

# Package ‘fitdistrplus’

July 2, 2014

**Title** Help to fit of a parametric distribution to non-censored or censored data

**Version** 1.0-2

**Date** 2014-02-11

**Description** Extends the `fitdistr` function (of the MASS package) with several functions to help the fit of a parametric distribution to non-censored or censored data. Censored data may contain left censored, right censored and interval censored values, with several lower and upper bounds. In addition to maximum likelihood estimation method the package provides moment matching, quantile matching and maximum goodness-of-fit estimation methods (available only for non-censored data).

**Depends** R (>= 2.10.0), survival

**Imports** stats

**Suggests** actuar, rgenoud, mc2d

**License** GPL (>= 2)

**URL** <http://riskassessment.r-forge.r-project.org>

**Author** Marie Laure Delignette-Muller [aut, cre], Christophe Dutang [aut], Regis Pouillot [ctb], Jean-Baptiste Denis [ctb]

**Maintainer** Marie Laure Delignette-Muller <mariel Laure.delignette@muller@vetagro-sup.fr>

**Repository** CRAN

**Repository/R-Forge/Project** riskassessment

**Repository/R-Forge/Revision** 273

**Repository/R-Forge/DateTimeStamp** 2014-02-11 15:16:02

**Date/Publication** 2014-02-12 15:40:38

**NeedsCompilation** no

**R topics documented:**

bootdist . . . . .	2
bootdistcens . . . . .	5
cdfcompdens . . . . .	7
danish . . . . .	9
descdist . . . . .	10
endosulfan . . . . .	13
fitdist . . . . .	14
fitdistcens . . . . .	23
fluazinam . . . . .	27
gofstat . . . . .	29
graphcomp . . . . .	32
groundbeef . . . . .	36
mgedist . . . . .	37
mledist . . . . .	40
mmedist . . . . .	43
plotdist . . . . .	46
plotdistcens . . . . .	48
qmedist . . . . .	50
quantiles . . . . .	53
salinity . . . . .	55
smokedfish . . . . .	57
toxocara . . . . .	58
<b>Index</b>	<b>60</b>

---

bootdist	<i>Bootstrap simulation of uncertainty for non-censored data</i>
----------	--

---

**Description**

Uses parametric or nonparametric bootstrap resampling in order to simulate uncertainty in the parameters of the distribution fitted to non-censored data.

**Usage**

```
bootdist(f, bootmethod="param", niter=1001)
## S3 method for class 'bootdist'
print(x, ...)
## S3 method for class 'bootdist'
plot(x, ...)
## S3 method for class 'bootdist'
summary(object, ...)
```

**Arguments**

f	An object of class "fitdist", output of the <code>fitdist</code> function.
bootmethod	A character string coding for the type of resampling : "param" for a parametric resampling and "nonparam" for a nonparametric resampling of data.
niter	The number of samples drawn by bootstrap.
x	An object of class "bootdist".
object	An object of class "bootdist".
...	Further arguments to be passed to generic methods

**Details**

Samples are drawn by parametric bootstrap (resampling from the distribution fitted by `fitdist`) or nonparametric bootstrap (resampling with replacement from the data set). On each bootstrap sample the function `mledist` (or `mmedist`, `qmedist`, `mgedist` according to the component `f$method` of the object of class "fitdist") is used to estimate bootstrapped values of parameters. When that function fails to converge, NA values are returned. Medians and 2.5 and 97.5 percentiles are computed by removing NA values. The medians and the 95 percent confidence intervals of parameters (2.5 and 97.5 percentiles) are printed in the summary. If inferior to the whole number of iterations, the number of iterations for which the function converges is also printed in the summary.

The plot of an object of class "bootdist" consists in a scatterplot or a matrix of scatterplots of the bootstrapped values of parameters. It uses the function `stripchart` when the fitted distribution is characterized by only one parameter, and the function `plot` in other cases. In these last cases, it provides a representation of the joint uncertainty distribution of the fitted parameters.

**Value**

`bootdist` returns an object of class "bootdist", a list with 6 components,

<code>estim</code>	a data frame containing the bootstrapped values of parameters.
<code>converg</code>	a vector containing the codes for convergence obtained if an iterative method is used to estimate parameters on each bootstrapped data set (and 0 if a closed formula is used).
<code>method</code>	A character string coding for the type of resampling : "param" for a parametric resampling and "nonparam" for a nonparametric resampling.
<code>nbboot</code>	The number of samples drawn by bootstrap.
<code>CI</code>	bootstrap medians and 95 percent confidence percentile intervals of parameters.
<code>fitpart</code>	The object of class "fitdist" on which the bootstrap procedure was applied.

Generic functions:

`print` The print of a "bootdist" object shows the bootstrap parameter estimates. If inferior to the whole number of bootstrap iterations, the number of iterations for which the estimation converges is also printed.

`summary` The summary provides the median and 2.5 and 97.5 percentiles of each parameter. If inferior to the whole number of bootstrap iterations, the number of iterations for which the estimation converges is also printed in the summary.

`plot` The plot shows the bootstrap estimates with `stripchart` function for univariate parameters and `plot` function for multivariate parameters.

### Author(s)

Marie-Laure Delignette-Muller <marIELaure.delignetteMuller@vetagro-sup.fr>.

### References

Cullen AC and Frey HC (1999) Probabilistic techniques in exposure assessment. Plenum Press, USA, pp. 181-241.

### See Also

`fitdist`, `mledist`, `qmedist`, `mmedist`, `mgedist` and `quantile.bootdist` for another generic function to calculate quantiles from the fitted distribution and its bootstrap results.

### Examples

```
# We choose a low number of bootstrap replicates in order to satisfy CRAN running times constraint.
# For practical applications, we recommend to use at least niter=501 or niter=1001.
```

```
# (1) Fit of a gamma distribution to serving size data
# using default method (maximum likelihood estimation)
# followed by parametric bootstrap
#
data(groundbeef)
x1 <- groundbeef$serving
f1 <- fitdist(x1, "gamma")
b1 <- bootdist(f1, niter=51)
print(b1)
plot(b1)
summary(b1)
quantile(b1)
```

```
# (2) non parametric bootstrap on the same fit
# with less iterations
#
b1b <- bootdist(f1, bootmethod="nonparam", niter=51)
summary(b1b)
quantile(b1b)
```

```
# (3) Fit of a normal distribution on acute toxicity values of endosulfan in log10 for
# nonarthropod invertebrates, using maximum likelihood estimation
# to estimate what is called a species sensitivity distribution
# (SSD) in ecotoxicology, followed by estimation of the 5 percent quantile value of
# the fitted distribution, what is called the 5 percent hazardous concentration (HC5)
# in ecotoxicology, with its two-sided 95 percent confidence interval calculated by
# parametric bootstrap
#
```

```

data(endosulfan)
ATV <- subset(endosulfan, group == "NonArthroInvert")$ATV
log10ATV <- log10(subset(endosulfan, group == "NonArthroInvert")$ATV)
fln <- fitdist(log10ATV, "norm")
bln <- bootdist(fln, bootmethod = "param", niter=51)
quantile(bln, probs = c(0.05, 0.1, 0.2))

```

---

bootdistcens

*Bootstrap simulation of uncertainty for censored data*


---

### Description

Uses nonparametric bootstrap resampling in order to simulate uncertainty in the parameters of the distribution fitted to censored data.

### Usage

```

bootdistcens(f, niter=1001)
## S3 method for class 'bootdistcens'
print(x, ...)
## S3 method for class 'bootdistcens'
plot(x, ...)
## S3 method for class 'bootdistcens'
summary(object, ...)

```

### Arguments

f	An object of class "fitdistcens", output of the <a href="#">fitdistcens</a> function.
niter	The number of samples drawn by bootstrap.
x	An object of class "bootdistcens".
object	An object of class "bootdistcens".
...	Further arguments to be passed to generic methods

### Details

Samples are drawn by nonparametric bootstrap (resampling with replacement from the data set). On each bootstrap sample the function [mledist](#) is used to estimate bootstrapped values of parameters. When [mledist](#) fails to converge, NA values are returned. Medians and 2.5 and 97.5 percentiles are computed by removing NA values. The medians and the 95 percent confidence intervals of parameters (2.5 and 97.5 percentiles) are printed in the summary. If inferior to the whole number of iterations, the number of iterations for which [mledist](#) converges is also printed in the summary.

The plot of an object of class "bootdistcens" consists in a scatterplot or a matrix of scatterplots of the bootstrapped values of parameters. It uses the function [stripchart](#) when the fitted distribution is characterized by only one parameter, and the function [plot](#) in other cases. In these last cases, it provides a representation of the joint uncertainty distribution of the fitted parameters.

**Value**

bootdistcens returns an object of class "bootdistcens", a list with 6 components,

estim	a data frame containing the bootstrapped values of parameters.
converg	a vector containing the codes for convergence of the iterative method used to estimate parameters on each bootstrapped data set.
method	A character string coding for the type of resampling : in this case "nonparam" as it is the only available method for censored data.
nbboot	The number of samples drawn by bootstrap.
CI	bootstrap medians and 95 percent confidence percentile intervals of parameters.
fitpart	The object of class "fitdistcens" on which the bootstrap procedure was applied.

Generic functions:

`print` The print of a "bootdistcens" object shows the bootstrap parameter estimates. If inferior to the whole number of bootstrap iterations, the number of iterations for which the estimation converges is also printed.

`summary` The summary provides the median and 2.5 and 97.5 percentiles of each parameter. If inferior to the whole number of bootstrap iterations, the number of iterations for which the estimation converges is also printed in the summary.

`plot` The plot shows the bootstrap estimates with the `stripchart` function for univariate parameters and `plot` function for multivariate parameters.

**Author(s)**

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr>.

**References**

Cullen AC and Frey HC (1999) Probabilistic techniques in exposure assessment. Plenum Press, USA, pp. 181-241.

**See Also**

`fitdistcens`, `mledist` and `quantile.bootdistcens` for another generic function to calculate quantiles from the fitted distribution and its bootstrap results.

**Examples**

```
# We choose a low number of bootstrap replicates in order to satisfy CRAN running times constraint.
# For practical applications, we recommend to use at least niter=501 or niter=1001.

# (1) Fit of a normal distribution to fluazinam data in log10
# followed by nonparametric bootstrap and calculation of quantiles
# with 95 percent confidence intervals
#
data(fluazinam)
```

```

(d1 <-log10(fluazinam))
f1 <- fitdistcens(d1, "norm")
b1 <- bootdistcens(f1, niter = 101)
b1
summary(b1)
plot(b1)
quantile(b1)

# (2) Estimation of the mean of the normal distribution
# by maximum likelihood with the standard deviation fixed at 1
# using the argument fix.arg
# followed by nonparametric bootstrap
# and calculation of quantiles with 95 percent confidence intervals
#
f1b <- fitdistcens(d1, "norm", start = list(mean = 1),fix.arg = list(sd = 1))
b1b <- bootdistcens(f1b, niter = 101)
summary(b1b)
plot(b1b)
quantile(b1b)

```

---

cdfcompcens

*Compares various fitted distributions to censored data on a cdf plot*


---

## Description

Plots the empirical cumulative distribution (censored continuous data) with theoretical ones corresponding to various fitted distributions.

## Usage

```

cdfcompcens(ft, xlim, ylim, xlogscale = FALSE, ylogscale = FALSE, main, xlab, ylab,
  datacol, fitlty, fitcol, addlegend = TRUE, legendtext, xlegend = "bottomright",
  ylegend = NULL, lines01 = FALSE, Turnbull.confint = FALSE, ...)

```

## Arguments

ft	One "fitdistcens" object or a list of objects of class "fitdistcens".
xlim	The $x$ -limits of the plot.
ylim	The $y$ -limits of the plot.
xlogscale	If TRUE, uses a logarithmic scale for the $x$ -axis.
ylogscale	If TRUE, uses a logarithmic scale for the $y$ -axis.
main	A main title for the plot, see also <a href="#">title</a> .
xlab	A label for the $x$ -axis, defaults to a description of $x$ .
ylab	A label for the $y$ -axis, defaults to a description of $y$ .
datacol	A specification of the color to be used in plotting data points.

<code>fitcol</code>	A (vector of) color(s) to plot fitted distributions. If there are fewer colors than fits they are recycled in the standard fashion.
<code>fitlty</code>	A (vector of) line type(s) to plot fitted distributions. If there are fewer colors than fits they are recycled in the standard fashion. See also <a href="#">par</a> .
<code>addlegend</code>	If TRUE, a legend is added to the plot.
<code>legendtext</code>	A character or expression vector of length $\geq 1$ to appear in the legend, see also <a href="#">legend</a> .
<code>xlegend, ylegend</code>	The $x$ and $y$ co-ordinates to be used to position the legend. They can be specified by keyword or in any way which is accepted by 'xy.coords': see <a href="#">legend</a> for details.
<code>lines01</code>	A logical to plot two horizontal lines at $h=0$ and $h=1$ .
<code>Turnbull.confint</code>	if TRUE, confidence intervals will be added to the Turnbull plot of the empirical distribution.
<code>...</code>	Further graphical arguments passed to graphical functions used in <a href="#">cdfcompens</a> .

## Details

Empirical and theoretical distributions are plotted in cdf. The EM approach of Turnbull (Turnbull, 1974) is used to compute the overall empirical cdf curve, with confidence intervals if `Turnbull.confint` is TRUE, by calls to functions [survfit](#) and [plot.survfit](#) from the survival package. By default a legend is added to the plot. Many graphical arguments are optional, dedicated to personalize the plot, and fixed to default values if omitted.

## Author(s)

Marie-Laure Delignette-Muller <marIELaure.delignetteMuller@vetagro-sup.fr>.

## References

Turnbull BW (1974) Nonparametric estimation of a survivorship function with doubly censored data. *Journal of American Statistical Association*, **69**, 169-173.

## See Also

[plotdistcens](#), [survfit.formula](#), [legend](#) and [par](#).

