

Package: distspec (via r-universe)

October 12, 2026

Type Package

Title Probability Distributions with Certain or Uncertain Parameters

Version 0.3.0

Description Represents probability distributions with fixed or potentially uncertain parameters, with tools to discretise, convolve, sample from and summarise them.

License MIT + file LICENSE

URL <https://epiforecasts.io/distspec/>,
<https://github.com/epiforecasts/distspec>

BugReports <https://github.com/epiforecasts/distspec/issues>

Depends R (>= 3.5.0)

Imports cli, lifecycle, methods, primarycensored, rlang, stats, truncnorm, utils

Suggests covr, ggplot2, knitr, rmarkdown, spelling, testthat

VignetteBuilder knitr

Encoding UTF-8

Language en-GB

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Repository <https://epiforecasts.r-universe.dev>

Date/Publication 2026-10-12 19:11:01 UTC

RemoteUrl <https://github.com/epiforecasts/distspec>

RemoteRef HEAD

RemoteSha d8b8ca2c4842f0cfcfc3139f7a0759d93b4e141c

Contents

+.dist_spec	2
==.dist_spec	3
as_dist_spec	4
Beta	5
bound_dist	6
c.dist_spec	7
cdf	8
collapse	9
convert_to_logmean	10
convert_to_logsd	10
Dirichlet	11
discretise	12
Distributions	13
Exponential	14
fix_parameters	15
Fixed	16
Gamma	16
get_distribution	17
get_parameters	18
get_pmf	18
has_uncertainty	19
is_constrained	20
LogNormal	20
max.dist_spec	21
mean.dist_spec	22
new_dist_spec	23
NonParametric	24
Normal	25
plot.dist_spec	26
print.dist_spec	27
quantile.dist_spec	28
sample_dist	29
Weibull	30
Index	31

+.dist_spec	<i>Creates a delay distribution as the sum of two other delay distributions.</i>
-------------	--

Description

Creates a delay distribution as the sum of two other delay distributions.

`==.dist_spec`

3

Usage

```
## S3 method for class 'dist_spec'  
e1 + e2
```

Arguments

e1 The first delay distribution (of type `<dist_spec>`) to combine.
e2 The second delay distribution (of type `<dist_spec>`) to combine.

Value

A delay distribution representing the sum of the two delays

Examples

```
# A fixed lognormal distribution with mean 5 and sd 1.  
dist1 <- LogNormal(  
  meanlog = 1.6, sdlog = 1, max = 20  
)  
dist1 + dist1  
  
# An uncertain gamma distribution with shape and rate normally distributed  
# as Normal(3, 0.5) and Normal(2, 0.5) respectively  
dist2 <- Gamma(  
  shape = Normal(3, 0.5),  
  rate = Normal(2, 0.5),  
  max = 20  
)  
dist1 + dist2
```

<code>==.dist_spec</code>	<i>Compares two delay distributions</i>
---------------------------	---

Description

Compares two delay distributions

Usage

```
## S3 method for class 'dist_spec'  
e1 == e2  
  
## S3 method for class 'dist_spec'  
e1 != e2
```

Arguments

e1 The first delay distribution (of type `<dist_spec>`) to combine.
e2 The second delay distribution (of type `<dist_spec>`) to combine.

Value

TRUE or FALSE

Examples

```
Fixed(1) == Normal(1, 0.5)
```

as_dist_spec

Coerce an object to a <dist_spec>

Description

Converts other representations of a probability distribution into a <dist_spec>. A <dist_spec> is returned unchanged. A numeric vector is interpreted as a probability mass function and becomes a [NonParametric\(\)](#) distribution; a single number becomes a [Fixed\(\)](#) point mass at that value.

Other packages can provide methods for their own distribution classes by registering an `as_dist_spec.<class>` method, without `distspec` depending on them or them re-implementing the <dist_spec> format.

Usage

```
as_dist_spec(x, ...)

## S3 method for class 'dist_spec'
as_dist_spec(x, ...)

## S3 method for class 'numeric'
as_dist_spec(x, ...)

## Default S3 method:
as_dist_spec(x, ...)
```

Arguments

x	Object to convert: a <dist_spec>, a single number, or a numeric vector representing a probability mass function (zero-indexed, i.e. the first entry is the mass at zero; normalised to sum to one).
...	Additional arguments passed to methods. Every method forwards them to bound_dist() (directly for a <dist_spec>, via the constructor otherwise), so limits such as <code>max</code> apply regardless of the input class.

Value

A <dist_spec>.

See Also

[Distributions](#) for the constructors this converts to.

Examples

```
# a PMF vector becomes a nonparametric distribution
as_dist_spec(c(0.1, 0.4, 0.3, 0.2))

# a single number becomes a fixed (point mass) distribution
as_dist_spec(3)

# a <dist_spec> is returned unchanged
as_dist_spec(Gamma(mean = 4, sd = 1))
```

Beta

*Beta distribution***Description**

A beta distribution as a <dist_spec>, given either by its shape parameters shape1/shape2 or by its mean/sd. It is not discretised.

Usage

```
Beta(shape1, shape2, mean, sd, ...)
```

Arguments

shape1, shape2	Shape parameters of the beta distribution.
mean, sd	Mean and standard deviation of the distribution, as an alternative to shape1/shape2.
...	Limits of the distribution, passed to bound_dist() .

