 10)

density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
```

---

### 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))
```
The Cauchy distribution

Description

Maturing

Usage

dist_cauchy(location, scale)

Arguments

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

Details

The Cauchy distribution is the student’s t distribution with one degree of freedom. The Cauchy
distribution does not have a well defined mean or variance. Cauchy distributions often appear as
priors in Bayesian contexts due to their heavy tails.

We recommend reading this documentation on https://pkg.mitchelloharawild.com/distributional/,
where the math will render nicely.

In the following, let $X$ be a Cauchy variable with mean location $= x_0$ and scale $= \gamma$.

**Support**: $R$, the set of all real numbers

**Mean**: Undefined.

**Variance**: Undefined.

**Probability density function (p.d.f):**

$$f(x) = \frac{1}{\pi \gamma \left[ 1 + \left( \frac{x - x_0}{\gamma} \right)^2 \right]}$$

**Cumulative distribution function (c.d.f):**

$$F(t) = \frac{1}{\pi} \arctan \left( \frac{t - x_0}{\gamma} \right) + \frac{1}{2}$$

**Moment generating function (m.g.f):**

Does not exist.

**See Also**

stats::Cauchy
Examples

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

mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)

generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)

---

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).

Details

Chi-square distributions show up often in frequentist settings as the sampling distribution of test statistics, especially in maximum likelihood estimation settings.

We recommend reading this documentation on https://pkg.mitchelloharawild.com/distributional/, where the math will render nicely.

In the following, let $X$ be a $\chi^2$ random variable with $df = k$.

Support: $R^+$, the set of positive real numbers

Mean: $k$

Variance: $2k$

Probability density function (p.d.f):

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$
Cumulative distribution function (c.d.f):

The cumulative distribution function has the form

\[ F(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx \]

but this integral does not have a closed form solution and must be approximated numerically. The c.d.f. of a standard normal is sometimes called the "error function". The notation \( \Phi(t) \) also stands for the c.d.f. of a standard normal evaluated at \( t \). Z-tables list the value of \( \Phi(t) \) for various \( t \).

Moment generating function (m.g.f):

\[ E(e^{tX}) = e^{\mu t + \sigma^2 t^2 / 2} \]

See Also

 stats::Chisquare

Examples

dist <- dist_chisq(df = c(1,2,3,4,6,9))

dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)

---

**dist_degenerate**

The degenerate distribution

Description

Stable

Usage

`dist_degenerate(x)`

Arguments

- `x` The value of the distribution.
Details

The degenerate distribution takes a single value which is certain to be observed. It takes a single parameter, which is the value that is observed by the distribution.

We recommend reading this documentation on https://pkg.mitchelloharawild.com/distributional/, where the math will render nicely.

In the following, let $X$ be a degenerate random variable with value $x = k_0$.

**Support:** $\mathbb{R}$, the set of all real numbers

**Mean:** $k_0$

**Variance:** 0

**Probability density function (p.d.f):**

$$f(x) = \begin{cases} 1 & \text{for } x = k_0 \\ 0 & \text{for } x \neq k_0 \end{cases}$$

**Cumulative distribution function (c.d.f):**

The cumulative distribution function has the form

$$F(x) = \begin{cases} 0 & \text{for } x < k_0 \\ 1 & \text{for } x \geq k_0 \end{cases}$$

**Moment generating function (m.g.f):**

$$E(e^{tX}) = e^{k_0 t}$$

Examples

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

---

**dist_exponential**

*The Exponential Distribution*

**Description**

Stable

**Usage**

```
dist_exponential(rate)
```

**Arguments**

```
rate \quad \text{vector of rates.}
```

**See Also**

```
stats::Exponential
```
Examples

dist <- dist_exponential(rate = c(2, 1, 2/3))
dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)

---

dist_f

The F 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.

Details

We recommend reading this documentation on https://pkg.mitchelloharawild.com/distributional/, where the math will render nicely.

In the following, let $X$ be a Gamma random variable with parameters shape $= \alpha$ and rate $= \beta$.

