

# Package ‘ssdtools’

May 17, 2023

**Title** Species Sensitivity Distributions

**Version** 1.0.4

**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 gamma, log-logistic, log-normal and Weibull to censored and/or weighted data. Multiple distributions can be averaged using Akaike Information Criteria. Confidence intervals on hazard concentrations and proportions are produced by parametric bootstrapping.

**License** Apache License (== 2.0) | file LICENSE

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

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

**Depends** R (>= 4.1)

**Imports** abind, chk, doFuture, foreach, furrr, generics, ggplot2, goftest, graphics, grid, lifecycle, parallel, plyr, purrr, Rcpp, scales, ssddata, stats, stringr, tibble, TMB, universals, utils, VGAM

**Suggests** covr, dplyr, fitdistrplus, future, glue, grDevices, knitr, magrittr, mle.tools, R.rsp, readr, reshape2, rlang, rmarkdown, testthat, tidyr, tidyselect, withr

**LinkingTo** Rcpp, RcppEigen, TMB

**VignetteBuilder** knitr, R.rsp

**Config/testthat/edition** 3

**Encoding** UTF-8

**Language** en-US

**LazyData** true

**RoxygenNote** 7.2.3

**NeedsCompilation** yes

**Author** Joe Thorley [aut, cre, ctr] (<<https://orcid.org/0000-0002-7683-4592>>),  
 Carl Schwarz [aut, ctr],  
 Rebecca Fisher [ctb],  
 David Fox [ctb],  
 Angeline Tillmanns [ctb],  
 Ali Azizishirazi [ctb],  
 Kathleen McTavish [ctb],  
 Heather Thompson [ctb],  
 Doug Spry [ctb],  
 Rick van Dam [ctb],  
 Graham Batley [ctb],  
 Heather Thompson [ctb],  
 Andy Teucher [ctb],  
 Seb Dalgarno [ctb] (<<https://orcid.org/0000-0002-3658-4517>>),  
 Emilie Doussantousse [ctb],  
 Stephanie Hazlitt [ctb],  
 Nadine Hussein [ctb],  
 Nan-Hung Hsieh [ctb],  
 Sergio Ibarra Espinosa [ctb],  
 Province of British Columbia [cph],  
 Environment and Climate Change Canada [cph]

**Maintainer** Joe Thorley <joe@poissonconsulting.ca>

**Repository** CRAN

**Date/Publication** 2023-05-17 04:40:02 UTC

## R topics documented:

augment.fitdists . . . . .	3
autoplot.fitdists . . . . .	4
boron_data . . . . .	5
boron_pred . . . . .	6
ccme_data . . . . .	6
coef.fitdists . . . . .	7
comma_signif . . . . .	8
dgompertz . . . . .	8
dist_data . . . . .	9
dlgumbel . . . . .	10
estimates.fitdists . . . . .	10
geom_hcintersect . . . . .	11
geom_ssdpoint . . . . .	12
geom_ssdsegment . . . . .	14
geom_xribbon . . . . .	15
glance.fitdists . . . . .	17
is.fitdists . . . . .	18
is_censored . . . . .	18
predict.fitburrrlioz . . . . .	19
predict.fitdists . . . . .	20

scale_colour_ssd . . . . .	21
ssdtools-ggproto . . . . .	22
ssd_data . . . . .	22
ssd_dists . . . . .	23
ssd_dists_all . . . . .	24
ssd_dists_bcanz . . . . .	24
ssd_ecd . . . . .	25
ssd_ecd_data . . . . .	25
ssd_exposure . . . . .	26
ssd_fit_bcanz . . . . .	27
ssd_fit_burrlioz . . . . .	28
ssd_fit_dists . . . . .	28
ssd_gof . . . . .	30
ssd_hc . . . . .	31
ssd_hc_bcanz . . . . .	33
ssd_hc_burrlioz . . . . .	34
ssd_hp . . . . .	35
ssd_is_censored . . . . .	37
ssd_match_moments . . . . .	38
ssd_pal . . . . .	39
ssd_pburrrIII3 . . . . .	39
ssd_plot . . . . .	42
ssd_plot_cdf . . . . .	44
ssd_plot_cf . . . . .	45
ssd_plot_data . . . . .	45
ssd_qburrrIII3 . . . . .	47
ssd_rburrrIII3 . . . . .	50
ssd_sort_data . . . . .	52
ssd_wqg_bc . . . . .	53
ssd_wqg_burrlioz . . . . .	54
stat_ssd . . . . .	55
subset.fitdists . . . . .	56
tidy.fitdists . . . . .	57
<b>Index</b>	<b>58</b>

---

augment.fitdists	<i>Augmented Data from fitdists Object</i>
------------------	--

---

## Description

Get a tibble of the original data with augmentation.

## Usage

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

**Arguments**

x	The object.
...	Unused.

**Value**

A tibble of the augmented data.

**See Also**

[ssd\\_data\(\)](#)

Other generics: [glance.fitdists\(\)](#), [tidy.fitdists\(\)](#)

**Examples**

```
fits <- ssd_fit_dists(ssdata::ccme_boron)
augment(fits)
```

---

autoplot.fitdists      *Plot a fitdists Object*

---

**Description**

A wrapper on [ssd\\_plot\\_cdf\(\)](#).

**Usage**

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

**Arguments**

object	The object.
...	Unused.

**Value**

A ggplot object.

**See Also**

[ssd\\_plot\\_cdf\(\)](#)

**Examples**

```
fits <- ssd_fit_dists(ssdata::ccme_boron)
autoplot(fits)
```

---

`boron_data`*CCME Species Sensitivity Data for Boron*

---

**Description**

Species Sensitivity Data from the Canadian Council of Ministers of the Environment. Please use `[ssddata::ccme_data]` instead.

**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\(\)](#)

Other boron: [boron\\_pred](#)

**Examples**

```
head(ccme_data)
```

---

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 8 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).

**See Also**

Other boron: [boron\\_data](#)

**Examples**

```
head(boron_pred)
```

---

ccme_data	<i>CCME Species Sensitivity Data</i>
-----------	--------------------------------------

---

**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. Please use `[ssddata::ccme_data]` instead.

**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)
```

---

`coef.fitdists`*Turn a fitdists Object into a Tidy Tibble*

---

**Description**

A wrapper on `tidy.fitdists()`.

**Usage**

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

**Arguments**

<code>object</code>	The object.
<code>...</code>	Unused.

**See Also**

[tidy.fitdists\(\)](#)

**Examples**

```
fits <- ssd_fit_dists(ssddata::cme_boron)
coef(fits)
```

---

code: comma\_signif

*Comma 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))
```

---

code: dgompertz

*Gompertz Probability Density*


---

**Description**

**[Deprecated]**

**Usage**

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



**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).

**Value**

A numeric vector.

---

dist_data	<i>Distribution Data</i>
-----------	--------------------------

---

**Description**

A data frame of information on the implemented distributions.

**Usage**

```
dist_data
```

**Format**

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 10 rows and 4 columns.

**Details**

**dist** The distribution (chr).

**npars** The number of parameters (int).

**tails** Whether the distribution has both tails (flag).

**stable** Whether the distribution is numerically stable (flag).

**bcanz** Whether the distribution belongs to the set of distributions approved by BC, Canada, Australia and New Zealand for official guidelines (flag).

**See Also**

Other dists: [ssd\\_dists\\_all\(\)](#), [ssd\\_dists\(\)](#)

**Examples**

```
dist
```

---

dlgumbel *Log-Gumbel (Inverse Weibull) Probability Density*

---

### Description

**[Deprecated]**

### Usage

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

### 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).

### Value

A numeric vector.

---

estimates.fitdists *Estimates for fitdists Object*

---

### Description

Gets a named list of the estimated values by distribution and term.

### Usage

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

### Arguments

x	The object.
...	Unused.

### Value

A named list of the estimates.

### See Also

[tidy.fitdists\(\)](#), [ssd\\_match\\_moments\(\)](#), [ssd\\_hc\(\)](#) and [ssd\\_plot\\_cdf\(\)](#)

**Examples**

```
fits <- ssd_fit_dists(ssddata::cme_boron)
estimates <- estimates(fits)
print(estimates)
ssd_hc(estimates)
ssd_plot_cdf(estimates)
```

geom\_hcintersect

*Species Sensitivity Hazard Concentration Intersection***Description**

Plots the intersection between each xintercept and yintercept value.