Value

A <dist_spec>.

See Also

[Distributions](#) for an overview and the other distributions.

Examples

```
Beta(shape1 = 2, shape2 = 5)
Beta(mean = 0.3, sd = 0.15)
```

bound_dist	<i>Define bounds of a <dist_spec></i>
------------	---

Description

Set the bounds that constrain a distribution: `max` truncates the support at that value, while `cdf_max` trims the tail by keeping the distribution only up to its `cdf_max` quantile. Either bound drops the mass beyond it and renormalises what remains. `discretise()` applies the bounds when producing a PMF, and `sample_dist()` draws from the bounded distribution. A bound set on a composite distribution refers to the sum of its components: `sample_dist()` rejects such a bound, and `discretise()` does not apply it.

Usage

```
bound_dist(x, max = Inf, cdf_max = 1, cdf_cutoff = deprecated())
```

Arguments

<code>x</code>	A <dist_spec>.
<code>max</code>	Numeric, maximum value of the distribution. The distribution will be truncated at this value. Default: <code>Inf</code> , i.e. no maximum.
<code>cdf_max</code>	Numeric in $(0, 1]$; the cumulative probability up to which the distribution is kept, i.e. it is truncated at the <code>cdf_max</code> quantile. For example <code>cdf_max = 0.999</code> keeps the distribution up to its 99.9th percentile. Default: <code>1</code> , i.e. keep the full distribution. A value below <code>0.5</code> is rejected, as it is almost certainly the tail probability to <i>drop</i> rather than the CDF level to keep (use <code>1 - x</code> instead).
<code>cdf_cutoff</code>	[Deprecated] Renamed to <code>cdf_max</code> . Supplying it warns and the value is interpreted as <code>cdf_max</code> , i.e. the CDF level to keep (the <code>distspec 0.1.0</code> convention). A value in the historical <code>EpiNow2</code> convention (the tail probability to drop) is caught by the guard against values below <code>0.5</code> ; pass <code>cdf_max = 1 - cdf_cutoff</code> instead.

Value

a <dist_spec> with relevant attributes set that define its bounds

See Also

`discretise()` and `sample_dist()`, which consume these bounds.

Examples

```
# Truncate a gamma distribution at 20
bound_dist(Gamma(mean = 5, sd = 1), max = 20)
# Keep it up to its 99.9th percentile
bound_dist(Gamma(mean = 5, sd = 1), cdf_max = 0.999)
```

c.dist_spec	<i>Combines multiple delay distributions for further processing</i>
-------------	---

Description

This combines the given distributions into a single composite <dist_spec> holding multiple delay distributions.

Note that distributions that already are combinations of other distributions cannot be combined with other combinations of distributions.

Usage

```
## S3 method for class 'dist_spec'  
c(...)
```

Arguments

... The delay distributions to combine

Value

Combined delay distributions (with class <dist_spec>)

Examples

```
# A fixed lognormal distribution with mean 5 and sd 1.  
dist1 <- LogNormal(  
  meanlog = 1.6, sdlog = 1, max = 20  
)  
dist1 + dist1  
  
# An uncertain gamma distribution with shape and rate normally distributed  
# as Normal(3, 0.5) and Normal(2, 0.5) respectively  
dist2 <- Gamma(  
  shape = Normal(3, 0.5),  
  rate = Normal(2, 0.5),  
  max = 20  
)  
c(dist1, dist2)
```

cdf

*Returns the cumulative distribution function of a delay distribution***Description**

Evaluates the CDF of a `<dist_spec>` with fixed (non-uncertain) parameters at the given values, respecting any `max/cdf_max` bound set with `bound_dist()`: this is the CDF of the truncated distribution, not the unbounded one, so it is 1 for any value at or beyond the bound. A `Fixed()` (point-mass) distribution has a step-function CDF: 0 below its value and 1 at or above it.

Only a distribution with fixed parameters can have its CDF evaluated. If any parameter is itself a distribution (a prior), there is no single distribution to evaluate and an error is raised; resolve it first with `fix_parameters()`.

A composite (multi-component) distribution returns one set of values per component, in keeping with `mean()/sd()/quantile.dist_spec()`. A `max/cdf_max` bound set on the composite itself (with `bound_dist()` on the sum) refers to that combined distribution, which has no closed-form CDF, so this raises an error. Bound the components individually to evaluate their CDF under a bound.

Usage

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

```
## S3 method for class 'dist_spec'
```

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

```
## S3 method for class 'multi_dist_spec'
```

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

Arguments

<code>x</code>	A <code><dist_spec></code> with fixed parameters.
<code>...</code>	Not used.
<code>q</code>	Numeric vector of values to evaluate the CDF at.

Value

For a single distribution, a numeric vector the same length as `q`. For a composite distribution of `k` components, a `length(q)` by `k` matrix, one column per component.

See Also

`quantile.dist_spec()` for the corresponding quantile function, and `fix_parameters()` to resolve an uncertain distribution first.

Examples

```
# The CDF of a fixed-parameter gamma distribution
cdf(Gamma(shape = 2, rate = 1), c(1, 2, 3))

# A `max` bound truncates the CDF accordingly: it reaches 1 at the bound
cdf(Gamma(shape = 2, rate = 1, max = 3), c(1, 2, 3))

# A fixed (point-mass) distribution has a step-function CDF
cdf(Fixed(3), c(2, 3, 4))
```

collapse	<i>Collapse nonparametric distributions in a <dist_spec></i>
----------	--

Description

This convolves any consecutive nonparametric distributions contained in the <dist_spec>.

