Package ‘ssdtools’

November 25, 2018

Version 0.0.3
Title Species Sensitivity Distributions
Description Species sensitivity distributions are cumulative probability distributions which are fitted to toxicity concentrations for multiple species. The ssdtools package uses Maximum Likelihood to fit log-normal, log-logistic, log-Gumbel, Gompertz, gamma or Weibull distributions. Multiple distributions can be averaged using Information Criteria. Confidence intervals can be calculated for the fitted cumulative distribution function or specific hazard concentrations (percentiles). Confidence intervals are currently produced by bootstrapping.

URL https://github.com/bcgov/ssdtools
BugReports https://github.com/bcgov/ssdtools/issues
License Apache License (== 2.0) | file LICENSE
Depends R (>= 3.4.0)
Imports checkr, fitdistrplus, FAdist, ggplot2, graphics, grid, scales, stats, VGAM
Suggests covr, knitr, rmarkdown, testthat, tibble, tidyr, readr, purrr, dplyr
Encoding UTF-8
LazyData true
RoxygenNote 6.1.1
VignetteBuilder knitr
Language en-US
NeedsCompilation no

Author Joe Thorley [aut, cre, ctr] (https://orcid.org/0000-0002-7683-4592), Carl Schwarz [aut, ctr], Ali Azizishirazi [ctb], Angeline Tillmanns [ctb], Stephanie Hazlitt [ctb], Kathleen McTavish [ctb], Emilie Doussantousse [ctb],
Heather Thompson [ctb],
Andy Teucher [ctb],
Province of British Columbia [cph]

Maintainer Joe Thorley <joe@poissonconsulting.ca>
Repository CRAN
Date/Publication 2018-11-25 19:10:03 UTC

R topics documented:

autoplast.fitdist ........................................... 3
autoplast.fitdistcens ...................................... 3
autoplast.fitdists .......................................... 4
autoplast.fitdistscens .................................... 5
boron_data ..................................................... 5
boron_dists .................................................... 6
boron_hc5 ..................................................... 7
boron_lnorm .................................................. 7
boron_pred ..................................................... 8
ccme_data ..................................................... 8
comma_signif .................................................. 9
fluazinam_dist .............................................. 10
fluazinam_lnorm ............................................. 10
fluazinam_pred .............................................. 11
geom_hcintersect .......................................... 11
gnom_ssd ..................................................... 12
gnom_xribbon ................................................. 13
gompertz ....................................................... 14
is.fitdist ...................................................... 15
is.fitdistcens ............................................... 16
is.fitdists ..................................................... 16
is.fitdistscens .............................................. 17
lgumbel ........................................................ 17
lllog ........................................................... 18
nobs.fitdist ................................................... 19
nobs.fitdistcens ............................................ 20
npars ........................................................... 20
pareto .......................................................... 21
predict.fitdist ............................................... 22
predict.fitdistcens ....................................... 23
predict.fitdists .............................................. 23
predict.fitdistscens ..................................... 24
ssdtools-ggproto ........................................... 25
ssd_cfplot .................................................... 25
ssd_ecd ........................................................ 26
ssd_fit_dists ............................................... 26
ssd_gof ........................................................ 27
ssd_hc .......................................................... 28
### autoplot.fitdist

**Description**

Autoplot

**Usage**

```r
## S3 method for class 'fitdist'
autoplot(object, ci = FALSE, hc = 5L,
         xlab = "Concentration", ylab = "Species Affected", ...)
```

**Arguments**

- `object` The object to plot.
- `ci` A flag indicating whether to plot confidence intervals
- `hc` A count between 1 and 99 indicating the percent hazard concentration to plot (or NULL).
- `xlab` A string of the x-axis label.
- `ylab` A string of the x-axis label.
- `...` Unused.

