Package ‘ssdtools’

October 27, 2021

Version  0.3.7
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 (>= 4.1)
Imports  chk, fitdistrplus, abind, actuar, ggplot2, graphics, grid, lifecycle, tibble, utils, 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  7.1.2
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],
Nadine Hussein [ctb],
Nan-Hung Hsieh [ctb],
Sergio Ibarra Espinosa [ctb],
Province of British Columbia [cph]

Maintainer Joe Thorley <joe@poissonconsulting.ca>
Repository CRAN
Date/Publication 2021-10-27 21:30:03 UTC

**R topics documented:**

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoplot.fitdist</td>
<td>3</td>
</tr>
<tr>
<td>boron_data</td>
<td>4</td>
</tr>
<tr>
<td>boron_dists</td>
<td>5</td>
</tr>
<tr>
<td>boron_HC5</td>
<td>5</td>
</tr>
<tr>
<td>boron_INorm</td>
<td>6</td>
</tr>
<tr>
<td>boron_pred</td>
<td>6</td>
</tr>
<tr>
<td>burrIII2</td>
<td>7</td>
</tr>
<tr>
<td>burrIII3</td>
<td>8</td>
</tr>
<tr>
<td>ccme_data</td>
<td>10</td>
</tr>
<tr>
<td>comma_signif</td>
<td>11</td>
</tr>
<tr>
<td>dllog</td>
<td>11</td>
</tr>
<tr>
<td>fluazinam_dists</td>
<td>13</td>
</tr>
<tr>
<td>fluazinam_INorm</td>
<td>13</td>
</tr>
<tr>
<td>fluazinam_pred</td>
<td>14</td>
</tr>
<tr>
<td>gamma</td>
<td>14</td>
</tr>
<tr>
<td>geom_HCintersect</td>
<td>15</td>
</tr>
<tr>
<td>geom_ssd</td>
<td>17</td>
</tr>
<tr>
<td>geom_Xribbon</td>
<td>18</td>
</tr>
<tr>
<td>gompertz</td>
<td>19</td>
</tr>
<tr>
<td>is.fitdist</td>
<td>20</td>
</tr>
<tr>
<td>is.fitdistcens</td>
<td>21</td>
</tr>
<tr>
<td>is.fitdists</td>
<td>21</td>
</tr>
<tr>
<td>is.fitdistcens</td>
<td>22</td>
</tr>
<tr>
<td>lgumbel</td>
<td>23</td>
</tr>
<tr>
<td>lnorm</td>
<td>24</td>
</tr>
<tr>
<td>nobs.fitdist</td>
<td>25</td>
</tr>
<tr>
<td>npars</td>
<td>25</td>
</tr>
<tr>
<td>pareto</td>
<td>26</td>
</tr>
</tbody>
</table>
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.

See Also
ggplot2::autoplot() and ssd_plot_cdf()
```
Examples

ggplot2::autoplot(boron_lnorm)
ggplot2::autoplot(boron_dists)
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**


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`

Other boron: `boron_dists`, `boron_hc5`, `boron_lnorm`, `boron_pred`

**Examples**

```
head(ccme_data)
```
`boron_dists`

`fitdists for CCME Boron Data`

**Description**

A `fitdists` object for Species Sensitivity Data for Boron.

**Usage**

`boron_dists`

**Format**

An object of class `fitdists` of length 3.

**See Also**

Other `boron`: `boron_data`, `boron_hc5`, `boron_lnorm`, `boron_pred`

**Examples**

```r
boron_dists
```

---

`boron_hc5`

*Model Averaged 5 Hazard Concentration for CCME Boron Data*

**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).
- **ucl**  The upper confidence limit (dbl).
- **dist**  The distribution (chr).
See Also
Other boron: boron_data, boron_dists, boron_lnorm, boron_pred

Examples
boron_hc5

---

boron_lnorm fitdist for CCME Boron Data

---

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.

See Also
Other boron: boron_data, boron_dists, boron_hc5, boron_pred

Examples
boron_lnorm

---

boron_pred Model Averaged Predictions for CCME Boron Data

---

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.
**burrIII2**

**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).
- **ucl**: The upper confidence limit (dbl).
- **dist**: The distribution (chr).

**See Also**

Other boron: `boron_data`, `boron_dists`, `boron_hc5`, `boron_lnorm`

**Examples**