Usage

```
## S3 method for class 'dist_spec'
collapse(x, ...)
```

Arguments

x	A <dist_spec>
...	ignored

Value

A <dist_spec> where consecutive nonparametric distributions have been convolved

See Also

[discretise\(\)](#) to produce the nonparametric components this convolves. The `vignette("distspec")` shows the full `get_pmf(collapse(discretise(d1 + d2)))` pipeline.

Examples

```
# A fixed gamma distribution with mean 5 and sd 1.
dist1 <- Gamma(mean = 5, sd = 1, max = 20)

# An uncertain lognormal distribution with meanlog and sdlog normally
# distributed as Normal(3, 0.5) and Normal(2, 0.5) respectively
dist2 <- LogNormal(
  meanlog = Normal(3, 0.5),
  sdlog = Normal(2, 0.5),
  max = 20
)
```

```
# The sum of two distributions
collapse(discretise(dist1 + dist2, strict = FALSE))
```

convert_to_logmean	<i>Convert mean and sd to log mean for a log normal distribution</i>
--------------------	--

Description

Convert from mean and standard deviation to the log mean of the lognormal distribution. Useful for defining a `LogNormal()` distribution from a mean and standard deviation on the natural scale.

Usage

```
convert_to_logmean(mean, sd)
```

Arguments

mean	Numeric, mean of a distribution
sd	Numeric, standard deviation of a distribution

Value

The log mean of a lognormal distribution

Examples

```
convert_to_logmean(2, 1)
```

convert_to_logsd	<i>Convert mean and sd to log standard deviation for a log normal distribution</i>
------------------	--

Description

Convert from mean and standard deviation to the log standard deviation of the lognormal distribution. Useful for defining a `LogNormal()` distribution from a mean and standard deviation on the natural scale.

Usage

```
convert_to_logsd(mean, sd)
```

Arguments

mean	Numeric, mean of a distribution
sd	Numeric, standard deviation of a distribution

Value

The log standard deviation of a lognormal distribution

Examples

```
convert_to_logsd(2, 1)
```

Dirichlet

Dirichlet prior for a nonparametric distribution

Description

A Dirichlet prior over the weights of a nonparametric probability mass function, used to specify an uncertain [NonParametric\(\)](#) distribution whose PMF is left uncertain given the Dirichlet prior. Give either alpha directly, or a reference prior PMF together with a concentration.

Usage

```
Dirichlet(alpha, prior, concentration, ...)
```

Arguments

alpha	A positive numeric vector of concentration parameters.
prior	Either a numeric PMF vector (zero-indexed, i.e. the first entry represents probability mass at zero) or a <code>dist_spec</code> object. If a <code>dist_spec</code> object is provided it will be discretised and the PMF extracted. If numeric, it will be normalised to sum to one internally.
concentration	A positive scalar controlling how tightly the Dirichlet prior concentrates around the supplied PMF. The Dirichlet alpha vector is computed as $\alpha_i = \text{concentration} \times p_i$ where p_i is the prior PMF. Guidance on values: • concentration = 1: weak prior, each alpha equals the PMF value (near-uniform for roughly equal PMF entries) • concentration = 5-20: moderate flexibility around the reference shape • concentration = 50+: strong anchoring to the reference PMF
...	Not used.

Value

A `<dist_spec>`.

See Also

[NonParametric\(\)](#) to use the prior, and [Distributions](#) for an overview.

Examples

```
Dirichlet(c(1, 1, 1, 1))
Dirichlet(prior = c(0.1, 0.3, 0.4, 0.2), concentration = 10)
```

discretise	<i>Discretise a <dist_spec></i>
------------	---------------------------------------

Description

Discretise a <dist_spec>

Usage

```
## S3 method for class 'dist_spec'
discretise(x, strict = TRUE, remove_trailing_zeros = TRUE, ...)

discretize(x, ...)
```

Arguments

x	A <dist_spec>
strict	Logical; If TRUE (default) an error will be thrown if a distribution cannot be discretised (e.g., because no finite maximum has been specified or parameters are uncertain). If FALSE then any distribution that cannot be discretised will be returned as is.
remove_trailing_zeros	Logical; If TRUE (default), trailing zeroes in the resulting PMF will be removed. If FALSE, trailing zeroes will be retained.
...	ignored

Value

A <dist_spec> where all distributions with constant parameters are nonparametric. Extract the resulting PMF vector with [get_pmf\(\)](#).

Methodological details

The probability mass function is computed using the {primarycensored} package, which provides double censored PMF calculations. This correctly represents the probability mass function of a double censored distribution arising from the difference of two censored events.

The probability mass function of the discretised probability distribution is a vector where the first entry corresponds to the integral over the (0,1] interval of the corresponding continuous distribution (probability of integer 0), the second entry corresponds to the (0,2] interval (probability mass of integer 1), the third entry corresponds to the (1, 3] interval (probability mass of integer 2), etc.

The maximum value truncates the distribution: mass beyond it is dropped and the remaining PMF is renormalised to sum to one. A cdf_max below 1 additionally trims the tail, keeping the distribution only up to its cdf_max quantile.

