

Package ‘ssdtools’

September 2, 2020

Version 0.3.2

Title Species Sensitivity Distributions

Description Species sensitivity distributions are cumulative probability distributions which are fitted to toxicity concentrations for different species as described by Posthuma et al.(2001) <isbn:9781566705783>. The ssdtools package uses Maximum Likelihood to fit distributions such as the log-normal, gamma, log-logistic, log-Gumbel, Gompertz and Weibull. The user can provide custom distributions. Multiple distributions can be averaged using Information Criteria. Confidence intervals on hazard concentrations and proportions are produced by parametric bootstrapping.

URL <https://github.com/bcgov/ssdtools>

BugReports <https://github.com/bcgov/ssdtools/issues>

License Apache License (== 2.0) | file LICENSE

Depends R (>= 3.4)

Imports chk, fitdistrplus, abind, actuar, ggplot2, graphics, grid, lifecycle, tibble, scales, stats, VGAM, Rcpp

Suggests covr, knitr, rmarkdown, testthat, readr, rlang, purrr, tidyr, dplyr, R.rsp, mle.tools, reshape2

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1.9000

VignetteBuilder knitr, R.rsp

Language en-US

LinkingTo Rcpp

NeedsCompilation yes

Author Joe Thorley [aut, cre, ctr] (<<https://orcid.org/0000-0002-7683-4592>>), Carl Schwarz [aut, ctr],

Angeline Tillmanns [ctb],
 Ali Azizishirazi [ctb],
 Rebecca Fisher [ctb],
 David Fox [ctb],
 Kathleen McTavish [ctb],
 Heather Thompson [ctb],
 Andy Teucher [ctb],
 Emilie Doussantousse [ctb],
 Stephanie Hazlitt [ctb],
 Nan-Hung Hsieh [ctb],
 Sergio Ibarra Espinosa [ctb],
 Province of British Columbia [cph]

Maintainer Joe Thorley <joe@poissonconsulting.ca>

Repository CRAN

Date/Publication 2020-09-02 06:50:09 UTC

R topics documented:

autoplot.fitdist	3
boron_data	4
boron_dists	5
boron_hc5	5
boron_lnorm	6
boron_pred	6
burrIII2	7
burrIII3	8
ccme_data	9
comma_signif	10
fluazinam_dists	11
fluazinam_lnorm	11
fluazinam_pred	12
gamma	12
geom_hcintersect	13
geom_ssd	14
geom_xribbon	16
gompertz	17
is.fitdist	18
is.fitdistcens	18
is.fitdists	19
is.fitdistscens	19
lgumbel	20
lnorm	21
nobs.fitdist	22
nobs.fitdistcens	22
npars	23
pareto	24
predict.fitdist	25

predict.fitdistcens	26
predict.fitdists	27
predict.fitdistscens	28
sburrIII2	29
ssdtools-ggproto	30
ssd_ecd	30
ssd_exposure	31
ssd_fit_dists	32
ssd_gof	33
ssd_hc	34
ssd_hp	37
ssd_match_moments	39
ssd_plot	40
ssd_plot_cdf	41
ssd_plot_cf	42
stat_ssd	42
subset.fitdists	44
test_data	44
weibull	45

Index **46**

autoplot.fitdist *Autoplot fitdist*

Description

Plots the cumulative distribution function (cdf) using the ggplot2 generic.

Usage

```
## S3 method for class 'fitdist'
autoplot(object, ...)

## S3 method for class 'fitdists'
autoplot(object, ...)

## S3 method for class 'fitdistcens'
autoplot(object, ...)
```

Arguments

object The object.
 ... Unused.

Functions

- autoplot.fitdists: Autoplot fitdists
- autoplot.fitdistcens: Autoplot fitdistcens

Examples

```
ggplot2::autoplot(boron_lnorm)
ggplot2::autoplot(boron_dists)
fluazinam_lnorm$censdata$right[3] <- fluazinam_lnorm$censdata$left[3] * 1.5
fluazinam_lnorm$censdata$left[5] <- NA
ggplot2::autoplot(fluazinam_lnorm)
```

boron_data

CCME Species Sensitivity Data for Boron

Description

Species Sensitivity Data from the Canadian Council of Ministers of the Environment.

Usage

```
boron_data
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 28 rows and 5 columns.

Details

Additional information is available from <http://ceqg-rcqe.ccme.ca/download/en/324/>.

The columns are as follows

Chemical The chemical (chr).

Species The species binomial name (chr).

Concentration The chemical concentration (dbl).

Units The units (chr).

Group The taxonomic group (fctr).

See Also

```
ccme_data
```

Examples

```
head(ccme_data)
```

boron_dists	<i>fitdists for CCME Boron Data</i>
-------------	-------------------------------------

Description

A fitdists object for Species Sensitivity Data for Boron.

Usage

```
boron_dists
```

Format

An object of class fitdists of length 3.

Examples

```
boron_dists
```

boron_hc5	<i>Model averaged 5 hazard concentration for CCME Boron Data</i>
-----------	--

Description

A data frame of the predictions based on 10000 bootstrap.

Usage

```
boron_hc5
```

Format

An object of class tbl_df (inherits from tbl, data.frame) with 1 rows and 6 columns.

Details

percent The percent of species affected (int).

est The estimated concentration (dbl).

se The standard error of the estimate (dbl).

lcl The lower confidence limit (dbl).

se The upper confidence limit (dbl).

dist The distribution (chr).

Examples

```
boron_hc5
```

boron_lnorm	<i>fitdist for CCME Boron Data</i>
-------------	------------------------------------

Description

A fitdist object for Species Sensitivity Data for Boron with the lnorm distribution.

Usage

```
boron_lnorm
```

Format

An object of class fitdist of length 17.

Examples

```
boron_lnorm
```

boron_pred	<i>Model averaged predictions for CCME Boron Data</i>
------------	---

Description

A data frame of the predictions based on 1,000 bootstrap iterations.

Usage