**Usage**

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

**Arguments**

mapping	Set of aesthetic mappings created by <a href="#">aes()</a> . 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 <a href="#">ggplot()</a> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <a href="#">fortify()</a> 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> ).
...	Other arguments passed on to <a href="#">layer()</a> . 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.
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.

**See Also**

[ssd\\_plot\\_cdf\(\)](#)

Other ggplot: [geom\\_ssdpoint\(\)](#), [geom\\_ssdsegment\(\)](#), [geom\\_xribbon\(\)](#), [scale\\_colour\\_ssd\(\)](#), [ssd\\_pal\(\)](#)

**Examples**

```
ggplot2::ggplot(ssddata::ccme_boron, ggplot2::aes(x = Conc)) +
  geom_ssdpoint() +
  geom_hcintersect(xintercept = 1.5, yintercept = 0.05)
```

---

geom_ssdpoint	<i>Species Sensitivity Data Points</i>
---------------	--

---

**Description**

Uses the empirical cumulative distribution to create scatterplot of points x.

**Usage**

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

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

**Arguments**

mapping	Set of aesthetic mappings created by <a href="#">aes()</a> . 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 <a href="#">ggplot()</a> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <a href="#">fortify()</a> 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, either as a <code>ggproto</code> <code>Geom</code> subclass or as a string naming the stat stripped of the <code>stat_</code> prefix (e.g. "count" rather than "stat_count")
position	Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use <code>position_jitter</code> ), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.
...	Other arguments passed on to <a href="#">layer()</a> . 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.
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. <a href="#">borders()</a> .

**Details**

`geom_ssd()` has been deprecated for `geom_ssdpoint()`.

**Functions**

- `geom_ssd()`: Species Sensitivity Data Points  
Uses the empirical cumulative distribution to create scatterplot of points x. **[Deprecated]**

**See Also**

[ssd\\_plot\\_cdf\(\)](#)

Other `ggplot`: [geom\\_hcintersect\(\)](#), [geom\\_ssdsegment\(\)](#), [geom\\_xribbon\(\)](#), [scale\\_colour\\_ssd\(\)](#), [ssd\\_pal\(\)](#)

**Examples**

```
ggplot2::ggplot(ssddata::ccme_boron, ggplot2::aes(x = Conc)) +
  geom_ssdpoint()

## Not run:
ggplot2::ggplot(ssddata::ccme_boron, ggplot2::aes(x = Conc)) +
  geom_ssd()

## End(Not run)
```

---

geom\_ssdsegment

*Species Sensitivity Censored Segments*


---

**Description**

Uses the empirical cumulative distribution to draw lines between points `x` and `xend`.

**Usage**

```
geom_ssdsegment(
  mapping = NULL,
  data = NULL,
  stat = "ssdsegment",
  position = "identity",
  ...,
  arrow = NULL,
  arrow.fill = NULL,
  lineend = "butt",
  linejoin = "round",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

**Arguments**

mapping	Set of aesthetic mappings created by <a href="#">aes()</a> . 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.
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 <a href="#">ggplot()</a> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <a href="#">fortify()</a> 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, either as a ggproto Geom subclass or as a string naming the stat stripped of the stat_ prefix (e.g. "count" rather than "stat_count")
position	Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use position_jitter), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.
...	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 geom/stat.
arrow	specification for arrow heads, as created by <code>grid::arrow()</code> .
arrow.fill	fill colour to use for the arrow head (if closed). NULL means use colour aesthetic.
lineend	Line end style (round, butt, square).
linejoin	Line join style (round, mitre, bevel).
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. <code>borders()</code> .

### See Also

[ssd\\_plot\\_cdf\(\)](#)

Other ggplot: [geom\\_hcintersect\(\)](#), [geom\\_ssdpoint\(\)](#), [geom\\_xribbon\(\)](#), [scale\\_colour\\_ssd\(\)](#), [ssd\\_pal\(\)](#)

### Examples

```
ggplot2::ggplot(ssddata::ccme_boron, ggplot2::aes(x = Conc, xend = Conc * 2)) +
  geom_ssdsegment()
```

---

geom\_xribbon

*Ribbon on X-Axis*

---

### Description

Plots the 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**

mapping	Set of aesthetic mappings created by <a href="#">aes()</a> . 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 <a href="#">ggplot()</a> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <a href="#">fortify()</a> 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, either as a <code>ggproto</code> <code>Geom</code> subclass or as a string naming the stat stripped of the <code>stat_</code> prefix (e.g. "count" rather than "stat_count")
position	Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use <code>position_jitter</code> ), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.
...	Other arguments passed on to <a href="#">layer()</a> . 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> .
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. <a href="#">borders()</a> .



**See Also**[ssd\\_plot\\_cdf\(\)](#)Other ggplot: [geom\\_hcintersect\(\)](#), [geom\\_ssdpoint\(\)](#), [geom\\_ssdsegment\(\)](#), [scale\\_colour\\_ssd\(\)](#), [ssd\\_pal\(\)](#)**Examples**

```
gp <- ggplot2::ggplot(boron_pred) +  
  geom_xribbon(ggplot2::aes(xmin = lcl, xmax = ucl, y = percent))
```

---

glance.fitdists	<i>Get a tibble summarizing each distribution</i>
-----------------	---

---

**Description**

Gets a tibble with a single row for each distribution.

**Usage**

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

**Arguments**

x	The object.
...	Unused.

**Value**

A tidy tibble of the distributions.

**See Also**[ssd\\_gof\(\)](#)Other generics: [augment.fitdists\(\)](#), [tidy.fitdists\(\)](#)**Examples**

```
fits <- ssd_fit_dists(ssddata::cme_boron)  
glance(fits)
```

---

`is.fitdists`*Is fitdists Object*

---

**Description**

Tests whether x is a fitdists Object.

**Usage**

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

**Arguments**

x                    The object.

**Value**

A flag specifying whether x is a fitdists Object.

**Examples**

```
fits <- ssd_fit_dists(ssdata::cme_boron)
is.fitdists(fits)
```

---

`is_censored`*Is Censored*

---

**Description**

Deprecated for [ssd\\_is\\_censored\(\)](#).

**Usage**

```
is_censored(x)
```

**Arguments**

x                    A fitdists object.

**Value**

A flag indicating if the data is censored.

**See Also**

[ssd\\_is\\_censored\(\)](#)

**Examples**

```
fits <- ssd_fit_dists(ssddata::cme_boron)
is_censored(fits)
```

---

predict.fitburrliz    *Predict Hazard Concentrations of fitburrliz Object*

---

**Description**

A wrapper on `ssd_hc()` that by default calculates all hazard concentrations from 1 to 99%.

**Usage**

```
## S3 method for class 'fitburrliz'
predict(
  object,
  percent = 1:99,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  min_pboot = 0.99,
  parametric = TRUE,
  ...
)
```

**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. A value of 10000 is recommended for official guidelines.
min_pboot	A number of the minimum proportion of bootstrap samples that must successfully fit in the sense of returning a likelihood.
parametric	A flag specifying whether to perform parametric as opposed to non-parametric bootstrapping.
...	Unused.

**Details**

It is useful for plotting purposes.

**See Also**

[ssd\\_hc\(\)](#) and [ssd\\_plot\(\)](#)

**Examples**

```
fits <- ssd_fit_burrlioz(ssddata::ccme_boron)
predict(fits)
```

---

predict.fitdists      *Predict Hazard Concentrations of fitdists Object*

---

**Description**

A wrapper on [ssd\\_hc\(\)](#) that by default calculates all hazard concentrations from 1 to 99%.

**Usage**