```r
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**

```r
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)
sburrIII2(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).
- **q**: vector of quantiles.
- **lower.tail**: logical; if TRUE (default), probabilities are P[X <= 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.

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
)
Arguments

- `x` The object.
- `lshape1` shape1 parameter on the log scale.
- `lshape2` shape2 parameter on the log scale.
- `lscale` scale parameter on the log scale.
- `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 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/refman2/auxillar/bu3pdf.htm 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

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


- **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

```r
head(ccme_data)
```
comma_signif

Colon and Significance Formatter

Description

By default the numeric vectors are first rounded to three significant figures. Then scales::comma is only applied to values greater than or equal to 1000 to ensure that labels are permitted to have different numbers of decimal places.

Usage

comma_signif(x, digits = 3, ...)

Arguments

x A numeric vector to format.
digits A whole number specifying the number of significant figures
... Additional arguments passed to scales::comma.

Value

A character vector.

Examples

comma_signif(c(0.1, 1, 10, 1000))
scales::comma(c(0.1, 1, 10, 1000))

dllog

Log-Logistic Distribution

Description

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

Usage

dllog(x, locationlog = 0, scalelog = 1, log = FALSE)

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

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

dllog(n, locationlog = 0, scalelog = 1)
sllng(x)

dllogis(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)

sllng(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 <= 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")
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()
Other fluazinam: fluazinam_lnorm, fluazinam_pred

Examples
fluazinam_dists

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

Usage
fluazinam_lnorm

Format
An object of class fitdistcens of length 17.

See Also
fitdistrplus::fluazinam()
Other fluazinam: fluazinam_dists, fluazinam_pred

Examples
fluazinam_lnorm
Model Averaged Predictions for Fluazinam

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).
- **ucl**: The upper confidence limit (dbl).
- **dist**: The distribution (chr).

See Also
- `fitdistrplus::fluazinam()`
- Other fluazinam: `fluazinam_dists, fluazinam_lnorm`

Examples
head(fluazinam_pred)

Gamma Distribution

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")

Description

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

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

Arguments

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

See Also

Other ggplot: `geom_ssd()`, `geom_xribbon()`

Examples

```r
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**

```r
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 `aes()` or `aes()`. If specified and `inherit.aes` = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

- `data`  
  The data to be displayed in this layer. There are three options:
  - If NULL, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.
  - A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.
  - A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. `~ head(.x,10)`).

- `stat`  
  The statistical transformation to use on the data for this layer, as a string.

- `position`  
  Position adjustment, either as a string, or the result of a call to a position adjustment function.

- `na.rm`  
  If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.

- `show.legend`  
  Logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.

- `inherit.aes`  
  If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behaviour from the default plot specification, e.g. `borders()`.
Other arguments passed on to layer(). These are often aesthetics, used to set an aesthetic to a fixed value, like colour = "red" or size = 3. They may also be parameters to the paired geom/stat.

See Also

ssd_plot_cdf()

Other ggplot: geom_hcintersect(), geom_xribbon()

Examples

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

geom_xribbon

Ribbons Plot

Description

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

Usage

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapping</td>
<td>Set of aesthetic mappings created by aes() or aes_. If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.</td>
</tr>
<tr>
<td>data</td>
<td>The data to be displayed in this layer. There are three options: If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot(). A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x, 10)).</td>
</tr>
</tbody>
</table>
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**

- `x`: A numeric vector of values.
- `llocation`: location parameter on the log scale.
- `lshape`: shape parameter on the log scale.
- `log`: logical; if TRUE, probabilities p are given as log(p).
is.fitdist

Tests whether an object is a fitdist.

Usage

is.fitdist(x)

Arguments

x

The object.

Value

A flag.

See Also

Other is: is.fitdistcens(), is.fitdistcens(), is.fitdists()
**is.fitdistcens**  
*Is censored fitdist*

**Description**
Tests whether an object is a censored fitdist.

**Usage**

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

**Arguments**

- `x`  
  The object.

**Value**
A flag.

**See Also**
Other is: `is.fitdistscens()`, `is.fitdist()`

**Examples**