Support: $x \in (0, \infty)$

Mean: $\frac{\alpha}{\beta}$

Variance: $\frac{\alpha}{\beta^2}$

Probability density function (p.m.f):

$$f(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$$

Cumulative distribution function (c.d.f):
\[ f(x) = \frac{\Gamma(\alpha, \beta x)}{\Gamma \alpha} \]

Moment generating function (m.g.f):

\[ E(e^{tX}) = \left( \frac{\beta}{\beta - t} \right)^\alpha, \quad t < \beta \]

See Also

stats::FDist

Examples

dist <- dist_f(df1 = c(1,2,5,10,100), df2 = c(1,1,2,1,100))
dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)

\begin{itemize}
  \item \textbf{dist.gamma} \quad \textit{The Gamma distribution}
\end{itemize}

Description

Stable

Usage

dist.gamma(shape, rate)

Arguments

\begin{itemize}
  \item \texttt{shape} \quad \text{shape and scale parameters. Must be positive, scale strictly.}
  \item \texttt{rate} \quad \text{an alternative way to specify the scale.}
\end{itemize}
Several important distributions are special cases of the Gamma distribution. When the shape parameter is 1, the Gamma is an exponential distribution with parameter $1/\beta$. When the shape $= n/2$ and rate $= 1/2$, the Gamma is an equivalent to a chi squared distribution with $n$ degrees of freedom. Moreover, if we have $X_1$ is $\text{Gamma}(\alpha_1, \beta)$ and $X_2$ is $\text{Gamma}(\alpha_2, \beta)$, a function of these two variables of the form $\frac{X_1}{X_1 + X_2} = \text{Beta}(\alpha_1, \alpha_2)$. This last property frequently appears in another distributions, and it has extensively been used in multivariate methods. More about the Gamma distribution will be added soon.

In the following, let $X$ be a Gamma random variable with parameters shape $= \alpha$ and rate $= \beta$.

**Support**: $x \in (0, \infty)$

**Mean**: $\frac{\alpha}{\beta}$

**Variance**: $\frac{\alpha}{\beta^2}$

**Probability density function (p.m.f):**

$$f(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$$

**Cumulative distribution function (c.d.f):**

$$f(x) = \frac{\Gamma(\alpha, \beta x)}{\Gamma(\alpha)}$$

**Moment generating function (m.g.f):**

$$E(e^{tX}) = \left( \frac{\beta}{\beta - t} \right)^\alpha, \ t < \beta$$

**See Also**

`stats::GammaDist`

**Examples**

```r
dist <- 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
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
```
The Geometric Distribution

Description

The Geometric distribution can be thought of as a generalization of the \texttt{dist_bernoulli()} distribution where we ask: "if I keep flipping a coin with probability $p$ of heads, what is the probability I need $k$ flips before I get my first heads?" The Geometric distribution is a special case of Negative Binomial distribution. \textbf{Stable}

Usage

\begin{verbatim}
dist_geometric(prob)
\end{verbatim}

Arguments

\begin{itemize}
  \item \texttt{prob} \hspace{1cm} probability of success in each trial. \(0 < \text{prob} \leq 1\).
\end{itemize}

Details

We recommend reading this documentation on \texttt{https://pkg.mitchelloharawild.com/distributional/}, where the math will render nicely.

In the following, let $X$ be a Geometric random variable with success probability $p = p$. Note that there are multiple parameterizations of the Geometric distribution.

\textbf{Support}: \(0 < p < 1, \; x = 0, 1, \ldots\)

\textbf{Mean}: \(\frac{1-p}{p}\)

\textbf{Variance}: \(\frac{1-p}{p^2}\)

\textbf{Probability mass function (p.m.f)}:

\[ P(X = x) = p(1 - p)^x, \]

\textbf{Cumulative distribution function (c.d.f)}:

\[ P(X \leq x) = 1 - (1 - p)^{x+1} \]

\textbf{Moment generating function (m.g.f)}:

\[ E(e^{tX}) = \frac{pe^t}{1 - (1 - p)e^t} \]

See Also

\texttt{stats::Geometric}
Examples

```r
dist <- dist_geometric(prob = c(0.2, 0.5, 0.8))

dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)

generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
```

---

**dist_gumbel**

*The Gumbel distribution*

**Description**

Stable

**Usage**

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

**Arguments**

- `alpha`: location parameter.
- `scale`: parameter. Must be strictly positive.

**Details**

The Gumbel distribution is a special case of the Generalized Extreme Value distribution, obtained when the GEV shape parameter $\xi$ is equal to 0. It may be referred to as a type I extreme value distribution.

We recommend reading this documentation on [https://pkg.mitchelloharawild.com/distributional/](https://pkg.mitchelloharawild.com/distributional/), where the math will render nicely.

In the following, let $X$ be a Gumbel random variable with location parameter $\mu = \mu$, scale parameter $\sigma = \sigma$.