```
## S3 method for class 'fitdists'
predict(
  object,
  percent = 1:99,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  average = TRUE,
  delta = 7,
  min_pboot = 0.99,
  parametric = TRUE,
  control = NULL,
  ...
)
```

**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. A value of 10000 is recommended for official guidelines.
average	A flag specifying whether to model average the estimates.
delta	A non-negative number specifying the maximum absolute Akaike Information-theoretic Criterion difference cutoff. Distributions with an absolute difference from the best model greater than the cutoff are excluded.

min_pboot	A number of the minimum proportion of bootstrap samples that must successfully fit in the sense of returning a likelihood.
parametric	A flag specifying whether to perform parametric as opposed to non-parametric bootstrapping.
control	A list of control parameters passed to <code>stats::optim()</code> .
...	Unused.

**Details**

It is useful for plotting purposes.

**See Also**

[ssd\\_hc\(\)](#) and [ssd\\_plot\(\)](#)

**Examples**

```
fits <- ssd_fit_dists(ssddata::ccme_boron)
predict(fits)
```

---

scale\_colour\_ssd      *Discrete color-blind scale for SSD Plots*

---

**Description**

Discrete color-blind scale for SSD Plots

**Usage**

```
scale_colour_ssd(...)
```

```
scale_color_ssd(...)
```

**Arguments**

...                    Arguments passed to `ggplot2::discrete_scale()`.

**Functions**

- `scale_color_ssd()`: Discrete color-blind scale for SSD Plots

**See Also**

Other ggplot: [geom\\_hcintersect\(\)](#), [geom\\_ssdpoint\(\)](#), [geom\\_ssdsegment\(\)](#), [geom\\_xribbon\(\)](#), [ssd\\_pal\(\)](#)

**Examples**

```
ssd_plot(ssddata::ccme_boron, boron_pred, shape = "Group") +
  scale_colour_ssd()
```

---

ssdtools-ggproto	<i>ggproto Classes for Plotting Species Sensitivity Data and Distributions</i>
------------------	--

---

**Description**

ggproto Classes for Plotting Species Sensitivity Data and Distributions

**Usage**

StatSsdpoint

StatSsdsegment

GeomSsdpoint

GeomSsdsegment

GeomHcintersect

GeomXribbon

**Format**

An object of class StatSsdpoint (inherits from Stat, ggproto, gg) of length 4.

An object of class StatSsdsegment (inherits from Stat, ggproto, gg) of length 4.

An object of class GeomSsdpoint (inherits from GeomPoint, Geom, ggproto, gg) of length 1.

An object of class GeomSsdsegment (inherits from GeomSegment, Geom, ggproto, gg) of length 1.

An object of class GeomHcintersect (inherits from Geom, ggproto, gg) of length 5.

An object of class GeomXribbon (inherits from Geom, ggproto, gg) of length 6.

**See Also**

[ggplot2::ggproto\(\)](#) and [ssd\\_plot\\_cdf\(\)](#)

---

ssd_data	<i>Data from fitdists Object</i>
----------	----------------------------------

---

**Description**

Get a tibble of the original data.

**Usage**

ssd\_data(x)

**Arguments**

x                    The object.

**Value**

A tibble of the original data.

**See Also**

[augment.fitdists\(\)](#), [ssd\\_ecd\\_data\(\)](#) and [ssd\\_sort\\_data\(\)](#)

**Examples**

```
fits <- ssd_fit_dists(ssddata::cme_boron)
ssd_data(fits)
```

---

ssd\_dists

*Species Sensitivity Distributions*

---

**Description**

Gets a character vector of the names of the available distributions.

**Usage**

```
ssd_dists(bcanz = NULL, tails = NULL, npars = 2:5)
```

**Arguments**

bcanz                A flag or NULL specifying whether to only include distributions in the set that is approved by BC, Canada, Australia and New Zealand for official guidelines.

tails                A flag or NULL specifying whether to only include distributions with both tails.

npars                A whole numeric vector specifying which distributions to include based on the number of parameters.

**Value**

A unique, sorted character vector of the distributions.

**See Also**

Other dists: [dist\\_data](#), [ssd\\_dists\\_all\(\)](#)

**Examples**

```
ssd_dists()
ssd_dists(bcanz = TRUE)
ssd_dists(tails = FALSE)
ssd_dists(npars = 5)
```

---

ssd_dists_all	<i>All Species Sensitivity Distributions</i>
---------------	--

---

**Description**

Gets a character vector of the names of all the available distributions.

**Usage**

```
ssd_dists_all()
```

**Value**

A unique, sorted character vector of the distributions.

**See Also**

Other dists: [dist\\_data](#), [ssd\\_dists\(\)](#)

**Examples**

```
ssd_dists_all()
```

---

ssd_dists_bcanz	<i>BCANZ Distributions</i>
-----------------	----------------------------

---

**Description**

Gets a character vector of the names of the distributions adopted by BC, Canada, Australia and New Zealand for official guidelines.

**Usage**

```
ssd_dists_bcanz()
```

**Value**

A unique, sorted character vector of the distributions.

**See Also**

[ssd\\_dists\(\)](#)

**Examples**

```
ssd_dists_bcanz()
```



---

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

---

**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_ecd_data	<i>Empirical Cumulative Density for Species Sensitivity Data</i>
--------------	--

---

**Description**

Empirical Cumulative Density for Species Sensitivity Data

**Usage**

```
ssd_ecd_data(  
  data,  
  left = "Conc",  
  right = left,  
  bounds = c(left = 1, right = 1)  
)
```

**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.
bounds	A named non-negative numeric vector of the left and right bounds for uncensored missing (0 and Inf) data in terms of the orders of magnitude relative to the extremes for non-missing values.

**Value**

A numeric vector of the empirical cumulative density for the rows in data.

**See Also**

[ssd\\_ecd\(\)](#) and [ssd\\_data\(\)](#)

**Examples**

```
ssd_ecd_data(ssdata::ccme_boron)
```

---

ssd\_exposure

*Percent Exposure*

---

**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	The mean of the exposure concentrations on the log scale.
sdlog	The standard deviation of the exposure concentrations on the log scale.
nboot	The number of samples to use to calculate the exposure.

**Value**

The proportion exposed.

## Examples

```
fits <- ssd_fit_dists(ssddata::ccme_boron, dists = "lnorm")
set.seed(10)
ssd_exposure(fits)
ssd_exposure(fits, meanlog = 1)
ssd_exposure(fits, meanlog = 1, sdlog = 1)
```

---

ssd\_fit\_bcanz

*Fit BCANZ Distributions*

---

## Description

Fits distributions using settings adopted by BC, Canada, Australia and New Zealand for official guidelines.

## Usage

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

## Arguments

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

## Value

An object of class `fitdists`.

## See Also

[ssd\\_fit\\_dists\(\)](#)

Other BCANZ: [ssd\\_hc\\_bcanz\(\)](#)

## Examples

```
ssd_fit_bcanz(ssddata::ccme_boron)
```

---

ssd\_fit\_burrlioz      *Fit Burrlioz Distributions*

---

### Description

Fits 'burrIII3' distribution. If shape1 parameter is at boundary returns 'lgumbel' (which is equivalent to inverse Weibull). Else if shape2 parameter is at a boundary returns 'invpareto'. Otherwise returns 'burrIII3'

### Usage

```
ssd_fit_burrlioz(data, left = "Conc", rescale = FALSE, silent = FALSE)
```

### Arguments

data	A data frame.
left	A string of the column in data with the concentrations.
rescale	A flag specifying whether to rescale concentration values by dividing by the largest finite value.
silent	A flag indicating whether fits should fail silently.

### Value

An object of class fitdists.

### See Also

[ssd\\_fit\\_dists\(\)](#)

### Examples

```
ssd_fit_burrlioz(ssddata::ccme_boron)
```

---

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 = ssd_dists_bcanz(),
  nrow = 6L,
  rescale = FALSE,
  reweight = FALSE,
  computable = TRUE,
  at_boundary_ok = FALSE,
  min_pmix = 0,
  range_shape1 = c(0.05, 20),
  range_shape2 = range_shape1,
  control = list(),
  silent = FALSE
)

```

**Arguments**

<code>data</code>	A data frame.
<code>left</code>	A string of the column in data with the concentrations.
<code>right</code>	A string of the column in data with the right concentration values.
<code>weight</code>	A string of the numeric column in data with positive weights less than or equal to 1,000 or NULL.
<code>dists</code>	A character vector of the distribution names.
<code>nrow</code>	A positive whole number of the minimum number of non-missing rows.
<code>rescale</code>	A flag specifying whether to rescale concentration values by dividing by the largest finite value.
<code>reweight</code>	A flag specifying whether to reweight weights by dividing by the largest weight.
<code>computable</code>	A flag specifying whether to only return fits with numerically computable standard errors.
<code>at_boundary_ok</code>	A flag specifying whether a model with one or more parameters at the boundary should be considered to have converged (default = FALSE).
<code>min_pmix</code>	A number between 0 and 0.5 specifying the minimum proportion in mixture models.
<code>range_shape1</code>	A numeric vector of length two of the lower and upper bounds for the shape1 parameter.
<code>range_shape2</code>	shape2 parameter.
<code>control</code>	A list of control parameters passed to <code>stats::optim()</code> .
<code>silent</code>	A flag indicating whether fits should fail silently.