```r
is.fitdistcens(boron_lnorm)
is.fitdistcens(fluazinam_lnorm)
```

---

**is.fitdists**  
*Is fitdists*

**Description**
Tests whether an object is a fitdists.

**Usage**

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

**Arguments**

- `x`  
  The object.

**Value**
A flag.
See Also

Other is: `is.fitdistcens()`, `is.fitdistscens()`, `is.fitdist()`

Examples

```r
is.fitdist(boron_lnorm)
is.fitdist(boron_dists)
```

---

### is.fitdistscens

**Is censored fitdists**

- **is.fitdistscens**

**Description**

Tests whether an object is a censored fitdists.

**Usage**

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

**Arguments**

- `x` The object.

**Value**

A flag.

**See Also**

Other is: `is.fitdistcens()`, `is.fitdists()`, `is.fitdist()`

**Examples**

```r
is.fitdistscens(boron_dists)
is.fitdistscens(fluazinam_lnorm)
is.fitdistscens(fluazinam_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**

- **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).
- **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.

**Examples**

```r
x <- seq(0.01, 5, by = 0.01)
plot(x, dlgumbel(x), type = "l")
```
**Log-Normal Distribution**

**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 <= 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**
```R
x <- seq(0.01, 5, by = 0.01)
plot(x, dlnorm(x), type = "l")
```
### nobs.fitdist

#### Number of Observations

**Description**

Number of Observations

**Usage**

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

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

**Arguments**

- `object`: The object.
- `...`: Unused.

**See Also**

`stats::nobs()`

**Examples**

```r
stats::nobs(boron_lnorm)
stats::nobs(fluazinam_lnorm)
```

### npars

#### Number of Parameters

**Description**

Get the Number of Parameters

**Usage**

```r
npars(x, ...)
```

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

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

```r
## 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(fluazinam_lnorm)
npars(fluazinam_dists)

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)
pppareto(q, scale = 1, shape = 1, lower.tail = TRUE, log.p = FALSE)
rrpareto(n, scale = 1, shape = 1)
spareto(x)
predict.fitdist

Arguments

x          A numeric vector of values.
scale      scale parameter.
shape      A string of the column in data for the shape aesthetic.
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 <= 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 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

Description

Predict

Usage

## S3 method for class 'fitdist'
predict(
  object,
  percent = 1:99,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
### S3 method for class 'fitdistcens'

predict(
  object,
  percent = 1:99,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
  ...  
)

### S3 method for class 'fitdistcens'

predict(
  object,
  percent = 1:99,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  parallel = NULL,
  ncpus = 1,
  average = TRUE,
  ic = "aic",
  ...  
)

### S3 method for class 'fitdistcens'

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.
... Unused.
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.

See Also

stats::predict()

Examples

predict(boron_lnorm, percent = c(5L, 50L))
predict(fluazinam_lnorm, percent = c(5L, 50L))
predict(boron_dists)
predict(fluazinam_dists)
### ssd_ecd

**Empirical Cumulative Density**

**Description**

Empirical Cumulative Density

**Usage**

```r
gssd_ecd(x, ties.method = "first")
```

**Arguments**

- **x**: a numeric, complex, character or logical vector.
- **ties.method**: a character string specifying how ties are treated, see ‘Details’; can be abbreviated.

**Value**

A numeric vector of the empirical cumulative density.

**Examples**

```r
gssd_ecd(1:10)
```

### ssd_exposure

**Percent Exposure**

**Description**

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

**Usage**

```r
gssd_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.
**ssd_fit_dists**

**Value**

A number of the proportion exposed.

**Examples**

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

---

**ssd_fit_dists**  
*Fit Distributions*

**Description**

Fits one or more distributions to species sensitivity data.