## Examples

```
# (1) Plot various distributions fitted to bacterial contamination data
#
data(smokedfish)
Clog10 <- log10(smokedfish)

fitsfn <- fitdistcens(Clog10,"norm")
summary(fitsfn)
```



```

fitsfl <- fitdistcens(Clog10,"logis")
summary(fitsfl)

dgumbel <- function(x,a,b) 1/b*exp((a-x)/b)*exp(-exp((a-x)/b))
pgumbel <- function(q,a,b) exp(-exp((a-q)/b))
qgumbel <- function(p,a,b) a-b*log(-log(p))
fitsfg<-fitdistcens(Clog10,"gumbel",start=list(a=-3,b=3))
summary(fitsfg)

cdfcompcens(list(fitsfn,fitsfl,fitsfg))
cdfcompcens(list(fitsfn,fitsfl,fitsfg),datacol="orange",
  legendtext=c("normal","logistic","Gumbel"),
  main="bacterial contamination fits",
  xlab="bacterial concentration (CFU/g)",ylab="F",
  xlegend = "center",lines01 = TRUE)

```

---

danish

*Danish reinsurance claim dataset*


---

### Description

The univariate dataset was collected at Copenhagen Reinsurance and comprise 2167 fire losses over the period 1980 to 1990. They have been adjusted for inflation to reflect 1985 values and are expressed in millions of Danish Krone.

The multivariate data set is the same data as above but the total claim has been divided into a building loss, a loss of contents and a loss of profits.

### Usage

```

data(danishuni)
data(danishmulti)

```

### Format

danishuni contains two columns:

Date The day of claim occurrence.

Loss The total loss amount in millions of Danish Krone (DKK).

danishmulti contains five columns:

Date The day of claim occurrence.

Building The loss amount (mDKK) of the building coverage.

Contents The loss amount (mDKK) of the contents coverage.

Profits The loss amount (mDKK) of the profit coverage.

Total The total loss amount (mDKK).

All columns are numeric except Date columns of class Date.

**Source**

Alexander McNeil, see <http://www.ma.hw.ac.uk/~mcneil/data.html>

**References**

Dataset used in McNeil (1996), *Estimating the Tails of Loss Severity Distributions using Extreme Value Theory*, ASTIN Bull.

**Examples**

```
# (1) load of data
#
data(danishuni)

# (2) plot and description of data
#
plotdist(danishuni$Loss)

# (3) load of data
#
data(danishmulti)

# (4) plot and description of data
#
idx <- sample(1:NROW(danishmulti), 10)
barplot(danishmulti$Building[idx], col="grey25",
        ylim=c(0, max(danishmulti$Total[idx])), main="Some claims of danish data set")
barplot(danishmulti$Content[idx], add=TRUE, col="grey50", axes=FALSE)
barplot(danishmulti$Profits[idx], add=TRUE, col="grey75", axes=FALSE)
legend("topleft", legend=c("Building", "Content", "Profits"), fill=c("grey25",
"grey50", "grey75"))
```

---

descdist

*Description of an empirical distribution for non-censored data*

---

**Description**

Computes descriptive parameters of an empirical distribution for non-censored data and provides a skewness-kurtosis plot.

**Usage**

```
descdist(data, discrete = FALSE, boot = NULL, method = "unbiased",
graph = TRUE, obs.col = "darkblue", boot.col = "orange")
```

**Arguments**

<code>data</code>	A numeric vector.
<code>discrete</code>	If TRUE, the distribution is considered as discrete.
<code>boot</code>	If not NULL, boot values of skewness and kurtosis are plotted from bootstrap samples of data. boot must be fixed in this case to an integer above 10.
<code>method</code>	"unbiased" for unbiased estimated values of statistics or "sample" for sample values.
<code>graph</code>	If FALSE, the skewness-kurtosis graph is not plotted.
<code>obs.col</code>	Color used for the observed point on the skewness-kurtosis graph.
<code>boot.col</code>	Color used for bootstrap sample of points on the skewness-kurtosis graph.

**Details**

Minimum, maximum, median, mean, sample sd, and sample (if `method=="sample"`) or by default unbiased estimations of skewness and Pearson's kurtosis values are printed (Sokal and Rohlf, 1995). A skewness-kurtosis plot such as the one proposed by Cullen and Frey (1999) is given for the empirical distribution. On this plot, values for common distributions are also displayed as tools to help the choice of distributions to fit to data. For some distributions (normal, uniform, logistic, exponential for example), there is only one possible value for the skewness and the kurtosis (for a normal distribution for example, skewness = 0 and kurtosis = 3), and the distribution is thus represented by a point on the plot. For other distributions, areas of possible values are represented, consisting in lines (gamma and lognormal distributions for example), or larger areas (beta distribution for example). The Weibull distribution is not represented on the graph but it is indicated on the legend that shapes close to lognormal and gamma distributions may be obtained with this distribution.

In order to take into account the uncertainty of the estimated values of kurtosis and skewness from data, the data set may be bootstrapped by fixing the argument `boot` to an integer above 10. boot values of skewness and kurtosis corresponding to the boot bootstrap samples are then computed and reported in blue color on the skewness-kurtosis plot.

If `discrete` is TRUE, the represented distributions are the Poisson, negative binomial distributions, and the normal distribution to which previous discrete distributions may converge. If `discrete` is FALSE, these are uniform, normal, logistic, lognormal, beta and gamma distributions.

**Value**

`descdist` returns a list with 7 components,

<code>min</code>	the minimum value
<code>max</code>	the maximum value
<code>median</code>	the median value
<code>mean</code>	the mean value
<code>sd</code>	the standard deviation sample or estimated value
<code>skewness</code>	the skewness sample or estimated value
<code>kurtosis</code>	the kurtosis sample or estimated value

**Author(s)**

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr>.

**References**

Cullen AC and Frey HC (1999) Probabilistic techniques in exposure assessment. Plenum Press, USA, pp. 81-159.

Evans M, Hastings N and Peacock B (2000) Statistical distributions. John Wiley and Sons Inc.

Sokal RR and Rohlf FJ (1995) Biometry. W.H. Freeman and Company, USA, pp. 111-115.

**See Also**

[plotdist](#)

**Examples**

```
# (1) Description of a sample from a normal distribution
# with and without uncertainty on skewness and kurtosis estimated by bootstrap
#
set.seed(1234)
x1 <- rnorm(100)
descdist(x1)
descdist(x1,boot=500)

# (2) Description of a sample from a beta distribution
# with uncertainty on skewness and kurtosis estimated by bootstrap
# with changing of default colors
#
descdist(rbeta(100,shape1=0.05,shape2=1),boot=500,
obs.col="blue",boot.col="yellow")

# (3) Description of a sample from a gamma distribution
# with uncertainty on skewness and kurtosis estimated by bootstrap
# without plotting
#
descdist(rgamma(100,shape=2,rate=1),boot=500,graph=FALSE)

# (3) Description of a sample from a Poisson distribution
# with uncertainty on skewness and kurtosis estimated by bootstrap
#
descdist(rpois(100,lambda=2),discrete=TRUE,boot=500)

# (4) Description of serving size data
# with uncertainty on skewness and kurtosis estimated by bootstrap
#
data(groundbeef)
serving <- groundbeef$serving
descdist(serving, boot=500)
```

---

endosulfan

*Species Sensitivity Distribution (SSD) for endosulfan*

---

## Description

Summary of 48- to 96-hour acute toxicity values (LC50 and EC50 values) for exposure of Australian and Non-Australian taxa to endosulfan.

## Usage

```
data(endosulfan)
```

## Format

endosulfan is a data frame with 3 columns, named ATV for Acute Toxicity Value (geometric mean of LC50 or EC50 values in micrograms per liter), Australian (coding for Australian or another origin) and group (arthropods, fish or non-arthropod invertebrates).

## Source

Hose, G.C., Van den Brink, P.J. 2004. Confirming the Species-Sensitivity Distribution Concept for Endosulfan Using Laboratory, Mesocosms, and Field Data. *Archives of Environmental Contamination and Toxicology*, **47**, 511-520.

## Examples

```
# (1) load of data
#
data(endosulfan)

# (2) plot and description of data for non Australian fish in decimal logarithm
#
log10ATV <- log10(subset(endosulfan, (Australian == "no") & (group == "Fish"))$ATV)
plotdist(log10ATV)
descdist(log10ATV, boot=1000)

# (3) fit of a normal and a logistic distribution to data in log10
# (classical distributions used for SSD)
# and visual comparison of the fits
#
f1n <- fitdist(log10ATV, "norm")
summary(f1n)

f1l <- fitdist(log10ATV, "logis")
summary(f1l)

cdfcomp(list(f1n, f1l), legendtext=c("normal", "logistic"),
xlab="log10ATV")
```

```

denscomp(list(fln,fll),legendtext=c("normal","logistic"),
xlab="log10ATV")

qqcomp(list(fln,fll),legendtext=c("normal","logistic"))
ppcomp(list(fln,fll),legendtext=c("normal","logistic"))

gofstat(list(fln,fll))

# (4) estimation of the 5 percent quantile value of
# logistic fitted distribution (5 percent hazardous concentration : HC5)
# with its two-sided 95 percent confidence interval calculated by
# parametric bootstrap
# with a small number of iterations to satisfy CRAN running times constraint.
# For practical applications, we recommend to use at least niter=501 or niter=1001.
#
# in log10(ATV)
bll <- bootdist(fll,niter=101)
HC5l1 <- quantile(bll,probs = 0.05)
# in ATV
10^(HC5l1$quantiles)
10^(HC5l1$quantCI)

# (5) estimation of the 5 percent quantile value of
# the fitted logistic distribution (5 percent hazardous concentration : HC5)
# with its one-sided 95 percent confidence interval (type "greater")
# calculated by
# nonparametric bootstrap
# with a small number of iterations to satisfy CRAN running times constraint.
# For practical applications, we recommend to use at least niter=501 or niter=1001.
#
# in log10(ATV)
bllnonpar <- bootdist(fll,niter=101,bootmethod = "nonparam")
HC5l1greater <- quantile(bllnonpar,probs = 0.05, CI.type="greater")
# in ATV
10^(HC5l1greater$quantiles)
10^(HC5l1greater$quantCI)

# (6) fit of a logistic distribution
# by minimizing the modified Anderson-Darling AD2L distance
# cf. ?mgdist for definition of this distance
#
fllAD2L <- fitdist(log10ATV,"logis",method="mge",gof="AD2L")
summary(fllAD2L)
plot(fllAD2L)

```

**Description**

Fit of univariate distributions to non-censored data by maximum likelihood, moment matching, quantile matching or maximizing goodness-of-fit (minimizing distance). Generic methods are `print`, `plot`, `summary` and `quantile`.

**Usage**

```
fitdist(data, distr, method = c("mle", "mme", "qme", "mge"),
        start=NULL, fix.arg=NULL, ...)
```

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

```
## S3 method for class 'fitdist'
plot(x, breaks="default", ...)
```

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

**Arguments**

<code>data</code>	A numeric vector.
<code>distr</code>	A character string "name" naming a distribution for which the corresponding density function <code>dname</code> , the corresponding distribution function <code>pname</code> and the corresponding quantile function <code>qname</code> must be defined, or directly the density function.
<code>method</code>	A character string coding for the fitting method: "mle" for 'maximum likelihood estimation', "mme" for 'moment matching estimation', "qme" for 'quantile matching estimation' and "mge" for 'maximum goodness-of-fit estimation'.
<code>start</code>	A named list giving the initial values of parameters of the named distribution. This argument may be omitted for some distributions for which reasonable starting values are computed (see details), and will not be taken into account if a closed formula is used to estimate parameters.
<code>fix.arg</code>	An optional named list giving the values of parameters of the named distribution that must be kept fixed rather than estimated. The use of this argument is not possible if <code>method="mme"</code> and a closed formula is used.
<code>x</code>	An object of class "fitdist".
<code>object</code>	An object of class "fitdist".
<code>breaks</code>	If "default" the histogram is plotted with the function <code>hist</code> with its default breaks definition. Else <code>breaks</code> is passed to the function <code>hist</code> . This argument is not taken into account with discrete distributions: "binom", "nbinom", "geom", "hyper" and "pois".
<code>...</code>	Further arguments to be passed to generic functions, or to one of the functions "mledist", "mmedist", "qmedist" or "mgedist" depending of the chosen method (see the help pages of these functions for details).

## Details

Fitting methods:

**When** `method="mle"` maximum likelihood estimations of the distribution parameters are computed using the function `mledist`.

**When** `method="mme"` the estimated values of the distribution parameters are computed by a closed formula for the following distributions: `"norm"`, `"lnorm"`, `"pois"`, `"exp"`, `"gamma"`, `"nbinom"`, `"geom"`, `"beta"`, `"unif"` and `"logis"`. For distributions characterized by one parameter (`"geom"`, `"pois"` and `"exp"`), this parameter is simply estimated by matching theoretical and observed means, and for distributions characterized by two parameters, these parameters are estimated by matching theoretical and observed means and variances (Vose, 2000). For other distributions, the theoretical and the empirical moments are matched numerically, by minimization of the sum of squared differences between observed and theoretical moments. In this last case, further arguments are needed in the call to `fitdist`: `order` and `memp` (see `mmedist` for details).

**When** `method = "qme"` the function carries out the quantile matching numerically, by minimization of the sum of squared differences between observed and theoretical quantiles. The use of this method requires an additional argument `probs`, defined as the numeric vector of the probabilities for which the quantile matching is done, of length equal to the number of parameters to estimate (see `qmedist` for details).

**When** `method = "mge"` the distribution parameters are estimated by maximization of goodness-of-fit (or minimization of a goodness-of-fit distance). The use of this method requires an additional argument `gof` coding for the goodness-of-fit distance chosen. One can use the classical Cramer-von Mises distance (`"CvM"`), the classical Kolmogorov-Smirnov distance (`"KS"`), the classical Anderson-Darling distance (`"AD"`) which gives more weight to the tails of the distribution, or one of the variants of this last distance proposed by Luceno (2006) (see `mgedist` for more details). This method is not suitable for discrete distributions.