### Details

By default the 'llogis', 'gamma' and 'lnorm' distributions are fitted to the data. For a complete list of the implemented distributions see [ssd\\_dists\\_all\(\)](#).

If weight specifies a column in the data frame with positive numbers, weighted estimation occurs. However, currently only the resultant parameter estimates are available.

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

### Value

An object of class fitdists.

### See Also

[ssd\\_plot\\_cdf\(\)](#) and [ssd\\_hc\(\)](#)

### Examples

```
fits <- ssd_fit_dists(ssdata::ccme_boron)
fits
ssd_plot_cdf(fits)
ssd_hc(fits)
```

---

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 'fitdists'  
ssd_gof(x, pvalue = FALSE, ...)
```

**Arguments**

x	The object.
...	Unused.
pvalue	A flag specifying whether to return p-values or the statistics (default) for the various tests.

**Value**

A tbl data frame of the gof statistics.

**Methods (by class)**

- `ssd_gof(fitdists)`: Goodness of Fit

**See Also**

[glance.fitdists\(\)](#)

**Examples**

```
fits <- ssd_fit_dists(ssddata::ccme_boron)  
ssd_gof(fits)  
ssd_gof(fits)
```

---

ssd\_hc

*Hazard Concentrations for Species Sensitivity Distributions*

---

**Description**

Gets concentration(s) that protect specified percentage(s) of species.

**Usage**

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

```

```

    x,
    percent = 5,
    hc = 5,
    ci = FALSE,
    level = 0.95,
    nboot = 1000,
    average = TRUE,
    delta = 7,
    min_pboot = 0.99,
    parametric = TRUE,
    control = NULL,
    ...
)

## S3 method for class 'fitburrlioz'
ssd_hc(
  x,
  percent = 5,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  min_pboot = 0.99,
  parametric = FALSE,
  ...
)

```

### Arguments

x	The object.
...	Unused.
percent	A numeric vector of percentages.
hc	A whole numeric vector between 1 and 99 indicating the percent hazard concentrations (deprecated for percent).
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. A value of 10000 is recommended for official guidelines.
average	A flag specifying whether to model average the estimates.
delta	A non-negative number specifying the maximum absolute Akaike Information-theoretic Criterion difference cutoff. Distributions with an absolute difference from the best model greater than the cutoff are excluded.
min_pboot	A number of the minimum proportion of bootstrap samples that must successfully fit in the sense of returning a likelihood.
parametric	A flag specifying whether to perform parametric as opposed to non-parametric bootstrapping.
control	A list of control parameters passed to <code>stats::optim()</code> .



**Details**

If `ci = TRUE` uses parametric bootstrapping to get confidence intervals on the hazard concentrations(s).

**Value**

A tibble of corresponding hazard concentrations.

**Methods (by class)**

- `ssd_hc(list)`: Hazard Concentrations for Distributional Estimates
- `ssd_hc(fitdists)`: Hazard Concentrations for `fitdists` Object
- `ssd_hc(fitburrrlioz)`: Hazard Concentrations for `fitburrrlioz` Object

**See Also**

[predict.fitdists\(\)](#) and [ssd\\_hp\(\)](#).

**Examples**

```
fits <- ssd_fit_dists(ssddata::ccme_boron)
ssd_hc(fits)
ssd_hc(estimate(fits))
ssd_hc(ssd_match_moments())
fit <- ssd_fit_burrrlioz(ssddata::ccme_boron)
ssd_hc(fit)
```

---

ssd\_hc\_bcanz

*BCANZ Hazard Concentrations*

---

**Description**

Gets hazard concentrations with confidence intervals that protect 1, 5, 10 and 20% of species using settings adopted by BC, Canada, Australia and New Zealand for official guidelines. This function can take several minutes to run with required 10,000 iterations.

**Usage**

```
ssd_hc_bcanz(x, nboot = 10000, delta = 10, min_pboot = 0.9)
```

**Arguments**

x	The object.
nboot	A count of the number of bootstrap samples to use to estimate the se and confidence limits. A value of 10000 is recommended for official guidelines.
delta	A non-negative number specifying the maximum absolute Akaike Information-theoretic Criterion difference cutoff. Distributions with an absolute difference from the best model greater than the cutoff are excluded.
min_pboot	A number of the minimum proportion of bootstrap samples that must successfully fit in the sense of returning a likelihood.

**Value**

A tibble of corresponding hazard concentrations.

**See Also**

[ssd\\_hc\(\)](#).

Other BCANZ: [ssd\\_fit\\_bcanz\(\)](#)

**Examples**

```
fits <- ssd_fit_bcanz(ssdata::cme_boron)
ssd_hc_bcanz(fits, nboot = 100)
```

---

ssd\_hc\_burrlioz      *Hazard Concentrations for Burrlioz Fit*

---

**Description**

Deprecated for [ssd\\_hc\(\)](#).

**Usage**

```
ssd_hc_burrlioz(  
  x,  
  percent = 5,  
  ci = FALSE,  
  level = 0.95,  
  nboot = 1000,  
  min_pboot = 0.99,  
  parametric = FALSE  
)
```

**Arguments**

x	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. A value of 10000 is recommended for official guidelines.
min_pboot	A number of the minimum proportion of bootstrap samples that must successfully fit in the sense of returning a likelihood.
parametric	A flag specifying whether to perform parametric as opposed to non-parametric bootstrapping.

**Value**

A tibble of corresponding hazard concentrations.

**Examples**

```
fit <- ssd_fit_burrlioz(ssdata::ccme_boron)
ssd_hc_burrlioz(fit)
```

---

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

---

**Description**

Gets percent of species affected at specified concentration(s).

**Usage**

```
ssd_hp(x, ...)

## S3 method for class 'fitdists'
ssd_hp(
  x,
  conc,
  ci = FALSE,
  level = 0.95,
  nboot = 1000,
  average = TRUE,
  delta = 7,
  min_pboot = 0.99,
  parametric = TRUE,
```

```

    control = NULL,
    ...
  )

```

### Arguments

<code>x</code>	The object.
<code>...</code>	Unused.
<code>conc</code>	A numeric vector of concentrations.
<code>ci</code>	A flag specifying whether to estimate confidence intervals (by parametric bootstrapping).
<code>level</code>	A number between 0 and 1 of the confidence level.
<code>nboot</code>	A count of the number of bootstrap samples to use to estimate the se and confidence limits. A value of 10000 is recommended for official guidelines.
<code>average</code>	A flag specifying whether to model average the estimates.
<code>delta</code>	A non-negative number specifying the maximum absolute Akaike Information-theoretic Criterion difference cutoff. Distributions with an absolute difference from the best model greater than the cutoff are excluded.
<code>min_pboot</code>	A number of the minimum proportion of bootstrap samples that must successfully fit in the sense of returning a likelihood.
<code>parametric</code>	A flag specifying whether to perform parametric as opposed to non-parametric bootstrapping.
<code>control</code>	A list of control parameters passed to <code>stats::optim()</code> .

### Details

If `ci = TRUE` uses parameteric bootstrapping to get confidence intervals on the hazard percent(s).

### Value

A tibble of corresponding hazard percents.