```
boron_pred
```

Format

An object of class tbl_df (inherits from tbl, data.frame) with 99 rows and 6 columns.

Details

percent The percent of species affected (int).

est The estimated concentration (dbl).

se The standard error of the estimate (dbl).

lcl The lower confidence limit (dbl).

se The upper confidence limit (dbl).

dist The distribution (chr).

Examples

```
head(boron_pred)
```

`burrIII2`*Burr Type III Two-Parameter Distribution*

Description

Probability density, cumulative distribution, inverse cumulative distribution, random sample and starting values functions.

Usage

```
dburrIII2(x, locationlog = 0, scalelog = 1, log = FALSE)
```

```
pburrIII2(q, locationlog = 0, scalelog = 1, lower.tail = TRUE, log.p = FALSE)
```

```
qburrIII2(p, locationlog = 0, scalelog = 1, lower.tail = TRUE, log.p = FALSE)
```

```
rburrIII2(n, locationlog = 0, scalelog = 1)
```

Arguments

<code>x</code>	A numeric vector of values.
<code>locationlog</code>	location on log scale parameter.
<code>scalelog</code>	scale on log scale parameter.
<code>log</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>q</code>	vector of quantiles.
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
<code>log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

Details

The `burrIII2` distribution has been deprecated for the identical `llogis` distribution.

Value

A numeric vector.

See Also

[llogis\(\)](#)

Examples

```
x <- seq(0.01, 5, by = 0.01)
plot(x, dburrIII2(x), type = "l")
```

burrIII3

*Burr Type III Three-Parameter Distribution***Description**

Density, distribution function, quantile function and random generation for the Burr Type III Three-Parameter distribution with `lshape` and `lscale` parameters.

Usage

```
dburrIII3(x, lshape1 = 0, lshape2 = 0, lscale = 0, log = FALSE)
```

```
qburrIII3(
  p,
  lshape1 = 0,
  lshape2 = 0,
  lscale = 0,
  lower.tail = TRUE,
  log.p = FALSE
)
```

```
pburrIII3(
  q,
  lshape1 = 0,
  lshape2 = 0,
  lscale = 0,
  lower.tail = TRUE,
  log.p = FALSE
)
```

```
rburrIII3(n, lshape1 = 0, lshape2 = 0, lscale = 0)
```

```
sburrIII3(x)
```

Arguments

<code>x</code>	The object.
<code>lshape1</code>	shape1 parameter on the log scale.
<code>lshape2</code>	shape2 parameter on the log scale.
<code>lscale</code>	scale parameter on the log scale.
<code>log</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>p</code>	vector of probabilities.
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
<code>log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>q</code>	vector of quantiles.
<code>n</code>	number of observations.

Details

The Burr 12 distribution from the actuar package is used as a base. The Burr III distribution is the distribution of $1/x$ where x has the Burr Type 12 distribution. refer to <https://www.itl.nist.gov/div898/software/dataplot/refman> for details. The shape1, shape2, and scale parameters are on the log(scale) as these must be positive.

Value

dburrIII3 gives the density, pburrIII3 gives the distribution function, qburrIII3 gives the quantile function, and rburrIII3 generates random samples.

See Also

[actuar::dburr\(\)](#)

Examples

```
x <- rburrIII3(1000)
hist(x, freq = FALSE, col = "gray", border = "white")
curve(dburrIII3(x), add = TRUE, col = "red4", lwd = 2)
```

ccme_data

CCME Species Sensitivity Data

Description

Species Sensitivity Data from the Canadian Council of Ministers of the Environment. The taxonomic groups are Amphibian, Fish, Invertebrate and Plant. Plants includes freshwater algae.

Usage

```
ccme_data
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 144 rows and 5 columns.

Details

Additional information on each of the chemicals is available from the CCME website.

Boron <http://ceqg-rcqe.ccme.ca/download/en/324/>

Cadmium <http://ceqg-rcqe.ccme.ca/download/en/148/>

Chloride <http://ceqg-rcqe.ccme.ca/download/en/337/>

Endosulfan <http://ceqg-rcqe.ccme.ca/download/en/327/>

Glyphosate <http://ceqg-rcqe.ccme.ca/download/en/182/>

Uranium <http://ceqg-rcqe.ccme.ca/download/en/328/>

Silver <http://ceqg-rcqe.ccme.ca/download/en/355/>

Chemical The chemical (chr).

Species The species binomial name (chr).

Conc The chemical concentration (dbl).

Group The taxonomic group (fctr).

Units The units (chr).

Examples

```
head(ccme_data)
```

comma_signif

Comma and Significance Formatter

Description

Comma and Significance Formatter

Usage

```
comma_signif(x, digits = 1, ...)
```

Arguments

x A numeric vector to format.

digits Deprecated, use accuracy instead.

... Other arguments passed on to `base::format()`.

Value

A function that returns a character vector.

See Also

[scales::comma\(\)](#)

Examples

```
comma_signif(1199)
```

fluazinam_dists	<i>fitdists for fitdistrplus fluazinam Data</i>
-----------------	---

Description

A fitdists object for Species Sensitivity Data for Fluazinam.

Usage

```
fluazinam_dists
```

Format

An object of class fitdistscens (inherits from fitdists) of length 3.

See Also

[fitdistrplus::fluazinam\(\)](#)

Examples

```
fluazinam_dists
```

fluazinam_lnorm	<i>fitdist for CCME Boron Data</i>
-----------------	------------------------------------

Description

A fitdist object for Species Sensitivity Data for Boron with the lnorm distribution.

Usage

```
fluazinam_lnorm
```

Format

An object of class fitdistcens of length 17.

See Also

[fitdistrplus::fluazinam\(\)](#)

Examples

```
fluazinam_lnorm
```

fluazinam_pred	<i>Model averaged predictions for fluazinam</i>
----------------	---

Description

A data frame of the predictions.