**Examples**

```r
ggplot2::autoplot(boron_lnorm)
```

### autoplot.fitdistcens

**Description**

Autoplot

**Usage**

```r
## S3 method for class 'fitdistcens'
autoplot(object, ci = FALSE, hc = 5L,
         xlab = "Concentration", ylab = "Species Affected", ...)
```
Arguments

- **object**: The object to plot.
- **ci**: A flag indicating whether to plot confidence intervals.
- **hc**: A count between 1 and 99 indicating the percent hazard concentration to plot (or NULL).
- **xlab**: A string of the x-axis label.
- **ylab**: A string of the x-axis label.
- **...**: Unused.

Examples

```r
fluazinam_lnorm$censdata$left[5] <- NA
ggplot2::autoplot(fluazinam_lnorm)
```

---

**Description**

Autoplot

**Usage**

```r
## S3 method for class 'fitdists'
autoplot(object, xlab = "Concentration", ylab = "Species Affected", ic = "aic", ...)
```

**Arguments**

- **object**: The object to plot.
- **xlab**: A string of the x-axis label.
- **ylab**: A string of the x-axis label.
- **ic**: A string indicating which information-theoretic criterion ('aic', 'aicc' or 'bic') to use for model averaging.
- **...**: Unused.

**Examples**

```r
## Not run:
ggplot2::autoplot(boron_dists)
## End(Not run)
```
autoplot.fitdistscens  Autoplot

**Description**

Autoplot

**Usage**

```r
## S3 method for class 'fitdistscens'
autoplot(object, xlab = "Concentration",
         ylab = "Species Affected", ic = "aic", ...)
```

**Arguments**

- `object` The object to plot.
- `xlab` A string of the x-axis label.
- `ylab` A string of the x-axis label.
- `ic` A string indicating which information-theoretic criterion ('aic', 'aicc' or 'bic') to use for model averaging.
- `...` Unused.

**Examples**

```r
## Not run:
ggplot2::autoplot(boron_dists)
## End(Not run)
```

---

**boron_data  CCME Species Sensitivity Data for Boron**

**Description**

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

**Usage**

`boron_data`
boron_dists

**Format**

A tbl data frame:

- **Chemical** The chemical <chr>.
- **Species** The species binomial name <chr>.
- **Concentration** The chemical concentration <dbl>.
- **Units** The units <chr>.
- **Group** The taxonomic group <fctr>.

**Details**


**See Also**

ccme_data

**Examples**

```r
head(ccme_data)
```

---

**Description**

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

**Usage**

`boron_dists`

**Format**

An object of class `fitdists` of length 6.

**Examples**

`boron_dists`
Model averaged 5% hazard concentration for CCME Boron Data

Description
A data frame of the predictions based on 10000 bootstrap.

Usage
boron_hc5

Format
A tbl data frame:
- **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>.

Examples
boron_hc5

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.

Examples
boron_lnorm
Model averaged predictions for CCME Boron Data

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

Usage
boron_pred

Format
A tbl data frame:

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>.

Examples
head(boron_pred)

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
A tbl data frame:

Chemical The chemical <chr>.
Species The species binomial name <chr>.
Concentration The chemical concentration <dbl>.
Units The units <chr>.
Group The taxonomic group <fctr>.
**Details**

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


**Examples**

```r
head(ccme_data)
```

<table>
<thead>
<tr>
<th>comma_signif</th>
<th>Comma and Significance Formatter</th>
</tr>
</thead>
</table>

**Description**

Comma and Significance Formatter

**Usage**

```r
coma_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**

`comma`

**Examples**

```r
coma_signif(1199)
```
Description

A fitdists object for Species Sensitivity Data for Fluazinam.

Usage

fluazinam_dists

Format

An object of class fitdistscens (inherits from fitdists) of length 5.

See Also

fluazinam

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

fluazinam

Examples

fluazinam_lnorm
**fluazinam_pred**

Model averaged predictions for fluazinam

---

**Description**

A data frame of the predictions.

**Usage**

```r
fluazinam_pred
```

**Format**

A tbl data frame:

- **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>.

**Examples**

```r
head(fluazinam_pred)
```

---

**geom_hcintersect**

*Hazard Concentration Intersection*

---

**Description**

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

**Usage**

```r
gem.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.

