Package ‘univariateML’

January 25, 2022

Type Package
Title Maximum Likelihood Estimation for Univariate Densities
Version 1.1.1
Description User-friendly maximum likelihood estimation (Fisher (1921) <doi:10.1098/rsta.1922.0009>) of univariate densities.
License MIT + file LICENSE
Encoding UTF-8
LazyData true
Depends R (>= 2.10)
Imports assertthat, extraDistr, tibble, logitnorm, actuar, nakagami, fGarch
Suggests testthat, knitr, rmarkdown, markdown, copula, dplyr, covr
RoxygenNote 7.1.2
VignetteBuilder knitr
URL https://github.com/JonasMoss/univariateML,
https://jonasmoss.github.io/univariateML/
BugReports https://github.com/JonasMoss/univariateML/issues
NeedsCompilation no
Author Jonas Moss [aut, cre] (<https://orcid.org/0000-0002-6876-6964>),
Thomas Nagler [ctb]
Maintainer Jonas Moss <jonas.gjertsen@gmail.com>
Repository CRAN
Date/Publication 2022-01-25 13:30:02 UTC

R topics documented:

univariateML-package .............................................. 2
abalone .................................................................. 3
bootstrapml .......................................................... 4
### univariateML-package

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>confint.univariateML</td>
<td>6</td>
</tr>
<tr>
<td>egypt</td>
<td>7</td>
</tr>
<tr>
<td>MaximumLikelihoodDistribution</td>
<td>8</td>
</tr>
<tr>
<td>mlbeta</td>
<td>9</td>
</tr>
<tr>
<td>mlbetapr</td>
<td>10</td>
</tr>
<tr>
<td>mlcauchy</td>
<td>11</td>
</tr>
<tr>
<td>mlexp</td>
<td>12</td>
</tr>
<tr>
<td>mlgamma</td>
<td>13</td>
</tr>
<tr>
<td>mlged</td>
<td>14</td>
</tr>
<tr>
<td>mlgumbel</td>
<td>15</td>
</tr>
<tr>
<td>mlinvgamma</td>
<td>16</td>
</tr>
<tr>
<td>mlinvgauss</td>
<td>17</td>
</tr>
<tr>
<td>mlinvweibull</td>
<td>18</td>
</tr>
<tr>
<td>mlkumar</td>
<td>19</td>
</tr>
<tr>
<td>mllaplace</td>
<td>20</td>
</tr>
<tr>
<td>mlgamma</td>
<td>21</td>
</tr>
<tr>
<td>mllogis</td>
<td>22</td>
</tr>
<tr>
<td>mlnorm</td>
<td>23</td>
</tr>
<tr>
<td>mllogis</td>
<td>24</td>
</tr>
<tr>
<td>mllogitnorm</td>
<td>25</td>
</tr>
<tr>
<td>mllomax</td>
<td>26</td>
</tr>
<tr>
<td>mlnaka</td>
<td>28</td>
</tr>
<tr>
<td>mlnorm</td>
<td>29</td>
</tr>
<tr>
<td>mlpareto</td>
<td>30</td>
</tr>
<tr>
<td>mlpower</td>
<td>31</td>
</tr>
<tr>
<td>mlrayleigh</td>
<td>32</td>
</tr>
<tr>
<td>mlsged</td>
<td>33</td>
</tr>
<tr>
<td>mlsnorm</td>
<td>34</td>
</tr>
<tr>
<td>mlsstd</td>
<td>35</td>
</tr>
<tr>
<td>mlsstd</td>
<td>36</td>
</tr>
<tr>
<td>mlunif</td>
<td>37</td>
</tr>
<tr>
<td>mlweibull</td>
<td>38</td>
</tr>
<tr>
<td>ml_input_checker</td>
<td>39</td>
</tr>
<tr>
<td>model_select</td>
<td>39</td>
</tr>
<tr>
<td>plot.univariateML</td>
<td>40</td>
</tr>
<tr>
<td>ProbabilityPlots</td>
<td>41</td>
</tr>
<tr>
<td>univariateML_models</td>
<td>42</td>
</tr>
</tbody>
</table>

### Index

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>univariateML-package</td>
<td>univariateML</td>
</tr>
</tbody>
</table>

### Description

An R-package for fast, easy, and reliable maximum likelihood estimation for a selection of parametric univariate densities.
Details

Data analysis often requires the estimation of univariate densities. Maximum likelihood estimation is sensible for almost every univariate density used in practice. Moreover, the maximum likelihood estimator is typically consistent and efficient.