Usage

```
fluazinam_pred
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 99 rows and 6 columns.

Details

percent The percent of species affected (int).

est The estimated concentration (dbl).

se The standard error of the estimate (dbl).

lcl The lower confidence limit (dbl).

se The upper confidence limit (dbl).

dist The distribution (chr).

Examples

```
head(fluazinam_pred)
```

gamma	<i>Gamma Distribution</i>
-------	---------------------------

Description

Probability density, cumulative distribution, inverse cumulative distribution, random sample and starting values functions.

Usage

```
dgamma(x, shape = 1, scale = 1, log = FALSE)
```

```
pgamma(q, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)
```

```
qgamma(p, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)
```

```
rgamma(n, shape = 1, scale = 1)
```

```
sgamma(x)
```

Arguments

x	A numeric vector of values.
shape	A string of the column in data for the shape aesthetic.
scale	scale parameter.
log	logical; if TRUE, probabilities p are given as log(p).
q	vector of quantiles.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities p are given as log(p).
p	vector of probabilities.
n	number of observations.

Value

A numeric vector.

See Also

[stats::dgamma\(\)](#)

Examples

```
x <- seq(0.01, 5, by = 0.01)
plot(x, dgamma(x), type = "l")
```

geom_hcintersect

Hazard Concentration Intersection

Description

For each x and y value, geom_hcintersect() plots the intersection.

Usage

```
geom_hcintersect(
  mapping = NULL,
  data = NULL,
  xintercept,
  yintercept,
  na.rm = FALSE,
  show.legend = NA,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by <code>aes()</code> or <code>aes_()</code> . If specified and <code>inherit.aes = TRUE</code> (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 <code>NULL</code> , the default, the data is inherited from the plot data as specified in the call to <code>ggplot()</code> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <code>fortify()</code> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code> , and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
xintercept	The x-value for the intersect
yintercept	The y-value for the intersect.
na.rm	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
...	Other arguments passed on to <code>layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>colour = "red"</code> or <code>size = 3</code> . They may also be parameters to the paired <code>geom/stat</code> .

Examples

```
ggplot2::ggplot(boron_data, ggplot2::aes(x = Conc)) +
  geom_ssd() +
  geom_hcintersect(xintercept = 1.5, yintercept = 0.05)
```

 geom_ssd

Plot Species Sensitivity Data

Description

Uses the empirical cumulative density/distribution to visualize species sensitivity data.

Usage

```
geom_ssd(
  mapping = NULL,
  data = NULL,
  stat = "ssd",
  position = "identity",
```

```

na.rm = FALSE,
show.legend = NA,
inherit.aes = TRUE,
...
)

```

Arguments

mapping	Set of aesthetic mappings created by <code>aes()</code> or <code>aes_()</code> . If specified and <code>inherit.aes = TRUE</code> (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 <code>NULL</code> , the default, the data is inherited from the plot data as specified in the call to <code>ggplot()</code> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <code>fortify()</code> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code> , and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
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 <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If <code>FALSE</code> , 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. <code>borders()</code> .
...	Other arguments passed on to <code>layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>colour = "red"</code> or <code>size = 3</code> . They may also be parameters to the paired <code>geom/stat</code> .

Examples

```

ggplot2::ggplot(boron_data, ggplot2::aes(x = Conc)) +
  geom_ssd()

```

geom_xribbon

*Ribbons Plot***Description**

For each y value, `geom_xribbon` displays an x interval defined by `xmin` and `xmax`.

Usage

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

Arguments

<code>mapping</code>	Set of aesthetic mappings created by <code>aes()</code> or <code>aes_()</code> . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply <code>mapping</code> if there is no plot mapping.
<code>data</code>	The data to be displayed in this layer. There are three options: If <code>NULL</code> , the default, the data is inherited from the plot data as specified in the call to <code>ggplot()</code> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <code>fortify()</code> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code> , and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
<code>stat</code>	The statistical transformation to use on the data for this layer, as a string.
<code>position</code>	Position adjustment, either as a string, or the result of a call to a position adjustment function.
<code>na.rm</code>	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
<code>show.legend</code>	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
<code>inherit.aes</code>	If <code>FALSE</code> , 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. <code>borders()</code> .

... 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.

gompertz *Gompertz Distribution*

Description

Probability density, cumulative distribution, inverse cumulative distribution, random sample and starting values functions.

Usage

```
dgompertz(x, llocation = 0, lshape = 0, log = FALSE)
```

```
pgompertz(q, llocation = 0, lshape = 0, lower.tail = TRUE, log.p = FALSE)
```

```
qgompertz(p, llocation = 0, lshape = 0, lower.tail = TRUE, log.p = FALSE)
```

```
rgompertz(n, llocation = 0, lshape = 0)
```

```
sgompertz(x)
```

Arguments

<code>x</code>	A numeric vector of values.
<code>llocation</code>	location parameter on the log scale.
<code>lshape</code>	shape parameter on the log scale.
<code>log</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>q</code>	vector of quantiles.
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
<code>log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

Value

A numeric vector.

See Also

[stats::dgamma\(\)](#)

Examples

```
x <- seq(0.01, 5, by = 0.01)
plot(x, dgompertz(x), type = "l")
```

is.fitdist	<i>Is fitdist</i>
------------	-------------------

Description

Tests whether an object is a fitdist.

Usage

```
is.fitdist(x)
```

Arguments

x The object.

Value

A flag.

Examples

```
is.fitdist(boron_lnorm)
is.fitdist(boron_dists)
is.fitdist(boron_dists[["lnorm"]])
```

is.fitdistcens	<i>Is censored fitdist</i>
----------------	----------------------------

Description

Tests whether an object is a censored fitdist.

Usage

```
is.fitdistcens(x)
```

Arguments

x The object.

Value

A flag.

Examples

```
is.fitdistcens(boron_lnorm)
is.fitdistcens(fluzinam_lnorm)
```

is.fitdists	<i>Is fitdists</i>
-------------	--------------------

Description

Tests whether an object is a fitdists.

Usage

```
is.fitdists(x)
```

Arguments

x The object.

Value

A flag.

Examples

```
is.fitdists(boron_lnorm)
is.fitdists(boron_dists)
```

is.fitdistscens	<i>Is censored fitdists</i>
-----------------	-----------------------------

Description

Tests whether an object is a censored fitdists.

Usage

```
is.fitdistscens(x)
```

Arguments

x The object.

Value

A flag.