Fixed distributions:

A `Fixed()` (point-mass) distribution is not discretised through a CDF but by its own method: an integer value places all of the mass on that integer, while a fractional value splits the mass proportionally across the two adjacent integers. For example `Fixed(2.25)` places 0.75 on 2 and 0.25 on 3.

References

Charniga, K., et al. "Best practices for estimating and reporting epidemiological delay distributions of infectious diseases using public health surveillance and healthcare data", *arXiv e-prints*, 2024. [doi:10.48550/arXiv.2405.08841](https://doi.org/10.48550/arXiv.2405.08841) Park, S. W., et al., "Estimating epidemiological delay distributions for infectious diseases", *medRxiv*, 2024. [doi:10.1101/2024.01.12.24301247](https://doi.org/10.1101/2024.01.12.24301247) Abbott S., et al., "primarycensored: Primary Event Censored Distributions", 2025. [doi:10.5281/zenodo.13632839](https://doi.org/10.5281/zenodo.13632839)

See Also

`collapse()` to convolve the discretised components of a composite distribution into a single PMF, and `sample_dist()` to draw random samples. The vignette("distspec") shows the full `get_pmf(collapse(discretise + d2))` pipeline.

Examples

```
# A fixed gamma distribution with mean 5 and sd 1, discretised to a PMF.
dist1 <- Gamma(mean = 5, sd = 1, max = 20)
get_pmf(discretise(dist1))

# An uncertain lognormal distribution cannot be discretised, so with
# `strict = FALSE` it is returned unchanged.
dist2 <- LogNormal(
  meanlog = Normal(3, 0.5),
  sdlog = Normal(2, 0.5),
  max = 20
)
discretise(dist2, strict = FALSE)

# A fractional fixed value splits its mass across the two adjacent integers.
get_pmf(discretise(Fixed(2.25)))
```

Description

`distspec` represents probability distributions (typically epidemiological delays, such as generation times or reporting delays) as `<dist_spec>` objects. Each supported distribution has its own constructor: `LogNormal()`, `Gamma()`, `Normal()`, `Exponential()`, `Weibull()`, `Beta()`, `Fixed()`, `NonParametric()` and `Dirichlet()`.

Details

A parameter can be given either as a fixed numeric value or as an uncertain value (another `<dist_spec>`); currently only normally distributed uncertain parameters (from [Normal\(\)](#)) are supported.

Each distribution has a "natural" (canonical) parameterisation, such as shape and rate for [Gamma\(\)](#) or meanlog and sdlog for [LogNormal\(\)](#). It can sometimes also be specified using other parameters, such as its mean and standard deviation, which are then converted to the natural parameters (propagating any uncertainty with a first-order delta-method approximation).

See Also

[discretise\(\)](#) and [collapse\(\)](#) to discretise and convolve distributions, [sample_dist\(\)](#) to draw samples, and [get_parameters\(\)](#) / [get_pmf\(\)](#) to inspect them.

Exponential	<i>Exponential distribution</i>
-------------	---------------------------------

Description

An exponential distribution as a `<dist_spec>`, given either by its rate or by its mean.

Usage

```
Exponential(rate, mean, ...)
```

```
Exp(...)
```

Arguments

rate	vector of rates.
mean	Mean of the distribution, as an alternative to rate.
...	Limits of the distribution, passed to bound_dist() .

Value

A `<dist_spec>`.

See Also

[Distributions](#) for an overview and the other distributions.

Examples

```
Exponential(rate = 1)
Exponential(mean = 4)
```

fix_parameters	<i>Fix the parameters of a <dist_spec></i>
----------------	--

Description

If the given <dist_spec> has any uncertainty, it is removed and the corresponding distribution converted into a fixed one.

Call this before `sample_dist()` or `get_pmf()` on an uncertain distribution, as neither can operate on a distribution that still carries a prior.

Usage

```
## S3 method for class 'dist_spec'
fix_parameters(x, strategy = c("mean", "sample"), ...)
```

Arguments

x	A <dist_spec>
strategy	Character; either "mean" (use the mean estimates of the mean and standard deviation) or "sample" (randomly sample mean and standard deviation from uncertainty given in the <dist_spec>)
...	ignored

Value

A <dist_spec> object without uncertainty

Examples

```
# An uncertain gamma distribution with shape and rate normally distributed
# as Normal(3, 0.5) and Normal(2, 0.5) respectively
dist <- Gamma(
  shape = Normal(3, 0.5),
  rate = Normal(2, 0.5),
  max = 20
)

fix_parameters(dist)
```

Fixed	<i>Fixed (point-mass) distribution</i>
-------	--

Description

A fixed (delta) distribution as a `<dist_spec>`, placing all of its mass on a single value.

Usage

```
Fixed(value, ...)
```

Arguments

value	Value of the fixed (delta) distribution.
...	Limits of the distribution, passed to bound_dist() .

Value

A `<dist_spec>`.

See Also

[Distributions](#) for an overview and the other distributions.

Examples

```
Fixed(value = 3)
Fixed(value = 3.5)
```

Gamma	<i>Gamma distribution</i>
-------	---------------------------

Description

A gamma distribution as a `<dist_spec>`, given either by its natural parameters shape/rate (or shape/scale) or by its mean/sd.