The purpose of this package is to

• Support maximum likelihood estimation of a large selection of densities.
• Supports plenty of generics such as plot and AIC to aid your data analysis.

Read the vignettes to learn more about univariateML: browseVignettes(package = "univariateML")

Author(s)

Maintainer: Jonas Moss <jonas.gjertsen@gmail.com> (ORCID)

Other contributors:

• Thomas Nagler <mail@tnagler.com> [contributor]

See Also

Useful links:

• https://github.com/JonasMoss/univariateML
• https://jonasmoss.github.io/univariateML/
• Report bugs at https://github.com/JonasMoss/univariateML/issues

abalone

Abalone data

Description

Physical measurements of 4177 abalones, a species of sea snail.

Usage

abalone

Format

A tibble with 4,177 observations and 9 variables:

sex  Sex of the abalone, F is female, M male, and I infant.
length Longest shell measurement.
diameter Diameter perpendicular to length.
height Height with with meat in shell.
whole_weight Grams whole abalone.
**shucked_weight**  Grams weight of meat.

**viscera_weight**  Grams gut weight (after bleeding).

**shell_weight**  Grams after being dried.

**rings**  +1.5 gives the age in years.

**Details**

See the web page [https://archive.ics.uci.edu/ml/datasets/Abalone](https://archive.ics.uci.edu/ml/datasets/Abalone) for more information about the data set.

**Source**


**References**


**Examples**

`abalone`

---

```r
bootstrapml
```

**Parametric Bootstrap on Distributions Fitted with Maximum Likelihood**

**Description**

The parametric bootstrap is a resampling technique using random variates from a known parametric distribution. In this function the distribution of the random variates is completely determined by the `univariateML` object `object`.

**Usage**

```r
bootstrapml(
  object,
  reps = 1000,
  map = identity,
  reducer = stats::quantile,
  ...
)
```
**Arguments**

- **object**
  A univariateML object.

- **reps**
  Positive integer. The number of bootstrap samples.

- **map**
  A function of the parameters of the univariateML object. Defaults to the identity.

- **reducer**
  A reducer function. Defaults to `stats::quantile` with default argument `probs = c(0.025, 0.975)`.

- **...**
  Passed to `reducer`.

**Details**

For each bootstrap iteration a maximum likelihood estimate is calculated using the `ml***` function specified by `object`. The resulting numeric vector is then passed to `map`. The values returned by `map` is collected in an array and the `reducer` is called on each row of the array.

By default the `map` function is the identity and the default `reducer` is the quantile function taking the argument `probs`, which defaults to `c(0.025, 0.975)`. This corresponds to a 95\% basic percentile confidence interval and is also reported by `confint()`.

*Note:* The default confidence intervals are percentile intervals, not empirical intervals. These confidence intervals will in some cases have poor coverage as they are not studentized, see e.g. Carpenter, J., & Bithell, J. (2000).

**Value**

The transposed map-reduced bootstrap samples.

**References**


**See Also**

`confint()` for an application of `bootstrapml`.