Examples

```
is.fitdistscens(boron_dists)
is.fitdistscens(fluzinam_lnorm)
is.fitdistscens(fluzinam_dists)
```

`lgumbel`*Log-Gumbel Distribution*

Description

Probability density, cumulative distribution, inverse cumulative distribution, random sample and starting values functions.

Usage

```
dlgumbel(x, locationlog = 0, scalelog = 1, log = FALSE)
```

```
plgumbel(q, locationlog = 0, scalelog = 1, lower.tail = TRUE, log.p = FALSE)
```

```
qlgumbel(p, locationlog = 0, scalelog = 1, lower.tail = TRUE, log.p = FALSE)
```

```
rlgumbel(n, locationlog = 0, scalelog = 1)
```

```
slgumbel(x)
```

Arguments

<code>x</code>	A numeric vector of values.
<code>locationlog</code>	location on log scale parameter.
<code>scalelog</code>	scale on log scale parameter.
<code>log</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>q</code>	vector of quantiles.
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
<code>log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

Value

A numeric vector.

Examples

```
x <- seq(0.01, 5, by = 0.01)
plot(x, dlgumbel(x), type = "l")
```

lnorm	<i>Log-Normal Distribution</i>
-------	--------------------------------

Description

Probability density, cumulative distribution, inverse cumulative distribution, random sample and starting values functions.

Usage

```
dlnorm(x, meanlog = 0, sdlog = 1, log = FALSE)
plnorm(q, meanlog = 0, sdlog = 1, lower.tail = TRUE, log.p = FALSE)
qlnorm(p, meanlog = 0, sdlog = 1, lower.tail = TRUE, log.p = FALSE)
rlnorm(n, meanlog = 0, sdlog = 1)
slnorm(x)
```

Arguments

x	A numeric vector of values.
meanlog	mean on log scale parameter.
sdlog	standard deviation on log scale parameter.
log	logical; if TRUE, probabilities p are given as log(p).
q	vector of quantiles.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities p are given as log(p).
p	vector of probabilities.
n	number of observations.

Value

A numeric vector.

See Also

[stats::dlnorm\(\)](#)

Examples

```
x <- seq(0.01, 5, by = 0.01)
plot(x, dlnorm(x), type = "l")
```

nobs.fitdist	<i>Number of Observations</i>
--------------	-------------------------------

Description

Number of Observations

Usage

```
## S3 method for class 'fitdist'
nobs(object, ...)
```

Arguments

object	The object.
...	Unused.

Examples

```
stats::nobs(boron_1norm)
```

nobs.fitdistcens	<i>Number of Observations</i>
------------------	-------------------------------

Description

Number of Observations

Usage

```
## S3 method for class 'fitdistcens'
nobs(object, ...)
```

Arguments

object	The object.
...	Unused.

Examples

```
stats::nobs(fluzazinam_1norm)
```

npars

Get the Number of Parameters

Description

Get the Number of Parameters

Usage

```
npars(x, ...)  
  
## S3 method for class 'fitdist'  
npars(x, ...)  
  
## S3 method for class 'fitdistcens'  
npars(x, ...)  
  
## S3 method for class 'fitdists'  
npars(x, ...)
```

Arguments

x	The object.
...	Unused.

Value

A count indicating the number of parameters.

Methods (by class)

- fitdist: Get the Number of parameters
- fitdistcens: Get the Number of parameters
- fitdists: Get the Number of parameters

Examples

```
npars(boron_lnorm)  
npars(boron_dists)  
npars(fluzinam_lnorm)  
npars(fluzinam_dists)
```

pareto

*Pareto Distribution***Description**

Probability density, cumulative distribution, inverse cumulative distribution, random sample and starting values functions.

Usage

```
dpareto(x, scale = 1, shape = 1, log = FALSE)
```

```
qpareto(p, scale = 1, shape = 1, lower.tail = TRUE, log.p = FALSE)
```

```
ppareto(q, scale = 1, shape = 1, lower.tail = TRUE, log.p = FALSE)
```

```
rpareto(n, scale = 1, shape = 1)
```

```
spareto(x)
```

Arguments

<code>x</code>	A numeric vector of values.
<code>scale</code>	scale parameter.
<code>shape</code>	A string of the column in data for the shape aesthetic.
<code>log</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>p</code>	vector of probabilities.
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
<code>log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$.
<code>q</code>	vector of quantiles.
<code>n</code>	number of observations.

Details

The pareto distribution has been deprecated as it is not suitable for SSD data. The functions are wrappers on the equivalent VGAM functions.

Value

A numeric vector.

See Also

[VGAM::dpareto\(\)](#)

Examples

```
x <- seq(0.01, 5, by = 0.01)
plot(x, dpareto(x), type = "l")
```

predict.fitdist	<i>Predict fitdist</i>
-----------------	------------------------

Description

Predict fitdist

Usage

```
## S3 method for class 'fitdist'
predict(
  object,
  percent = 1:99,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
  ...
)
```

Arguments

object	The object.
percent	A numeric vector of percentages.
ci	A flag specifying whether to estimate confidence intervals (by parametric bootstrapping).
level	A number between 0 and 1 of the confidence level.
nboot	A count of the number of bootstrap samples to use to estimate the se and confidence limits.
parallel	A string specifying the type of parallel operation to be used ('no', 'snow' or 'multicore').
ncpus	A count of the number of parallel processes to use.
...	Unused.

Examples

```
predict(boron_lnorm, percent = c(5L, 50L))
```

predict.fitdistcens *Predict censored fitdist*

Description

Predict censored fitdist

Usage

```
## S3 method for class 'fitdistcens'
predict(
  object,
  percent = 1:99,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
  ...
)
```

Arguments

object	The object.
percent	A numeric vector of percentages.
ci	A flag specifying whether to estimate confidence intervals (by parametric bootstrapping).
level	A number between 0 and 1 of the confidence level.
nboot	A count of the number of bootstrap samples to use to estimate the se and confidence limits.
parallel	A string specifying the type of parallel operation to be used ('no', 'snow' or 'multicore').
ncpus	A count of the number of parallel processes to use.
...	Unused.