By default direct optimization of the log-likelihood (or other criteria depending of the chosen method) is performed using `optim`, with the "Nelder-Mead" method for distributions characterized by more than one parameter and the "BFGS" method for distributions characterized by only one parameter. The optimization algorithm used in `optim` can be chosen or another optimization function can be specified using `...argument` (see `mledist` for details). For the following named distributions, reasonable starting values will be computed if `start` is omitted: `"norm"`, `"lnorm"`, `"exp"` and `"pois"`, `"cauchy"`, `"gamma"`, `"logis"`, `"nbinom"` (parametrized by `mu` and `size`), `"geom"`, `"beta"`, `"unif"`, and `"weibull"`. Note that these starting values may not be good enough if the fit is poor. With the parameter estimates, the function returns the log-likelihood whatever the estimation method and for maximum likelihood estimation the standard errors of the estimates calculated from the Hessian at the solution found by `optim` or by the user-supplied function passed to `mledist`.

NB: if your data values are particularly small or large, a scaling may be needed before the optimization process. See example (14).

## Value

`fitdist` returns an object of class `"fitdist"`, a list with the following components:

`estimate`            the parameter estimates



method	the character string coding for the fitting method : "mle" for 'maximum likelihood estimation', "mme" for 'matching moment estimation', "qme" for 'matching quantile estimation' and "mge" for 'maximum goodness-of-fit estimation'
sd	the estimated standard errors or NULL if not available
cor	the estimated correlation matrix or NULL if not available
loglik	the log-likelihood
aic	the Akaike information criterion
bic	the the so-called BIC or SBC (Schwarz Bayesian criterion)
n	the length of the data set
data	the data set
distname	the name of the distribution
fix.arg	the named list giving the values of parameters of the named distribution that must be kept fixed rather than estimated by maximum likelihood or NULL if there are no such parameters.
dots	the list of further arguments passed in ... to be used in <a href="#">bootdist</a> in iterative calls to <a href="#">mledist</a> , <a href="#">mmedist</a> , <a href="#">qmedist</a> , <a href="#">mgedist</a> or NULL if no such arguments.

#### Generic functions:

**print** The print of a "fitdist" object shows few traces about the fitting method and the fitted distribution.

**summary** The summary provides the parameter estimates of the fitted distribution, the log-likelihood, AIC and BIC statistics and when the maximum likelihood is used, the standard errors of the parameter estimates and the correlation matrix between parameter estimates.

**plot** The plot of an object of class "fitdist" returned by [fitdist](#) uses the function [plotdist](#). An object of class "fitdist" or a list of objects of class "fitdist" corresponding to various fits using the same data set may also be plotted using a cdf plot (function [cdfcomp](#)), a density plot (function [denscomp](#)), a density Q-Q plot (function [qqcomp](#)), or a P-P plot (function [ppcomp](#)).

#### Author(s)

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr> and Christophe Dutang.

#### References

Cullen AC and Frey HC (1999) Probabilistic techniques in exposure assessment. Plenum Press, USA, pp. 81-155.

Venables WN and Ripley BD (2002) Modern applied statistics with S. Springer, New York, pp. 435-446.

Vose D (2000) Risk analysis, a quantitative guide. John Wiley & Sons Ltd, Chichester, England, pp. 99-143.

**See Also**

See [mledist](#), [mmedist](#), [qmedist](#), [mgedist](#) for details on parameter estimation. See [gofstat](#) for goodness-of-fit statistics. See [plotdist](#), [graphcomp](#) for graphs. See [bootdist](#) for bootstrap procedures and [fitdistcens](#) for censored-data fitting methods. See [optim](#) for base R optimization procedures. See [quantile.fitdist](#), another generic function, which calculates quantiles from the fitted distribution.

**Examples**

```
# (1) fit of a gamma distribution by maximum likelihood estimation
#

data(groundbeef)
serving <- groundbeef$serving
fitg <- fitdist(serving, "gamma")
summary(fitg)
plot(fitg)
plot(fitg, demp = TRUE)
plot(fitg, histo = FALSE, demp = TRUE)
cdfcomp(fitg, addlegend=FALSE)
denscomp(fitg, addlegend=FALSE)
ppcomp(fitg, addlegend=FALSE)
qqcomp(fitg, addlegend=FALSE)

# (2) use the moment matching estimation (using a closed formula)
#

fitgmme <- fitdist(serving, "gamma", method="mme")
summary(fitgmme)

# (3) Comparison of various fits
#

fitW <- fitdist(serving, "weibull")
fitg <- fitdist(serving, "gamma")
fitln <- fitdist(serving, "lnorm")
summary(fitW)
summary(fitg)
summary(fitln)
cdfcomp(list(fitW, fitg, fitln), legendtext=c("Weibull", "gamma", "lognormal"))
denscomp(list(fitW, fitg, fitln), legendtext=c("Weibull", "gamma", "lognormal"))
qqcomp(list(fitW, fitg, fitln), legendtext=c("Weibull", "gamma", "lognormal"))
ppcomp(list(fitW, fitg, fitln), legendtext=c("Weibull", "gamma", "lognormal"))
gofstat(list(fitW, fitg, fitln), fitnames=c("Weibull", "gamma", "lognormal"))

# (4) defining your own distribution functions, here for the Gumbel distribution
# for other distributions, see the CRAN task view
# dedicated to probability distributions
#
```

```

dgumbel <- function(x, a, b) 1/b*exp((a-x)/b)*exp(-exp((a-x)/b))
pgumbel <- function(q, a, b) exp(-exp((a-q)/b))
qgumbel <- function(p, a, b) a-b*log(-log(p))

fitgumbel <- fitdist(serving, "gumbel", start=list(a=10, b=10))
summary(fitgumbel)
plot(fitgumbel)

# (5) fit discrete distributions (Poisson and negative binomial)
#

data(toxocara)
number <- toxocara$number
fitp <- fitdist(number,"pois")
summary(fitp)
plot(fitp)
fitnb <- fitdist(number,"nbinom")
summary(fitnb)
plot(fitnb)

cdfcomp(list(fitp,fitnb))
gofstat(list(fitp,fitnb))

# (6) how to change the optimisation method?
#

data(groundbeef)
serving <- groundbeef$serving
fitdist(serving, "gamma", optim.method="Nelder-Mead")
fitdist(serving, "gamma", optim.method="BFGS")
fitdist(serving, "gamma", optim.method="SANN")

# (7) custom optimization function
#

#create the sample
set.seed(1234)
mysample <- rexp(100, 5)
mystart <- list(rate=8)

res1 <- fitdist(mysample, dexp, start= mystart, optim.method="Nelder-Mead")

#show the result
summary(res1)

#the warning tell us to use optimise, because the Nelder-Mead is not adequate.

#to meet the standard 'fn' argument and specific name arguments, we wrap optimize,
myoptimize <- function(fn, par, ...)
{
  res <- optimize(f=fn, ..., maximum=FALSE)
  #assume the optimization function minimize

```

```

        standardres <- c(res, convergence=0, value=res$objective,
            par=res$minimum, hessian=NA)

    return(standardres)
}

#call fitdist with a 'custom' optimization function
res2 <- fitdist(mysample, dexp, start=mystart, custom.optim=myoptimize,
    interval=c(0, 100))

#show the result
summary(res2)

# (8) custom optimization function - another example with the genetic algorithm
#
## Not run:
#set a sample
fit1 <- fitdist(serving, "gamma")
summary(fit1)

#wrap genoud function rgenoud package
mygenoud <- function(fn, par, ...)
{
    require(rgenoud)
    res <- genoud(fn, starting.values=par, ...)
    standardres <- c(res, convergence=0)

    return(standardres)
}

#call fitdist with a 'custom' optimization function
fit2 <- fitdist(serving, "gamma", custom.optim=mygenoud, nvars=2,
    Domains=cbind(c(0, 0), c(10, 10)), boundary.enforcement=1,
    print.level=1, hessian=TRUE)

summary(fit2)

## End(Not run)

# (9) estimation of the standard deviation of a gamma distribution
# by maximum likelihood with the shape fixed at 4 using the argument fix.arg
#

data(groundbeef)
serving <- groundbeef$serving
f1c <- fitdist(serving, "gamma", start=list(rate=0.1), fix.arg=list(shape=4))
summary(f1c)
plot(f1c)

# (10) fit of a Weibull distribution to serving size data
# by maximum likelihood estimation
# or by quantile matching estimation (in this example

```

```

# matching first and third quartiles)
#

data(groundbeef)
serving <- groundbeef$serving
fWmle <- fitdist(serving, "weibull")
summary(fWmle)
plot(fWmle)
gofstat(fWmle)

fWqme <- fitdist(serving, "weibull", method="qme", probs=c(0.25, 0.75))
summary(fWqme)
plot(fWqme)
gofstat(fWqme)

# (11) Fit of a Pareto distribution by numerical moment matching estimation
#

## Not run:
require(actuar)
#simulate a sample
x4 <- rpareto(1000, 6, 2)

#empirical raw moment
memp <- function(x, order)
  ifelse(order == 1, mean(x), sum(x^order)/length(x))

#fit
fP <- fitdist(x4, "pareto", method="mme", order=c(1, 2), memp="memp",
start=c(shape=10, scale=10), lower=1, upper=Inf)
summary(fP)
plot(fP)

## End(Not run)

# (12) Fit of a Weibull distribution to serving size data by maximum
# goodness-of-fit estimation using all the distances available
#

data(groundbeef)
serving <- groundbeef$serving
(f1 <- fitdist(serving, "weibull", method="mge", gof="CvM"))
(f2 <- fitdist(serving, "weibull", method="mge", gof="KS"))
(f3 <- fitdist(serving, "weibull", method="mge", gof="AD"))
(f4 <- fitdist(serving, "weibull", method="mge", gof="ADR"))
(f5 <- fitdist(serving, "weibull", method="mge", gof="ADL"))
(f6 <- fitdist(serving, "weibull", method="mge", gof="AD2R"))
(f7 <- fitdist(serving, "weibull", method="mge", gof="AD2L"))
(f8 <- fitdist(serving, "weibull", method="mge", gof="AD2"))
cdfcomp(list(f1, f2, f3, f4, f5, f6, f7, f8))
cdfcomp(list(f1, f2, f3, f4, f5, f6, f7, f8), xlogscale=TRUE, xlim=c(8, 250), verticals=TRUE)

```

```

denscomp(list(f1, f2, f3, f4, f5, f6, f7, f8))

# (13) Fit of a uniform distribution using maximum likelihood
# (a closed formula is used in this special case where the loglikelihood is not defined),
# or maximum goodness-of-fit with Cramer-von Mises or Kolmogorov-Smirnov distance
#

set.seed(1234)
u <- runif(50, min=5, max=10)

fumle <- fitdist(u, "unif", method="mle")
summary(fumle)
plot(fumle)
gofstat(fumle)

fuCvM <- fitdist(u, "unif", method="mge", gof="CvM")
summary(fuCvM)
plot(fuCvM)
gofstat(fuCvM)

fuKS <- fitdist(u, "unif", method="mge", gof="KS")
summary(fuKS)
plot(fuKS)
gofstat(fuKS)

# (14) scaling problem
# the simulated dataset (below) has particularly small values, hence without scaling (10^0),
# the optimization raises an error. The for loop shows how scaling by 10^i
# for i=1,...,6 makes the fitting procedure work correctly.

set.seed(1234)
x2 <- rnorm(100, 1e-4, 2e-4)

for(i in 0:6)
  cat(i, try(fitdist(x2*10^i, "cauchy", method="mle")$estimate, silent=TRUE), "\n")

# (15) Fit of a normal distribution on acute toxicity values of endosulfan in log10 for
# nonarthropod invertebrates, using maximum likelihood estimation
# to estimate what is called a species sensitivity distribution
# (SSD) in ecotoxicology, followed by estimation of the 5 percent quantile value of
# the fitted distribution (which is called the 5 percent hazardous concentration, HC5,
# in ecotoxicology) and estimation of other quantiles.
#
data(endosulfan)
ATV <- subset(endosulfan, group == "NonArthroInvert")$ATV
log10ATV <- log10(subset(endosulfan, group == "NonArthroInvert")$ATV)
fln <- fitdist(log10ATV, "norm")

quantile(fln, probs = 0.05)
quantile(fln, probs = c(0.05, 0.1, 0.2))

# (16) Fit of a triangular distribution using Cramer-von Mises or

```

```

# Kolmogorov-Smirnov distance
#

## Not run:
set.seed(1234)
require(mc2d)
t <- rtriang(100, min=5, mode=6, max=10)
fCvM <- fitdist(t, "triang", method="mge", start = list(min=4, mode=6,max=9), gof="CvM")
fKS <- fitdist(t, "triang", method="mge", start = list(min=4, mode=6,max=9), gof="KS")
cdfcomp(list(fCvM,fKS))

## End(Not run)

```

---

fitdistcens

*Fitting of univariate distributions to censored data*


---

## Description

Fits a univariate distribution to censored data by maximum likelihood.

## Usage

```
fitdistcens(censdata, distr, start=NULL, fix.arg=NULL, ...)
```

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

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

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

## Arguments

censdata	A dataframe of two columns respectively named <code>left</code> and <code>right</code> , describing each observed value as an interval. The left column contains either NA for left censored observations, the left bound of the interval for interval censored observations, or the observed value for non-censored observations. The right column contains either NA for right censored observations, the right bound of the interval for interval censored observations, or the observed value for non-censored observations.
distr	A character string "name" naming a distribution, for which the corresponding density function <code>dname</code> and the corresponding distribution function <code>pname</code> must be defined, or directly the density function.

start	A named list giving the initial values of parameters of the named distribution. This argument may be omitted for some distributions for which reasonable starting values are computed (see details).
fix.arg	An optional named list giving the values of parameters of the named distribution that must be kept fixed rather than estimated by maximum likelihood.
x	an object of class "fitdistcens".
object	an object of class "fitdistcens".
...	further arguments to be passed to generic functions, to the function plotdistcens in order to control the type of ecdf-plot used for censored data, or to the function mledist in order to control the optimization method.

### Details

Maximum likelihood estimations of the distribution parameters are computed using the function `mledist`. By default direct optimization of the log-likelihood is performed using `optim`, with the "Nelder-Mead" method for distributions characterized by more than one parameter and the "BFGS" method for distributions characterized by only one parameter. The algorithm used in `optim` can be chosen or another optimization function can be specified using `...` argument (see `mledist` for details). For the following named distributions, reasonable starting values will be computed if `start` is omitted: "norm", "lnorm", "exp" and "pois", "cauchy", "gamma", "logis", "nbinom" (parametrized by mu and size), "geom", "beta" and "weibull". Note that these starting values may not be good enough if the fit is poor. The function is not able to fit a uniform distribution. With the parameter estimates, the function returns the log-likelihood and the standard errors of the estimates calculated from the Hessian at the solution found by `optim` or by the user-supplied function passed to `mledist`.