**Examples**

```r
set.seed(1)
object <- mlgamma(mtcars$qsec)

## Calculate c(0.025, 0.975) confidence interval for the gamma parameters.
bootstrapml(object)
```

```
# 2.5% 97.5%
# shape 68.62945 160.841557
# rate 3.896915 9.089194
```
## The mean of a gamma distribution is shape/rate. Now we calculate a
## parametric bootstrap confidence interval for the mean with confidence
## limits c(0.05, 0.95)

bootstrapml(object, map = function(x) x[1] / x[2], probs = c(0.05, 0.95))

# 5%   95%
# 17.33962 18.31253

## Print a histogram of the bootstrapped estimates from an exponential.
object <- mlexp(mtcars$qsec)
hist(bootstrapml(object, reducer = identity))

---

### confint.univariateML

**Confidence Intervals for Maximum Likelihood Estimates**

**Description**

Computes a confidence interval for one or more parameters in a `univariateML` object.

**Usage**

```r
## S3 method for class 'univariateML'
confint(object, parm = NULL, level = 0.95, Nreps = 1000, ...)
```

**Arguments**

- `object` An object of class `univariateML`.
- `parm` Vector of strings; the parameters to calculate a confidence interval for. Each parameter must be a member of `names(object)`.
- `level` The confidence level.
- `Nreps` Number of bootstrap iterations. Passed to `bootstrapml()`.
- `...` Additional arguments passed to `bootstrapml()`.

**Details**

`confint.univariateML` is a wrapper for `bootstrapml()` that computes confidence intervals for the main parameters of `object`. The main parameters of `object` are the members of `names(object)`. For instance, the main parameters of an object obtained from `mlnorm` are mean and sd. The confidence intervals are parametric bootstrap percentile intervals with limits $(1\text{-level})/2$ and $1-(1\text{-level})$.

**Value**

A matrix or vector with columns giving lower and upper confidence limits for each parameter in `parm`. 
egypt

See Also

stats::confint() for the generic function and bootstrapml() for the function used to calculate the confidence intervals.

Examples

object <- mlinvgauss(airquality$Wind)
confint(object) # 95% confidence interval for mean and shape
confint(object, "mean") # 95% confidence interval for the mean parameter
# confint(object, "variance") # Fails since 'variance isn't a main parameter.

description

Mortality data from ancient Egypt

Usage

egypt

Format

A tibble with 141 observations and 2 variables:

age  Age at death.
sex  Sex of deceased; 82 males and 49 females.

details

This data was collected by Spiegelberg (1901) and analyzed by Karl Pearson (1902) in the first volume of Biometrika. It was analyzed by Claeskens & Hjort (2008) and the data is based on their transcription.

References


See Also

The source of the data is https://feb.kuleuven.be/public/u043181/modelselection/datasets/egyptlives_data.txt
Examples

egypt

MaximumLikelihoodDistribution

Maximum likelihood estimated distribution

Description

Density, distribution function, quantile function and random generation for a univariate distribution estimated by maximum likelihood.

Usage

dml(x, obj, log = FALSE)
pml(q = q, obj, lower.tail = TRUE, log.p = FALSE)
qml(p = p, obj, lower.tail = TRUE, log.p = FALSE)
rml(n = n, obj)

Arguments

x, q vector of quantiles.
obj an univariateML object.
log, log.p logical; if TRUE, the probabilities p are gives as log(p).
lower.tail logical; if TRUE (default), the probabilities are P[X ≤ x] otherwise, P[X > x]
p vector of probabilities.
n number of observations. If length(n) > 1, the length is taken to be the number required.

Details

dml is the density, pml is the distribution function, qml is the quantile function, and rml is the random variable generator.

These functions work like their counterparts in stats, e.g. Normal. The univariateML object contains both maximum likelihood estimates and the identity of the model these estimates were calculated under. These functions are wrappers around underlying density, distribution, quantile and random generation functions where unknown parameters are filled with the maximum likelihood estimates. See the example.

Value

dml gives the density, pml gives the distribution function, qml gives the quantile function, and rml generates random deviates.
Examples

```r
## Simple example
obj <- mlnorm(airquality$Wind)
dml(0.5, obj) == dnorm(0.5, mean = obj[1], sd = obj[2])

obj <- mlbetapr(airquality$Wind)
# Plot the logarithm of the beta prime distribution.
plot(function(x) dml(x, obj, log = TRUE),
     from = 0, to = 20,
     main = "Logarithm of Density", ylab = NA, lwd = 2)
```

mlbeta

### Beta distribution maximum likelihood estimation

Description

Uses `stats::nlm` to estimate the parameters of the Beta distribution.