Examples

```
predict(fluazinam_lnorm, percent = c(5L, 50L))
```

predict.ftdists *Predict ftdists*

Description

Predict ftdists

Usage

```
## S3 method for class 'ftdists'
predict(
  object,
  percent = 1:99,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
  average = TRUE,
  ic = "aicc",
  ...
)
```

Arguments

object	The object.
percent	A numeric vector of percentages.
ci	A flag specifying whether to estimate confidence intervals (by parametric bootstrapping).
level	A number between 0 and 1 of the confidence level.
nboot	A count of the number of bootstrap samples to use to estimate the se and confidence limits.
parallel	A string specifying the type of parallel operation to be used ('no', 'snow' or 'multicore').
ncpus	A count of the number of parallel processes to use.
average	A flag specifying whether to model average the estimates.
ic	A string specifying which information-theoretic criterion ('aic', 'aicc' or 'bic') to use for model averaging .
...	Unused.

Examples

```
predict(boron_dists)
```

predict.fitdistscens *Predict censored fitdists*

Description

Predict censored fitdists

Usage

```
## S3 method for class 'fitdistscens'
predict(
  object,
  percent = 1:99,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
  average = TRUE,
  ic = "aic",
  ...
)
```

Arguments

object	The object.
percent	A numeric vector of percentages.
ci	A flag specifying whether to estimate confidence intervals (by parametric bootstrapping).
level	A number between 0 and 1 of the confidence level.
nboot	A count of the number of bootstrap samples to use to estimate the se and confidence limits.
parallel	A string specifying the type of parallel operation to be used ('no', 'snow' or 'multicore').
ncpus	A count of the number of parallel processes to use.
average	A flag specifying whether to model average the estimates.
ic	A string specifying which information-theoretic criterion ('aic', 'aicc' or 'bic') to use for model averaging .
...	Unused.

Examples

```
predict(fluazinam_dists)
```

sburrIII2

Log-Logistic Distribution

Description

Probability density, cumulative distribution, inverse cumulative distribution, random sample and starting values functions.

Usage

```
sburrIII2(x)
```

```
dlllog(x, locationlog = 0, scalelog = 1, log = FALSE)
```

```
qllog(p, locationlog = 0, scalelog = 1, lower.tail = TRUE, log.p = FALSE)
```

```
pllog(q, locationlog = 0, scalelog = 1, lower.tail = TRUE, log.p = FALSE)
```

```
rllog(n, locationlog = 0, scalelog = 1)
```

```
slllog(x)
```

```
dlllogis(x, locationlog = 0, scalelog = 1, log = FALSE)
```

```
pllogis(q, locationlog = 0, scalelog = 1, lower.tail = TRUE, log.p = FALSE)
```

```
qllogis(p, locationlog = 0, scalelog = 1, lower.tail = TRUE, log.p = FALSE)
```

```
rllogis(n, locationlog = 0, scalelog = 1)
```

```
slllogis(x)
```

Arguments

x	A numeric vector of values.
locationlog	location on log scale parameter.
scalelog	scale on log scale parameter.
log	logical; if TRUE, probabilities p are given as log(p).
p	vector of probabilities.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities p are given as log(p).
q	vector of quantiles.
n	number of observations.

Details

The llog distribution has been deprecated for the identical llogis distribution.

Value

A numeric vector.

See Also

[stats::dlogis\(\)](#)

Examples

```
x <- seq(0.01, 5, by = 0.01)
plot(x, dllogis(x), type = "l")
```

<code>ssdtools-ggproto</code>	<i>Base ggproto classes for ggplot2</i>
-------------------------------	---

Description

Base ggproto classes for ggplot2

See Also

[ggplot2::ggplot2-ggproto\(\)](#)

<code>ssd_ecd</code>	<i>Empirical Cumulative Density</i>
----------------------	-------------------------------------

Description

Empirical Cumulative Density

Usage

```
ssd_ecd(x, ties.method = "first")
```

Arguments

<code>x</code>	a numeric, complex, character or logical vector.
<code>ties.method</code>	a character string specifying how ties are treated, see ‘Details’; can be abbreviated.

Value

A numeric vector of the empirical cumulative density.

Examples

```
ssd_eed(1:10)
```

ssd_exposure	<i>Percent Exposure</i>
--------------	-------------------------

Description

Calculates average proportion exposed based on log-normal distribution of concentrations.

Usage

```
ssd_exposure(x, meanlog = 0, sdlog = 1, nboot = 1000)
```

Arguments

x	The object.
meanlog	A number of the mean of the exposure concentrations on the log scale.
sdlog	A number of the standard deviation of the exposure concentrations on the log scale.
nboot	The number of samples to use to calculate the exposure.

Value

A number of the proportion exposed.

Examples

```
set.seed(10)
ssd_exposure(boron_1norm)
ssd_exposure(boron_1norm, meanlog = 1)
ssd_exposure(boron_1norm, meanlog = 1, sdlog = 1)
```

 ssd_fit_dists

Fit Distributions

Description

Fits one or more distributions to species sensitivity data.

Usage

```
ssd_fit_dists(
  data,
  left = "Conc",
  right = left,
  weight = NULL,
  dists = c("llogis", "gamma", "lnorm"),
  computable = FALSE,
  silent = FALSE
)
```

Arguments

data	A data frame.
left	A string of the column in data with the concentrations.
right	A string of the column in data with the right concentration values.
weight	A string of the column in data with the weightings (or NULL)
dists	A character vector of the distribution names.
computable	A flag specifying whether to only return fits with numerically computable standard errors.
silent	A flag indicating whether fits should fail silently.

Details

By default the 'llogis', 'gamma' and 'lnorm' distributions are fitted to the data. The `ssd_fit_dists` function has also been tested with the 'gompertz', 'lgumbel' and 'weibull' distributions.

If `weight` specifies a column in the data frame with positive integers, weighted estimation occurs. However, currently only the resultant parameter estimates are available (via `coef`).

If the `right` argument is different to the `left` argument then the data are considered to be censored.