### Methods (by class)

- `ssd_hp(fitdists)`: Hazard Percents for fitdists Object

### See Also

[ssd\\_hc\(\)](#)

### Examples

```

fits <- ssd_fit_dists(ssddata::cme_boron)
ssd_hp(fits, conc = 1)

```

---

ssd_is_censored	<i>Is Censored</i>
-----------------	--------------------

---

### Description

Tests if an object has censored data.

Test if a data frame is censored.

Test if a fitdists object is censored.

### Usage

```
ssd_is_censored(x, ...)
```

```
## S3 method for class 'data.frame'  
ssd_is_censored(x, left = "Conc", right = left, ...)
```

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

### Arguments

x	The object.
...	Unused.
left	A string of the column in data with the concentrations.
right	A string of the column in data with the right concentration values.

### Value

A flag indicating whether an object is censored.

### Examples

```
ssd_is_censored(ssddata::ccme_boron)  
ssd_is_censored(data.frame(Conc = 1, right = 2), right = "right")  
  
fits <- ssd_fit_dists(ssddata::ccme_boron)  
ssd_is_censored(fits)
```

---

ssd\_match\_moments      *Match Moments*

---

### Description

Gets a named list of the values that produce the moment values (meanlog and sdlog) by distribution and term.

### Usage

```
ssd_match_moments(  
  dists = ssd_dists_bcanz(),  
  meanlog = 1,  
  sdlog = 1,  
  nsim = 1e+05  
)
```

### Arguments

dists	A character vector of the distribution names.
meanlog	The mean on the log scale.
sdlog	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 values that produce the moment values by distribution and term.

### See Also

[estimates.fitdists\(\)](#), [ssd\\_hc\(\)](#) and [ssd\\_plot\\_cdf\(\)](#)

### Examples

```
moments <- ssd_match_moments()  
print(moments)  
ssd_hc(moments)  
ssd_plot_cdf(moments)
```

---

ssd_pal	<i>Color-blind Palette for SSD Plots</i>
---------	--

---

**Description**

Color-blind Palette for SSD Plots

**Usage**

```
ssd_pal()
```

**Value**

A character vector of a color blind palette with 8 colors.

**See Also**

Other ggplot: [geom\\_hcintersect\(\)](#), [geom\\_ssdpoint\(\)](#), [geom\\_ssdsegment\(\)](#), [geom\\_xribbon\(\)](#), [scale\\_colour\\_ssd\(\)](#)

**Examples**

```
ssd_pal()
```

---

ssd_pburrrIII3	<i>Cumulative Distribution Function</i>
----------------	---

---

**Description**

Cumulative Distribution Function

**Usage**

```
ssd_pburrrIII3(  
  q,  
  shape1 = 1,  
  shape2 = 1,  
  scale = 1,  
  lower.tail = TRUE,  
  log.p = FALSE  
)
```

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

```
ssd_pgompertz(q, location = 1, shape = 1, lower.tail = TRUE, log.p = FALSE)
```

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

ssd_pinvpareto(q, shape = 3, scale = 1, lower.tail = TRUE, log.p = FALSE)

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

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

ssd_pllogis_llogis(
  q,
  locationlog1 = 0,
  scalelog1 = 1,
  locationlog2 = 1,
  scalelog2 = 1,
  pmix = 0.5,
  lower.tail = TRUE,
  log.p = FALSE
)

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

ssd_plnorm_lnorm(
  q,
  meanlog1 = 0,
  sdlog1 = 1,
  meanlog2 = 1,
  sdlog2 = 1,
  pmix = 0.5,
  lower.tail = TRUE,
  log.p = FALSE
)

ssd_plnorm(q, meanlog = 0, sdlog = 1, lower.tail = TRUE, log.p = FALSE)

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

### Arguments

q	vector of quantiles.
shape1	shape1 parameter.
shape2	shape2 parameter.
scale	scale parameter.



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 $\log(p)$ .
shape	A string of the column in data for the shape aesthetic.
location	location parameter.
llocation	location parameter on the log scale.
lshape	shape parameter on the log scale.
locationlog	location on log scale parameter.
scalelog	scale on log scale parameter.
locationlog1	locationlog1 parameter.
scalelog1	scalelog1 parameter.
locationlog2	locationlog2 parameter.
scalelog2	scalelog2 parameter.
pmix	Proportion mixture parameter.
meanlog1	mean on log scale parameter.
sdlog1	standard deviation on log scale parameter.
meanlog2	mean on log scale parameter.
sdlog2	standard deviation on log scale parameter.
meanlog	mean on log scale parameter.
sdlog	standard deviation on log scale parameter.

### Functions

- `ssd_pburrrIII3()`: Cumulative Distribution Function for BurrIII Distribution
- `ssd_pgamma()`: Cumulative Distribution Function for Gamma Distribution
- `ssd_pgompertz()`: Cumulative Distribution Function for Gompertz Distribution
- `pgompertz()`: Cumulative Distribution Function for Gompertz Distribution **[Deprecated]**
- `ssd_pinvpareto()`: Cumulative Distribution Function for Inverse Pareto Distribution
- `ssd_plgumbel()`: Cumulative Distribution Function for Log-Gumbel Distribution
- `plgumbel()`: Cumulative Distribution Function for Log-Gumbel Distribution **[Deprecated]**
- `ssd_pllogis_llogis()`: Cumulative Distribution Function for Log-Logistic/Log-Logistic Mixture Distribution
- `ssd_pllogis()`: Cumulative Distribution Function for Log-Logistic Distribution
- `ssd_plnorm_lnorm()`: Cumulative Distribution Function for Log-Normal/Log-Normal Mixture Distribution
- `ssd_plnorm()`: Cumulative Distribution Function for Log-Normal Distribution
- `ssd_pweibull()`: Cumulative Distribution Function for Weibull Distribution

### See Also

[ssd\\_q](#) and [ssd\\_r](#)

**Examples**

```
ssd_pburrrIII3(1)
ssd_pgamma(1)
ssd_pgompertz(1)
ssd_pinvpareto(1)
ssd_plgumbel(1)
ssd_pllogis_lllogis(1)
ssd_pllogis(1)
ssd_plnorm_lnorm(1)
ssd_plnorm(1)
ssd_pweibull(1)
```

---

ssd\_plot

*Plot Species Sensitivity Data and Distributions*

---

**Description**

Plots species sensitivity data and distributions.

**Usage**

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

```

    bounds = c(left = 1, right = 1),
    xbreaks = waiver()
)

```

### 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.
linetype	A string of the column in pred to use for the linetype.
linecolor	A string of the column in pred to use for the line color.
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.
bounds	A named non-negative numeric vector of the left and right bounds for uncensored missing (0 and Inf) data in terms of the orders of magnitude relative to the extremes for non-missing values.
xbreaks	The x-axis breaks as one of: <ul style="list-style-type: none"> <li>• NULL for no breaks</li> <li>• waiver() for the default breaks</li> <li>• A numeric vector of positions</li> </ul>

### See Also

[ssd\\_plot\\_cdf\(\)](#) and [geom\\_ssdpoint\(\)](#)

### Examples

```
ssd_plot(ssddata::ccme_boron, boron_pred, label = "Species", shape = "Group")
```

---

ssd\_plot\_cdf

*Plot Cumulative Distribution Function (CDF)*


---

### Description

Generic function to plots the cumulative distribution function (CDF).

### Usage

```
ssd_plot_cdf(x, ...)

## S3 method for class 'fitdists'
ssd_plot_cdf(x, average = FALSE, delta = 7, ...)

## S3 method for class 'list'
ssd_plot_cdf(x, ...)
```

### Arguments

x	The object.
...	Additional arguments passed to <a href="#">ssd_plot()</a> .
average	A flag specifying whether to model average the estimates.
delta	A non-negative number specifying the maximum absolute Akaike Information-theoretic Criterion difference cutoff. Distributions with an absolute difference from the best model greater than the cutoff are excluded.

### Methods (by class)

- `ssd_plot_cdf(fitdists)`: Plot CDF for fitdists object
- `ssd_plot_cdf(list)`: Plot CDF for named list of distributional parameter values

### See Also

[ssd\\_plot\(\)](#)  
[estimates.fitdists\(\)](#) and [ssd\\_match\\_moments\(\)](#)

### Examples

```
fits <- ssd_fit_dists(ssdata::ccme_boron)
ssd_plot_cdf(fits)

ssd_plot_cdf(list(
  llogis = c(locationlog = 2, scalelog = 1),
  lnorm = c(meanlog = 2, sdlog = 2))
)
```

---

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.