### Value

`fitdistcens` returns an object of class "fitdistcens", a list with the following components:

estimate	the parameter estimates
sd	the estimated standard errors
cor	the estimated correlation matrix
loglik	the log-likelihood
aic	the Akaike information criterion
bic	the the so-called BIC or SBC (Schwarz Bayesian criterion)
censdata	the censored data set
distname	the name of the distribution
fix.arg	the named list giving the values of parameters of the named distribution that must be kept fixed rather than estimated by maximum likelihood or NULL if there are no such parameters.
dots	the list of further arguments passed in <code>...</code> to be used in <code>bootdistcens</code> to control the optimization method used in iterative calls to <code>mledist</code> or NULL if no such arguments.

Generic functions:



`print` The print of a "fitdist" object shows few traces about the fitting method and the fitted distribution.

`summary` The summary provides the parameter estimates of the fitted distribution, the log-likelihood, AIC and BIC statistics, the standard errors of the parameter estimates and the correlation matrix between parameter estimates.

`plot` The plot of an object of class "fitdistcens" returned by `fitdistcens` uses the function `plotdistcens`.

### Author(s)

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr>.

### References

Venables WN and Ripley BD (2002) Modern applied statistics with S. Springer, New York, pp. 435-446.

### See Also

`plotdistcens`, `optim`, `mledist`, `fitdist` and `quantile.fitdistcens` for another generic function to calculate quantiles from the fitted distribution.

### Examples

```
# (1) Fit of a lognormal distribution to bacterial contamination data
#
data(smokedfish)
fitsf <- fitdistcens(smokedfish,"lnorm")
summary(fitsf)
# default plot using the Turnbull algorithm
plot(fitsf)
# plot using the Turnbull algorithm with confidence intervals for the
# empirical distribution
plot(fitsf, Turnbull.confint = TRUE)
# basic plot using intervals and points (see ?plotdistcens for details)
plot(fitsf, Turnbull = FALSE)
# plot of the same fit using the Turnbull algorithm in logscale
cdfcompdens(fitsf,main="bacterial contamination fits",
             xlab="bacterial concentration (CFU/g)",ylab="F",
             addlegend = FALSE,lines01 = TRUE, xlogscale = TRUE, xlim = c(1e-2,1e2))
# zoom on large values of F
cdfcompdens(fitsf,main="bacterial contamination fits",
             xlab="bacterial concentration (CFU/g)",ylab="F",
             addlegend = FALSE,lines01 = TRUE, xlogscale = TRUE,
             xlim = c(1e-2,1e2),ylim=c(0.4,1))

# (2) Fit of a normal distribution on acute toxicity values
# of fluazinam (in decimal logarithm) for
# macroinvertebrates and zooplankton, using maximum likelihood estimation
# to estimate what is called a species sensitivity distribution
# (SSD) in ecotoxicology
```

```

#

data(fluazinam)
log10EC50 <- log10(fluazinam)
fln <- fitdistcens(log10EC50, "norm")
fln
summary(fln)
plot(fln)

# (3) defining your own distribution functions, here for the Gumbel distribution
# for other distributions, see the CRAN task view dedicated to
# probability distributions
#

dgumbel <- function(x,a,b) 1/b*exp((a-x)/b)*exp(-exp((a-x)/b))
pgumbel <- function(q,a,b) exp(-exp((a-q)/b))
qgumbel <- function(p,a,b) a-b*log(-log(p))
fg <- fitdistcens(log10EC50, "gumbel", start=list(a=1,b=1))
summary(fg)
plot(fg)

# (4) comparison of fits of various distributions
#

fll <- fitdistcens(log10EC50, "logis")
summary(fll)

cdfcompdens(list(fln,fll,fg), legendtext=c("normal", "logistic", "gumbel"),
xlab = "log10(EC50)")

# (5) how to change the optimisation method?
#

fitdistcens(log10EC50, "logis", optim.method="Nelder-Mead")
fitdistcens(log10EC50, "logis", optim.method="BFGS")
fitdistcens(log10EC50, "logis", optim.method="SANN")

# (6) custom optimisation function - example with the genetic algorithm
#
## Not run:

#wrap genoud function rgenoud package
mygenoud <- function(fn, par, ...)
{
  require(rgenoud)
  res <- genoud(fn, starting.values=par, ...)
  standardres <- c(res, convergence=0)

  return(standardres)
}

# call fitdistcens with a 'custom' optimization function
fit.with.genoud <- fitdistcens(log10EC50, "logis", custom.optim=mygenoud, nvars=2,

```

```

    Domains=cbind(c(0,0), c(5, 5)), boundary.enforcement=1,
    print.level=1, hessian=TRUE)

summary(fit.with.genoud)

## End(Not run)

# (7) estimation of the mean of a normal distribution
# by maximum likelihood with the standard deviation fixed at 1 using the argument fix.arg
#
flnb <- fitdistcens(log10EC50, "norm", start = list(mean = 1), fix.arg = list(sd = 1))

# (8) Fit of a lognormal distribution on acute toxicity values of fluazinam for
# macroinvertebrates and zooplankton, using maximum likelihood estimation
# to estimate what is called a species sensitivity distribution
# (SSD) in ecotoxicology, followed by estimation of the 5 percent quantile value of
# the fitted distribution (which is called the 5 percent hazardous concentration, HC5,
# in ecotoxicology) and estimation of other quantiles.

data(fluazinam)
log10EC50 <- log10(fluazinam)
fln <- fitdistcens(log10EC50, "norm")

quantile(fln, probs = 0.05)
quantile(fln, probs = c(0.05, 0.1, 0.2))

```

---

fluazinam

*Species-Sensitivity Distribution (SSD) for Fluazinam*


---

## Description

48-hour acute toxicity values (EC50 values) for exposure of macroinvertebrates and zooplankton to fluazinam.

## Usage

```
data(fluazinam)
```

## Format

fluazinam is a data frame with 2 columns named left and right, describing each observed EC50 value (in micrograms per liter) as an interval. The left column contains either NA for left censored observations, the left bound of the interval for interval censored observations, or the observed value for non-censored observations. The right column contains either NA for right censored observations, the right bound of the interval for interval censored observations, or the observed value for noncensored observations.

## Source

Hose, G.C., Van den Brink, P.J. 2004. The species sensitivity distribution approach compared to a microcosm study: A case study with the fungicide fluazinam. *Ecotoxicology and Environmental Safety*, **73**, 109-122.

## Examples

```
# (1) load of data
#
data(fluazinam)

# (2) plot of data using Turnbull cdf plot
#
log10EC50 <- log10(fluazinam)
plotdistcens(log10EC50)

# (3) fit of a lognormal and a logistic distribution to data
# (classical distributions used for species sensitivity
# distributions, SSD, in ecotoxicology)
# and visual comparison of the fits using Turnbull cdf plot
#
f1n <- fitdistcens(log10EC50, "norm")
summary(f1n)

f1l <- fitdistcens(log10EC50, "logis")
summary(f1l)

cdfcompdens(list(f1n, f1l), legendtext=c("normal", "logistic"),
xlab = "log10(EC50)")

# (4) estimation of the 5 percent quantile value of
# the normal fitted distribution (5 percent hazardous concentration : HC5)
# with its two-sided 95 percent confidence interval calculated by
# non parametric bootstrap
# with a small number of iterations to satisfy CRAN running times constraint.
# For practical applications, we recommend to use at least niter=501 or niter=1001.
#
# in log10(EC50)
b1n <- bootdistcens(f1n, niter=101)
HC51n <- quantile(b1n, probs = 0.05)
# in EC50
10^(HC51n$quantiles)
10^(HC51n$quantCI)

# (5) estimation of the HC5 value
# with its one-sided 95 percent confidence interval (type "greater")
#
# in log10(EC50)
HC51nb <- quantile(b1n, probs = 0.05, CI.type="greater")

# in LC50
10^(HC51nb$quantiles)
```

```
10^(HC51nb$quantCI)
```

---

gofstat

*Goodness-of-fit statistics*


---

### Description

Computes goodness-of-fit statistics for parametric distributions fitted to a same non-censored data set.

### Usage

```
gofstat(f, chisqbreaks, meancount, discrete, fitnames=NULL)
```

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

### Arguments

f	An object of class "fitdist", output of the function fitdist, or a list of "fitdist" objects.
chisqbreaks	A numeric vector defining the breaks of the cells used to compute the chi-squared statistic. If omitted, these breaks are automatically computed from the data in order to reach roughly the same number of observations per cell, roughly equal to the argument meancount, or slightly more if there are some ties.
meancount	The mean number of observations per cell expected for the definition of the breaks of the cells used to compute the chi-squared statistic. This argument will not be taken into account if the breaks are directly defined in the argument chisqbreaks. If chisqbreaks and meancount are both omitted, meancount is fixed in order to obtain roughly $(4n)^{2/5}$ cells with $n$ the length of the dataset.
discrete	If TRUE, only the Chi-squared statistic is computed. If missing, discrete is set to TRUE when the distribution name of the first fit belongs to "binom", "nbinom", "geom", "hyper", "pois".
fitnames	A vector defining the names of the fits.
x	An object of class "fitdist".
...	Further arguments to be passed to generic functions.

### Details

Goodness-of-fit statistics are computed. The Chi-squared statistic is computed using cells defined by the argument chisqbreaks or cells automatically defined from data, in order to reach roughly the same number of observations per cell, roughly equal to the argument meancount, or slightly more if there are some ties. The choice to define cells from the empirical distribution (data), and not from the theoretical distribution, was done to enable the comparison of Chi-squared values obtained with

different distributions fitted on a same data set. If `chisqbreaks` and `meancount` are both omitted, `meancount` is fixed in order to obtain roughly  $(4n)^{2/5}$  cells, with  $n$  the length of the data set (Vose, 2000). The Chi-squared statistic is not computed if the program fails to define enough cells due to a too small dataset. When the Chi-squared statistic is computed, and if the degree of freedom (nb of cells - nb of parameters - 1) of the corresponding distribution is strictly positive, the p-value of the Chi-squared test is returned.

For continuous distributions, Kolmogorov-Smirnov, Cramer-von Mises and Anderson-Darling and statistics are also computed, as defined by Stephens (1986).

An approximate Kolmogorov-Smirnov test is performed by assuming the distribution parameters known. The critical value defined by Stephens (1986) for a completely specified distribution is used to reject or not the distribution at the significance level 0.05. Because of this approximation, the result of the test (decision of rejection of the distribution or not) is returned only for data sets with more than 30 observations. Note that this approximate test may be too conservative.

For data sets with more than 5 observations and for distributions for which the test is described by Stephens (1986) for maximum likelihood estimations ("`exp`", "`cauchy`", "`gamma`" and "`weibull`"), the Cramer-von Mises and Anderson-darling tests are performed as described by Stephens (1986). Those tests take into account the fact that the parameters are not known but estimated from the data by maximum likelihood. The result is the decision to reject or not the distribution at the significance level 0.05. Those tests are available only for maximum likelihood estimations.

Only recommended statistics are automatically printed, i.e. Cramer-von Mises, Anderson-Darling and Kolmogorov statistics for continuous distributions and Chi-squared statistics for discrete ones ("`binom`", "`nbinom`", "`geom`", "`hyper`" and "`pois`").

Results of the tests are not printed but stored in the output of the function.

## Value

`gof.stat` returns an object of class "`gofstat.fitdist`" with following components,

<code>chisq</code>	a named vector with the Chi-squared statistics or NULL if not computed
<code>chisqbreaks</code>	common breaks used to define cells in the Chi-squared statistic
<code>chisqpvalue</code>	a named vector with the p-values of the Chi-squared statistic or NULL if not computed
<code>chisqdf</code>	a named vector with the degrees of freedom of the Chi-squared distribution or NULL if not computed
<code>chisqtable</code>	a table with observed and theoretical counts used for the Chi-squared calculations
<code>cvm</code>	a named vector of the Cramer-von Mises statistics or "not computed" if not computed
<code>cvmtest</code>	a named vector of the decisions of the Cramer-von Mises test or "not computed" if not computed
<code>ad</code>	a named vector with the Anderson-Darling statistics or "not computed" if not computed
<code>adtest</code>	a named vector with the decisions of the Anderson-Darling test or "not computed" if not computed

ks	a named vector with the Kolmogorov-Smirnov statistic or "not computed" if not computed
kstest	a named vector with the decisions of the Kolmogorov-Smirnov test or "not computed" if not computed
aic	a named vector with the values of the Aikake's Information Criterion.
bic	a named vector with the values of the Bayesian Information Criterion.
discrete	the input argument or the automatic choice by the function.
nbfit	Number of fits in argument.

### Author(s)

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr> and Christophe Dutang.

### References

- Cullen AC and Frey HC (1999) Probabilistic techniques in exposure assessment. Plenum Press, USA, pp. 81-155.
- Stephens MA (1986) Tests based on edf statistics. In Goodness-of-fit techniques (D'Agostino RB and Stephens MA, eds), Marcel dekker, New York, pp. 97-194.
- Venables WN and Ripley BD (2002) Modern applied statistics with S. Springer, New York, pp. 435-446.
- Vose D (2000) Risk analysis, a quantitative guide. John Wiley & Sons Ltd, Chichester, England, pp. 99-143.

### See Also

[fitdist](#).