**Support**: $R$, the set of all real numbers.

**Mean**: $\mu + \sigma \gamma$, where $\gamma$ is Euler’s constant, approximately equal to 0.57722.

**Median**: $\mu - \sigma \ln(\ln 2)$.

**Variance**: $\sigma^2 \pi^2 / 6$.

**Probability density function (p.d.f):**

$$f(x) = \sigma^{-1} \exp[-(x - \mu)/\sigma] \exp\{-\exp[-(x - \mu)/\sigma]\}$$
for \( x \) in \( R \), the set of all real numbers.

**Cumulative distribution function (c.d.f):**

In the \( \xi = 0 \) (Gumbel) special case

\[
F(x) = \exp\{- \exp[-(x - \mu)/\sigma]\}
\]

for \( x \) in \( R \), the set of all real numbers.

See Also

actuar::Gumbel

Examples

```r
dist <- dist_gumbel(alpha = c(0.5, 1, 1.5, 3), scale = c(2, 2, 3, 4))
dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
```

---

**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.
Details

To understand the HyperGeometric distribution, consider a set of \( r \) objects, of which \( m \) are of the type I and \( n \) are of the type II. A sample with size \( k \) \((k < r)\) with no replacement is randomly chosen. The number of observed type I elements observed in this sample is set to be our random variable \( X \).

We recommend reading this documentation on https://pkg.mitchelloharawild.com/distributional/, where the math will render nicely.

In the following, let \( X \) be a HyperGeometric random variable with success probability \( p = p = \frac{m}{m+n} \).

**Support:** \( x \in \{ \max (0, k-n), \ldots, \min (k, m) \} \)

**Mean:** \( \frac{km}{n+m} = kp \)

**Variance:** \( \frac{km(n)(n+m-k)}{(n+m)(n+m-1)} = kp(1-p)(1 - \frac{k-1}{m+n-1}) \)

**Probability mass function (p.m.f):**

\[
P(X = x) = \binom{m}{x} \binom{n}{k-x} \binom{m+n}{k}
\]

**Cumulative distribution function (c.d.f):**

\[
P(X \leq k) \approx \Phi\left( \frac{x - kp}{\sqrt{kp(1-p)}} \right)
\]

See Also

stats::Hypergeometric

Examples

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

dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
```
**dist_inflated**

**Inflate a value of a probability distribution**

**Description**

Maturing

**Usage**

```r
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**

```r
dist_inverse_exponential(rate)
```

**Arguments**

- `rate`: an alternative way to specify the scale.

**See Also**

`actuar::InverseExponential`

**Examples**

```r
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

\texttt{dist_logarithmic(prob)}

Arguments

\begin{itemize}
  \item \texttt{prob} \quad \text{parameter. } 0 \leq \text{prob} < 1.
\end{itemize}

See Also

\texttt{actuar::Logarithmic}

Examples

\begin{verbatim}
dist_logarithmic(prob = c(0.33, 0.66, 0.99))
\end{verbatim}

dist_logistic

The Logistic distribution

Description
Stable

Usage

\texttt{dist_logistic(location, scale)}

Arguments

\begin{itemize}
  \item \texttt{location} \quad \text{location and scale parameters.}
  \item \texttt{scale} \quad \text{location and scale parameters.}
\end{itemize}

Details

A continuous distribution on the real line. For binary outcomes the model given by \( P(Y = 1 | X) = F(X \beta) \) where \( F \) is the Logistic \texttt{cdf()} is called logistic regression.

We recommend reading this documentation on \url{https://pkg.mitchelloharawild.com/distributional/}, where the math will render nicely.

In the following, let \( X \) be a Logistic random variable with \texttt{location} = \( \mu \) and \texttt{scale} = \( s \).

Support: \( R \), the set of all real numbers

Mean: \( \mu \)
Variance: \( s^2 \pi^2 / 3 \)

Probability density function (p.d.f):

\[
f(x) = \frac{e^{-\left(\frac{z-x}{s}\right)}}{s[1 + \exp\left(-\left(\frac{z-x}{s}\right)\right)]^2}
\]

Cumulative distribution function (c.d.f):

\[
F(t) = \frac{1}{1 + e^{-\left(\frac{t-\mu}{s}\right)}}
\]

Moment generating function (m.g.f):

\[
E(e^{tX}) = e^{\mu t} \beta(1 - st, 1 + st)
\]

where \( \beta(x, y) \) is the Beta function.

See Also

stats::Logistic

Examples

dist <- dist_logistic(location = c(5,9,9,6,2), scale = c(2,3,4,2,1))

dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)

generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quartile(dist, 0.7)

---

dist_missing

Missing distribution

Description

Experimental

Usage

dist_missing(length = 1)
**dist_mixture**

**Arguments**

**length**  
The number of missing distributions

**Details**

A placeholder distribution for handling missing values in a vector of distributions.

**Examples**