Usage

```
Gamma(shape, rate, scale, mean, sd, ...)
```

Arguments

shape, scale	shape and scale parameters. Must be positive, scale strictly.
rate	an alternative way to specify the scale.
mean, sd	Mean and standard deviation of the distribution, as an alternative to shape/rate.
...	Limits of the distribution, passed to bound_dist() .

Value

A <dist_spec>.

See Also

[Distributions](#) for an overview and the other distributions.

Examples

```
Gamma(mean = 4, sd = 1)
Gamma(shape = 16, rate = 4)
Gamma(shape = Normal(16, 2), rate = Normal(4, 1))
```

get_distribution	<i>Get the distribution of a <dist_spec></i>
------------------	--

Description

Get the distribution of a <dist_spec>

Usage

```
get_distribution(x, id = NULL)
```

Arguments

x	A <dist_spec>.
id	Integer; the id of the distribution to use (if x is a composite distribution). If x is a single distribution this is ignored and can be left at its default value of NULL.

Value

A character string naming the distribution (or "nonparametric")

Examples

```
dist <- Gamma(shape = 3, rate = 2, max = 10)
get_distribution(dist)
```

get_parameters	<i>Get parameters of a parametric distribution</i>
----------------	--

Description

Generic function to extract the distribution parameters (e.g. shape and rate for Gamma) from a `dist_spec` object.

Usage

```
get_parameters(x, ...)

## S3 method for class 'dist_spec'
get_parameters(x, id = NULL, ...)
```

Arguments

<code>x</code>	A <code>dist_spec</code> object
<code>...</code>	Additional arguments passed to methods
<code>id</code>	Integer; the id of the distribution to use (if <code>x</code> is a composite distribution). If <code>x</code> is a single distribution this is ignored and can be left at its default value of <code>NULL</code> .

Value

A list of parameters of the distribution.

Examples

```
dist <- Gamma(shape = 3, rate = 2)
get_parameters(dist)
```

get_pmf	<i>Get the probability mass function of a nonparametric distribution</i>
---------	--

Description

Get the probability mass function of a nonparametric distribution

Usage

```
get_pmf(x, id = NULL)
```

Arguments

<code>x</code>	A <code><dist_spec></code> .
<code>id</code>	Integer; the id of the distribution to use (if <code>x</code> is a composite distribution). If <code>x</code> is a single distribution this is ignored and can be left at its default value of <code>NULL</code> .

Details

An uncertain (Dirichlet-backed) nonparametric distribution has no concrete PMF, so calling `get_pmf()` on one is an error. Resolve it to a fixed PMF first with `fix_parameters()` (e.g. `strategy = "mean"`).

Value

The pmf of the distribution

Examples

```
dist <- discretise(Gamma(shape = 3, rate = 2, max = 10))
get_pmf(dist)
```

has_uncertainty	<i>Check whether a <dist_spec> is uncertain</i>
-----------------	---

Description

A distribution is uncertain when it carries a prior: a parametric distribution with a `<dist_spec>` (rather than numeric) parameter, or a nonparametric distribution whose PMF is given by a `<dist_spec>` (a `Dirichlet()` prior) rather than a numeric vector.

Usage

```
has_uncertainty(x, id = NULL)
```

Arguments

x	A <code><dist_spec></code> .
id	Integer; the id of the distribution to use (if x is a composite distribution). If x is a single distribution this is ignored and can be left at its default value of NULL.

Value

TRUE if the (component) distribution is uncertain.

Examples

```
has_uncertainty(Gamma(shape = 1, rate = 1))
has_uncertainty(Gamma(shape = Normal(1, 0.5), rate = 1))
```

<code>is_constrained</code>	<i>Check if a <dist_spec> is constrained, i.e. has a finite maximum or nonzero CDF cutoff.</i>
-----------------------------	--

Description

Check if a <dist_spec> is constrained, i.e. has a finite maximum or nonzero CDF cutoff.

Usage

```
## S3 method for class 'dist_spec'
is_constrained(x, ...)
```

Arguments

<code>x</code>	A <dist_spec>
<code>...</code>	ignored

Value

Logical; TRUE if x is constrained

Examples

```
# A fixed gamma distribution with mean 5 and sd 1.
dist1 <- Gamma(mean = 5, sd = 1, max = 20)

# An uncertain lognormal distribution with meanlog and sdlog normally
# distributed as Normal(3, 0.5) and Normal(2, 0.5) respectively
dist2 <- LogNormal(
  meanlog = Normal(3, 0.5),
  sdlog = Normal(2, 0.5),
  max = 20
)

# both distributions are constrained and therefore so is the sum
is_constrained(dist1 + dist2)
```

<code>LogNormal</code>	<i>Lognormal distribution</i>
------------------------	-------------------------------

Description

A lognormal distribution as a <dist_spec>, given either by its natural parameters meanlog/sdlog or by its mean/sd.

Usage

```
LogNormal(meanlog, sdlog, mean, sd, ...)
```

Arguments

meanlog, sdlog mean and standard deviation of the distribution on the log scale with default values of 0 and 1 respectively.

mean, sd Mean and standard deviation of the distribution, as an alternative to meanlog/sdlog.

... Limits of the distribution, passed to [bound_dist\(\)](#).

Value

A <dist_spec>.

See Also

[Distributions](#) for an overview and the other distributions.