**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.

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.

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.

Examples

```r
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

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

**mapping**  
Set of aesthetic mappings created by `aes()` or `aes_()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

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

**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.

---

**gompertz**  
*Gompertz Distribution*

Description

Density, distribution function, quantile function and random generation for the Gompertz distribution. The functions are wrappers on the equivalent VGAM functions that return a zero length numeric vector if `x`, `q` or `p` are zero length.

Usage

```r
  dgompertz(x, scale = 1, shape, log = FALSE)

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

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

  rgompertz(n, scale = 1, shape)
```
is.fitdist

Arguments

- `x, q`: vector of quantiles.
- `scale`: scale parameter.
- `shape`: shape parameter.
- `log, log.p`: logical; if TRUE, probabilities p are given as log(p).
- `lower.tail`: logical; if TRUE (default), probabilities are P[X ≤ x], otherwise, P[X > x].
- `p`: vector of probabilities.
- `n`: number of observations.

Value

dgompertz gives the density, pgompertz gives the distribution function, qgompertz gives the quantile function, and rgompertz generates random deviates.

See Also
dgompertz

Examples

```r
x <- rgompertz(1000,1,0.1)
hist(log(x),freq=FALSE,col='gray',border='white')
hist(x,freq=FALSE,col='gray',border='white')
curve(dgompertz(x,1,0.1),add=TRUE,col='red4',lwd=2)
```

Description

Tests whether an object is a fitdist.

Usage

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

Arguments

- `x`: The object to test.

Value

A flag.
Examples

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

---

is.fitdistcens  Is censored fitdist

Description

Tests whether an object is a censored fitdist.

Usage

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

Arguments

- `x`  The object to test.

Value

A flag.

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 to test.

Value

A flag.
is.fitdistscens

Examples

is.fitdist(boron_lnorm)
is.fitdist(boron_dists)

is.fitdistscens  Is censored fitdist

Description

Tests whether an object is a censored fitdist.

Usage

is.fitdistscens(x)

Arguments

x  The object to test.

Value

A flag.

Examples

is.fitdistscens(boron_dists)
is.fitdistscens(fluazinam_lnorm)
is.fitdistscens(fluazinam_dists)

lgumbel  Log-Gumbel Distribution

Description

Density, distribution function, quantile function and random generation for the Log-Gumbel distribution with scale and location parameters.

Usage

dlgumbel(x, location = 0, scale = 0, log = FALSE)
qlgumbel(p, location = 0, scale = 0, lower.tail = TRUE, log.p = FALSE)
plgumbel(q, location = 0, scale = 0, lower.tail = TRUE, log.p = FALSE)
rlgumbel(n, location = 0, scale = 0)
Arguments

- **x, q**: vector of quantiles.
- **location**: location parameter.
- **scale**: scale parameter.
- **log, log.p**: 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].
- **n**: number of observations.

Value

dlgumbel gives the density, plgumbel gives the distribution function, qlgumbel gives the quantile function, and rlgumbel generates random deviates.

Examples

```r
x <- rlgumbel(1000, 1, 0.1)
hist(log(x), freq=FALSE, col='gray', border='white')
hist(x, freq=FALSE, col='gray', border='white')
curve(dlgumbel(x, 1, 0.1), add=TRUE, col='red', lwd=2)
```

---

**llog**

*Log-Logistic Distribution*

**Description**

Density, distribution function, quantile function and random generation for the log-logistic distribution with shape and scale parameters.

**Usage**

- `dllog(x, shape = 1, scale = 1, log = FALSE)`
- `qllog(p, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)`
- `pllog(q, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)`
- `rllog(n, shape = 1, scale = 1)`
Arguments

- **x, q**: vector of quantiles.
- **shape**: shape parameter.
- **scale**: scale parameter.
- **log, log.p**: 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].
- **n**: number of observations.

Details

The functions are wrappers to export the identical functions from the FAdist package.

Value

dllog gives the density, pllog gives the distribution function, qllog gives the quantile function, and rllog generates random deviates.

See Also

dllog

Examples