```r
dist <- dist_missing(3L)
dist
mean(dist)
variance(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
```

---

**dist_mixture**  
*Create a mixture of distributions*

**Description**

**Experimental**

**Usage**

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

**Arguments**

...  
Distributions to be used in the mixture.

weights  
The weight of each distribution passed to ....

**Examples**

```r
dist_mixture(dist_normal(0, 1), dist_normal(5, 2), weights = c(0.3, 0.7))
```
The Multinomial distribution

Description

Maturing

Usage

```r
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 \( \text{dmultinom} \), it defaults to \( \text{sum}(x) \).
- `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.

Details

The multinomial distribution is a generalization of the binomial distribution to multiple categories. It is perhaps easiest to think that we first extend a \( \text{dist_bernoulli()} \) distribution to include more than two categories, resulting in a categorical distribution. We then extend repeat the Categorical experiment several (\( n \)) times.

We recommend reading this documentation on https://pkg.mitchelloharawild.com/distributional/, where the math will render nicely.

In the following, let \( X = (X_1, ..., X_k) \) be a Multinomial random variable with success probability \( p = \mathbf{p} \). Note that \( \mathbf{p} \) is vector with \( k \) elements that sum to one. Assume that we repeat the Categorical experiment size = \( n \) times.

Support: Each \( X_i \) is in \( 0, 1, 2, ..., n \).

Mean: The mean of \( X_i \) is \( np_i \).

Variance: The variance of \( X_i \) is \( np_i(1 - p_i) \). For \( i \neq j \), the covariance of \( X_i \) and \( X_j \) is \( -np_ip_j \).

Probability mass function (p.m.f):

\[
P(X_1 = x_1, ..., X_k = x_k) = \frac{n!}{x_1!x_2!...x_k!} p_1^{x_1} \cdot p_2^{x_2} \cdot ... \cdot p_k^{x_k}
\]

Cumulative distribution function (c.d.f):

Omitted for multivariate random variables for the time being.

Moment generating function (m.g.f):

\[
E(e^{tX}) = \left( \sum_{i=1}^{k} p_i e^{i t} \right)^n
\]

See Also

`stats::Multinomial`
Examples

```
dist <- dist_multinomial(size = c(4, 3), prob = list(c(0.3, 0.5, 0.2), c(0.1, 0.5, 0.4)))
dist
mean(dist)
variance(dist)
generate(dist, 10)
```

# TODO: Needs fixing to support multiple inputs
# density(dist, 2)
# density(dist, 2, log = TRUE)

---

`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.

Details

A generalization of the geometric distribution. It is the number of failures in a sequence of i.i.d. Bernoulli trials before a specified number of successes (size) occur. The probability of success in each trial is given by prob.

We recommend reading this documentation on https://pkg.mitchelloharawild.com/distributional/, where the math will render nicely.

In the following, let X be a Negative Binomial random variable with success probability prob = p and the number of successes size = r.

Support: \{0, 1, 2, 3, ...\}

Mean: \frac{pr}{1-p}

Variance: \frac{pr}{(1-p)^2}

Probability mass function (p.m.f):

\[ f(k) = \binom{k+r-1}{k} \cdot (1-p)^r \cdot p^k \]

Cumulative distribution function (c.d.f):

Too nasty, omitted.

Moment generating function (m.g.f):

\[ \left( \frac{1-p}{1-pte^t} \right)^r, t < -\log p \]

See Also

stats::NegBinomial

Examples

dist <- dist_negative_binomial(size = 10, prob = 0.5)

dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)
generate(dist, 10)

density(dist, 2)
density(dist, 2, log = TRUE)

cdf(dist, 4)
quantile(dist, 0.7)

### Description

**Stable**

#### Usage

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

#### Arguments

- **mu**: The mean (location parameter) of the distribution, which is also the mean of the distribution. Can be any real number.
- **sigma**: The standard deviation (scale parameter) of the distribution. Can be any positive number. If you would like a Normal distribution with **variance** $\sigma^2$, be sure to take the square root, as this is a common source of errors.

#### Details

The Normal distribution is ubiquitous in statistics, partially because of the central limit theorem, which states that sums of i.i.d. random variables eventually become Normal. Linear transformations of Normal random variables result in new random variables that are also Normal. If you are taking an intro stats course, you’ll likely use the Normal distribution for Z-tests and in simple linear regression. Under regularity conditions, maximum likelihood estimators are asymptotically Normal. The Normal distribution is also called the gaussian distribution.

