

Package ‘fitdistrplus’

February 9, 2012

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

Version 0.2-2

Date 2011-04-07

Author Marie Laure Delignette-Muller <ml.delignette@vetagro-sup.fr>, Regis Pouillot <rpouillot@yahoo.fr>, Jean-Baptiste Denis <jbdenis@jouy.inra.fr> and Christophe Dutang <christophe.dutang@ensimag.fr>

Maintainer Marie Laure Delignette-Muller <ml.delignette@vetagro-sup.fr>

Depends R (>= 2.9.2)

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).

License GPL (>= 2)

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

Repository CRAN

Repository/R-Forge/Project riskassessment

Repository/R-Forge/Revision 130

Date/Publication 2011-04-27 17:19:07

R topics documented:

bootdist	2
bootdistcens	5
descdist	7
fitdist	10
fitdistcens	16
gofstat	19
groundbeef	22
mgedist	23
mledist	26
mmedist	29
plotdist	32
plotdistcens	33
qmedist	35
smokedfish	37

Index 39

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' result of the function fitdist.
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 non parametric bootstrap (resampling with replacement from the data set). On each bootstrap sample the function `mledist` (or `mmedist`, `qmedist` or `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 4 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 of data.
<code>CI</code>	bootstrap medians and 95 percent confidence percentile intervals of parameters.

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@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](#), and [mgedist](#).

Examples

```
# (1) basic fit of a normal distribution with maximum likelihood estimation
# followed by parametric bootstrap
#
x1<-c(6.4,13.3,4.1,1.3,14.1,10.6,9.9,9.6,15.3,22.1,13.4,
13.2,8.4,6.3,8.9,5.2,10.9,14.4)
f1<-fitdist(x1,"norm",method="mle")
b1<-bootdist(f1)
print(b1)
```

```

plot(b1)
summary(b1)

# (2) non parametric bootstrap
#
b1np<-bootdist(f1,bootmethod="nonparam")
summary(b1np)

# (3) fit of a gamma distribution followed by parametric bootstrap
#
f1b<-fitdist(x1,"gamma",method="mle")
b1b<-bootdist(f1b)
summary(b1b)

# (4) fit of a gamma distribution with control of the optimization
# method, followed by parametric bootstrap
#
f1c <- fitdist(x1,"gamma",optim.method="L-BFGS-B",lower=c(0,0))
b1c <- bootdist(f1c)
summary(b1c)

# (5) estimation of the standard deviation of a normal distribution
# by maximum likelihood with the mean fixed at 10 using the argument fix.arg
# followed by parametric bootstrap
#
f1d <-fitdist(x1,"norm",start=list(sd=5),fix.arg=list(mean=10))
b1d <- bootdist(f1d)
summary(b1d)
plot(b1d)

# (6) fit of a discrete distribution by matching moment estimation
# (using a closed formula) followed by parametric bootstrap
#
x2<-c(rep(4,1),rep(2,3),rep(1,7),rep(0,12))
f2<-fitdist(x2,"pois",method="mme")
b2<-bootdist(f2)
plot(b2,pch=16)
summary(b2)

# (7) 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)
# followed by parametric bootstrap
#
data(groundbeef)
serving <- groundbeef$serving

fWmle <- fitdist(serving,"weibull")
bWmle <- bootdist(fWmle,niter=101)
summary(bWmle)

fWqme <- fitdist(serving,"weibull",method="qme",probs=c(0.25,0.75))
bWqme <- bootdist(fWqme,niter=101)
summary(bWqme)

```

```

# (8) Fit of a Pareto distribution by numerical moment matching estimation
# followed by parametric bootstrap
#
## Not run:
  require(actuar)
  #simulate a sample
  x4 <- rpareto(1000, 6, 2)
  memp <- function(x, order)
    ifelse(order == 1, mean(x), sum(x^order)/length(x))

  f4 <- fitdist(x4, "pareto", "mme", order=1:2,
    start=c(shape=10, scale=10),
    lower=1, memp="memp", upper=50)

  b4 <- bootdist(f4, niter=101)
  summary(b4)

  b4npar <- bootdist(f4, niter=101, bootmethod="nonparam")
  summary(b4npar)

## End(Not run)

# (9) Fit of a uniform distribution using Cramer-von Mises
# followed by parametric bootstrap
#
u <- runif(50,min=5,max=10)

fu <- fitdist(u,"unif",method="mge",gof="CvM")
bu <- bootdist(fu, bootmethod="param")
summary(bu)
plot(bu)