```r
x <- rllog(r1000)
hist(x,freq=FALSE,col='gray',border='white')
curve(dllog(x),add=TRUE,col='red4',lwd=2)
```

<table>
<thead>
<tr>
<th>nobs.fitdist</th>
<th>Number of Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Description

Number of Observations

Usage

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

Arguments

- **object**: The object.
- **...**: Unused.

Examples

```r
stats::nobs(boron_lnorm)
```
**nobs.fitdistcens**  
*Number of Observations*

**Description**
Number of Observations

**Usage**

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

**Arguments**
- `object` The object.
- `...` Unused.

**Examples**

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

---

**npars**  
*Get the Number of Parameters*

**Description**
Get the Number of Parameters

**Usage**

```r
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**

```r
npars(boron_lnorm)
npars(boron_dists)
```

---

**Description**

Density, distribution function, quantile function and random generation for the Pareto distribution with parameters `scale` and `shape`. The functions are wrappers on the equivalent VGAM functions.

**Usage**

```r
dpareto(x, scale = 1, shape, log = FALSE)
qpareto(q, scale = 1, shape, lower.tail = TRUE, log.p = FALSE)
ppareto(p, scale = 1, shape, lower.tail = TRUE, log.p = FALSE)
rpareto(n, scale = 1, shape)
```

**Arguments**

- `x`, `q` vector of quantiles.
- `scale` alpha parameter.
- `shape` k parameter.
- `log`, `log.p` logical; if TRUE, probabilities `p` are given as `log(p)`.
- `lower.tail` logical; if TRUE (default), probabilities are $P[X \leq x]$; otherwise, $P[X > x]$.
- `p` vector of probabilities.
- `n` number of observations.

**Value**

dpareto gives the density, ppareto gives the distribution function, qpareto gives the quantile function, and rpareto generates random deviates.
predict.fitdist

See Also
dpareto

Examples

```r
x <- rpareto(1000, 1, 0.1)
hist(log(x), freq=FALSE, col='gray', border='white')
hist(x, freq=FALSE, col='gray', border='white')
curve(dpareto(x, 1, 0.1), add=TRUE, col='red4', lwd=2)
```

predict.fitdist

**Description**

Predict fitdist

**Usage**

```r
## S3 method for class 'fitdist'
predict(object, percent = 1:99, nboot = 1000,
        level = 0.95, ...)
```

**Arguments**

- `object`: The object.
- `percent`: A numeric vector of the densities to calculate the estimated concentrations for.
- `nboot`: A count of the number of bootstrap samples to use to estimate the se and confidence limits.
- `level`: The confidence level.
- `...`: Unused

**Examples**

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

**Predict censored fitdist**

**Description**

Predict censored fitdist

**Usage**

```r
## S3 method for class 'fitdistcens'
predict(object, percent = 1:99, nboot = 1000,
         level = 0.95, ...)
```

**Arguments**

- **object**  
  The object.

- **percent**  
  A numeric vector of the densities to calculate the estimated concentrations for.

- **nboot**  
  A count of the number of bootstrap samples to use to estimate the se and confidence limits.

- **level**  
  The confidence level.

- **...**  
  Unused

**Examples**

```r
## Not run:
predict(fluazinam_lnorm, percent = c(5L, 50L))

## End(Not run)
```

---

predict.fitdists  

**Predict fitdist**

**Description**

Predict fitdist

**Usage**

```r
## S3 method for class 'fitdists'
predict(object, percent = 1:99, nboot = 1000,
         ic = "aicc", average = TRUE, level = 0.95, ...)
```
Arguments