### Examples

```
# (1) fit of two distributions to the serving size data
# by maximum likelihood estimation
# and comparison of goodness-of-fit statistics
#

data(groundbeef)
serving <- groundbeef$serving
(fitg <- fitdist(serving, "gamma"))
gofstat(fitg)
(fitln <- fitdist(serving, "lnorm"))
gofstat(fitln)

gofstat(list(fitg, fitln))

# (2) fit of two discrete distributions to toxocara data
```

```

# and comparison of goodness-of-fit statistics
#

data(toxocara)
number <- toxocara$number

fitp <- fitdist(number,"pois")
summary(fitp)
plot(fitp)

fitnb <- fitdist(number,"nbinom")
summary(fitnb)
plot(fitnb)

gofstat(list(fitp, fitnb),fitnames = c("Poisson","negbin"))

# (3) Use of Chi-squared results in addition to
#     recommended statistics for continuous distributions
#

set.seed(1234)
x4 <- rweibull(n=1000,shape=2,scale=1)
# fit of the good distribution
f4 <- fitdist(x4,"weibull")

# fit of a bad distribution
f4b <- fitdist(x4,"cauchy")

gofstat(list(f4,f4b),fitnames=c("Weibull", "Cauchy"))

```

---

graphcomp

*Graphical comparison of multiple fitted distributions (for non-censored continuous data)*


---

## Description

cdfcomp plots the empirical cumulative distribution against fitted distribution functions, denscomp plots the histogram against fitted density functions, qqcomp plots theoretical quantiles against empirical ones, ppcomp plots theoretical probabilities against empirical ones.

## Usage

```

cdfcomp(ft, xlim, ylim, xlogscale = FALSE, ylogscale = FALSE, main, xlab, ylab,
  datapch, datacol, fitlty, fitcol, addlegend = TRUE, legendtext,
  xlegend = "bottomright", ylegend = NULL, horizontals = TRUE,
  verticals = FALSE, use.ppoints = TRUE, a.ppoints = 0.5,
  lines01 = FALSE, discrete, ...)

```



```
denscomp(ft, xlim, ylim, probability = TRUE, main, xlab, ylab, datapch,
  datacol, fitlty, fitcol, addlegend = TRUE, legendtext, xlegend = "topright",
  ylegend = NULL, demp = FALSE, dempcol = "grey", ...)
```

```
qqcomp(ft, xlim, ylim, xlogscale = FALSE, ylogscale = FALSE, main, xlab, ylab,
  fitpch, fitcol, addlegend = TRUE, legendtext, xlegend = "bottomright",
  ylegend = NULL, use.ppoints = TRUE, a.ppoints = 0.5, line01 = TRUE,
  line01col = "black", line01lty = 1, ynoise = TRUE, ...)
```

```
ppcomp(ft, xlim, ylim, xlogscale = FALSE, ylogscale = FALSE, main, xlab, ylab,
  fitpch, fitcol, addlegend = TRUE, legendtext, xlegend = "bottomright",
  ylegend = NULL, use.ppoints = TRUE, a.ppoints = 0.5, line01 = TRUE,
  line01col = "black", line01lty = 1, ynoise = TRUE, ...)
```

### Arguments

ft	One "fitdist" object or a list of objects of class "fitdist".
xlim	The $x$ -limits of the plot.
ylim	The $y$ -limits of the plot.
xlogscale	If TRUE, uses a logarithmic scale for the $x$ -axis.
ylogscale	If TRUE, uses a logarithmic scale for the $y$ -axis.
main	A main title for the plot, see also <a href="#">title</a> .
xlab	A label for the $x$ -axis, defaults to a description of $x$ .
ylab	A label for the $y$ -axis, defaults to a description of $y$ .
datapch	An integer specifying a symbol to be used in plotting data points, see also <a href="#">points</a> .
datacol	A specification of the color to be used in plotting data points.
fitcol	A (vector of) color(s) to plot fitted distributions. If there are fewer colors than fits they are recycled in the standard fashion.
fitlty	A (vector of) line type(s) to plot fitted distributions/densities. If there are fewer colors than fits they are recycled in the standard fashion. See also <a href="#">par</a> .
fitpch	A (vector of) line type(s) to plot fitted quantiles/probabilities. If there are fewer colors than fits they are recycled in the standard fashion.
addlegend	If TRUE, a legend is added to the plot.
legendtext	A character or expression vector of length $\geq 1$ to appear in the legend, see also <a href="#">legend</a> .
xlegend, ylegend	The $x$ and $y$ co-ordinates to be used to position the legend. They can be specified by keyword or in any way which is accepted by 'xy.coords': see <a href="#">legend</a> for details.
horizontals	If TRUE, draws horizontal lines for the step empirical cdf function. See also <a href="#">plot.stepfun</a> .
verticals	If TRUE, draws also vertical lines for the empirical cdf function. Only taken into account if horizontals=TRUE.

<code>use.ppoints</code>	If TRUE, probability points of the empirical distribution are defined as $(1:n - a.ppoints)/(n - 2a.ppoints)$ using function <code>ppoints</code> . If FALSE, probability points are simply defined as $1:n / n$ . This argument is ignored for discrete data.
<code>a.ppoints</code>	If <code>use.ppoints=TRUE</code> , this is passed to function <code>ppoints</code> .
<code>lines01</code>	A logical to plot two horizontal lines at $h=0$ and $h=1$ for <code>cdfcomp</code> .
<code>line01</code>	A logical to plot an horizontal line $y = x$ for <code>qqcomp</code> and <code>ppcomp</code> .
<code>line01col</code> , <code>line01lty</code>	Color and line type for <code>line01</code> .
<code>demp</code>	A logical to add the empirical density on the plot using the <code>density</code> function.
<code>dempcol</code>	A color for the empirical density in case it is added on the plot.
<code>ynoise</code>	A logical to add a small Gaussian noise when plotting empirical quantiles/probabilities for <code>qqcomp</code> and <code>ppcomp</code> .
<code>probability</code>	A logical to use the probability scale for <code>denscomp</code> , see also <code>hist</code> .
<code>...</code>	Further graphical arguments passed to graphical functions used in <code>cdfcomp</code> , <code>denscomp</code> , <code>ppcomp</code> and <code>qqcomp</code> .
<code>discrete</code>	If TRUE, the distribution is considered as discrete. If missing, <code>discrete</code> is set to TRUE when the distribution name of the first fit belongs to "binom", "nbinom", "geom", "hyper", "pois"; otherwise <code>discrete</code> is set to FALSE.

## Details

`cdfcomp` provides a plot of the empirical distribution and each fitted distribution in cdf, by default using the Hazen's rule for the empirical distribution, with probability points defined as  $(1:n - 0.5)/n$ . If `discrete` is TRUE probability points are always defined as  $(1:n)/n$ .

`denscomp` provides a density plot of each fitted distribution with the histogram of the data (data are assumed continuous).

`ppcomp` provides a plot of the probabilities of each fitted distribution (x-axis) against the empirical probabilities (y-axis) by default defined as  $(1:n - 0.5)/n$  (data are assumed continuous).

`qqcomp` provides a plot of the quantiles of each theoretical distribution (x-axis) against the empirical quantiles of the data (y-axis), by default defining probability points as  $(1:n - 0.5)/n$  for theoretical quantile calculation (data are assumed continuous).

By default a legend is added to these plots. Many graphical arguments are optional, dedicated to personalize the plots, and fixed to default values if omitted.

## Author(s)

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr> and Christophe Dutang.

## See Also

See `plot`, `legend`, `ppoints`, `plot.stepfun` for classic plotting functions and `plotdist`.

## Examples

```

# (1) Plot various distributions fitted to serving size data
#
data(groundbeef)
serving <- groundbeef$serving
fitW <- fitdist(serving, "weibull")
fitln <- fitdist(serving, "lnorm")
fitg <- fitdist(serving, "gamma")
cdfcomp(list(fitW, fitln, fitg), horizontals = FALSE)
cdfcomp(list(fitW, fitln, fitg), horizontals = TRUE)
cdfcomp(list(fitW, fitln, fitg), horizontals = TRUE, verticals=TRUE, datacol="purple")
cdfcomp(list(fitW, fitln, fitg), legendtext=c("Weibull", "lognormal", "gamma"),
  main="ground beef fits", xlab="serving sizes (g)",
  ylab="F", xlim = c(0,250), xlegend = "center", lines01 = TRUE)
denscomp(list(fitW, fitln, fitg), legendtext=c("Weibull", "lognormal", "gamma"),
  main="ground beef fits", xlab="serving sizes (g)", xlim = c(0,250),
  xlegend = "topright")
ppcomp(list(fitW, fitln, fitg), legendtext=c("Weibull", "lognormal", "gamma"),
  main="ground beef fits", xlegend = "bottomright", line01 = TRUE)
qqcomp(list(fitW, fitln, fitg), legendtext=c("Weibull", "lognormal", "gamma"),
  main="ground beef fits", xlegend = "bottomright", line01 = TRUE,
  xlim = c(0,300), ylim = c(0,300), fitpch=16)

# (2) Plot lognormal distributions fitted by
# maximum goodness-of-fit estimation
# using various distances (data plotted in log scale)
#
data(endosulfan)
ATV <- subset(endosulfan, group == "NonArthroInvert")$ATV
flnMGEKS <- fitdist(ATV, "lnorm", method="mge", gof="KS")
flnMGEAD <- fitdist(ATV, "lnorm", method="mge", gof="AD")
flnMGEADL <- fitdist(ATV, "lnorm", method="mge", gof="ADL")
flnMGEAD2L <- fitdist(ATV, "lnorm", method="mge", gof="AD2L")
cdfcomp(list(flnMGEKS, flnMGEAD, flnMGEADL, flnMGEAD2L),
  xlogscale=TRUE, main="fits of a lognormal dist. using various GOF dist.",
  legendtext=c("MGE KS", "MGE AD", "MGE ADL", "MGE AD2L"),
  verticals=TRUE, xlim=c(10,100000))
qqcomp(list(flnMGEKS, flnMGEAD, flnMGEADL, flnMGEAD2L),
  main="fits of a lognormal dist. using various GOF dist.",
  legendtext=c("MGE KS", "MGE AD", "MGE ADL", "MGE AD2L"),
  xlogscale = TRUE, ylogscale = TRUE)
ppcomp(list(flnMGEKS, flnMGEAD, flnMGEADL, flnMGEAD2L),
  main="fits of a lognormal dist. using various GOF dist.",
  legendtext=c("MGE KS", "MGE AD", "MGE ADL", "MGE AD2L"))

# (3) Plot normal and logistic distributions fitted by
# maximum likelihood estimation
# using various plotting positions in cdf plots
#
data(endosulfan)
log10ATV <- log10(subset(endosulfan, group == "NonArthroInvert")$ATV)
fln <- fitdist(log10ATV, "norm")

```

```

fll <- fitdist(log10ATV,"logis")
# default plot using Hazen plotting position: (1:n - 0.5)/n
cdfcomp(list(fln,fll),legendtext=c("normal","logistic"),xlab="log10ATV")
# plot using mean plotting position (named also Gumbel plotting position)
# (1:n)/(n + 1)
cdfcomp(list(fln,fll),legendtext=c("normal","logistic"),xlab="log10ATV",
  use.ppoints = TRUE, a.ppoints = 0)
# plot using basic plotting position: (1:n)/n
cdfcomp(list(fln,fll),legendtext=c("normal","logistic"),xlab="log10ATV",
  use.ppoints = FALSE)

# (4) Comparison of fits of two distributions fitted to discrete data
#
data(toxocara)
number <- toxocara$number
fitp <- fitdist(number,"pois")
fitnb <- fitdist(number,"nbinom")
cdfcomp(list(fitp,fitnb),legendtext=c("Poisson","negative binomial"))

```

---

groundbeef

*Ground beef serving size data set*


---

### Description

Serving sizes collected in a French survey, for ground beef patties consumed by children under 5 years old.

### Usage

```
data(groundbeef)
```

### Format

groundbeef is a data frame with 1 column (serving: serving sizes in grams)

### Source

Delignette-Muller, M.L., Cornu, M. 2008. Quantitative risk assessment for *Escherichia coli* O157:H7 in frozen ground beef patties consumed by young children in French households. *International Journal of Food Microbiology*, **128**, 158-164.

### Examples

```

# (1) load of data
#
data(groundbeef)

# (2) description and plot of data

```

```
#
serving <- groundbeef$serving
descdist(serving)
plotdist(serving)

# (3) fit of a Weibull distribution to data
#
fitW <- fitdist(serving,"weibull")
summary(fitW)
plot(fitW)
gofstat(fitW)
```

---

mgedist

*Maximum goodness-of-fit fit of univariate continuous distributions*


---

### Description

Fit of univariate continuous distribution by maximizing goodness-of-fit (or minimizing distance) for non censored data.

### Usage

```
mgedist(data, distr, gof = "CvM", start = NULL, fix.arg = NULL,
        optim.method = "default", lower = -Inf, upper = Inf, custom.optim = NULL, ...)
```

### Arguments

data	A numeric vector for non censored data.
distr	A character string "name" naming a distribution for which the corresponding quantile function <code>qname</code> and the corresponding density distribution <code>dname</code> must be classically defined.
gof	A character string coding for the name of the goodness-of-fit distance used : "CvM" for Cramer-von Mises distance, "KS" for Kolmogorov-Smirnov distance, "AD" for Anderson-Darling distance, "ADR", "ADL", "AD2R", "AD2L" and "AD2" for variants of Anderson-Darling distance described by Luceno (2006).
start	A named list giving the initial values of parameters of the named distribution. This argument may be omitted for some distributions for which reasonable starting values are computed (see details).
fix.arg	An optional named list giving the values of parameters of the named distribution that must kept fixed rather than estimated.
optim.method	"default" or optimization method to pass to <a href="#">optim</a> .
lower	Left bounds on the parameters for the "L-BFGS-B" method (see <a href="#">optim</a> ).
upper	Right bounds on the parameters for the "L-BFGS-B" method (see <a href="#">optim</a> ).
custom.optim	a function carrying the optimization.
...	further arguments passed to the <code>optim</code> or <code>custom.optim</code> function.

## Details

The `mgedist` function numerically maximizes goodness-of-fit, or minimizes a goodness-of-fit distance coded by the argument `gof`. One may use one of the classical distances defined in Stephens (1986), the Cramer-von Mises distance ("`CvM`"), the Kolmogorov-Smirnov distance ("`KS`") or the Anderson-Darling distance ("`AD`") which gives more weight to the tails of the distribution, or one of the variants of this last distance proposed by Luceno (2006). The right-tail AD ("`ADR`") gives more weight only to the right tail, the left-tail AD ("`ADL`") gives more weight only to the left tail. Either of the tails, or both of them, can receive even larger weights by using second order Anderson-Darling Statistics (using "`AD2R`", "`AD2L`" or "`AD2`"). The optimization process is the same as `mledist`, see the 'details' section of `mledist`.