**Details**

Soft deprecated for direct call to [fitdistrplus::descdist\(\)](#).

**Functions**

- `ssd_cfplot()`: Defunct Cullen and Frey Plot

**Examples**

```
ssd_plot_cf(ssddata::ccme_boron)
```

---

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

---

**Description**

Plots species sensitivity data.

**Usage**

```

ssd_plot_data(
  data,
  left = "Conc",
  right = left,
  label = NULL,
  shape = NULL,
  color = NULL,
  size = 2.5,
  xlab = "Concentration",
  ylab = "Species Affected",
  shift_x = 3,
  bounds = c(left = 1, right = 1),
  xbreaks = waiver()
)

```

**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.
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.
shift_x	The value to multiply the label x values by.
bounds	A named non-negative numeric vector of the left and right bounds for uncensored missing (0 and Inf) data in terms of the orders of magnitude relative to the extremes for non-missing values.
xbreaks	The x-axis breaks as one of: <ul style="list-style-type: none"> <li>• NULL for no breaks</li> <li>• <code>waiver()</code> for the default breaks</li> <li>• A numeric vector of positions</li> </ul>

**See Also**

[ssd\\_plot\(\)](#) and [geom\\_ssdpoint\(\)](#)

**Examples**

```

ssd_plot_data(ssddata::cme_boron, label = "Species", shape = "Group")

```

---

`ssd_qburrIII3`*Quantile Function*

---

**Description**

Quantile Function

**Usage**

```
ssd_qburrIII3(  
  p,  
  shape1 = 1,  
  shape2 = 1,  
  scale = 1,  
  lower.tail = TRUE,  
  log.p = FALSE  
)  
  
ssd_qgamma(p, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)  
  
ssd_qgompertz(p, location = 1, shape = 1, lower.tail = TRUE, log.p = FALSE)  
  
qgompertz(p, llocation = 0, lshape = 0, lower.tail = TRUE, log.p = FALSE)  
  
ssd_qinvpareto(p, shape = 3, scale = 1, lower.tail = TRUE, log.p = FALSE)  
  
ssd_qlgumbel(  
  p,  
  locationlog = 0,  
  scalelog = 1,  
  lower.tail = TRUE,  
  log.p = FALSE  
)  
  
qlgumbel(p, locationlog = 0, scalelog = 1, lower.tail = TRUE, log.p = FALSE)  
  
ssd_qllogis_llogis(  
  p,  
  locationlog1 = 0,  
  scalelog1 = 1,  
  locationlog2 = 1,  
  scalelog2 = 1,  
  pmix = 0.5,  
  lower.tail = TRUE,  
  log.p = FALSE  
)
```

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

```
ssd_qlnorm_lnorm(
  p,
  meanlog1 = 0,
  sdlog1 = 1,
  meanlog2 = 1,
  sdlog2 = 1,
  pmix = 0.5,
  lower.tail = TRUE,
  log.p = FALSE
)
```

```
ssd_qlnorm(p, meanlog = 0, sdlog = 1, lower.tail = TRUE, log.p = FALSE)
```

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

### Arguments

p	vector of probabilities.
shape1	shape1 parameter.
shape2	shape2 parameter.
scale	scale parameter.
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 $\log(p)$ .
shape	A string of the column in data for the shape aesthetic.
location	location parameter.
llocation	location parameter on the log scale.
lshape	shape parameter on the log scale.
locationlog	location on log scale parameter.
scalelog	scale on log scale parameter.
locationlog1	locationlog1 parameter.
scalelog1	scalelog1 parameter.
locationlog2	locationlog2 parameter.
scalelog2	scalelog2 parameter.
pmix	Proportion mixture parameter.
meanlog1	mean on log scale parameter.
sdlog1	standard deviation on log scale parameter.
meanlog2	mean on log scale parameter.
sdlog2	standard deviation on log scale parameter.
meanlog	mean on log scale parameter.
sdlog	standard deviation on log scale parameter.



**Functions**

- `ssd_qburrIII3()`: Quantile Function for BurrIII Distribution
- `ssd_qgamma()`: Quantile Function for Gamma Distribution
- `ssd_qgompertz()`: Quantile Function for Gompertz Distribution
- `qgompertz()`: Quantile Function for Gompertz Distribution [**Deprecated**]
- `ssd_qinvpareto()`: Quantile Function for Inverse Pareto Distribution
- `ssd_qlgumbel()`: Quantile Function for Log-Gumbel Distribution
- `qlgumbel()`: Quantile Function for Log-Gumbel Distribution [**Deprecated**]
- `ssd_qllogis_lllogis()`: Cumulative Distribution Function for Log-Logistic/Log-Logistic Mixture Distribution
- `ssd_qllogis()`: Cumulative Distribution Function for Log-Logistic Distribution
- `ssd_qlnorm_lnorm()`: Cumulative Distribution Function for Log-Normal/Log-Normal Mixture Distribution
- `ssd_qlnorm()`: Cumulative Distribution Function for Log-Normal Distribution
- `ssd_qweibull()`: Cumulative Distribution Function for Weibull Distribution

**See Also**

[ssd\\_p](#) and [ssd\\_r](#)

**Examples**

```
ssd_qburrIII3(0.5)
```

```
ssd_qgamma(0.5)
```

```
ssd_qgompertz(0.5)
```

```
ssd_qinvpareto(0.5)
```

```
ssd_qlgumbel(0.5)
```

```
ssd_qllogis_lllogis(0.5)
```

```
ssd_qllogis(0.5)
```

```
ssd_qlnorm_lnorm(0.5)
```

```
ssd_qlnorm(0.5)
```

```
ssd_qweibull(0.5)
```

---

`ssd_rburrrIII3`*Random Number Generation*

---

**Description**

Random Number Generation

**Usage**`ssd_rburrrIII3(n, shape1 = 1, shape2 = 1, scale = 1, chk = TRUE)``ssd_rgamma(n, shape = 1, scale = 1, chk = TRUE)``ssd_rgompertz(n, location = 1, shape = 1, chk = TRUE)``rgompertz(n, llocation = 0, lshape = 0)``ssd_rinvpareto(n, shape = 3, scale = 1, chk = TRUE)``ssd_rlgumbel(n, locationlog = 0, scalelog = 1, chk = TRUE)``rlgumbel(n, locationlog = 0, scalelog = 1)`

```
ssd_rllogis_lllogis(  
  n,  
  locationlog1 = 0,  
  scalelog1 = 1,  
  locationlog2 = 1,  
  scalelog2 = 1,  
  pmix = 0.5,  
  chk = TRUE  
)
```

`ssd_rllogis(n, locationlog = 0, scalelog = 1, chk = TRUE)`

```
ssd_rlnorm_lnorm(  
  n,  
  meanlog1 = 0,  
  sdlog1 = 1,  
  meanlog2 = 1,  
  sdlog2 = 1,  
  pmix = 0.5,  
  chk = TRUE  
)
```

`ssd_rlnorm(n, meanlog = 0, sdlog = 1, chk = TRUE)`

```
ssd_rweibull(n, shape = 1, scale = 1, chk = TRUE)
```

### Arguments

n	number of observations.
shape1	shape1 parameter.
shape2	shape2 parameter.
scale	scale parameter.
chk	A flag specifying whether to check the arguments.
shape	A string of the column in data for the shape aesthetic.
location	location parameter.
llocation	location parameter on the log scale.
lshape	shape parameter on the log scale.
locationlog	location on log scale parameter.
scalelog	scale on log scale parameter.
locationlog1	locationlog1 parameter.
scalelog1	scalelog1 parameter.
locationlog2	locationlog2 parameter.
scalelog2	scalelog2 parameter.
pmix	Proportion mixture parameter.
meanlog1	mean on log scale parameter.
sdlog1	standard deviation on log scale parameter.
meanlog2	mean on log scale parameter.
sdlog2	standard deviation on log scale parameter.
meanlog	mean on log scale parameter.
sdlog	standard deviation on log scale parameter.