```

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

<code>f</code>	An object of class 'fitdistcens' result of the function <code>fitdistcens</code> .
<code>niter</code>	The number of samples drawn by bootstrap.
<code>x</code>	an object of class 'bootdistcens'.
<code>object</code>	an object of class 'bootdistcens'.
<code>...</code>	further arguments to be passed to generic methods

Details

Samples are drawn by non parametric 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 3 components,

<code>estim</code>	a data frame containing the bootstrapped values of parameters.
<code>converg</code>	a vector containing the codes for convergence obtained when using <code>mledist</code> on each bootstrapped data set.
<code>CI</code>	bootstrap medians and 95 percent confidence percentile intervals of parameters.

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@vetagro-sup.fr>

References

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

See Also

[fitdistcens](#) and [mledist](#).

Examples

```

# (1) Fit of a normal distribution followed by nonparametric bootstrap
#
d1<-data.frame(
left=c(1.73,1.51,0.77,1.96,1.96,-1.4,-1.4,NA,-0.11,0.55,
      0.41,2.56,NA,-0.53,0.63,-1.4,-1.4,-1.4,NA,0.13),
right=c(1.73,1.51,0.77,1.96,1.96,0,-0.7,-1.4,-0.11,0.55,
      0.41,2.56,-1.4,-0.53,0.63,0,-0.7,NA,-1.4,0.13))
f1<-fitdistcens(d1, "norm")
b1<-bootdistcens(f1)
b1
summary(b1)
plot(b1)

# (2) Fit of a gamma distribution followed by nonparametric bootstrap
#
d3<-data.frame(left=10^(d1$left),right=10^(d1$right))
f3 <- fitdistcens(d3,"gamma")
b3 <- bootdistcens(f3,niter=101)
summary(b3)
plot(b3)

# (3) Fit of a gamma distribution followed by nonparametric bootstrap
# with control of the optimization method
#
f3BFGS <- fitdistcens(d3,"gamma",optim.method="L-BFGS-B",lower=c(0,0))
b3BFGS <- bootdistcens(f3BFGS,niter=101)
summary(b3BFGS)
plot(b3BFGS)

# (4) Estimation of the standard deviation of a normal distribution
# by maximum likelihood with the mean fixed at 0.1 using the argument fix.arg
# followed by nonparametric bootstrap
#
f1b <- fitdistcens(d1, "norm", start=list(sd=1.5),fix.arg=list(mean=0.1))
b1b<-bootdistcens(f1b,niter=101)
summary(b1b)
plot(b1b)

```

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="red", boot.col="pink")
```

Arguments

data	A numeric vector.
discrete	If TRUE, the distribution is considered as discrete.
boot	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.
method	"unbiased" for unbiased estimated values of statistics or "sample" for sample values.
graph	If FALSE, the skewness-kurtosis graph is not plotted.
obs.col	Color used for the observed point on the skewness-kurtosis graph.
boot.col	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 (Fisher, 1930) are printed. Be careful, estimations of skewness and kurtosis are unbiased only for normal distributions and estimated values are thus only indicative. 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 a 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 and normal distributions. If discrete is FALSE, these are uniform, normal, logistic, lognormal, beta and gamma distributions.

Value

descdist returns a list with 7 components,

min	the minimum value
max	the maximum value
median	the median value

mean	the mean value
sd	the standard deviation sample or estimated value
skewness	the skewness sample or estimated value
kurtosis	the kurtosis sample or estimated value

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@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.

Fisher RA (1930) The moments of the distribution for normal samples of measures of departures from normality. Proc. R. Soc. London, Series A 130, 16-28.

See Also

[plotdist](#)

Examples

```
# (1) Description of a sample from a normal distribution
# with and without uncertainty on skewness and kurtosis estimated by bootstrap
#
x1 <- rnorm(100)
descdist(x1)
descdist(x1,boot=1000)

# (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=1000,
obs.col="blue",boot.col="orange")

# (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=1000,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=1000)