This function is not intended to be called directly but is internally called in `fitdist` and `bootdist`. This function is intended to be used only with continuous distributions.

NB: if your data values are particularly small or large, a scaling may be needed before the optimization process. See example (4).

## Value

`mgedist` returns a list with following components,

<code>estimate</code>	the parameter estimates.
<code>convergence</code>	an integer code for the convergence of <code>optim</code> defined as below or defined by the user in the user-supplied optimization function. $\emptyset$ indicates successful convergence. 1 indicates that the iteration limit of <code>optim</code> has been reached. 10 indicates degeneracy of the Nelder-Mead simplex. 100 indicates that <code>optim</code> encountered an internal error.
<code>value</code>	the value of the statistic distance corresponding to <code>estimate</code> .
<code>hessian</code>	a symmetric matrix computed by <code>optim</code> as an estimate of the Hessian at the solution found or computed in the user-supplied optimization function.
<code>gof</code>	the code of the goodness-of-fit distance maximized.
<code>optim.function</code>	the name of the optimization function used.
<code>loglik</code>	the log-likelihood.
<code>fix.arg</code>	the named list giving the values of parameters of the named distribution that must kept fixed rather than estimated by maximum likelihood or <code>NULL</code> if there are no such parameters.

## Author(s)

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr>.

## References

- Luceno, A. 2006. Fitting the generalized Pareto distribution to data using maximum goodness-of-fit estimators. *Computational Statistics and Data Analysis*, **51**, 904-917.
- Stephens MA (1986) Tests based on edf statistics. In Goodness-of-fit techniques (D'Agostino RB and Stephens MA, eds), Marcel Dekker, New York, pp. 97-194.

**See Also**

[mmedist](#), [mledist](#), [qmedist](#), [fitdist](#) for other estimation methods.

**Examples**

```
# (1) Fit of a Weibull distribution to serving size data by maximum
# goodness-of-fit estimation using all the distances available
#

data(groundbeef)
serving <- groundbeef$serving
mgedist(serving, "weibull", gof="CvM")
mgedist(serving, "weibull", gof="KS")
mgedist(serving, "weibull", gof="AD")
mgedist(serving, "weibull", gof="ADR")
mgedist(serving, "weibull", gof="ADL")
mgedist(serving, "weibull", gof="AD2R")
mgedist(serving, "weibull", gof="AD2L")
mgedist(serving, "weibull", gof="AD2")

# (2) Fit of a uniform distribution using Cramer-von Mises or
# Kolmogorov-Smirnov distance
#

set.seed(1234)
u <- runif(100,min=5,max=10)
mgedist(u,"unif",gof="CvM")
mgedist(u,"unif",gof="KS")

# (3) Fit of a triangular distribution using Cramer-von Mises or
# Kolmogorov-Smirnov distance
#

## Not run:
require(mc2d)
set.seed(1234)
t <- rtriang(100,min=5,mode=6,max=10)
mgedist(t,"triang",start = list(min=4, mode=6,max=9),gof="CvM")
mgedist(t,"triang",start = list(min=4, mode=6,max=9),gof="KS")

## End(Not run)

# (4) scaling problem
# the simulated dataset (below) has particularly small values, hence without scaling (10^0),
# the optimization raises an error. The for loop shows how scaling by 10^i
# for i=1,...,6 makes the fitting procedure work correctly.

set.seed(1234)
x2 <- rnorm(100, 1e-4, 2e-4)
for(i in 6:0)
```

```
cat(i, try(mgedist(x*10^i,"cauchy")$estimate, silent=TRUE), "\n")
```

---

mledist

*Maximum likelihood fit of univariate distributions*


---

## Description

Fit of univariate distributions using maximum likelihood for censored or non censored data.

## Usage

```
mledist(data, distr, start = NULL, fix.arg = NULL, optim.method = "default",
        lower = -Inf, upper = Inf, custom.optim = NULL, ...)
```

## Arguments

data	A numeric vector for non censored data or a dataframe of two columns respectively named <code>left</code> and <code>right</code> , describing each observed value as an interval for censored data. In that case the <code>left</code> column contains either NA for left censored observations, the left bound of the interval for interval censored observations, or the observed value for non-censored observations. The <code>right</code> column contains either NA for right censored observations, the right bound of the interval for interval censored observations, or the observed value for non-censored observations.
distr	A character string "name" naming a distribution for which the corresponding density function <code>dname</code> and the corresponding distribution <code>pname</code> must be classically defined.
start	A named list giving the initial values of parameters of the named distribution. This argument may be omitted for some distributions for which reasonable starting values are computed (see details).
fix.arg	An optional named list giving the values of parameters of the named distribution that must kept fixed rather than estimated by maximum likelihood.
optim.method	"default" (see details) or optimization method to pass to <a href="#">optim</a> .
lower	Left bounds on the parameters for the "L-BFGS-B" method (see <a href="#">optim</a> ).
upper	Right bounds on the parameters for the "L-BFGS-B" method (see <a href="#">optim</a> ).
custom.optim	a function carrying the MLE optimisation (see details).
...	further arguments passed to the <code>optim</code> or <code>custom.optim</code> function.



## Details

When `custom.optim=NULL` (the default), maximum likelihood estimations of the distribution parameters are computed with the R base `optim`. Direct optimization of the log-likelihood is performed (using `optim`) by default with the "Nelder-Mead" method for distributions characterized by more than one parameter and the "BFGS" method for distributions characterized by only one parameter, or with the method specified in the argument "optim.method" if not "default". Box-constrained optimization may be used with the method "L-BFGS-B", using the constraints on parameters specified in arguments `lower` and `upper`. If non-trivial bounds are supplied, this method will be automatically selected, with a warning.

For the following named distributions, reasonable starting values will be computed if `start` is omitted: "norm", "lnorm", "exp" and "pois", "cauchy", "gamma", "logis", "nbinom" (parametrized by `mu` and `size`), "geom", "beta" and "weibull". Note that these starting values may not be good enough if the fit is poor. The function is not able to fit a uniform distribution.

If `custom.optim` is not `NULL`, then the user-supplied function is used instead of the R base `optim`. The `custom.optim` must have (at least) the following arguments `fn` for the function to be optimized, `par` for the initialized parameters. Internally the function to be optimized will also have other arguments, such as `obs` with observations and `ddistname` with distribution name for non censored data (Beware of potential conflicts with optional arguments of `custom.optim`). It is assumed that `custom.optim` should carry out a MINIMIZATION. Finally, it should return at least the following components `par` for the estimate, convergence for the convergence code, `value` for `fn(par)` and `hessian`. See examples in `fitdist` and `fitdistcens`.

This function is not intended to be called directly but is internally called in `fitdist` and `bootdist` when used with the maximum likelihood method and `fitdistcens` and `bootdistcens`.

NB: if your data values are particularly small or large, a scaling may be needed before the optimization process. See example (7).

## Value

`mledist` returns a list with following components,

<code>estimate</code>	the parameter estimates
<code>convergence</code>	an integer code for the convergence of <code>optim</code> defined as below or defined by the user in the user-supplied optimization function. $\emptyset$ indicates successful convergence. 1 indicates that the iteration limit of <code>optim</code> has been reached. 10 indicates degeneracy of the Nealder-Mead simplex. 100 indicates that <code>optim</code> encountered an internal error.
<code>loglik</code>	the log-likelihood
<code>hessian</code>	a symmetric matrix computed by <code>optim</code> as an estimate of the Hessian at the solution found or computed in the user-supplied optimization function. It is used in <code>fitdist</code> to estimate standard errors.
<code>optim.function</code>	the name of the optimization function used for maximum likelihood
<code>fix.arg</code>	the named list giving the values of parameters of the named distribution that must kept fixed rather than estimated by maximum likelihood or <code>NULL</code> if there are no such parameters.

**Author(s)**

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr> and Christophe Dutang.

**References**

Venables W.N. and Ripley B.D. (2002) Modern applied statistics with S. Springer, New York, pp. 435-446.

**See Also**

[mmedist](#), [qmedist](#), [mgedist](#), [fitdist](#), [fitdistcens](#), [optim](#), [bootdistcens](#) and [bootdist](#).

**Examples**

```
# (1) basic fit of a normal distribution with maximum likelihood estimation
#
set.seed(1234)
x1 <- rnorm(n=100)
mledist(x1, "norm")

# (2) defining your own distribution functions, here for the Gumbel distribution
# for other distributions, see the CRAN task view dedicated to probability distributions

dgumbel <- function(x,a,b) 1/b*exp((a-x)/b)*exp(-exp((a-x)/b))
mledist(x1, "gumbel", start=list(a=10,b=5))

# (3) fit of a discrete distribution (Poisson)
#
set.seed(1234)
x2 <- rpois(n=30, lambda = 2)
mledist(x2, "pois")

# (4) fit a finite-support distribution (beta)
#
set.seed(1234)
x3 <- rbeta(n=100, shape1=5, shape2=10)
mledist(x3, "beta")

# (5) fit frequency distributions on USArrests dataset.
#
x4 <- USArrests$Assault
mledist(x4, "pois")
mledist(x4, "nbinom")

# (6) fit a continuous distribution (Gumbel) to censored data.
```

```

#

data(fluazinam)
log10EC50 <-log10(fluazinam)
# definition of the Gumbel distribution
dgumbel <- function(x,a,b) 1/b*exp((a-x)/b)*exp(-exp((a-x)/b))
pgumbel <- function(q,a,b) exp(-exp((a-q)/b))
qgumbel <- function(p,a,b) a-b*log(-log(p))

mledist(log10EC50,"gumbel",start=list(a=0,b=2),optim.method="Nelder-Mead")

# (7) scaling problem
# the simulated dataset (below) has particularly small values, hence without scaling (10^0),
# the optimization raises an error. The for loop shows how scaling by 10^i
# for i=1,...,6 makes the fitting procedure work correctly.

set.seed(1234)
x2 <- rnorm(100, 1e-4, 2e-4)
for(i in 6:0)
  cat(i, try(mledist(x*10^i, "cauchy")$estimate, silent=TRUE), "\n")

```

---

mmedist

*Matching moment fit of univariate distributions*


---

## Description

Fit of univariate distributions by matching moments (raw or centered) for non censored data.

## Usage

```

mmedist(data, distr, order, memp, start = NULL, fix.arg = NULL,
  optim.method = "default", lower = -Inf, upper = Inf, custom.optim = NULL, ...)

```

## Arguments

data	A numeric vector for non censored data.
distr	A character string "name" naming a distribution (see 'details').
order	A numeric vector for the moment order(s). The length of this vector must be equal to the number of parameters to estimate.
memp	A function implementing empirical moments, raw or centered but has to be consistent with distr argument. This function must have two arguments : as a first one the numeric vector of the data and as a second the order of the moment returned by the function.

<code>start</code>	A named list giving the initial values of parameters of the named distribution. This argument may be omitted for some distributions for which reasonable starting values are computed (see details).
<code>fix.arg</code>	An optional named list giving the values of parameters of the named distribution that must be kept fixed rather than estimated.
<code>optim.method</code>	"default" or optimization method to pass to <a href="#">optim</a> .
<code>lower</code>	Left bounds on the parameters for the "L-BFGS-B" method (see <a href="#">optim</a> ).
<code>upper</code>	Right bounds on the parameters for the "L-BFGS-B" method (see <a href="#">optim</a> ).
<code>custom.optim</code>	a function carrying the optimization .
<code>...</code>	further arguments passed to the <code>optim</code> or <code>custom.optim</code> function.

### Details

The argument `distr` can be one of the base R distributions: "norm", "lnorm", "exp" and "pois", "gamma", "logis", "nbinom", "geom", "beta" and "unif". In that case, no other arguments than `data` and `distr` are required, because the estimate is computed by a closed formula. For distributions characterized by one parameter ("geom", "pois" and "exp"), this parameter is simply estimated by matching theoretical and observed means, and for distributions characterized by two parameters, these parameters are estimated by matching theoretical and observed means and variances (Vose, 2000).

The argument `distr` can also be the distribution name as long as a corresponding `mdistr` function exists, e.g. "pareto" if "mpareto" exists. In that case arguments `arguments` order and `memp` have to be supplied in order to carry out the matching numerically, by minimization of the sum of squared differences between observed and theoretical moments. Optionnally other arguments can be supplied to control optimization (see the 'details' section of [mledist](#) for details about arguments for the control of optimization).

This function is not intended to be called directly but is internally called in [fitdist](#) and [bootdist](#) when used with the matching moments method.

### Value

`mmedist` returns a list with following components,

<code>estimate</code>	the parameter estimates.
<code>convergence</code>	(if appropriate) an integer code for the convergence of <code>optim</code> defined as below or defined by the user in the user-supplied optimization function. $\emptyset$ indicates successful convergence. 1 indicates that the iteration limit of <code>optim</code> has been reached. 10 indicates degeneracy of the Nelder-Mead simplex. 100 indicates that <code>optim</code> encountered an internal error.
<code>loglik</code>	the log-likelihood.
<code>hessian</code>	(if appropriate) a symmetric matrix computed by <code>optim</code> as an estimate of the Hessian at the solution found or computed in the user-supplied optimization function.
<code>optim.function</code>	(if appropriate) the name of the optimization function.

memp	(if appropriate) the empirical moment function.
order	the order of the moment(s) matched.
method	either "closed formula" or the name of the optimization method.
fix.arg	the named list giving the values of parameters of the named distribution that must kept fixed rather than estimated by maximum likelihood or NULL if there are no such parameters.

**Author(s)**

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr> and Christophe Dutang.

**References**

Vose D (2000) Risk analysis, a quantitative guide. John Wiley & Sons Ltd, Chichester, England, pp. 99-143.

Evans M, Hastings N and Peacock B (2000) Statistical distributions. John Wiley and Sons Inc.

**See Also**

[mmedist](#), [qmedist](#), [fitdist](#), [fitdistcens](#), [optim](#), [bootdistcens](#) and [bootdist](#).