The fits are performed using `fitdistrplus::fitdist()` (and `fitdistrplus::fitdistcens()` in the case of censored data). The method used is "mle" (maximum likelihood estimation) which means that numerical optimization is carried out in `fitdistrplus::mledist()` using `stats::optim()` unless finite bounds are supplied in the (lower and upper) in which it is carried out using `stats::constrOptim()`. In both cases the "Nelder-Mead" method is used.

Value

An object of class `fitdists` (a list of `fitdistrplus::fitdist()` objects).

Examples

```
ssd_fit_dists(boron_data)
data(fluazinam, package = "fitdistrplus")
ssd_fit_dists(fluazinam, left = "left", right = "right")
```

ssd_gof

Goodness of Fit

Description

Returns a `tbl` data frame with the following columns

dist The distribution name (chr)

aic Akaike's Information Criterion (dbl)

bic Bayesian Information Criterion (dbl)

and if the data are non-censored

aicc Akaike's Information Criterion corrected for sample size (dbl)

and if there are 8 or more samples

ad Anderson-Darling statistic (dbl)

ks Kolmogorov-Smirnov statistic (dbl)

cvm Cramer-von Mises statistic (dbl)

In the case of an object of class `fitdists` the function also returns

delta The Information Criterion differences (dbl)

weight The Information Criterion weights (dbl)

where `delta` and `weight` are based on `aic` for censored data and `aicc` for non-censored data.

Usage

```
ssd_gof(x, ...)
```

```
## S3 method for class 'fitdist'
ssd_gof(x, ...)
```

```
## S3 method for class 'fitdists'
ssd_gof(x, ...)
```

```
## S3 method for class 'fitdistcens'
```

```
ssd_gof(x, ...)

## S3 method for class 'fitdistscens'
ssd_gof(x, ...)
```

Arguments

```
x          The object.
...        Unused.
```

Value

A tbl data frame of the gof statistics.

Methods (by class)

- fitdist: Goodness of Fit
- fitdists: Goodness of Fit
- fitdistscens: Goodness of Fit
- fitdistscens: Goodness of Fit

Examples

```
ssd_gof(boron_lnorm)
ssd_gof(boron_dists)
ssd_gof(boron_lnorm)
ssd_gof(boron_dists)
ssd_gof(fluzazinam_lnorm)
ssd_gof(fluzazinam_lnorm)
```

ssd_hc	<i>Hazard Concentration</i>
--------	-----------------------------

Description

Gets concentrations that protect specified percentages of species.

Usage

```
ssd_hc(x, ...)

## S3 method for class 'list'
ssd_hc(x, percent = 5, hc = 5, ...)

## S3 method for class 'fitdist'
ssd_hc(
  x,
```

```
    percent = 5,
    hc = 5,
    ci = FALSE,
    level = 0.95,
    nboot = 1000,
    parallel = NULL,
    ncpus = 1,
    ...
)

## S3 method for class 'fitdistscens'
ssd_hc(
  x,
  percent = 5,
  hc = 5,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
  ...
)

## S3 method for class 'fitdists'
ssd_hc(
  x,
  percent = 5,
  hc = 5,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
  average = TRUE,
  ic = "aicc",
  ...
)

## S3 method for class 'fitdistscens'
ssd_hc(
  x,
  percent = 5,
  hc = 5,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
```

```

    average = TRUE,
    ic = "aic",
    ...
)

```

Arguments

x	The object.
...	Unused.
percent	A numeric vector of percentages.
hc	A numeric vector of percentages.
ci	A flag specifying whether to estimate confidence intervals (by parametric bootstrapping).
level	A number between 0 and 1 of the confidence level.
nboot	A count of the number of bootstrap samples to use to estimate the se and confidence limits.
parallel	A string specifying the type of parallel operation to be used ('no', 'snow' or 'multicore').
ncpus	A count of the number of parallel processes to use.
average	A flag specifying whether to model average the estimates.
ic	A string specifying which information-theoretic criterion ('aic', 'aicc' or 'bic') to use for model averaging .

Value

A data frame of the percent and concentrations.

Methods (by class)

- list: Hazard Percent list of distributions
- fitdist: Hazard Percent fitdist
- fitdistcens: Hazard Percent fitdistcens
- fitdists: Hazard Percent fitdists
- fitdistscens: Hazard Percent fitdistscens

Examples

```

ssd_hc(list("lnorm" = NULL))
ssd_hc(list("lnorm" = list(meanlog = 2, sdlog = 1)))
ssd_hc(boron_lnorm, c(0, 1, 30, Inf))
ssd_hc(fluzazinam_lnorm, c(0, 1, 30, Inf))
ssd_hc(boron_dists, c(0, 1, 30, Inf))
ssd_hc(fluzazinam_dists, c(0, 1, 30, Inf))

```

ssd_hp	<i>Hazard Percent</i>
--------	-----------------------

Description

Gets percent species protected at specified concentrations.

Usage

```
ssd_hp(x, ...)  
  
## S3 method for class 'fitdist'  
ssd_hp(  
  x,  
  conc,  
  ci = FALSE,  
  level = 0.95,  
  nboot = 1000,  
  parallel = NULL,  
  ncpus = 1,  
  ...  
)  
  
## S3 method for class 'fitdistcens'  
ssd_hp(  
  x,  
  conc,  
  ci = FALSE,  
  level = 0.95,  
  nboot = 1000,  
  parallel = NULL,  
  ncpus = 1,  
  ...  
)  
  
## S3 method for class 'fitdists'  
ssd_hp(  
  x,  
  conc,  
  ci = FALSE,  
  level = 0.95,  
  nboot = 1000,  
  parallel = NULL,  
  ncpus = 1,  
  average = TRUE,  
  ic = "aicc",  
  ...  
)
```

```

)

## S3 method for class 'fitdistscens'
ssd_hp(
  x,
  conc,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
  average = TRUE,
  ic = "aic",
  ...
)

```

Arguments

x	The object.
...	Unused.
conc	A numeric vector of concentrations.
ci	A flag specifying whether to estimate confidence intervals (by parametric bootstrapping).
level	A number between 0 and 1 of the confidence level.
nboot	A count of the number of bootstrap samples to use to estimate the se and confidence limits.
parallel	A string specifying the type of parallel operation to be used ('no', 'snow' or 'multicore').
ncpus	A count of the number of parallel processes to use.
average	A flag specifying whether to model average the estimates.
ic	A string specifying which information-theoretic criterion ('aic', 'aicc' or 'bic') to use for model averaging .