Examples

```
LogNormal(mean = 4, sd = 1)
LogNormal(mean = 4, sd = 1, max = 10)
# Uncertain parameters must be given as the natural parameters
LogNormal(meanlog = Normal(1.5, 0.5), sdlog = 0.25, max = 10)
```

max.dist_spec	<i>Returns the maximum of one or more delay distribution</i>
---------------	--

Description

This works out the maximum of all the (parametric / nonparametric) delay distributions combined in the passed <dist_spec> (ignoring any uncertainty in parameters)

Usage

```
## S3 method for class 'dist_spec'
max(x, ...)
```

Arguments

x The <dist_spec> to use

... Not used

Value

A numeric vector of maxima, one per component.

Examples

```
# A fixed gamma distribution with mean 5 and sd 1.
dist1 <- Gamma(mean = 5, sd = 1, max = 20)
max(dist1)

# An uncertain lognormal distribution with meanlog and sdlog normally
# distributed as Normal(3, 0.5) and Normal(2, 0.5) respectively
dist2 <- LogNormal(
  meanlog = Normal(3, 0.5),
  sdlog = Normal(2, 0.5),
  max = 20
)
max(dist2)

# The max the sum of two distributions
max(dist1 + dist2)
```

mean.dist_spec	<i>Returns the mean of one or more delay distribution</i>
----------------	---

Description

This works out the mean of all the (parametric / nonparametric) delay distributions combined in the passed `<dist_spec>`.

Usage

```
## S3 method for class 'dist_spec'
mean(x, ..., ignore_uncertainty = FALSE)
```

Arguments

<code>x</code>	The <code><dist_spec></code> to use
<code>...</code>	Not used
<code>ignore_uncertainty</code>	Logical; whether to ignore any uncertainty in parameters. If set to <code>FALSE</code> (the default) then the mean of any uncertain parameters will be returned as <code>NA</code> .

Value

A numeric vector of means, one per component of the `<dist_spec>`; `NA` for any component with uncertain parameters unless `ignore_uncertainty = TRUE`.

Examples

```
# A fixed lognormal distribution with mean 5 and sd 1.
dist1 <- LogNormal(mean = 5, sd = 1, max = 20)
mean(dist1)

# An uncertain gamma distribution with shape and rate normally distributed
# as Normal(3, 0.5) and Normal(2, 0.5) respectively
dist2 <- Gamma(
  shape = Normal(3, 0.5),
  rate = Normal(2, 0.5),
  max = 20
)
mean(dist2)

# The mean of the sum of two distributions
mean(dist1 + dist2)
```

new_dist_spec	<i>Internal function for generating a dist_spec given parameters and a distribution.</i>
---------------	--

Description

This will convert all parameters to natural parameters before generating a dist_spec. If they are uncertain the uncertainty is propagated to the natural parameters with a first-order (delta-method) approximation (see [convert_to_natural\(\)](#)).

Usage

```
new_dist_spec(
  params,
  distribution,
  max = Inf,
  cdf_max = 1,
  cdf_cutoff = deprecated()
)
```

Arguments

params	Parameters of the distribution (including max)
distribution	Character; the distribution type (e.g. "gamma", "lognormal", "nonparametric").
max	Numeric, maximum value of the distribution. The distribution will be truncated at this value. Default: Inf, i.e. no maximum.
cdf_max	Numeric in (0, 1]; the cumulative probability up to which the distribution is kept, i.e. it is truncated at the cdf_max quantile. For example cdf_max = 0.999 keeps the distribution up to its 99.9th percentile. Default: 1, i.e. keep the full distribution. A value below 0.5 is rejected, as it is almost certainly the tail probability to <i>drop</i> rather than the CDF level to keep (use 1 - x instead).

`cdf_cutoff` [Deprecated] Renamed to `cdf_max`. Supplying it warns and the value is interpreted as `cdf_max`, i.e. the CDF level to keep (the `distspec` 0.1.0 convention). A value in the historical `EpiNow2` convention (the tail probability to drop) is caught by the guard against values below 0.5; pass `cdf_max = 1 - cdf_cutoff` instead.

Value

A `dist_spec` of the given specification.

Examples

```
new_dist_spec(
  params = list(mean = 2, sd = 1),
  distribution = "normal"
)
```

NonParametric	<i>Nonparametric distribution</i>
---------------	-----------------------------------

Description

A nonparametric distribution as a `<dist_spec>`, defined directly by its probability mass function. The PMF can instead be left uncertain by passing a [Dirichlet\(\)](#) prior.

Usage

```
NonParametric(pmf, ...)
```

Arguments

<code>pmf</code>	Probability mass function, as a zero-indexed numeric vector (the first entry is the mass at zero) or a <code><dist_spec></code> (e.g. from Dirichlet()). A numeric vector is normalised to sum to one.
<code>...</code>	Limits of the distribution, passed to bound_dist() .

Details

A distribution constructed with a [Dirichlet\(\)](#) prior has no concrete PMF, in the same way that `Gamma(shape = Normal(3, 0.5), rate = 2)` has an uncertain shape in place of a fixed value. [get_pmf\(\)](#) errors and [has_uncertainty\(\)](#) returns TRUE until [fix_parameters\(\)](#) resolves the prior into a PMF (e.g. with `strategy = "mean"`).

Value

A `<dist_spec>`.

See Also

[Distributions](#) for an overview and the other distributions.