# (4) Description of serving size data
# with uncertainty on skewness and kurtosis estimated by bootstrap
```

```
#
data(groundbeef)
serving <- groundbeef$serving
descdist(serving, boot=1000)
```

fitdist

Fit of univariate distributions to non-censored data

Description

Fit of univariate distributions to non-censored data by maximum likelihood, quantile matching or moment matching.

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

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.
method	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'.
start	An 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.
fix.arg	An optional named list giving the values of parameters of the named distribution that must kept fixed rather than estimated. The use of this argument is not possible if <code>method="mme"</code> and a closed formula is used.
x	an object of class 'fitdist'.
object	an object of class 'fitdist'.

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 with discrete distributions: "binom", "nbinom", "geom", "hyper" and "pois".
...	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

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 may 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 method used in `optim` may be chosen or another optimization method may be chosen 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 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`.

The plot of an object of class "fitdist" returned by `fitdist` uses the function `plotdist`.

Value

fitdist returns an object of class 'fitdist', a list with 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' and "qme" for 'matching quantile 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 dataset
distname	the name of the distribution
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.
dots	the list of further arguments passed in ...to be used in bootdist in iterative calls to mledist, mmedist, qmedist, mgedist or NULL if no such arguments

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@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, Chischester, England, pp. 99-143.

See Also

[plotdist](#), [optim](#), [mledist](#), [mmedist](#), [qmedist](#), [mgedist](#), [gofstat](#) and [fitdistcens](#).

Examples

```
# (1) basic fit of a normal distribution with maximum likelihood estimation
#
x1 <- c(6.4,13.3,4.1,1.3,14.1,10.6,9.9,9.6,15.3,22.1,13.4,
13.2,8.4,6.3,8.9,5.2,10.9,14.4)
```

```

f1 <- fitdist(x1,"norm")
print(f1)
plot(f1)
summary(f1)
gofstat(f1)

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

f1b <- fitdist(x1,"norm",method="mme")
summary(f1b)

# (3) moment matching estimation (using a closed formula)
# for log normal distribution
#

f1c <- fitdist(x1,"lnorm",method="mme")
summary(f1c)

# (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))

f1c <- fitdist(x1,"gumbel",start=list(a=10,b=5))
print(f1c)
plot(f1c)

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

x2<-c(rep(4,1),rep(2,3),rep(1,7),rep(0,12))
f2<-fitdist(x2,"pois")
plot(f2)
summary(f2)
gofstat(f2)

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

fitdist(x1,"gamma",optim.method="Nelder-Mead")
fitdist(x1,"gamma",optim.method="BFGS")
fitdist(x1,"gamma",optim.method="L-BFGS-B",lower=c(0,0))
fitdist(x1,"gamma",optim.method="SANN")

# (7) custom optimization function
#

#create the sample

```

```

mysample <- rexp(100, 5)
mystart <- 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
x1 <- c(6.4, 13.3, 4.1, 1.3, 14.1, 10.6, 9.9, 9.6, 15.3, 22.1,
       13.4, 13.2, 8.4, 6.3, 8.9, 5.2, 10.9, 14.4)
fit1 <- fitdist(x1, "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(x1, "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 normal distribution
# by maximum likelihood with the mean fixed at 10 using the argument fix.arg
#
fitdist(x1,"norm",start=list(sd=5),fix.arg=list(mean=10))

# (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(10, 10), lower=1, upper=Inf)
summary(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
```

```

fitdist(serving,"weibull",method="mge",gof="CvM")
fitdist(serving,"weibull",method="mge",gof="KS")
fitdist(serving,"weibull",method="mge",gof="AD")
fitdist(serving,"weibull",method="mge",gof="ADR")
fitdist(serving,"weibull",method="mge",gof="ADL")
fitdist(serving,"weibull",method="mge",gof="AD2R")
fitdist(serving,"weibull",method="mge",gof="AD2L")
fitdist(serving,"weibull",method="mge",gof="AD2")

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

u <- runif(50,min=5,max=10)

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)

```

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 *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 <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 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, 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 method used in `optim` may be chosen or another optimization method may be chosen 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`.

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

Value

`fitdistcens` returns an object of class 'fitdistcens', a list with 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 dataset
distname	the name of the distribution

`dots` the list of further arguments passed in ... to be used in `bootdistcens` to control the optimization method used in iterative calls to `mledist` or `NULL` if no such arguments

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@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](#) and [fitdist](#).

Examples