**Examples**

```
# (1) basic fit of a normal distribution with moment matching estimation
#
set.seed(1234)
x1 <- rnorm(n=100)
mmedist(x1, "norm")

# (2) fit a discrete distribution (Poisson)
#
set.seed(1234)
x2 <- rpois(n=30, lambda = 2)
mmedist(x2, "pois")

# (3) fit a finite-support distribution (beta)
#
set.seed(1234)
x3 <- rbeta(n=100, shape1=5, shape2=10)
mmedist(x3, "beta")

# (4) fit a Pareto distribution
#
```

```
## Not run:
  require(actuar)
  #simulate a sample
  x4 <- rpareto(1000, 6, 2)

  #empirical raw moment
  memp <- function(x, order)
    ifelse(order == 1, mean(x), sum(x^order)/length(x))

  #fit
  mmedist(x4, "pareto", order=c(1, 2), memp="memp",
    start=c(shape=10, scale=10), lower=1, upper=Inf)

## End(Not run)
```

---

plotdist

---

*Plot of empirical and theoretical distributions for non-censored data*


---

### Description

Plots an empirical distribution (non-censored data) with a theoretical one if specified.

### Usage

```
plotdist(data, distr, para, histo = TRUE, breaks = "default", demp = FALSE, discrete, ...)
```

### Arguments

data	A numeric vector.
distr	A character string "name" naming a distribution for which the corresponding density function <code>dname</code> , the corresponding distribution function <code>pname</code> and the corresponding quantile function <code>qname</code> must be defined, or directly the density function. This argument may be omitted only if <code>para</code> is omitted.
para	A named list giving the parameters of the named distribution. This argument may be omitted only if <code>distr</code> is omitted.
histo	A logical to plot the histogram using the <code>hist</code> function.
breaks	If "default" the histogram is plotted with the function <code>hist</code> with its default breaks definition. Else <code>breaks</code> is passed to the function <code>hist</code> . This argument is not taken into account if <code>discrete</code> is TRUE.
demp	A logical to plot the empirical density on the first plot (alone or superimposed on the histogram depending of the value of the argument <code>histo</code> ) using the <code>density</code> function.

discrete	If TRUE, the distribution is considered as discrete. If both <code>distr</code> and <code>discrete</code> are missing, <code>discrete</code> is set to FALSE. If <code>discrete</code> is missing but not <code>distr</code> , <code>discrete</code> is set to TRUE when <code>distr</code> belongs to "binom", "nbinom", "geom", "hyper" or "pois".
...	further graphical arguments passed to graphical functions used in <code>plotdist</code> .

## Details

Empirical and, if specified, theoretical distributions are plotted in density and in cdf. For the plot in density, the user can use the arguments `histo` and `demp` to specify if he wants the histogram using the function `hist`, the density plot using the function `density`, or both (at least one of the two arguments must be put to "TRUE"). For continuous distributions, the function `hist` is used with its default breaks definition if `breaks` is "default" or passing `breaks` as an argument if it differs from "default". For continuous distribution and when a theoretical distribution is specified by both arguments `distname` and `para`, Q-Q plot (plot of the quantiles of the theoretical fitted distribution (x-axis) against the empirical quantiles of the data) and P-P plot (i.e. for each value of the data set, plot of the cumulative density function of the fitted distribution (x-axis) against the empirical cumulative density function (y-axis)) are also given (Cullen and Frey, 1999). The function `ppoints` (with default parameter for argument `a`) is used for the Q-Q plot, to generate the set of probabilities at which to evaluate the inverse distribution. NOTE THAT FROM VERSION 0.4-3, `ppoints` is also used for P-P plot and cdf plot for continuous data. To personalize the four plots proposed for continuous data, for example to change the plotting position, we recommend the use of functions `cdfcomp`, `denscomp`, `qqcomp` and `ppcomp`.

## Author(s)

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr>.

## References

Cullen AC and Frey HC (1999) Probabilistic techniques in exposure assessment. Plenum Press, USA, pp. 81-159.

## See Also

[graphcomp](#), [descdist](#), [hist](#), [plot](#), [plotdistcens](#) and [ppoints](#).

## Examples

```
# (1) Plot of an empirical distribution with changing
# of default line types for CDF and colors
# and optionally adding a density line
#
set.seed(1234)
x1 <- rnorm(n=30)
plotdist(x1)
plotdist(x1,demp = TRUE)
plotdist(x1,histo = FALSE, demp = TRUE)
plotdist(x1, col="blue", type="b", pch=16)
plotdist(x1, type="s")
```

```

# (2) Plot of a discrete distribution against data
#
set.seed(1234)
x2 <- rpois(n=30, lambda = 2)
plotdist(x2, discrete=TRUE)
plotdist(x2, "pois", para=list(lambda = mean(x2)))
plotdist(x2, "pois", para=list(lambda = mean(x2)), lwd="2")

# (3) Plot of a continuous distribution against data
#
xn <- rnorm(n=100, mean=10, sd=5)
plotdist(xn, "norm", para=list(mean=mean(xn), sd=sd(xn)))
plotdist(xn, "norm", para=list(mean=mean(xn), sd=sd(xn)), pch=16)
plotdist(xn, "norm", para=list(mean=mean(xn), sd=sd(xn)), demp = TRUE)
plotdist(xn, "norm", para=list(mean=mean(xn), sd=sd(xn)),
histo = FALSE, demp = TRUE)

# (4) Plot of serving size data
#
data(groundbeef)
plotdist(groundbeef$serving, type="s")

# (5) Plot of numbers of parasites with a Poisson distribution
data(toxocara)
number <- toxocara$number
plotdist(number, discrete = TRUE)
plotdist(number, "pois", para=list(lambda=mean(number)))

```

---

plotdistcens

*Plot of empirical and theoretical distributions for censored data*


---

## Description

Plots an empirical distribution for censored data with a theoretical one if specified.

## Usage

```
plotdistcens(censdata, distr, para, leftNA = -Inf, rightNA = Inf,
  Turnbull = TRUE, Turnbull.confint = FALSE, ...)
```

## Arguments

**censdata** A dataframe of two columns respectively named *left* and *right*, describing each observed value as an interval. The left column contains either NA for left censored observations, the left bound of the interval for interval censored observations, or the observed value for non-censored observations. The right column contains either NA for right censored observations, the right bound of



	the interval for interval censored observations, or the observed value for non-censored observations.
distr	A character string "name" naming a distribution, for which the corresponding density function dname and the corresponding distribution function pname must be defined, or directly the density function.
para	A named list giving the parameters of the named distribution. This argument may be omitted only if distr is omitted.
leftNA	the real value of the left bound of left censored observations : -Inf or a finite value such as 0 for positive data for example.
rightNA	the real value of the right bound of right censored observations : Inf or a finite value such as a realistic maximum value.
Turnbull	if TRUE the Turnbull algorithm is used to estimate the cdf curve of the censored data and previous arguments leftNA and rightNA are not used (see details)
Turnbull.confint	if TRUE confidence intervals will be added to the Turnbull plot.
...	further graphical arguments passed to other methods

### Details

Empirical and, if specified, theoretical distributions are plotted in cdf. If `Turnbull` is TRUE, the EM approach of Turnbull (Turnbull, 1974) is used to compute the overall empirical cdf curve, with confidence intervals if `Turnbull.confint` is TRUE, by calls to functions `survfit` and `plot.survfit` from the survival package.

Else data are reported directly as segments for interval, left and right censored data, and as points for non-censored data. Before plotting, observations are ordered and a rank  $r$  is associated to each of them. Left censored observations are ordered first, by their right bounds. Interval censored and non censored observations are then ordered by their mid-points and, at last, right censored observations are ordered by their left bounds. If `leftNA` (resp. `rightNA`) is finite, left censored (resp. right censored) observations are considered as interval censored observations and ordered by mid-points with non-censored and interval censored data. It is sometimes necessary to fix `rightNA` or `leftNA` to a realistic extreme value, even if not exactly known, to obtain a reasonable global ranking of observations. After ranking, each of the  $n$  observations is plotted as a point (one  $x$ -value) or a segment (an interval of possible  $x$ -values), with an  $y$ -value equal to  $r/n$ ,  $r$  being the rank of each observation in the global ordering previously described. This second method may be interesting but is certainly less rigorous than the Turnbull method that should be preferred.

### Author(s)

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr>.

### References

Turnbull BW (1974) Nonparametric estimation of a survivorship function with doubly censored data. *Journal of American Statistical Association*, **69**, 169-173.

### See Also

[plotdist](#), [survfit.formula](#).

**Examples**

```

# (1) Plot of an empirical censored distribution (censored data) as a CDF
# using the default Turnbull method
#
data(smokedfish)
d1 <- as.data.frame(log10(smokedfish))
plotdistcens(d1)

# (2) Add the CDF of a normal distribution
#
plotdistcens(d1,"norm",para=list(mean=-1.6,sd=1.5))

# (3) Various plots of the same empirical distribution
#
# default Turnbull plot
plotdistcens(d1,Turnbull = TRUE)
# Turnbull plot with confidence intervals
plotdistcens(d1,Turnbull = TRUE,Turnbull.confint = TRUE)
# with intervals and points
plotdistcens(d1,rightNA=3, Turnbull = FALSE)
# with intervals and points
# defining a minimum value for left censored values
plotdistcens(d1,leftNA=-3, Turnbull = FALSE)

```

---

qmedist

*Quantile matching fit of univariate distributions*


---

**Description**

Fit of univariate distribution by matching quantiles for non censored data.

**Usage**

```

qmedist(data, distr, probs, start = NULL, fix.arg = NULL, qtype = 7,
        optim.method = "default", lower = -Inf, upper = Inf, custom.optim = NULL, ...)

```

**Arguments**

data	A numeric vector for non censored data.
distr	A character string "name" naming a distribution for which the corresponding quantile function qname and the corresponding density distribution dname must be classically defined.
probs	A numeric vector of the probabilities for which the quantile matching is done. The length of this vector must be equal to the number of parameters to estimate.

start	A named list giving the initial values of parameters of the named distribution. This argument may be omitted for some distributions for which reasonable starting values are computed (see details).
fix.arg	An optional named list giving the values of parameters of the named distribution that must kept fixed rather than estimated.
qtype	The quantile type used by the R <a href="#">quantile</a> function to compute the empirical quantiles, (default 7 corresponds to the default quantile method in R).
optim.method	"default" or optimization method to pass to <a href="#">optim</a> .
lower	Left bounds on the parameters for the "L-BFGS-B" method (see <a href="#">optim</a> ).
upper	Right bounds on the parameters for the "L-BFGS-B" method (see <a href="#">optim</a> ).
custom.optim	a function carrying the optimization.
...	further arguments passed to the <a href="#">optim</a> or <a href="#">custom.optim</a> function.

### Details

The `qmedist` function carries out the quantile matching numerically, by minimization of the sum of squared differences between observed and theoretical quantiles. The optimization process is the same as [mledist](#), see the 'details' section of [mledist](#).

This function is not intended to be called directly but is internally called in [fitdist](#) and [bootdist](#).

### Value

`qmedist` returns a list with following components,

estimate	the parameter estimates.
convergence	an integer code for the convergence of <code>optim</code> defined as below or defined by the user in the user-supplied optimization function. 0 indicates successful convergence. 1 indicates that the iteration limit of <code>optim</code> has been reached. 10 indicates degeneracy of the Nelder-Mead simplex. 100 indicates that <code>optim</code> encountered an internal error.
value	the value of the statistic distance corresponding to estimate.
hessian	a symmetric matrix computed by <code>optim</code> as an estimate of the Hessian at the solution found or computed in the user-supplied optimization function.
probs	the probability vector on which quantiles are matched.
optim.function	the name of the optimization function used.
loglik	the log-likelihood.
fix.arg	the named list giving the values of parameters of the named distribution that must kept fixed rather than estimated by maximum likelihood or NULL if there are no such parameters.

### Author(s)

Christophe Dutang and Marie Laure Delignette-Muller.

## References

Klugman et al. (2004) *Loss Models: From Data to Decisions*, 2nd edition. Wiley Series in Probability and Statistics, p. 331.

## See Also

[mmedist](#), [mledist](#), [fitdist](#) for other estimation methods and [quantile](#) for empirical quantile estimation in R.

## Examples

```
# (1) basic fit of a normal distribution
#

set.seed(1234)
x1 <- rnorm(n=100)
qmedist(x1, "norm", probs=c(1/3, 2/3))

# (2) defining your own distribution functions, here for the Gumbel
# distribution for other distributions, see the CRAN task view dedicated
# to probability distributions

dgumbel <- function(x, a, b) 1/b*exp((a-x)/b)*exp(-exp((a-x)/b))
qgumbel <- function(p, a, b) a - b*log(-log(p))
qmedist(x1, "gumbel", probs=c(1/3, 2/3), start=list(a=10,b=5))

# (3) fit a discrete distribution (Poisson)
#

set.seed(1234)
x2 <- rpois(n=30,lambda = 2)
qmedist(x2, "pois", probs=1/2)

# (4) fit a finite-support distribution (beta)
#

set.seed(1234)
x3 <- rbeta(n=100,shape1=5, shape2=10)
qmedist(x3, "beta", probs=c(1/3, 2/3))

# (5) fit frequency distributions on USArrests dataset.
#

x4 <- USArrests$Assault
qmedist(x4, "pois", probs=1/2)
qmedist(x4, "nbinom", probs=c(1/3, 2/3))
```

---

 quantiles

*Quantile estimation from a fitted distribution*


---

**Description**

Quantile estimation from a fitted distribution, optionally with confidence intervals calculated from the bootstrap result.

**Usage**

```
## S3 method for class 'fitdist'
quantile(x, probs = seq(0.1, 0.9, by=0.1), ...)
## S3 method for class 'fitdistcens'
quantile(x, probs = seq(0.1, 0.9, by=0.1), ...)
## S3 method for class 'bootdist'
quantile(x, probs = seq(0.1, 0.9, by=0.1), CI.type = "two.sided", CI.level = 0.95, ...)
## S3 method for class 'bootdistcens'
quantile(x, probs = seq(0.1, 0.9, by=0.1), CI.type = "two.sided", CI.level = 0.95, ...)
## S3 method for class 'quantile.fitdist'
print(x, ...)
## S3 method for class 'quantile.fitdistcens'
print(x, ...)
## S3 method for class 'quantile.bootdist'
print(x, ...)
## S3 method for class 'quantile.bootdistcens'
print(x, ...)
```

**Arguments**

x	An object of class "fitdist", "fitdistcens", "bootdist", "bootdistcens" or "quantile.fitdist", "quantile.fitdistcens", "quantile.bootdist", "quantile.bootdistcens" for the print generic function.
probs	A numeric vector of probabilities with values in [0, 1] at which quantiles must be calculated.
CI.type	Type of confidence intervals: either "two.sided" or one-sided intervals ("less" or "greater").
CI.level	The confidence level.
...	Further arguments to be passed to generic functions.