**Usage**

```r
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

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

---

### Goodness of Fit

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.
**ssd_gof**

**Usage**

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

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

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

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

```r
## 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
- `fitdistcens`: Goodness of Fit
- `fitdistscens`: Goodness of Fit

**Examples**

```r
ssd_gof(boron_lnorm)
ssd_gof(boron_dists)
ssd_gof(fluazinam_lnorm)
```
ssd_hc  

**Hazard Concentration**

**Description**

Gets concentrations that protect specified percentages of species.

**Usage**

```r
ssd_hc(x, ...)
```

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

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

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

```r
## S3 method for class 'fitdists'
ssd_hc(
x,
    percent = 5,
    hc = 5,
    ci = FALSE,
    level = 0.95,
    nboot = 1000,
    parallel = NULL,
    ncpus = 1,
    ...
)
```
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
- fitdistscens: Hazard Percent fitdistscens
- fitdists: Hazard Percent fitdists
- fitdistscens: Hazard Percent fitdistscens

Examples

```r
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(fluazinam_lnorm, c(0, 1, 30, Inf))
ssd_hc(boron_dists, c(0, 1, 30, Inf))
ssd_hc(fluazinam_dists, c(0, 1, 30, Inf))
```

Description

Gets percent species protected at specified concentrations.

Usage

```r
ssd_hp(x, ...)
```
ssd_hp

```
parallel = NULL,
ncpus = 1,
...
```

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

```r
## 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.
ssd_match_moments

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
- fitdistcens: Hazard Percent fitdistcens
- 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))

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.
Description

Plots species sensitivity data.

Usage

```r
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 cumulative distribution function (cdf).

Usage

```r
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

```r
ssd_plot_cdf(boron_lnorm)
ssd_plot_cdf(boron_lnorm)
fluazinam_lnorm$censdata$left[5] <- NA
ssd_plot_cdf(fluazinam_lnorm)
ssd_plot_cdf(boron_dists)
```

---

**ssd_plot_cf**

_Cullen and Frey Plot_

**Description**

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

**Usage**

```r
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**

```r
ssd_plot_cf(boron_data)
```
**stat_ssd**

*Plot Species Sensitivity Data*

**Description**

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

**Usage**

```r
code
```

```r
def stat_ssd(
  mapping = NULL,
  data = NULL,
  geom = "point",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
code
```