### Functions

- `ssd_rburrrIII3()`: Random Generation for BurrIII Distribution
- `ssd_rgamma()`: Random Generation for Gamma Distribution
- `ssd_rgompertz()`: Random Generation for Gompertz Distribution
- `rgompertz()`: Random Generation for Gompertz Distribution **[Deprecated]**
- `ssd_rinvpareto()`: Random Generation for Inverse Pareto Distribution
- `ssd_rlgumbel()`: Random Generation for log-Gumbel Distribution
- `rlgumbel()`: Random Generation for log-Gumbel Distribution **[Deprecated]**
- `ssd_rllogis_llogis()`: Random Generation for Log-Logistic/Log-Logistic Mixture Distribution
- `ssd_rllogis()`: Random Generation for Log-Logistic Distribution
- `ssd_rlnorm_lnorm()`: Random Generation for Log-Normal/Log-Normal Mixture Distribution
- `ssd_rlnorm()`: Random Generation for Log-Normal Distribution
- `ssd_rweibull()`: Random Generation for Weibull Distribution

**See Also**

[ssd\\_p](#) and [ssd\\_q](#)

**Examples**

```
set.seed(50)
hist(ssd_rburrrIII3(10000), breaks = 1000)

set.seed(50)
hist(ssd_rgamma(10000), breaks = 1000)

set.seed(50)
hist(ssd_rgompertz(10000), breaks = 1000)

set.seed(50)
hist(ssd_rinvpareto(10000), breaks = 1000)

set.seed(50)
hist(ssd_rlgumbel(10000), breaks = 1000)

set.seed(50)
hist(ssd_rllogis_llogis(10000), breaks = 1000)

set.seed(50)
hist(ssd_rllogis(10000), breaks = 1000)

set.seed(50)
hist(ssd_rlnorm_lnorm(10000), breaks = 1000)

set.seed(50)
hist(ssd_rlnorm(10000), breaks = 1000)

set.seed(50)
hist(ssd_rweibull(10000), breaks = 1000)
```

---

ssd\_sort\_data

*Sort Species Sensitivity Data*

---

**Description**

Sorts Species Sensitivity Data by empirical cumulative density (ECD).

**Usage**

```
ssd_sort_data(data, left = "Conc", right = left)
```

**Arguments**

<code>data</code>	A data frame.
<code>left</code>	A string of the column in data with the concentrations.
<code>right</code>	A string of the column in data with the right concentration values.

**Details**

Useful for sorting data before using `geom_ssdpoint()` and `geom_ssdsegment()` to construct plots for censored data with `stat = identity` to ensure order is the same for the various components.

**Value**

data sorted by the empirical cumulative density.

**See Also**

[ssd\\_ecd\\_data\(\)](#) and [ssd\\_data\(\)](#)

**Examples**

```
ssd_sort_data(ssddata::cme_boron)
```

---

ssd\_wqg\_bc

*Water Quality Guideline for British Columbia*

---

**Description**

Calculates the 5% Hazard Concentration for British Columbia after rescaling the data based on the log-logistic, log-normal and gamma distributions using the parametric bootstrap and AICc model averaging.

**Usage**

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

**Arguments**

<code>data</code>	A data frame.
<code>left</code>	A string of the column in data with the concentrations.

**Details**

Returns a tibble the model averaged 5% hazard concentration with standard errors, 95% lower and upper confidence limits and the number of bootstrap samples as well as the proportion of bootstrap samples that successfully returned a likelihood (convergence of the bootstrap sample is not required).

**Value**

A tibble of the 5% hazard concentration with 95% confidence intervals.

**See Also**

[ssd\\_fit\\_dists\(\)](#) and [ssd\\_hc\(\)](#)

Other wqg: [ssd\\_wqg\\_burrliz\(\)](#)

**Examples**

```
## Not run:  
ssd_wqg_bc(ssdata: :ccme_boron)  
  
## End(Not run)
```

---

ssd_wqg_burrliz	<i>Water Quality Guideline for Burrliz</i>
-----------------	--

---

**Description**

Calculates the 5% Hazard Concentration (after rescaling the data) using the same approach as Burrliz based on 10,000 non-parametric bootstrap samples.

**Usage**

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

**Arguments**

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

**Details**

Returns a tibble the model averaged 5% hazard concentration with standard errors, 95% lower and upper confidence limits and the number of bootstrap samples as well as the proportion of bootstrap samples that successfully returned a likelihood (convergence of the bootstrap sample is not required).

**Value**

A tibble of the 5% hazard concentration with 95% confidence intervals.

**See Also**

[ssd\\_fit\\_burrliz\(\)](#) and [ssd\\_hc\\_burrliz\(\)](#)

Other wqg: [ssd\\_wqg\\_bc\(\)](#)

**Examples**

```
## Not run:
  ssd_wqg_burrlioz(ssddata::ccme_boron)

## End(Not run)
```

stat\_ssd

*Plot Species Sensitivity Data***Description**

Uses the empirical cumulative density/distribution to visualize species sensitivity data. **[Deprecated]**

**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 <a href="#">aes()</a> . 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 <a href="#">ggplot()</a> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <a href="#">fortify()</a> 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 to display the data, either as a ggproto <code>Geom</code> subclass or as a string naming the geom stripped of the <code>geom_</code> prefix (e.g. "point" rather than "geom_point")

position	Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use <code>position_jitter</code> ), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.
...	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> .
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. <code>borders()</code> .

**See Also**

[geom\\_ssdpoint\(\)](#)

**Examples**

```
## Not run:
ggplot2::ggplot(ssddata::ccme_boron, ggplot2::aes(x = Conc)) +
  stat_ssd()

## End(Not run)
```

---

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

---

**Description**

Select a subset of distributions from a `fitdists` object. The Akaike Information-theoretic Criterion differences are calculated after selecting the distributions named in `select`.

**Usage**

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

**Arguments**

x	The object.
select	A character vector of the distributions to select.
delta	A non-negative number specifying the maximum absolute Akaike Information-theoretic Criterion difference cutoff. Distributions with an absolute difference from the best model greater than the cutoff are excluded.
...	Unused.



**Examples**

```
fits <- ssd_fit_dists(ssdata::cme_boron)
subset(fits, c("gamma", "lnorm"))
```

---

tidy.fitdists	<i>Turn a fitdists Object into a Tibble</i>
---------------	---

---

**Description**

Turns a fitdists object into a tidy tibble of the estimates (est) and standard errors (se) by the terms (term) and distributions (dist).

**Usage**

```
## S3 method for class 'fitdists'
tidy(x, all = FALSE, ...)
```

**Arguments**

x	The object.
all	A flag specifying whether to also return transformed parameters.
...	Unused.

**Value**

A tidy tibble of the estimates and standard errors.

**See Also**

[coef.fitdists\(\)](#)

Other generics: [augment.fitdists\(\)](#), [glance.fitdists\(\)](#)