We recommend reading this documentation on [https://pkg.mitchelloharawild.com/distributional/](https://pkg.mitchelloharawild.com/distributional/), where the math will render nicely.

In the following, let $X$ be a Normal random variable with mean $\mu = \mu$ and standard deviation $\sigma = \sigma$.

**Support**: $\mathbb{R}$, the set of all real numbers

**Mean**: $\mu$

**Variance**: $\sigma^2$

**Probability density function (p.d.f):**

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-(x-\mu)^2 / 2\sigma^2}$$

**Cumulative distribution function (c.d.f):**

The cumulative distribution function has the form

$$F(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi\sigma^2}}e^{-(x-\mu)^2 / 2\sigma^2} dx$$
but this integral does not have a closed form solution and must be approximated numerically. The c.d.f. of a standard Normal is sometimes called the "error function". The notation \( \Phi(t) \) also stands for the c.d.f. of a standard Normal evaluated at \( t \). Z-tables list the value of \( \Phi(t) \) for various \( t \).

**Moment generating function (m.g.f):**

\[
E(e^{tX}) = e^{\mu t + \sigma^2 t^2 / 2}
\]

**See Also**

stats::Normal

**Examples**

```r
dist <- dist_normal(mu = 1:5, sigma = 3)
dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
```

---

**dist_pareto**

*The Pareto distribution*

**Description**

**Questioning**

**Usage**

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

**Arguments**

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

**See Also**

actuar::Pareto

**Examples**

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

**Percentile distribution**

**Description**

Maturing

**Usage**

\[
\text{dist\_percentile}(x, \text{percentile})
\]

**Arguments**

- \(x\)  
  A list of values

- \(\text{percentile}\)  
  A list of percentiles

**Examples**

\[
\begin{align*}
\text{dist} & \leftarrow \text{dist\_normal}() \\
\text{percentiles} & \leftarrow \text{seq}(0.01, 0.99, \text{by} = 0.01) \\
x & \leftarrow \text{vapply}(\text{percentiles}, \text{quantile}, \text{double(1L)}, x = \text{dist}) \\
\text{dist\_percentile}(\text{list}(x), \text{list}(\text{percentiles}\times100))
\end{align*}
\]

--

**dist_poisson**  

**The Poisson Distribution**

**Description**

Stable

**Usage**

\[
\text{dist\_poisson}(\lambda)
\]

**Arguments**

- \(\lambda\)  
  vector of (non-negative) means.

**Details**

Poisson distributions are frequently used to model counts.

We recommend reading this documentation on [https://pkg.mitchelloharawild.com/distributional/](https://pkg.mitchelloharawild.com/distributional/), where the math will render nicely.

In the following, let \(X\) be a Poisson random variable with parameter \(\lambda = \lambda\).

**Support:** \(\{0, 1, 2, 3, \ldots\}\)

**Mean:** \(\lambda\)

**Variance:** \(\lambda\)
**Probability mass function (p.m.f):**

\[ P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!} \]

**Cumulative distribution function (c.d.f):**

\[ P(X \leq k) = e^{-\lambda} \sum_{i=0}^{[k]} \frac{\lambda^i}{i!} \]

**Moment generating function (m.g.f):**

\[ E(e^{tX}) = e^{\lambda(e^t-1)} \]

**See Also**

stats::Poisson

**Examples**

```r
dist <- dist_poisson(lambda = c(1, 4, 10))
dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
```

---

**dist_poisson_inverse_gaussian**

*The Poisson-Inverse Gaussian distribution*

**Description**

Stable

**Usage**

```r
dist_poisson_inverse_gaussian(mean, shape)
```
dist_studentized_range

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

dist_sample(x) A list of sampled values.

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

dist_studentized_range The Studentized Range distribution

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.
**Details**

Tukey’s studentized range distribution, used for Tukey’s honestly significant differences test in ANOVA.

We recommend reading this documentation on [https://pkg.mitchelloharawild.com/distributional/](https://pkg.mitchelloharawild.com/distributional/), where the math will render nicely.

**Support:** $R^+$, the set of positive real numbers.

Other properties of Tukey’s Studentized Range Distribution are omitted, largely because the distribution is not fun to work with.