```
# (1) basic fit of a normal distribution on censored data
#

d1<-data.frame(
  left=c(1.73,1.51,0.77,1.96,1.96,-1.4,-1.4,NA,-0.11,0.55,0.41,
        2.56,NA,-0.53,0.63,-1.4,-1.4,-1.4,NA,0.13),
  right=c(1.73,1.51,0.77,1.96,1.96,0,-0.7,-1.4,-0.11,0.55,0.41,
        2.56,-1.4,-0.53,0.63,0,-0.7,NA,-1.4,0.13))
f1n<-fitdistcens(d1, "norm")
f1n
summary(f1n)
plot(f1n,rightNA=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))
pgumbel <- function(q,a,b) exp(-exp((a-q)/b))
qgumbel <- function(p,a,b) a-b*log(-log(p))
f1g<-fitdistcens(d1,"gumbel",start=list(a=0,b=2))
summary(f1g)
plot(f1g,rightNA=3)

# (3) comparison of fits of various distributions
#

d3<-data.frame(left=10^(d1$left),right=10^(d1$right))
f3w<-fitdistcens(d3,"weibull")
summary(f3w)
plot(f3w,leftNA=0)
f3l<-fitdistcens(d3,"lnorm")
```

```

summary(f3l)
plot(f3l,leftNA=0)
f3e<-fitdistcens(d3,"exp")
summary(f3e)
plot(f3e,leftNA=0)

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

fitdistcens(d3,"gamma",optim.method="Nelder-Mead")
fitdistcens(d3,"gamma",optim.method="BFGS")
fitdistcens(d3,"gamma",optim.method="SANN")
fitdistcens(d3,"gamma",optim.method="L-BFGS-B",lower=c(0,0))

# (5) 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(d3, "gamma", custom.optim=mygenoud, nvars=2,
  Domains=cbind(c(0,0), c(10, 10)), boundary.enforcement=1,
  print.level=1, hessian=TRUE)

summary(fit.with.genoud)

## End(Not run)

# (6) estimation of the standard deviation of a normal distribution
# by maximum likelihood with the mean fixed at 0.1 using the argument fix.arg
#
fitdistcens(d1, "norm", start=list(sd=1.5),fix.arg=list(mean=0.1))

# (7) Fit of a lognormal distribution to bacterial contamination data
#
data(smokedfish)
fitsf <- fitdistcens(smokedfish,"norm")
summary(fitsf)
plot(fitsf)

```

Description

Computes goodness-of-fit statistics for a fit of a parametric distribution on non-censored data.

Usage

```
gofstat(f, chisqbreaks, meancount, print.test = FALSE)
```

Arguments

<code>f</code>	An object of class 'fitdist' result of the function <code>fitdist</code> .
<code>chisqbreaks</code>	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 <code>meancount</code> , or slightly more if there are some ties.
<code>meancount</code>	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 <code>chisqbreaks</code> . If <code>chisqbreaks</code> and <code>meancount</code> are both omitted, <code>meancount</code> is fixed in order to obtain roughly $(4n)^{2/5}$ cells with n the length of the dataset.
<code>print.test</code>	If <code>FALSE</code> , the results of the tests are computed but not printed

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 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. 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 dataset. 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 (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 the distributions assumed continuous (all but "binom", "nbinom", "geom", "hyper" and "pois" for R base 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 datasets with more than 30 observations. Note that this approximate test may be too conservative.

For datasets with more than 5 observations and for distributions for which the test is described by Stephens (1986) ("norm", "lnorm", "exp", "cauchy", "gamma", "logis" 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.

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 printed only if `print.test=TRUE`. Even not printed, all the available results may be found in the list returned by the function.

Value

`gof` returns a list with following components,

<code>chisq</code>	the Chi-squared statistic or NULL if not computed
<code>chisqbreaks</code>	breaks used to define cells in the Chi-squared statistic
<code>chisqvalue</code>	p-value of the Chi-squared statistic or NULL if not computed
<code>chisqdf</code>	degree 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>	the Cramer-von Mises statistic or NULL if not computed
<code>cvmtest</code>	the decision of the Cramer-von Mises test or NULL if not computed
<code>ad</code>	the Anderson-Darling statistic or NULL if not computed
<code>adtest</code>	the decision of the Anderson-Darling test or NULL if not computed
<code>ks</code>	the Kolmogorov-Smirnov statistic or NULL if not computed
<code>kstest</code>	the decision of the Kolmogorov-Smirnov test or NULL if not computed

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@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) for a fit of a normal distribution
#

x1 <- c(6.4,13.3,4.1,1.3,14.1,10.6,9.9,9.6,15.3,22.1,13.4,
13.2,8.4,6.3,8.9,5.2,10.9,14.4)
print(f1 <- fitdist(x1,"norm"))
gofstat(f1)
gofstat(f1,print.test=TRUE)

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

x2<-c(rep(4,1),rep(2,3),rep(1,7),rep(0,12))
print(f2<-fitdist(x2,"pois"))
g2 <- gofstat(f2, chisqbreaks=c(0,1),print.test=TRUE)
g2$chisqtable

# (3) comparison of fits of various distributions
#

x3<-rweibull(n=100,shape=2,scale=1)
gofstat(f3a<-fitdist(x3,"weibull"))
gofstat(f3b<-fitdist(x3,"gamma"))
gofstat(f3c<-fitdist(x3,"exp"))

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

x4<-rweibull(n=100,shape=2,scale=1)
f4<-fitdist(x4,"weibull")
g4 <-gofstat(f4,meancount=10)
print(g4)

# (5) estimation of the standard deviation of a normal distribution
# by maximum likelihood with the mean fixed at 10 using the argument fix.arg
#
f1b <- fitdist(x1,"norm",start=list(sd=5),fix.arg=list(mean=10))
gofstat(f1b)
```