Examples

```
NonParametric(c(0.1, 0.3, 0.2, 0.4))

# With a Dirichlet prior (PMF left uncertain)
NonParametric(pmf = Dirichlet(c(1, 1, 1, 1)))
```

Normal

Normal distribution

Description

A normal distribution as a `<dist_spec>`, given by its mean and sd. Also used to give an uncertain parameter of another distribution.

Usage

```
Normal(mean, sd, ...)
```

Arguments

mean, sd	Mean and standard deviation of the distribution.
...	Limits of the distribution, passed to bound_dist() .

Value

A `<dist_spec>`.

See Also

[Distributions](#) for an overview and the other distributions.

Examples

```
Normal(mean = 4, sd = 1)
Normal(mean = 4, sd = 1, max = 10)
```

plot.dist_spec	<i>Plot PMF and CDF for a dist_spec object</i>
----------------	--

Description

This function takes a `<dist_spec>` object and plots its probability mass function (PMF) and cumulative distribution function (CDF) using `{ggplot2}`.

Usage

```
## S3 method for class 'dist_spec'
plot(x, samples = 50L, res = 1, cumulative = TRUE, ...)
```

Arguments

<code>x</code>	A <code><dist_spec></code> object
<code>samples</code>	Integer; Number of samples to generate for distributions with uncertain parameters (default: 50).
<code>res</code>	Numeric; Resolution of the PMF and CDF (default: 1, i.e. integer discretisation). This applies only to components discretised from a continuous distribution; a nonparametric component is already discretised on its integer support and is unaffected by <code>res</code> .
<code>cumulative</code>	Logical; whether to plot the cumulative distribution in addition to the probability mass function
<code>...</code>	ignored

Details

A component must have a finite range to be plotted. One with no finite max and no `cdf_max` of its own raises an error; bound it first (e.g. with `bound_dist()`).

Value

A `{ggplot2}` object showing the PMF (and, if `cumulative = TRUE`, the CDF) of each component, faceted by distribution.

Examples

```
# A fixed lognormal distribution with mean 5 and sd 1.
dist1 <- LogNormal(mean = 1.6, sd = 0.5, max = 20)
# Plot discretised distribution with 1 day discretisation window
plot(dist1)
# Plot discretised distribution with 0.01 day discretisation window
plot(dist1, res = 0.01, cumulative = FALSE)

# An uncertain gamma distribution with shape and rate normally distributed
# as Normal(3, 0.5) and Normal(2, 0.5) respectively
```

```

dist2 <- Gamma(
  shape = Normal(3, 0.5),
  rate = Normal(2, 0.5),
  max = 20
)
plot(dist2)

# Multiple distributions with 0.1 discretisation window and do not plot the
# cumulative distribution
plot(dist1 + dist2, res = 0.1, cumulative = FALSE)

```

print.dist_spec	<i>Prints the parameters of one or more delay distributions</i>
-----------------	---

Description

This displays the parameters of the uncertain and probability mass functions of fixed delay distributions combined in the passed <dist_spec>.

Usage

```

## S3 method for class 'dist_spec'
print(x, ...)

```

Arguments

x	The <dist_spec> to use
...	Not used

Value

invisible

Examples

```

#' # A fixed lognormal distribution with mean 5 and sd 1.
dist1 <- LogNormal(mean = 1.5, sd = 0.5, max = 20)
print(dist1)

# An uncertain gamma distribution with shape and rate normally distributed
# as Normal(3, 0.5) and Normal(2, 0.5) respectively
dist2 <- Gamma(
  shape = Normal(3, 0.5), rate = Normal(2, 0.5), max = 20
)
print(dist2)

```

quantile.dist_spec	Returns quantiles of a delay distribution
--------------------	---

Description

Computes quantiles of a `<dist_spec>` with fixed (non-uncertain) parameters, respecting any `max/cdf_max` bound set with `bound_dist()`: the quantiles are those of the truncated distribution, not the unbounded one. A `Fixed()` (point-mass) distribution has every quantile equal to its value.

Only a distribution with fixed parameters can have its quantiles computed. If any parameter is itself a distribution (a prior), there is no single distribution to compute quantiles of and an error is raised; resolve it first with `fix_parameters()`.

A composite (multi-component) distribution returns one set of quantiles per component, in keeping with `mean()/sd()`, which also return one value per component. A `max/cdf_max` bound set on the composite itself (with `bound_dist()` on the sum) refers to that combined distribution, which has no closed-form quantile function, so this raises an error. Bound the components individually to get their quantiles under a bound.

Usage

```
## S3 method for class 'dist_spec'
quantile(x, probs = seq(0, 1, 0.25), ...)

## S3 method for class 'multi_dist_spec'
quantile(x, probs = seq(0, 1, 0.25), ...)
```

Arguments

<code>x</code>	A <code><dist_spec></code> with fixed parameters.
<code>probs</code>	Numeric vector of probabilities in <code>[0, 1]</code> .
<code>...</code>	Not used.

Value

For a single distribution, a numeric vector of quantiles the same length as `probs`. For a composite distribution of `k` components, a `length(probs)` by `k` matrix, one column per component.

See Also

`cdf()` for the corresponding cumulative distribution function, and `fix_parameters()` to resolve an uncertain distribution first.