**Details**

Quantiles of the parametric distribution are calculated for each probability specified in probs, using the estimated parameters. When used with an object of class "bootdist" or "bootdistcens", percentile confidence intervals and medians estimates are also calculated from the bootstrap result. If CI.type is two.sided, the CI.level two-sided confidence intervals of quantiles are calculated.

If `CI.type` is `less` or `greater`, the `CI.level` one-sided confidence intervals of quantiles are calculated. The print functions show the estimated quantiles with percentile confidence intervals and median estimates when a bootstrap resampling has been done previously, and the number of bootstrap iterations for which the estimation converges if it is inferior to the whole number of bootstrap iterations.

### Value

`quantile` returns a list with 2 components (the first two described below) when called with an object of class `"fitdist"` or `"fitdistcens"` and 8 components (described below) when called with an object of class `"bootdist"` or `"bootdistcens"` :

<code>quantiles</code>	a dataframe containing the estimated quantiles for each probability value specified in the argument <code>probs</code> (one row, and as many columns as values in <code>probs</code> ).
<code>probs</code>	the numeric vector of probabilities at which quantiles are calculated.
<code>bootquant</code>	A data frame containing the bootstrapped values for each quantile (many rows, as specified in the call to <code>bootdist</code> in the argument <code>niter</code> , and as many columns as values in <code>probs</code> )
<code>quantCI</code>	If <code>CI.type</code> is <code>two.sided</code> , the two bounds of the <code>CI.level</code> percent <code>two.sided</code> confidence interval for each quantile (two rows and as many columns as values in <code>probs</code> ). If <code>CI.type</code> is <code>less</code> , right bound of the <code>CI.level</code> percent <code>one.sided</code> confidence interval for each quantile (one row). If <code>CI.type</code> is <code>greater</code> , left bound of the <code>CI.level</code> percent <code>one.sided</code> confidence interval for each quantile (one row).
<code>quantmedian</code>	Median of bootstrap estimates (per probability).
<code>CI.type</code>	Type of confidence interval: either <code>"two.sided"</code> or one-sided intervals ( <code>"less"</code> or <code>"greater"</code> ).
<code>CI.level</code>	The confidence level.
<code>nbboot</code>	The number of samples drawn by bootstrap.
<code>nbconverg</code>	The number of iterations for which the optimization algorithm converges.

### Author(s)

Marie-Laure Delignette-Muller <marie-laure.delignette-muller@vetagro-sup.fr> and Christophe Dutang.

### See Also

[fitdist](#), [bootdist](#), [fitdistcens](#) and [bootdistcens](#).

### Examples

```
# (1) Fit of a normal distribution on acute toxicity log-transformed values of
# endosulfan for nonarthropod invertebrates, using maximum likelihood estimation
# to estimate what is called a species sensitivity distribution
# (SSD) in ecotoxicology, followed by estimation of the 5, 10 and 20 percent quantile
# values of the fitted distribution, which are called the 5, 10, 20 percent hazardous
# concentrations (HC5, HC10, HC20) in ecotoxicology, followed with calculations of their
```

```

# confidence intervals with various definitions, from a small number of bootstrap
# iterations to satisfy CRAN running times constraint.
# For practical applications, we recommend to use at least niter=501 or niter=1001.
#
data(endosulfan)
ATV <- subset(endosulfan, group == "NonArthroInvert")$ATV
log10ATV <- log10(subset(endosulfan, group == "NonArthroInvert")$ATV)
fln <- fitdist(log10ATV, "norm")
quantile(fln, probs = c(0.05, 0.1, 0.2))
bln <- bootdist(fln, bootmethod="param", niter=101)
quantile(bln, probs = c(0.05, 0.1, 0.2))
quantile(bln, probs = c(0.05, 0.1, 0.2), CI.type = "greater")
quantile(bln, probs = c(0.05, 0.1, 0.2), CI.level = 0.9)

# (2) Draw of 95 percent confidence intervals on quantiles of the
# previously fitted distribution
#
cdfcomp(fln)
q1 <- quantile(bln, probs = seq(0,1,length=101))
points(q1$quantCI[1,],q1$probs,type="l")
points(q1$quantCI[2,],q1$probs,type="l")

# (3) Fit of a distribution on acute salinity log-transformed tolerance
# for riverine macro-invertebrates, using maximum likelihood estimation
# to estimate what is called a species sensitivity distribution
# (SSD) in ecotoxicology, followed by estimation of the 5, 10 and 20 percent quantile
# values of the fitted distribution, which are called the 5, 10, 20 percent hazardous
# concentrations (HC5, HC10, HC20) in ecotoxicology, followed with calculations of
# their confidence intervals with various definitions.
# from a small number of bootstrap iterations to satisfy CRAN running times constraint.
# For practical applications, we recommend to use at least niter=501 or niter=1001.
#
data(salinity)
log10LC50 <- log10(salinity)
flncens <- fitdistcens(log10LC50,"norm")
quantile(flncens, probs = c(0.05, 0.1, 0.2))
blncens <- bootdistcens(flncens, niter = 101)
quantile(blncens, probs = c(0.05, 0.1, 0.2))
quantile(blncens, probs = c(0.05, 0.1, 0.2), CI.type = "greater")
quantile(blncens, probs = c(0.05, 0.1, 0.2), CI.level = 0.9)

```

---

salinity

*Species-Sensitivity Distribution (SSD) for salinity tolerance*


---

### Description

72-hour acute salinity tolerance (LC50 values) of riverine macro-invertebrates.

**Usage**

```
data(salinity)
```

**Format**

salinity is a data frame with 2 columns named left and right, describing each observed LC50 value (in electrical conductivity, millisiemens per centimeter) as an interval. The left column contains either NA for left censored observations, the left bound of the interval for interval censored observations, or the observed value for non-censored observations. The right column contains either NA for right censored observations, the right bound of the interval for interval censored observations, or the observed value for noncensored observations.

**Source**

Kefford, B.J., Nuggeoda, D., Metzeling, L., Fields, E. 2006. Validating species sensitivity distributions using salinity tolerance of riverine macroinvertebrates in the southern Murray-darling Basin (Victoria, Australia). *Canadian Journal of Fisheries and Aquatic Science*, **63**, 1865-1877.

**Examples**

```
# (1) load of data
#
data(salinity)

# (2) plot of data using Turnbull cdf plot
#
log10LC50 <-log10(salinity)
plotdistcens(log10LC50)

# (3) fit of a normal and a logistic distribution to data in log10
# (classical distributions used for species sensitivity
# distributions, SSD, in ecotoxicology))
# and visual comparison of the fits using Turnbull cdf plot
#
f1n <- fitdistcens(log10LC50,"norm")
summary(f1n)

f1l <- fitdistcens(log10LC50,"logis")
summary(f1l)

cdfcompdens(list(f1n,f1l),legendtext=c("normal","logistic"),
  xlab = "log10(LC50)",xlim=c(0.5,2),lines01 = TRUE)

# (4) estimation of the 5 percent quantile value of
# the normal fitted distribution (5 percent hazardous concentration : HC5)
# with its two-sided 95 percent confidence interval calculated by
# non parametric bootstrap
# from a small number of bootstrap iterations to satisfy CRAN running times constraint.
# For practical applications, we recommend to use at least niter=501 or niter=1001.
#
# in log10(LC50)
```



```
b1n <- bootdistcens(fln, niter=101)
HC5ln <- quantile(b1n, probs = 0.05)
# in LC50
10^(HC5ln$quantiles)
10^(HC5ln$quantCI)

# (5) estimation of the HC5 value
# with its one-sided 95 percent confidence interval (type "greater")
#
# in log10(LC50)
HC5lnb <- quantile(b1n, probs = 0.05, CI.type="greater")

# in LC50
10^(HC5lnb$quantiles)
10^(HC5lnb$quantCI)
```

---

smokedfish

*Contamination data of Listeria monocytogenes in smoked fish*

---

## Description

Contamination data of *Listeria monocytogenes* in smoked fish on the Belgian market in the period 2005 to 2007.

## Usage

```
data(smokedfish)
```

## Format

smokedfish is a data frame with 2 columns named left and right, describing each observed value of *Listeria monocytogenes* concentration (in CFU/g) as an interval. The left column contains either NA for left censored observations, the left bound of the interval for interval censored observations, or the observed value for non-censored observations. The right column contains either NA for right censored observations, the right bound of the interval for interval censored observations, or the observed value for non-censored observations.

## Source

Busschaert, P., Geereard, A.H., Uyttendaele, M., Van Impe, J.F., 2010. Estimating distributions out of qualitative and (semi) quantitative microbiological contamination data for use in risk assessment. *International Journal of Food Microbiology*. **138**, 260-269.

**Examples**

```
# (1) load of data
#
data(smokedfish)

# (2) plot of data in CFU/g
#
plotdistcens(smokedfish)

# (3) plot of transformed data in log10[CFU/g]
#
Clog10 <- log10(smokedfish)
plotdistcens(Clog10)

# (4) Fit of a normal distribution to data in log10[CFU/g]
#
fitlog10 <- fitdistcens(Clog10,"norm")
summary(fitlog10)
plot(fitlog10)
```

---

toxocara

*Parasite abundance in insular feral cats*

---

**Description**

Toxocara cati abundance in feral cats living on Kerguelen island.

**Usage**

```
data(toxocara)
```

**Format**

toxocara is a data frame with 1 column (number: number of parasites in digestive tract)

**Source**

Fromont, E., Morvilliers, L., Artois, M., Pontier, D. 2001. Parasite richness and abundance in insular and mainland feral cats. *Parasitology*, **123**, 143-151.

**Examples**

```
# (1) load of data
#
data(toxocara)

# (2) description and plot of data
#
```

```
number <- toxocara$number
descdist(number,discrete=TRUE,boot=1000)
plotdist(number,discrete=TRUE)

# (3) fit of a Poisson distribution to data
#
fitp <- fitdist(number,"pois")
summary(fitp)
plot(fitp)

# (4) fit of a negative binomial distribution to data
#
fitnb <- fitdist(number,"nbinom")
summary(fitnb)
plot(fitnb)
```

# Index

## \*Topic **datasets**

danish, 9  
endosulfan, 13  
fluazinam, 27  
groundbeef, 36  
salinity, 55  
smokedfish, 57  
toxocara, 58

## \*Topic **distribution**

bootdist, 2  
bootdistcens, 5  
cdfcomp, 7  
descdist, 10  
fitdist, 14  
fitdistcens, 23  
gofstat, 29  
graphcomp, 32  
mgedist, 37  
mledist, 40  
mmedist, 43  
plotdist, 46  
plotdistcens, 48  
qmedist, 50  
quantiles, 53

bootdist, 2, 17, 18, 38, 41, 42, 44, 45, 51, 54  
bootdistcens, 5, 41, 42, 45, 54

cdfcomp, 17, 47  
cdfcomp (graphcomp), 32  
cdfcomp, 7, 8

danish, 9  
danishmulti (danish), 9  
danishuni (danish), 9  
denscomp, 17, 47  
denscomp (graphcomp), 32  
density, 34, 46, 47  
descdist, 10, 47  
endosulfan, 13

fitdist, 3, 4, 14, 25, 31, 38, 39, 41, 42, 44,  
45, 51, 52, 54

fitdistcens, 5, 6, 18, 23, 41, 42, 45, 54  
fitdistrplus (fitdist), 14  
fluazinam, 27

gofstat, 18, 29  
graphcomp, 18, 32, 47  
groundbeef, 36

hist, 34, 46, 47

legend, 8, 33, 34

mge (mgedist), 37  
mgedist, 3, 4, 16–18, 37, 42  
mle (mledist), 40  
mledist, 3–6, 16–18, 24, 25, 38, 39, 40, 44,  
51, 52

mme (mmedist), 43  
mmedist, 3, 4, 16–18, 39, 42, 43, 45, 52

optim, 16, 18, 24, 25, 37, 40–42, 44, 45, 51

par, 8, 33  
plot, 3–6, 34, 47  
plot.bootdist (bootdist), 2  
plot.bootdistcens (bootdistcens), 5  
plot.fitdist (fitdist), 14  
plot.fitdistcens (fitdistcens), 23  
plot.stepfun, 33, 34  
plot.survfit, 8, 49  
plotdist, 12, 17, 18, 34, 46, 49  
plotdistcens, 8, 25, 47, 48  
points, 33  
ppcomp, 17, 47  
ppcomp (graphcomp), 32  
ppoints, 34, 47  
print.bootdist (bootdist), 2  
print.bootdistcens (bootdistcens), 5  
print.fitdist (fitdist), 14

print.fitdistcens (fitdistcens), 23  
print.gofstat.fitdist (gofstat), 29  
print.quantile.bootdist (quantiles), 53  
print.quantile.bootdistcens  
    (quantiles), 53  
print.quantile.fitdist (quantiles), 53  
print.quantile.fitdistcens (quantiles),  
    53

qme (qmedist), 50  
qmedist, 3, 4, 16–18, 39, 42, 45, 50  
qqcomp, 17, 47  
qqcomp (graphcomp), 32  
quantile, 51, 52  
quantile.bootdist, 4  
quantile.bootdist (quantiles), 53  
quantile.bootdistcens, 6  
quantile.bootdistcens (quantiles), 53  
quantile.fitdist, 18  
quantile.fitdist (quantiles), 53  
quantile.fitdistcens, 25  
quantile.fitdistcens (quantiles), 53  
quantiles, 53

salinity, 55  
smokedfish, 57  
stripchart, 3–6  
summary.bootdist (bootdist), 2  
summary.bootdistcens (bootdistcens), 5  
summary.fitdist (fitdist), 14  
summary.fitdistcens (fitdistcens), 23  
survfit, 8, 49  
survfit.formula, 8, 49

title, 7, 33  
toxocara, 58