- `object`: The object.
- `percent`: A numeric vector of the densities to calculate the estimated concentrations for.
- `nboot`: A count of the number of bootstrap samples to use to estimate the se and confidence limits.
- `ic`: A string indicating which information-theoretic criterion ('aic', 'aicc' or 'bic') to use for model averaging.
- `average`: A flag indicating whether to model-average.
- `level`: The confidence level.
- `...`: Unused

Examples

```r
## Not run:
predict(boron_dists)
## End(Not run)
```

---

**predict.fitdistscens**

predict Censored fitdists

Description

Predict Censored fitdists

Usage

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

Arguments

- `object`: The object.
- `percent`: A numeric vector of the densities to calculate the estimated concentrations for.
- `nboot`: A count of the number of bootstrap samples to use to estimate the se and confidence limits.
- `ic`: A string indicating which information-theoretic criterion ('aic', 'aicc' or 'bic') to use for model averaging.
- `average`: A flag indicating whether to model-average.
- `level`: The confidence level.
- `...`: Unused
### Examples

```r
## Not run:
predict(flauazinam_dists)
```
```
## End(Not run)
```

---

### ssdtools-ggproto

**Base ggproto classes for ggplot2**

---

### Description

Base ggproto classes for ggplot2

### See Also

- ggplot2-ggproto

---

### ssd_cfplot

**Cullen and Frey Plot**

---

### Description

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

### Usage

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

### Arguments

- `data`: A data frame.
- `left`: A string of the column in data with the left concentration values.

### See Also

- descdist

### Examples

```r
ssd_cfplot(boron_data)
```
ssd_ecd

Empirical Cumulative Density

Description

Empirical Cumulative Density

Usage

ssd_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

ssd_ecd(1:10)

---

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("lnorm", "llog", "gompertz", "lgumbel", "gamma", "weibull"),
   silent = FALSE)

Arguments

data A data frame.
left A string of the column in data with the left concentration values.
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 distributions to fit.
silent A flag indicating whether fits should fail silently.
Details

By default the ‘lnorm’, ‘llog’, ‘gompertz’, ‘lgumbel’, ‘gamma’ and ‘weibull’ distributions are fitted to the data. The ssd_fit_dists function has also been tested with the ‘pareto’ distribution.

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. It may be possible to use artificial censoring to improve the estimates in the extreme tails (Liu et al 2018).

Value

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

References


Examples

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

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 'fitdistcens'
ssd_gof(x, ...)

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

Arguments

x  The object.
...

Value

A tbl data frame of the gof statistics.

Methods (by class)

• fitdist: Goodness of Fit
• fitdistcens: Goodness of Fit
• fitdists: Goodness of Fit

Examples

ssd_gof(boron_lnorm)
ssd_gof(boron_dists)

Description

Estimates with bootstrap confidence intervals the hazard concentration at which by default 5% of
the species are affected.
Usage

```
ssd_hc(x, ...)  

## S3 method for class 'fitdist'
ssd_hc(x, hc = 5L, nboot = 1000, level = 0.95, ...)

## S3 method for class 'fitdistcens'
ssd_hc(x, hc = 5L, nboot = 1000, level = 0.95, ...)

## S3 method for class 'fitdists'
ssd_hc(x, hc = 5L, ic = "aic", average = TRUE,
        nboot = 1000, level = 0.95, ...)

## S3 method for class 'fitdistscens'
ssd_hc(x, hc = 5L, ic = "aic", average = TRUE,
        nboot = 1000, level = 0.95, ...)
```

Arguments

- **x**: The object.
- **...**: Unused.
- **hc**: A number between 0 and 1 indicating the percent hazard concentration to plot (or NULL).
- **nboot**: A count of the number of bootstrap samples to use to estimate the se and confidence limits.
- **level**: The confidence level.
- **ic**: A string indicating which information-theoretic criterion ('aic', 'aicc' or 'bic') to use for model averaging.
- **average**: A flag indicating whether to model-average.

Value

A data frame of the estimate with the standard error and upper and lower confidence intervals.