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 optim .
lower	Left bounds on the parameters for the "L-BFGS-B" method (see optim).
upper	Right bounds on the parameters for the "L-BFGS-B" method (see optim).
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.

Value

`mgedist` 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.
gof	the code of the goodness-of-fit distance maximized.
optim.function	the name of the optimization function used.
loglik	the log-likelihood.

Author(s)

Marie Laure Delignette-Muller.

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
#

u <- runif(100,min=5,max=10)
mgedist(u,"unif",gof="CvM")
mgedist(u,"unif",gof="KS")
```

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 <code>optim</code> .
lower	Left bounds on the parameters for the "L-BFGS-B" method (see <code>optim</code>).
upper	Right bounds on the parameters for the "L-BFGS-B" method (see <code>optim</code>).
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`.

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. 0 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

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@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](#), [fitdist](#), [fitdistcens](#), [optim](#), [bootdistcens](#) and [bootdist](#).

Examples

```
# (1) basic fit of a normal distribution with maximum likelihood estimation
#
x1<-c(6.4,13.3,4.1,1.3,14.1,10.6,9.9,9.6,15.3,22.1,13.4,
13.2,8.4,6.3,8.9,5.2,10.9,14.4)
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 a discrete distribution (Poisson)
#
x2<-c(rep(4,1),rep(2,3),rep(1,7),rep(0,12))
mledist(x2,"pois")
mledist(x2,"nbinom")

# (4) fit a finite-support distribution (beta)
#
x3<-c(0.80,0.72,0.88,0.84,0.38,0.64,0.69,0.48,0.73,0.58,0.81,
0.83,0.71,0.75,0.59)
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.
#
d1<-data.frame(
left=c(1.73,1.51,0.77,1.96,1.96,-1.4,-1.4,NA,-0.11,0.55,0.41,
2.56,NA,-0.53,0.63,-1.4,-1.4,-1.4,NA,0.13),
```

```

right=c(1.73,1.51,0.77,1.96,1.96,0,-0.7,-1.4,-0.11,0.55,0.41,
        2.56,-1.4,-0.53,0.63,0,-0.7,NA,-1.4,0.13))
mledist(d1,"norm")