**See Also**

stats::Tukey

**Examples**

dist <- dist_studentized_range(nmeans = c(6, 2), df = c(5, 4), nranges = c(1, 1))
dist
cdf(dist, 4)
quantile(dist, 0.7)

---

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

**Description**

Stable

**Usage**

`dist_student_t(df, mu = 0, sigma = 1, ncp = NULL)`

**Arguments**

- `df`: degrees of freedom (> 0, maybe non-integer). `df = Inf` is allowed.
- `mu`: The location parameter of the distribution. If `ncp == 0` (or NULL), this is the median.
- `sigma`: The scale parameter of the distribution.
- `ncp`: non-centrality parameter $\delta$; currently except for `rt()`, only for $\text{abs}(ncp) \leq 37.62$. If omitted, use the central $t$ distribution.
The Student’s T distribution is closely related to the \texttt{Normal()} distribution, but has heavier tails. As $\nu$ increases to $\infty$, the Student’s T converges to a Normal. The T distribution appears repeatedly throughout classic frequentist hypothesis testing when comparing group means.

We recommend reading this documentation on \url{https://pkg.mitchelloharawild.com/distributional/}, where the math will render nicely.

In the following, let $X$ be a \textbf{central} Students T random variable with df = $\nu$.

\textbf{Support:} $R$, the set of all real numbers

\textbf{Mean:} Undefined unless $\nu \geq 2$, in which case the mean is zero.

\textbf{Variance:}

$$\frac{\nu}{\nu - 2}$$

Undefined if $\nu < 1$, infinite when $1 < \nu \leq 2$.

\textbf{Probability density function (p.d.f):}

$$f(x) = \frac{\Gamma\left(\frac{\nu + 1}{2}\right)}{\sqrt{\nu\pi\Gamma\left(\frac{\nu}{2}\right)}}\left(1 + \frac{x^2}{\nu}\right)^{-\frac{\nu + 1}{2}}$$

\textbf{See Also}

\texttt{stats::TDist}

\textbf{Examples}

```r
dist <- dist_student_t(df = c(1,2,5), mu = c(0,1,2), sigma = c(1,2,3))
dist
mean(dist)
variance(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
```

---

\textbf{dist_transformed} \hspace{1cm} \textit{Modify a distribution with a transformation}

\textbf{Description}

\textbf{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.
Examples

```r
dist <- dist_truncated(dist_normal(2,1), lower = 0)

dist
mean(dist)
variance(dist)
generate(dist, 10)
density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)
if(requireNamespace("ggdist")) {
  library(ggplot2)
  ggplot() +
    ggdist::stat_dist_halfeye(
      aes(y = c("Normal", "Truncated"),
          dist = c(dist_normal(2,1), dist_truncated(dist_normal(2,1), lower = 0)))
    )
}
```