**Arguments**

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

**See Also**

`geom_ssd()` and `ssd_plot_cdf()`

**Examples**

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

---

**subset.fitdists**

**Subset fitdists**

**Description**

Subset fitdists

**Usage**

```r
## 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**

```r
subset(boron_dists, c("gamma", "lnorm"))
```
test_data  Test Data

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)
weibull

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

* **boron**
  - boron_data, 4
  - boron_dists, 5
  - boron_hc5, 5
  - boron_lnorm, 6
  - boron_pred, 6

* **datasets**
  - boron_data, 4
  - boron_dists, 5
  - boron_hc5, 5
  - boron_lnorm, 6
  - boron_pred, 6
  - ccme_data, 10
  - fluazinam_dists, 13
  - fluazinam_lnorm, 13
  - fluazinam_pred, 14
  - ssdtools-ggproto, 29
  - test_data, 44

* **fluazinam**
  - fluazinam_dists, 13
  - fluazinam_lnorm, 13
  - fluazinam_pred, 14

* **ggplot**
  - geom_hcintersect, 15
  - geom_ssd, 17
  - geom_xribbon, 18

* **is**
  - is.fitdist, 20
  - is.fitdistcens, 21
  - is.fitdists, 21
  - is.fitdistscens, 22

* **predict**
  - predict.fitdist, 27

  actuar::dburr(), 9
  aes(), 16–18, 42
  aes_(), 16–18, 42
  autoplot.fitdist, 3
  autoplot.fitdistcens (autoplot.fitdist), 3
  autoplot.fitdists (autoplot.fitdist), 3
  borders(), 17, 19, 42
  boron_data, 4, 5–7
  boron_dists, 4, 5, 6, 7
  boron_hc5, 4, 5, 6, 7
  boron_lnorm, 4–6, 6, 7
  boron_pred, 4–6, 6
  burrIII2, 7
  burrIII3, 8
  ccme_data, 4, 10
  comma_signif, 11
  dburriI2 (burrIII2), 7
  dburriI3 (burrIII3), 8
  dgamma (gamma), 14
  dgompertz (gompertz), 19
  dlog (dllog), 11
  dlogis (dlogis), 11
  dlnorm (lnorm), 24
  dpareto (pareto), 26
  dweibull (weibull), 44

  fitdistrplus::descdist(), 41
  fitdistrplus::fitdist(), 32
  fitdistrplus::fitdistcens(), 32
  fitdistrplus::fluazinam(), 13, 14
  fitdistrplus::mledist(), 32
  fluazinam_dists, 13, 13, 14
  fluazinam_lnorm, 13, 13, 14
  fluazinam_pred, 13, 13, 14
  fortify(), 16–18, 42

  gamma, 14
  geom_hcintersect, 15, 18, 19
  geom_ssd, 16, 17, 19
  geom_ssd(), 43
  geom_xribbon, 16, 18, 18
  GeomHcintersect (ssdtools-ggproto), 29
GeomSsd (ssdtools-ggproto), 29
GeomSsdcens (ssdtools-ggproto), 29
GeomXribbon (ssdtools-ggproto), 29
ggplot(), 16–18, 42
ggplot2::autoplot(), 3
gompertz, 19
is.fitdist, 20, 21, 22
is.fitdistcens, 20, 21, 22
is.fitdists, 20, 21, 22
is.fitdistscens, 20–22, 22
layer(), 16, 18, 19, 43
lgumbel, 23
llogis (dlllog), 11
llogis(), 8
lnorm, 24

nobs.fitdist, 25
nobs.fitdistcens (nobs.fitdist), 25
npars, 25

pareto, 26
pburrIII2 (burrIII2), 7
pburrIII3 (burrIII3), 8
pgamma (gamma), 14
pgompertz (gompertz), 19
plgumbel (lgumbel), 23
pllog (dlllog), 11
pllogis (dlllog), 11
plnorm (lnorm), 24
ppareto (pareto), 26
predict.fitdist, 27
predict.fitdists (predict.fitdist), 27
predict.fitdistcens (predict.fitdist), 27
predict.fitdists (predict.fitdist), 27
predict.fitdistcens (predict.fitdist), 27
pweibull (weibull), 44

qburrIII2 (burrIII2), 7
qburrIII3 (burrIII3), 8
qgamma (gamma), 14
qgompertz (gompertz), 19
qlgumbel (lgumbel), 23
qllog (dlllog), 11
qllogis (dlllog), 11
qlnorm (lnorm), 24
qpareto (pareto), 26
rweibull (weibull), 44

rburrIII2 (burrIII2), 7
rburrIII3 (burrIII3), 8
rgamma (gamma), 14
rgompertz (gompertz), 19
rlgumbel (lgumbel), 23
rllog (dlllog), 11
rllogis (dlllog), 11
rlnorm (lnorm), 24
rpareto (pareto), 26

sbburrIII2 (burrIII2), 7
sbburrIII3 (burrIII3), 8
scales::comma, 11
sgamma (gamma), 14
sgompertz (gompertz), 19
slgumbel (lgumbel), 23
 sllog (dlllog), 11
slogis (dlllog), 11
slnorm (lnorm), 24
spareto (pareto), 26
ssd_cfpplot (ssd_plot_CF), 41
ssd_ecd, 30
ssd_exposure, 30
ssd_fit_dists, 31
ssd_gof, 32
ssd_hp, 34

ssd_match_moments, 38
ssd_plot, 39
ssd_plot_cdf, 40
ssd_plot_cdf(), 3, 18, 39, 43
ssd_plot_cf, 41
ssdtools-ggproto, 29
stat_ssd, 42
stats::constrOptim(), 32
stats::dgamma(), 15, 20
stats::dnorm(), 24
stats::dlogis(), 12
stats::dweibull(), 45
stats::nobs(), 25
stats::optim(), 32
stats::predict(), 29
StatSsd (ssdtools-ggproto), 29
StatSsdcens (ssdtools-ggproto), 29
subset.fitdists, 43
test_data, 44
VGAM::dpareto(), 27
weibull, 44