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))
mledist(d1,"gumbel",start=list(a=0,b=2),optim.method="Nelder-Mead")

```

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 <code>distr</code> 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.
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 optim .
lower	Left bounds on the parameters for the "L-BFGS-B" method (see optim).
upper	Right bounds on the parameters for the "L-BFGS-B" method (see optim).
custom.optim	a function carrying the optimization .
...	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 may 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 `fitdistr` and `bootdistr` 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. 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.
<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. It is used in <code>fitdistr</code> to estimate standard errors.
<code>optim.function</code>	(if appropriate) the name of the optimization function.
<code>memp</code>	(if appropriate) the empirical moment function.
<code>order</code>	the order of the moment(s) matched.
<code>method</code>	either <code>"closed formula"</code> or the name of the optimization method.

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@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
#
x1<-c(6.4,13.3,4.1,1.3,14.1,10.6,9.9,9.6,15.3,22.1,13.4,
13.2,8.4,6.3,8.9,5.2,10.9,14.4)
mmedist(x1,"norm")

# (2) fit a discrete distribution (Poisson)
#
x2<-c(rep(4,1),rep(2,3),rep(1,7),rep(0,12))
mmedist(x2,"pois")

# (3) fit a finite-support distribution (beta)
#
x3<-c(0.80,0.72,0.88,0.84,0.38,0.64,0.69,0.48,0.73,0.58,0.81,
0.83,0.71,0.75,0.59)
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(10, 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,breaks="default",discrete=FALSE,...)
```

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.
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.
discrete	If TRUE, the distribution is considered as discrete. This argument is not taken into account if <code>distr</code> is defined. In this last case, the distribution is automatically assumed discrete if and only if <code>distr</code> is "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 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` is used for the Q-Q plot, to generate the set of probabilities at which to evaluate the inverse distribution.

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@vetagro-sup.fr>

References

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

See Also

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

Examples

```
# (1) Plot of an empirical distribution with changing of default line types for CDF and colors
#
x1<-c(6.4,13.3,4.1,1.3,14.1,10.6,9.9,9.6,15.3,22.1,13.4,
13.2,8.4,6.3,8.9,5.2,10.9,14.4)
plotdist(x1)
plotdist(x1,col="red",type="b",pch=16)
plotdist(x1,type="s")

# (2) Plot of a discrete distribution against data
#
x2<-c(rep(4,1),rep(2,3),rep(1,7),rep(0,12))
plotdist(x2,discrete=TRUE)
plotdist(x2,"pois",para=list(lambda=mean(x2)))
plotdist(x2,"pois",para=list(lambda=mean(x2)),col="red",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,col="green")

# (4) Plot of serving size data
#
data(groundbeef)
plotdist(groundbeef$serving)
```

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,...)
```

Arguments

<code>censdata</code>	A dataframe of two columns respectively named <code>left</code> and <code>right</code> , describing each observed value as an interval. 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.
<code>distr</code>	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.
<code>para</code>	A named list giving the parameters of the named distribution. This argument may be omitted only if <code>distr</code> is omitted.
<code>leftNA</code>	the real value of the left bound of left censored observations : $-\text{Inf}$ or a finite value such as 0 for positive data for example.
<code>rightNA</code>	the real value of the right bound of right censored observations : Inf or a finite value such as a realistic maximum value.
<code>...</code>	further graphical arguments passed to other methods

Details

Empirical and, if specified, theoretical distributions are plotted in `cdf`. 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.

Author(s)

Marie-Laure Delignette-Muller <ml.delignette@vetagro-sup.fr>

See Also

[plotdist](#).

Examples

```
# (1) Plot of an empirical censored distribution (censored data) as a CDF
#
d1<-data.frame(
```

```

left=c(1.73,1.51,0.77,1.96,1.96,-1.4,-1.4,NA,-0.11,0.55,
       0.41,2.56,NA,-0.53,0.63,-1.4,-1.4,-1.4,NA,0.13),
right=c(1.73,1.51,0.77,1.96,1.96,0,-0.7,-1.4,-0.11,0.55,
       0.41,2.56,-1.4,-0.53,0.63,0,-0.7,NA,-1.4,0.13))
plotdistcens(d1)

# (2) Plot of the same empirical distribution defining a realistic maximum value
# for right censored values
#
plotdistcens(d1,rightNA=3)