dist_uniform

The Uniform distribution

Description

Stable

Usage

```r
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.

Details

A distribution with constant density on an interval.

We recommend reading this documentation on https://pkg.mitchelloharawild.com/distributional/, where the math will render nicely.

In the following, let $X$ be a Poisson random variable with parameter $\lambda = \lambda$.

Support: $[a, b]$

Mean: $\frac{1}{2}(a + b)$

Variance: $\frac{1}{12}(b - a)^2$
Probability mass function (p.m.f):

\[ f(x) = \frac{1}{b - a} \quad \text{for} \ x \in [a, b] \]

\[ f(x) = 0 \quad \text{otherwise} \]

Cumulative distribution function (c.d.f):

\[ F(x) = 0 \quad \text{for} \ x < a \]

\[ F(x) = \frac{x - a}{b - a} \quad \text{for} \ x \in [a, b] \]

\[ F(x) = 1 \quad \text{for} \ x > b \]

Moment generating function (m.g.f):

\[ E(e^{tX}) = \frac{e^{tb} - e^{ta}}{t(b - a)} \quad \text{for} \ t \neq 0 \]

\[ E(e^{tX}) = 1 \quad \text{for} \ t = 0 \]

See Also

stats::Uniform

Examples

\[
dist = \text{dist_uniform(min = c(3, -2), max = c(5, 4))}
\]

\[
dist
\]

\[
\text{mean}(dist)
\]

\[
\text{variance}(dist)
\]

\[
\text{skewness}(dist)
\]

\[
\text{kurtosis}(dist)
\]

\[
\text{generate}(dist, 10)
\]

\[
\text{density}(dist, 2)
\]

\[
\text{density}(dist, 2, \log = \text{TRUE})
\]

\[
\text{cdf}(dist, 4)
\]

\[
\text{quantile}(dist, 0.7)
\]
dist_weibull

The Weibull distribution

Description
Stable

Usage

\texttt{dist\_weibull(\text{shape, scale})}

Arguments

- \texttt{shape}: shape and scale parameters, the latter defaulting to 1.
- \texttt{scale}: shape and scale parameters, the latter defaulting to 1.

Details

Generalization of the gamma distribution. Often used in survival and time-to-event analyses.

We recommend reading this documentation on \url{https://pkg.mitchelloharawild.com/distributional/}, where the math will render nicely.

In the following, let $X$ be a Weibull random variable with success probability $p = p$.

\textbf{Support}: $R^+$ and zero.

\textbf{Mean}: $\lambda \Gamma(1 + 1/k)$, where $\Gamma$ is the gamma function.

\textbf{Variance}: $\lambda [\Gamma(1 + \frac{2}{k}) - (\Gamma(1 + \frac{1}{k}))^2]$

\textbf{Probability density function (p.d.f)}:

$$f(x) = \frac{k}{\lambda} \left(\frac{x}{\lambda}\right)^{k-1} e^{-\left(x/\lambda\right)^k}, x \geq 0$$

\textbf{Cumulative distribution function (c.d.f)}:

$$F(x) = 1 - e^{-\left(x/\lambda\right)^k}, x \geq 0$$

\textbf{Moment generating function (m.g.f)}:

$$\sum_{n=0}^{\infty} \frac{t^n \lambda^n}{n!} \Gamma(1 + n/k), k \geq 1$$

See Also

\texttt{stats::Weibull}
Examples

dist <- dist_weibull(shape = c(0.5, 1, 1.5, 5), scale = rep(1, 4))

dist
mean(dist)
variance(dist)
skewness(dist)
kurtosis(dist)

generate(dist, 10)

density(dist, 2)
density(dist, 2, log = TRUE)
cdf(dist, 4)
quantile(dist, 0.7)

dist_wrap

Create a distribution from p/d/q/r style functions

Description

Experimental

Usage

dist_wrap(dist, ..., package = "stats")

Arguments

dist The name of the distribution used in the functions (name that is prefixed by p/d/q/r)
...

package The package from which the distribution is provided.

Details

If a distribution is not yet supported, you can vectorise p/d/q/r functions using this function. dist_wrap() stores the distributions parameters, and provides wrappers which call the appropriate p/d/q/r functions.

Using this function to wrap a distribution should only be done if the distribution is not yet available in this package. If you need a distribution which isn’t in the package yet, consider making a request at https://github.com/mitchelloharawild/distributional/issues.
Examples

```r
dist <- dist_wrap("norm", mean = 1:3, sd = c(3, 9, 2))
density(dist, 1) # dnorm()
cdf(dist, 4) # pnorm()
quantile(dist, 0.975) # qnorm()
generate(dist, 10) # rnorm()

library(actuar)
dist <- dist_wrap("invparalogis", package = "actuar", shape = 2, rate = 2)
density(dist, 1) # actuar::dinvparalogis()
cdf(dist, 4) # actuar::pinvparalogis()
quantile(dist, 0.975) # actuar::qinvparalogis()
generate(dist, 10) # actuar::rinvparalogis()
```

generate.distribution  Randomly sample values from a distribution

Description

Stable

Usage

```r
## 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 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)
geom_hilo_ribbon(
data.frame(x = rep(1:3, 2), interval = c(hilo(dist, 80), hilo(dist, 95)))
) +
geom_hilo_linerange(aes(x = x, hilo = interval))

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))

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**

```R
## 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_distribution  Test if the object is a distribution

Description
This function returns TRUE for distributions and FALSE for all other objects. **Stable**

Usage

```r
is_distribution(x)
```

Arguments

- `x`  
  An object.

Value

TRUE if the object inherits from the distribution class.

Examples

```r
dist <- dist_normal()
is_distribution(dist)
is_distribution("distributional")
```

is_hdr  Is the object a hdr

Description

Is the object a hdr

Usage

```r
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.

kurtosis  Kurtosis of a probability distribution

Description
Stable

Usage
kurtosis(x, ...)

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

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

likelihood  The (log) likelihood of a sample matching a distribution

Description
Maturing

Usage
likelihood(x, ...)

### S3 method for class 'distribution'
likelihood(x, sample, ..., log = FALSE)
Arguments

x

The distribution(s).

... Additional arguments used by methods.

sample A list of sampled values to compare to distribution(s).

log If TRUE, the log-likelihood will be computed.

---

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 Unused, included for consistency with the generic function.