Value

A data frame of the conc and percent.

Methods (by class)

- `fitdist`: Hazard Percent fitdist
- `fitdistscens`: Hazard Percent fitdistscens
- `fitdists`: Hazard Percent fitdists
- `fitdistscens`: Hazard Percent fitdistscens

Examples

```
ssd_hp(boron_lnorm, c(0, 1, 30, Inf))
ssd_hp(fluazinam_lnorm, c(0, 1, 30, Inf))
ssd_hp(boron_dists, c(0, 1, 30, Inf))
ssd_hp(fluazinam_dists, c(0, 1, 30, Inf))
```

ssd_match_moments	<i>Match Moments</i>
-------------------	----------------------

Description

Match Moments

Usage

```
ssd_match_moments(
  dists = c("llogis", "gamma", "lnorm"),
  meanlog = 1,
  sdlog = 1,
  nsim = 1e+05
)
```

Arguments

dists	A character vector of the distribution names.
meanlog	A number of the mean on the log scale.
sdlog	A number of the standard deviation on the log scale.
nsim	A positive whole number of the number of simulations to generate.

Value

A named list of the parameter values that produce a distribution with moments closest to the meanlog and sdlog.

See Also

[ssd_plot_cdf\(\)](#).

Examples

```
ssd_match_moments()
```

 ssd_plot

SSD Plot

Description

SSD Plot

Usage

```
ssd_plot(
  data,
  pred,
  left = "Conc",
  right = left,
  label = NULL,
  shape = NULL,
  color = NULL,
  size = 2.5,
  xlab = "Concentration",
  ylab = "Percent of Species Affected",
  ci = TRUE,
  ribbon = FALSE,
  hc = 5L,
  shift_x = 3
)
```

Arguments

data	A data frame.
pred	A data frame of the predictions.
left	A string of the column in data with the concentrations.
right	A string of the column in data with the right concentration values.
label	A string of the column in data with the labels.
shape	A string of the column in data for the shape aesthetic.
color	A string of the column in data for the color aesthetic.
size	A number for the size of the labels.
xlab	A string of the x-axis label.
ylab	A string of the x-axis label.
ci	A flag specifying whether to estimate confidence intervals (by parametric bootstrapping).
ribbon	A flag indicating whether to plot the confidence interval as a grey ribbon as opposed to green solid lines.
hc	A count between 1 and 99 indicating the percent hazard concentration (or NULL).
shift_x	The value to multiply the label x values by.

Examples

```
ssd_plot(boron_data, boron_pred, label = "Species", shape = "Group")
```

ssd_plot_cdf

Plot Cumulative Distribution Function

Description

Plots the cdf.

Usage

```
ssd_plot_cdf(x, ...)
```

```
## S3 method for class 'list'
```

```
ssd_plot_cdf(x, xlab = "Concentration", ylab = "Species Affected", ...)
```

```
## S3 method for class 'fitdist'
```

```
ssd_plot_cdf(x, xlab = "Concentration", ylab = "Species Affected", ...)
```

```
## S3 method for class 'fitdistcens'
```

```
ssd_plot_cdf(x, xlab = "Concentration", ylab = "Species Affected", ...)
```

```
## S3 method for class 'fitdists'
```

```
ssd_plot_cdf(x, xlab = "Concentration", ylab = "Species Affected", ...)
```

Arguments

x	The object.
...	Unused.
xlab	A string of the x-axis label.
ylab	A string of the x-axis label.

Methods (by class)

- list: Plot list
- fitdist: Plot CDF fitdist
- fitdistcens: Plot CDF fitdistcens
- fitdists: Plot CDF fitdists

Examples

```
ssd_plot_cdf(boron_lnorm)
ssd_plot_cdf(boron_lnorm)
fluazinam_lnorm$censdata$right[3] <- fluazinam_lnorm$censdata$left[3] * 1.5
fluazinam_lnorm$censdata$left[5] <- NA
ssd_plot_cdf(fluazinam_lnorm)
ssd_plot_cdf(boron_dists)
```

ssd_plot_cf	<i>Cullen and Frey Plot</i>
-------------	-----------------------------

Description

Plots a Cullen and Frey graph of the skewness and kurtosis for non-censored data.

Usage

```
ssd_plot_cf(data, left = "Conc")
```

```
ssd_cfplot(data, left = "Conc")
```

Arguments

data A data frame.

left A string of the column in data with the concentrations.

Functions

- `ssd_cfplot`: Deprecated Cullen and Frey Plot

See Also

[fitdistrplus::descdist\(\)](#)

Examples

```
ssd_plot_cf(boron_data)
```

stat_ssd	<i>Plot Species Sensitivity Data</i>
----------	--------------------------------------

Description

Uses the empirical cumulative density/distribution to visualize species sensitivity data.

Usage

```
stat_ssd(
  mapping = NULL,
  data = NULL,
  geom = "point",
  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 <code>inherit.aes = TRUE</code> (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 <code>NULL</code> , the default, the data is inherited from the plot data as specified in the call to ggplot() . A <code>data.frame</code> , 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 <code>data.frame</code> , and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
geom	The geometric object to use display the data
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If <code>FALSE</code> , 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 <code>colour = "red"</code> or <code>size = 3</code> . They may also be parameters to the paired geom/stat.

See Also

[geom_ssd\(\)](#)

Examples

```
ggplot2::ggplot(boron_data, ggplot2::aes(x = Conc)) +
  stat_ssd()
```

subset.fitdists	<i>Subset fitdists</i>
-----------------	------------------------

Description

Subset fitdists

Usage

```
## S3 method for class 'fitdists'
subset(x, select = names(x), ...)
```

Arguments

x	The object.
select	A character vector of the distributions to select.
...	Unused.

Examples

```
subset(boron_dists, c("gamma", "lnorm"))
```

test_data	<i>Test Data</i>
-----------	------------------

Description

Data to test ssdtools.

Usage

```
test_data
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 141 rows and 2 columns.