Examples

```
# Quantiles of a fixed-parameter gamma distribution
quantile(Gamma(shape = 2, rate = 1), c(0.05, 0.5, 0.95))

# A `max` bound truncates the quantiles accordingly
```

```
quantile(Gamma(shape = 2, rate = 1, max = 3), c(0.05, 0.5, 0.95))

# A fixed (point-mass) distribution: every quantile equals its value
quantile(Fixed(3), c(0.1, 0.9))
```

sample_dist

Sample from a distribution

Description

Draws random samples from a `<dist_spec>` whose parameters are fixed numbers. An unbounded distribution uses the base-R random-generation function for its family (e.g. `rgamma()` for a gamma distribution). A discretised distribution is sampled on its integer support.

A `max/cdf_max` set with `bound_dist()` on a parametric distribution is respected: samples are drawn from the truncated distribution by inverse CDF rather than by discarding draws beyond the bound, so the cost does not depend on how much of the mass the bound cuts off.

Only distributions with fixed parameters can be sampled. If any parameter is itself a distribution (a prior), there is no single distribution to sample from and an error is raised.

A composite (multi-component) distribution is sampled per component, in keeping with `mean()/sd()`, which also return one value per component. Use `rowSums()` on the result to obtain samples of the combined (convolved) distribution. A `max/cdf_max` set on the composite itself (with `bound_dist()` on the sum) refers to that combined distribution, which has no closed-form distribution to draw from, so sampling such a composite raises an error. Bound the components individually to sample them under a bound.

Usage

```
sample_dist(x, n, ...)

## S3 method for class 'dist_spec'
sample_dist(x, n, ...)

## S3 method for class 'multi_dist_spec'
sample_dist(x, n, ...)
```

Arguments

<code>x</code>	A <code><dist_spec></code> .
<code>n</code>	The number of samples to draw.
<code>...</code>	Not used.

Value

For a single distribution, a numeric vector of `n` samples. For a composite distribution of `k` components, an `n` by `k` matrix, one column of `n` samples per component (`rowSums()` gives `n` samples of the combined distribution).

See Also

[fix_parameters\(\)](#) to resolve an uncertain distribution to fixed parameters before sampling, and [discretise\(\)](#) to obtain a PMF instead.

Examples

```
# Samples from a fixed gamma distribution
sample_dist(Gamma(shape = 2, rate = 1), 10)

# Samples from a discretised distribution, drawn on its integer support
sample_dist(discretise(Gamma(shape = 2, rate = 1, max = 20)), 10)

# A fixed distribution always returns the same value
sample_dist(Fixed(3), 5)

# A composite: an n-by-k matrix, one column per component
sample_dist(Gamma(shape = 2, rate = 1) + Gamma(shape = 3, rate = 1), 10)
```

Weibull

Weibull distribution

Description

A Weibull distribution as a `<dist_spec>`, given either by its shape/scale or by its mean/sd.

Usage

```
Weibull(shape, scale, mean, sd, ...)
```

Arguments

shape, scale	shape and scale parameters, the latter defaulting to 1.
mean, sd	Mean and standard deviation of the distribution, as an alternative to shape/scale.
...	Limits of the distribution, passed to bound_dist() .

Value

A `<dist_spec>`.

See Also

[Distributions](#) for an overview and the other distributions.

Examples

```
Weibull(shape = 1, scale = 1)
Weibull(shape = 1, scale = 1, max = 10)
Weibull(mean = 4, sd = 1)
```

Index

`!=.dist_spec(==.dist_spec)`, 3
`+.dist_spec`, 2
`==.dist_spec`, 3

`as_dist_spec`, 4

Beta, 5
`Beta()`, 13
`bound_dist`, 6
`bound_dist()`, 4, 5, 8, 14, 16, 21, 24–26, 28–30

`c.dist_spec`, 7
`cdf`, 8
`cdf()`, 28
`collapse`, 9
`collapse()`, 13, 14
`convert_to_logmean`, 10
`convert_to_logsd`, 10
`convert_to_natural()`, 23

Dirichlet, 11
`Dirichlet()`, 13, 19, 24
`discretise`, 12
`discretise()`, 6, 9, 14, 30
`discretize(discretise)`, 12
`Distributions`, 4, 5, 11, 13, 14, 16, 17, 21, 25, 30

Exp (Exponential), 14
Exponential, 14
`Exponential()`, 13

`fix_parameters`, 15
`fix_parameters()`, 8, 19, 24, 28, 30
Fixed, 16
`Fixed()`, 4, 8, 13, 28

Gamma, 16
`Gamma()`, 13, 14
`get_distribution`, 17

`get_parameters`, 18
`get_parameters()`, 14
`get_pmf`, 18
`get_pmf()`, 12, 14, 15, 24

`has_uncertainty`, 19
`has_uncertainty()`, 24

`is_constrained`, 20

LogNormal, 20
`LogNormal()`, 10, 13, 14

`max.dist_spec`, 21
`mean()`, 8, 28
`mean.dist_spec`, 22

`new_dist_spec`, 23
NonParametric, 24
`NonParametric()`, 4, 11, 13
Normal, 25
`Normal()`, 13, 14

`plot.dist_spec`, 26
`print.dist_spec`, 27

`quantile.dist_spec`, 28
`quantile.dist_spec()`, 8
`quantile.multi_dist_spec`
 (`quantile.dist_spec`), 28

`rgamma()`, 29

`sample_dist`, 29
`sample_dist()`, 6, 13–15
`sd()`, 8, 28

Weibull, 30
`Weibull()`, 13