# (3) Add the CDF of a normal distribution
#
plotdistcens(d1,"norm",para=list(mean=0.12,sd=1.4),rightNA=3)

# (4) Plot of the CDF of the same dataset after logarithmic transformation
# and add of a lognormal distribution
#
d3<-data.frame(left=10^(d1$left),right=10^(d1$right))
plotdistcens(d3,leftNA=0)
plotdistcens(d3,"lnorm",para=list(meanlog=0.27,sdlog=3.3),leftNA=0)

```

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).

<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>qtype</code>	The quantile type used by the R <code>quantile</code> function to compute the empirical quantiles, (default 7 corresponds to the default quantile method in R).
<code>optim.method</code>	"default" or optimization method to pass to <code>optim</code> .
<code>lower</code>	Left bounds on the parameters for the "L-BFGS-B" method (see <code>optim</code>).
<code>upper</code>	Right bounds on the parameters for the "L-BFGS-B" method (see <code>optim</code>).
<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 `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,

<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. 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.
<code>value</code>	the value of the statistic distance corresponding to estimate.
<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>probs</code>	the probability vector on which quantiles are matched.
<code>optim.function</code>	the name of the optimization function used.
<code>loglik</code>	the log-likelihood.

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
#
x1<-c(6.4,13.3,4.1,1.3,14.1,10.6,9.9,9.6,15.3,22.1,13.4,
13.2,8.4,6.3,8.9,5.2,10.9,14.4)
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)
#
x2<-c(rep(4,1),rep(2,3),rep(1,7),rep(0,12))
qmedist(x2, "pois", probs=1/2)
qmedist(x2, "nbinom", probs=c(1/3, 2/3))

# (4) fit a finite-support distribution (beta)
#
x3<-c(0.80,0.72,0.88,0.84,0.38,0.64,0.69,0.48,0.73,0.58,0.81,
0.83,0.71,0.75,0.59)
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))
```

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 noncensored 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 <- data.frame(left=log10(smokedfish$left),right=log10(smokedfish$right))
plotdistcens(Clog10)

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

Index

*Topic **datasets**

groundbeef, [22](#)
smokedfish, [37](#)

*Topic **distribution**

bootdist, [2](#)
bootdistcens, [5](#)
descdist, [7](#)
fitdist, [10](#)
fitdistcens, [16](#)
gofstat, [20](#)
mgedist, [23](#)
mledist, [26](#)
mmedist, [29](#)
plotdist, [32](#)
plotdistcens, [33](#)
qmedist, [35](#)

bootdist, [2](#), [24](#), [27](#), [28](#), [30](#), [31](#), [36](#)
bootdistcens, [5](#), [27](#), [28](#), [31](#)

descdist, [7](#), [33](#)

fitdist, [3](#), [10](#), [18](#), [21](#), [24](#), [25](#), [27](#), [28](#), [30](#), [31](#),
[36](#), [37](#)

fitdistcens, [6](#), [12](#), [16](#), [27](#), [28](#), [31](#)
fitdistrplus (fitdist), [10](#)

gofstat, [12](#), [19](#)
groundbeef, [22](#)

hist, [32](#), [33](#)

mge (mgedist), [23](#)
mgedist, [3](#), [11](#), [12](#), [23](#)
mle (mledist), [26](#)
mledist, [3](#), [6](#), [11](#), [12](#), [17](#), [18](#), [24](#), [25](#), [26](#), [30](#),
[36](#), [37](#)

mme (mmedist), [29](#)
mmedist, [3](#), [11](#), [12](#), [25](#), [28](#), [29](#), [31](#), [37](#)

optim, [11](#), [12](#), [17](#), [18](#), [24](#), [26–29](#), [31](#), [36](#)

plot, [3](#), [6](#), [33](#)

plot.bootdist (bootdist), [2](#)
plot.bootdistcens (bootdistcens), [5](#)
plot.fitdist (fitdist), [10](#)
plot.fitdistcens (fitdistcens), [16](#)
plotdist, [9](#), [11](#), [12](#), [32](#), [34](#)
plotdistcens, [17](#), [18](#), [33](#), [33](#)
ppoints, [32](#), [33](#)
print.bootdist (bootdist), [2](#)
print.bootdistcens (bootdistcens), [5](#)
print.fitdist (fitdist), [10](#)
print.fitdistcens (fitdistcens), [16](#)

qme (qmedist), [35](#)

qmedist, [3](#), [11](#), [12](#), [25](#), [28](#), [31](#), [35](#)
quantile, [36](#), [37](#)

smokedfish, [37](#)

stripchart, [3](#), [6](#)

summary.bootdist (bootdist), [2](#)
summary.bootdistcens (bootdistcens), [5](#)
summary.fitdist (fitdist), [10](#)
summary.fitdistcens (fitdistcens), [16](#)