Details

Chemical The chemical (`chr`).

Conc The chemical concentration (`dbl`).

Examples

```
head(test_data)
```

weibull

Weibull Distribution

Description

Density, distribution function, quantile function and random generation for the weibull distribution with parameters shape and scale.

Usage

```
dweibull(x, shape = 1, scale = 1, log = FALSE)
```

```
pweibull(q, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)
```

```
qweibull(p, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)
```

```
rweibull(n, shape = 1, scale = 1)
```

Arguments

x	A numeric vector of values.
shape	A string of the column in data for the shape aesthetic.
scale	scale parameter.
log	logical; if TRUE, probabilities p are given as log(p).
q	vector of quantiles.
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities p are given as log(p).
p	vector of probabilities.
n	number of observations.

Value

A numeric vector.

See Also

[stats::dweibull\(\)](#)

Examples

```
x <- seq(0.01, 5, by = 0.01)
plot(x, dweibull(x), type = "l")
```

Index

- * **datasets**
 - boron_data, 4
 - boron_dists, 5
 - boron_hc5, 5
 - boron_lnorm, 6
 - boron_pred, 6
 - ccme_data, 9
 - fluazinam_dists, 11
 - fluazinam_lnorm, 11
 - fluazinam_pred, 12
 - ssdtools-ggproto, 30
 - test_data, 44
- actuar::dburr(), 9
- aes(), 14–16, 43
- aes_(), 14–16, 43
- autoplot.fitdist, 3
- autoplot.fitdistcens
 - (autoplot.fitdist), 3
- autoplot.fitdists (autoplot.fitdist), 3
- base::format(), 10
- borders(), 15, 16, 43
- boron_data, 4
- boron_dists, 5
- boron_hc5, 5
- boron_lnorm, 6
- boron_pred, 6
- burrIII2, 7
- burrIII3, 8
- ccme_data, 9
- comma_signif, 10
- dburrIII2 (burrIII2), 7
- dburrIII3 (burrIII3), 8
- dgamma (gamma), 12
- dgompertz (gompertz), 17
- dlgumbel (lgumbel), 20
- dllog (sburrIII2), 29
- dllogis (sburrIII2), 29
- dlnorm (lnorm), 21
- dpareto (pareto), 24
- dweibull (weibull), 45
- fitdistrplus::descdist(), 42
- fitdistrplus::fitdist(), 32, 33
- fitdistrplus::fitdistcens(), 32
- fitdistrplus::fluazinam(), 11
- fitdistrplus::mledist(), 32
- fluazinam_dists, 11
- fluazinam_lnorm, 11
- fluazinam_pred, 12
- fortify(), 14–16, 43
- gamma, 12
- geom_hcintersect, 13
- geom_ssd, 14
- geom_ssd(), 43
- geom_xribbon, 16
- GeomHcintersect (ssdtools-ggproto), 30
- GeomSsd (ssdtools-ggproto), 30
- GeomSsdcens (ssdtools-ggproto), 30
- GeomXribbon (ssdtools-ggproto), 30
- ggplot(), 14–16, 43
- gompertz, 17
- is.fitdist, 18
- is.fitdistcens, 18
- is.fitdists, 19
- is.fitdistscens, 19
- layer(), 14, 15, 17, 43
- lgumbel, 20
- llogis (sburrIII2), 29
- llogis(), 7
- lnorm, 21
- nobs.fitdist, 22
- nobs.fitdistcens, 22
- npars, 23

pareto, 24
 pburrIII2 (burrIII2), 7
 pburrIII3 (burrIII3), 8
 pgamma (gamma), 12
 pgompertz (gompertz), 17
 plgumbel (lgumbel), 20
 pllog (sburrIII2), 29
 pllogis (sburrIII2), 29
 plnorm (lnorm), 21
 ppareto (pareto), 24
 predict.fitdist, 25
 predict.fitdistscens, 26
 predict.fitdists, 27
 predict.fitdistscens, 28
 pweibull (weibull), 45

 qburrIII2 (burrIII2), 7
 qburrIII3 (burrIII3), 8
 qgamma (gamma), 12
 qgompertz (gompertz), 17
 qlgumbel (lgumbel), 20
 qllog (sburrIII2), 29
 qllogis (sburrIII2), 29
 qlnorm (lnorm), 21
 qpareto (pareto), 24
 qweibull (weibull), 45

 rburrIII2 (burrIII2), 7
 rburrIII3 (burrIII3), 8
 rgamma (gamma), 12
 rgompertz (gompertz), 17
 rlgumbel (lgumbel), 20
 rllog (sburrIII2), 29
 rllogis (sburrIII2), 29
 rlnorm (lnorm), 21
 rpareto (pareto), 24
 rweibull (weibull), 45

 sburrIII2, 29
 sburrIII3 (burrIII3), 8
 scales::comma(), 10
 sgamma (gamma), 12
 sgompertz (gompertz), 17
 slgumbel (lgumbel), 20
 sllog (sburrIII2), 29
 sllogis (sburrIII2), 29
 slnorm (lnorm), 21
 spareto (pareto), 24
 ssd_cfplot (ssd_plot_cf), 42
 ssd_ecd, 30
 ssd_exposure, 31
 ssd_fit_dists, 32
 ssd_gof, 33
 ssd_hc, 34
 ssd_hp, 37
 ssd_match_moments, 39
 ssd_plot, 40
 ssd_plot_cdf, 41
 ssd_plot_cdf(), 39
 ssd_plot_cf, 42
 ssdtools-ggproto, 30
 stat_ssd, 42
 stats::constrOptim(), 32
 stats::dgamma(), 13, 17
 stats::dlnorm(), 21
 stats::dlogis(), 30
 stats::dweibull(), 45
 stats::optim(), 32
 StatSsd (ssdtools-ggproto), 30
 StatSsdscens (ssdtools-ggproto), 30
 subset.fitdists, 44

 test_data, 44

 VGAM::dpareto(), 24

 weibull, 45