Methods (by class)

- **fitdist**: Hazard Concentration
- **fitdistcens**: Hazard Concentration
- **fitdists**: Hazard Concentration
- **fitdistscens**: Hazard Concentration
Examples

ssd_hc(boron_lnorm)
## Not run:
ssd_hc(fluazinam_lnorm)

## End(Not run)
## Not run:
ssd_hc(boron_dists)

## End(Not run)
## Not run:
ssd_hc(fluazinam_dists)

## End(Not run)

### 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 indicating whether to plot confidence intervals.

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 to plot (or NULL).

shift_x
The value to multiply the label x values by.
**stat_ssd**

**Plot Species Sensitivity Data**

**Examples**

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

**Description**

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

**Usage**

```r
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 `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.
- `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.
subset.fitdists

See Also

geom_ssd

Examples

```r
ggplot2::ggplot(boron_data, ggrepplot2::aes(x = Conc)) +
stat_ssd()
```

```
subset.fitdists c("gamma", "weibull")
```

---

**subset.fitdists**  
Subset fitdists

**Usage**

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

**Arguments**

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

**Examples**

```r
subset(boron_dists, c("gamma", "weibull"))
```
Index

*Topic **datasets**
  - boron_data, 5
  - boron_dists, 6
  - boron_hc5, 7
  - boron_lnorm, 7
  - boron_pred, 8
  - ccme_data, 8
  - fluazinam_dists, 10
  - fluazinam_lnorm, 10
  - fluazinam_pred, 11
  - ssdtools-ggproto, 25

- aes(), 12–14, 31
- aes_(), 12–14, 31
- autoplot.fitdist, 3
- autoplot.fitdistcens, 3
- autoplot.fitdists, 4
- autoplot.fitdistscens, 5

- base::format(), 9
- borders(), 13, 14, 31
- boron_data, 5
- boron_dists, 6
- boron_hc5, 7
- boron_lnorm, 7
- boron_pred, 8

- ccme_data, 8
- comma, 9
- comma_signif, 9

- descdist, 25
- dgompertz, 15
- dgompertz(gompertz), 14
-dlgumbel(lgumbel), 17
- dlog, 19
- dlog(llog), 18
- dpareto, 22
- dpareto(pareto), 21

- fitdist, 27

- fluazinam, 10
- fluazinam_dists, 10
- fluazinam_lnorm, 10
- fluazinam_pred, 11
- fortify(), 12–14, 31

- geom_hcintersect, 11
- geom_ssd, 12, 32
- geom_xribbon, 13
-GeomHcintersect(ssdtools-ggproto), 25
-GeomSsd(ssdtools-ggproto), 25
-GeomSsdCens(ssdtools-ggproto), 25
-GeomXribbon(ssdtools-ggproto), 25

ggplot(), 12–14, 31

gompertz, 14

- is.fitdist, 15
- is.fitdistcens, 16
- is.fitdists, 16
- is.fitdistscens, 17

- layer(), 12–14, 31
- lgumbel, 17
- llog, 18

- nobs.fitdist, 19
- nobs.fitdistcens, 20
- npars, 20

- pareto, 21
- pgompertz(gompertz), 14
- plgumbel(lgumbel), 17
- plog(llog), 18
- ppareto(pareto), 21
- predict.fitdist, 22
- predict.fitdistcens, 23
- predict.fitdists, 23
- predict.fitdistscens, 24

- qgompertz(gompertz), 14
- qlgumbel(lgumbel), 17
qllog (llog), 18
qpareto (pareto), 21
gompertz (gompertz), 14
rlgumbel (lgumbel), 17
rllog (llog), 18
rpareto (pareto), 21

ssd_cfplot, 25
ssd_ecd, 26
ssd_fit_dists, 26
ssd_gof, 27
ssd_hc, 28
ssd_plot, 30
ssdtols-ggproto, 25
stat_ssd, 31
StatSsd (ssdtols-ggproto), 25
StatSsdcsens (ssdtols-ggproto), 25
subset.fitdists, 32