**Examples**

```
fits <- ssd_fit_dists(ssdata::cme_boron)
tidy(fits)
tidy(fits, all = TRUE)
```

# Index

- \* **BCANZ**
    - ssd\_fit\_bcanz, 27
    - ssd\_hc\_bcanz, 33
  - \* **boron**
    - boron\_data, 5
    - boron\_pred, 6
  - \* **datasets**
    - boron\_data, 5
    - boron\_pred, 6
    - ccme\_data, 6
    - dist\_data, 9
    - ssdtools-ggproto, 22
  - \* **dists BCANZ**
    - ssd\_dists\_bcanz, 24
  - \* **dists**
    - dist\_data, 9
    - ssd\_dists, 23
    - ssd\_dists\_all, 24
  - \* **generics**
    - augment.fitdists, 3
    - glance.fitdists, 17
    - tidy.fitdists, 57
  - \* **ggplot2**
    - stat\_ssd, 55
  - \* **ggplot**
    - geom\_hcintersect, 11
    - geom\_ssdpoint, 12
    - geom\_ssdsegment, 14
    - geom\_xribbon, 15
    - scale\_colour\_ssd, 21
    - ssd\_pal, 39
  - \* **wqg**
    - ssd\_wqg\_bc, 53
    - ssd\_wqg\_burrlioz, 54
- aes(), 11, 13, 14, 16, 55  
augment.fitdists, 3, 17, 57  
augment.fitdists(), 23  
autoplot.fitdists, 4  
borders(), 13, 15, 16, 56  
boron\_data, 5, 6  
boron\_pred, 5, 6  
ccme\_data, 6  
ccme\_data(), 5  
coef.fitdists, 7  
coef.fitdists(), 57  
comma\_signif, 8  
dgeomertz, 8  
dist\_data, 9, 23, 24  
dlgumbel, 10  
estimates.fitdists, 10  
estimates.fitdists(), 38, 44  
fitdistrplus::descdist(), 45  
fortify(), 11, 13, 14, 16, 55  
geom\_hcintersect, 11, 13, 15, 17, 21, 39  
geom\_ssd (geom\_ssdpoint), 12  
geom\_ssdpoint, 12, 12, 15, 17, 21, 39  
geom\_ssdpoint(), 43, 46, 53, 56  
geom\_ssdsegment, 12, 13, 14, 17, 21, 39  
geom\_ssdsegment(), 53  
geom\_xribbon, 12, 13, 15, 15, 21, 39  
GeomHcintersect (ssdtools-ggproto), 22  
GeomSsdpoint (ssdtools-ggproto), 22  
GeomSsdsegment (ssdtools-ggproto), 22  
GeomXribbon (ssdtools-ggproto), 22  
ggplot(), 11, 13, 14, 16, 55  
ggplot2::discrete\_scale(), 21  
ggplot2::ggproto(), 22  
glance.fitdists, 4, 17, 57  
glance.fitdists(), 31  
grid::arrow(), 15  
is.fitdists, 18  
is\_censored, 18

- layer(), [11](#), [13](#), [15](#), [16](#), [56](#)
- pgompertz (ssd\_pburrrIII3), [39](#)
- plgumbel (ssd\_pburrrIII3), [39](#)
- predict.fitburrrlioz, [19](#)
- predict.fitdists, [20](#)
- predict.fitdists(), [33](#)
- qgompertz (ssd\_qburrrIII3), [47](#)
- qlgumbel (ssd\_qburrrIII3), [47](#)
- rgompertz (ssd\_rburrrIII3), [50](#)
- rlgumbel (ssd\_rburrrIII3), [50](#)
- scale\_color\_ssd (scale\_colour\_ssd), [21](#)
- scale\_colour\_ssd, [12](#), [13](#), [15](#), [17](#), [21](#), [39](#)
- scales::comma, [8](#)
- ssd\_cfplot (ssd\_plot\_cf), [45](#)
- ssd\_data, [22](#)
- ssd\_data(), [4](#), [26](#), [53](#)
- ssd\_dists, [9](#), [23](#), [24](#)
- ssd\_dists(), [24](#)
- ssd\_dists\_all, [9](#), [23](#), [24](#)
- ssd\_dists\_all(), [30](#)
- ssd\_dists\_bcanz, [24](#)
- ssd\_ecd, [25](#)
- ssd\_ecd(), [26](#)
- ssd\_ecd\_data, [25](#)
- ssd\_ecd\_data(), [23](#), [53](#)
- ssd\_exposure, [26](#)
- ssd\_fit\_bcanz, [27](#), [34](#)
- ssd\_fit\_burrrlioz, [28](#)
- ssd\_fit\_burrrlioz(), [54](#)
- ssd\_fit\_dists, [28](#)
- ssd\_fit\_dists(), [27](#), [28](#), [54](#)
- ssd\_gof, [30](#)
- ssd\_gof(), [17](#)
- ssd\_hc, [31](#)
- ssd\_hc(), [10](#), [19–21](#), [30](#), [34](#), [36](#), [38](#), [54](#)
- ssd\_hc\_bcanz, [27](#), [33](#)
- ssd\_hc\_burrrlioz, [34](#)
- ssd\_hc\_burrrlioz(), [54](#)
- ssd\_hp, [35](#)
- ssd\_hp(), [33](#)
- ssd\_is\_censored, [37](#)
- ssd\_is\_censored(), [18](#)
- ssd\_match\_moments, [38](#)
- ssd\_match\_moments(), [10](#), [44](#)
- ssd\_p, [49](#), [52](#)
- ssd\_p (ssd\_pburrrIII3), [39](#)
- ssd\_pal, [12](#), [13](#), [15](#), [17](#), [21](#), [39](#)
- ssd\_pburrrIII3, [39](#)
- ssd\_pgamma (ssd\_pburrrIII3), [39](#)
- ssd\_pgompertz (ssd\_pburrrIII3), [39](#)
- ssd\_pinvpareto (ssd\_pburrrIII3), [39](#)
- ssd\_plgumbel (ssd\_pburrrIII3), [39](#)
- ssd\_pllogis (ssd\_pburrrIII3), [39](#)
- ssd\_pllogis\_llogis (ssd\_pburrrIII3), [39](#)
- ssd\_plnorm (ssd\_pburrrIII3), [39](#)
- ssd\_plnorm\_lnorm (ssd\_pburrrIII3), [39](#)
- ssd\_plot, [42](#)
- ssd\_plot(), [20](#), [21](#), [44](#), [46](#)
- ssd\_plot\_cdf, [44](#)
- ssd\_plot\_cdf(), [4](#), [10](#), [12](#), [13](#), [15](#), [17](#), [22](#), [30](#), [38](#), [43](#)
- ssd\_plot\_cf, [45](#)
- ssd\_plot\_data, [45](#)
- ssd\_pweibull (ssd\_pburrrIII3), [39](#)
- ssd\_q, [41](#), [52](#)
- ssd\_q (ssd\_qburrrIII3), [47](#)
- ssd\_qburrrIII3, [47](#)
- ssd\_qgamma (ssd\_qburrrIII3), [47](#)
- ssd\_qgompertz (ssd\_qburrrIII3), [47](#)
- ssd\_qinvpareto (ssd\_qburrrIII3), [47](#)
- ssd\_qlgumbel (ssd\_qburrrIII3), [47](#)
- ssd\_qllogis (ssd\_qburrrIII3), [47](#)
- ssd\_qllogis\_llogis (ssd\_qburrrIII3), [47](#)
- ssd\_qlnorm (ssd\_qburrrIII3), [47](#)
- ssd\_qlnorm\_lnorm (ssd\_qburrrIII3), [47](#)
- ssd\_qweibull (ssd\_qburrrIII3), [47](#)
- ssd\_r, [41](#), [49](#)
- ssd\_r (ssd\_rburrrIII3), [50](#)
- ssd\_rburrrIII3, [50](#)
- ssd\_rgamma (ssd\_rburrrIII3), [50](#)
- ssd\_rgompertz (ssd\_rburrrIII3), [50](#)
- ssd\_rinvpareto (ssd\_rburrrIII3), [50](#)
- ssd\_rlgumbel (ssd\_rburrrIII3), [50](#)
- ssd\_rllogis (ssd\_rburrrIII3), [50](#)
- ssd\_rllogis\_llogis (ssd\_rburrrIII3), [50](#)
- ssd\_rlnorm (ssd\_rburrrIII3), [50](#)
- ssd\_rlnorm\_lnorm (ssd\_rburrrIII3), [50](#)
- ssd\_rweibull (ssd\_rburrrIII3), [50](#)
- ssd\_sort\_data, [52](#)
- ssd\_sort\_data(), [23](#)
- ssd\_wqg\_bc, [53](#), [54](#)
- ssd\_wqg\_burrrlioz, [54](#), [54](#)
- ssdtools-ggproto, [22](#)

`stat_ssd`, [55](#)  
`stats::optim()`, [21](#), [29](#), [32](#), [36](#)  
`StatSsdpoint` (`ssdtools-ggproto`), [22](#)  
`StatSsdsegment` (`ssdtools-ggproto`), [22](#)  
`subset.fitdists`, [56](#)  
  
`tidy.fitdists`, [4](#), [17](#), [57](#)  
`tidy.fitdists()`, [7](#), [10](#)