... 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, ..., log = FALSE)

Arguments
x  The distribution(s).
p  The probability of the quantile.
...  Additional arguments passed to methods.
log  If TRUE, probabilities will be given as log probabilities.

Details
Computes the quantiles of a distribution.
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 coords::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 infinite 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**

### scale_level_continuous

#### 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 infinite 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

---

**skewness**  

*Skewness of a probability distribution*

---

**Description**

**Stable**

**Usage**

```r
skewness(x, ...)
```

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

**Arguments**

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

Description
A generic function for computing the variance of a probability distribution. The default method will use `stats::var()` to compute the variance.

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

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

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

scale_level, 52

actuar::Burr, 8
actuar::Gumbel, 18
actuar::InverseExponential, 20
actuar::InverseGamma, 21
actuar::InverseGaussian, 21
actuar::Logarithmic, 22
actuar::Pareto, 29
actuar::PoissonInverseGaussian, 32
aes(), 41, 42
aes(), 41, 42
autoplot.distribution, 3

Binomial(), 5
borders(), 41, 43

cdf(), 4
cdf(), 3, 22
cdf(), 3, 22
cdf(), 52, 53

density(), 3, 35
density_distribution, 4
dist_bernoulli, 16, 25
dist_bernoulli(), 16, 25
dist_beta, 6
dist_binomial, 7
dist_burr, 8
dist_cauchy, 9
dist_chisq, 10
dist_degenerate, 11
dist_exponential, 12
dist_f, 13
dist_gamma, 14
dist_geometric, 16
dist_gumbel, 17
dist_hypergeometric, 18
dist_inflated, 20
dist_inverse_exponential, 20
dist_inverse_gammainverse_exponential, 20
dist_inverse_gaussian, 21
dist_inverse_gaussian, 21
dist_logarithmic, 22
dist_logistic, 22
dist_missing, 23
dist_mixture, 24
dist_multinomial, 25
dist_multivariate_normal, 26
dist_negative_binomial, 26
dist_negative, 28
dist_negative, 28
dist_pareto, 29
dist_percentile, 30
dist_poisson, 30
dist_poisson_inverse_gaussian, 31
dist_sample, 32
dist_student_t, 33
dist_studentized_range, 32
dist_transformed, 34
dist_truncated, 35
dist_uniform, 36
dist_weibull, 38
dist_wrap, 39

expansion(), 52, 54

fortify(), 41, 42

generate.distribution, 40
generate.distribution(), 40
generate.distribution(), 40, 43
generate.distribution(), 40, 43
generate.distribution(), 40, 43
generate.distribution(), 40, 43

ggplot(), 41, 42
ggplot(), 41, 42
ggplot(), 41, 42
guide_colourbar, 43
guide_colourbar, 43
guide_colourbar, 43
guide_legend, 43
guide_legend, 43
guide_legend, 43
guides(), 52

hdr, 44
hdr.distribution, 44
hdr.distribution(), 45
hilo, 45
hilo, 45
hilo, 45
hilo, 45
hilo, 45
hilo, 45
hilo, 45
hilo, 45

is_distribution, 46
is_hdr, 46
is_hilo, 47

56
INDEX

kurtosis, 47
labs(), 43
layer(), 41, 43
likelihood, 47
mean(), 35
mean.distribution, 48
median.distribution, 48
new_dist, 49
new_hdr, 49
new_hilo, 50
Normal(), 34
quantile(), 45
quantile.distribution, 50
scale_colour_gradient2(), 53
scale_colour_gradientn(), 53
scale_hilo_continuous, 51
scale_level, 52
scale_level_continuous (scale_level), 52
scales::area_pal(), 53
scales::boxcox_trans(), 52, 54
scales::censor(), 52, 53
scales::extended_breaks(), 51, 53
scales::rescale(), 53
scales::squish(), 52, 53
scales::squish_infinite(), 52, 54
scales::trans_new(), 52, 54
sec_axis(), 52
skewness, 54
stats::Beta, 6
stats::binom.test(), 7
stats::Cauchy, 9
stats::Chisquare, 11
stats::Exponential, 12
stats::FDist, 14
stats::GammaDist, 15
stats::Geometric, 16
stats::Hypergeometric, 19
stats::Logistic, 23
stats::Multinomial, 25
stats::NegBinomial, 27
stats::Normal, 29
stats::Poisson, 31
stats::TDist, 34
stats::Tukey, 33
stats::Uniform, 37
stats::var(), 55
stats::Weibull, 38
transformation object, 51, 53

variance, 55
variance(), 35
variance.distribution, 55
variance.distribution(), 55
waiver(), 43