Usage

```r
mlbeta(x, na.rm = FALSE, ...)
```

Arguments

- `x`: a (non-empty) numeric vector of data values.
- `na.rm`: logical. Should missing values be removed?
- `...`: `start` contains optional starting parameter values for the minimization, passed to the `stats::nlm` function. `type` specifies whether a dedicated "gradient", "hessian", or "none" should be passed to `stats::nlm`.

Details

For the density function of the Beta distribution see `Beta`.

For `type`, the option none is fastest.

Value

`mlbeta` returns an object of class `univariateML`. This is a named numeric vector with maximum likelihood estimates for `shape1` and `shape2` and the following attributes:

- `model`: The name of the model.
- `density`: The density associated with the estimates.
- `logLik`: The loglikelihood at the maximum.
- `support`: The support of the density.
- `n`: The number of observations.
- `call`: The call as captured by `match.call`
References


See Also

Beta for the Beta density, nlm for the optimizer this function uses.

Examples

AIC(mlbeta(USArrests$Rape / 100))

Description

This function does not estimate the scale parameter for the BetaPrime distribution. Transforms the data and uses stats::nlm to estimate the parameters of the Beta distribution.

Usage

mlbetapr(x, na.rm = FALSE, ...)

Arguments

x
  a (non-empty) numeric vector of data values.
na.rm
  logical. Should missing values be removed?
...
  passed to mlbeta.

Details

For the density function of the Beta prime distribution see BetaPrime.
For type, the option none is fastest.

Value

mlbetapr returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for shape1 and shape2 and the following attributes:

model
  The name of the model.
density
  The density associated with the estimates.
logLik
  The loglikelihood at the maximum.
support
  The support of the density.
n
  The number of observations.
call
  The call as captured by match.call.
References


See Also

BetaPrime for the Beta prime density, nlm for the optimizer this function uses, mlbeta for the Beta distribution maximum likelihood estimator.

Examples

AIC(mlbetapr(USArrests$Rape))

Description

Calculates the estimates using nlm and an exponential transform of the location parameter. If n < 5, an exact solution is reported. In the edge case where no maximum likelihood estimator exists and error is thrown.

Usage

mlcauchy(x, na.rm = FALSE, ...)

Arguments

x a (non-empty) numeric vector of data values.
na.rm logical. Should missing values be removed?
... currently affects nothing.

Details

For the density function of the Cauchy distribution see Cauchy.

Value

mlcauchy returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for location and scale and the following attributes:

model The name of the model.
density The density associated with the estimates.
logLik The loglikelihood at the maximum.
support The support of the density.
n The number of observations.
call The call as captured my match.call
References


See Also

Cauchy for the Cauchy density, nlm for the optimizer this function uses.

Examples

mlcauchy(airquality$Temp)

---

mlexp

Exponential distribution maximum likelihood estimation

Description

The maximum likelihood estimate of rate is the inverse sample mean.

Usage

mlexp(x, na.rm = FALSE, ...)

Arguments

x a (non-empty) numeric vector of data values.
na.rm logical. Should missing values be removed? If FALSE, the function fails when x contains missing values.
... currently affects nothing.

Details

For the density function of the exponential distribution see Exponential.

Value

mlexp returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for rate and the following attributes:

- model The name of the model.
- density The density associated with the estimates.
- logLik The loglikelihood at the maximum.
- support The support of the density.
- n The number of observations.
- call The call as captured by match.call.
mlgamma

References


See Also

Exponential for the exponential density.

Examples

mlexp(precip)

Description

Uses Newton-Raphson to estimate the parameters of the Gamma distribution.

Usage

mlgamma(x, na.rm = FALSE, ...)

Arguments

x

a (non-empty) numeric vector of data values.

na.rm

logical. Should missing values be removed?

... rel.tol is the relative accuracy requested, defaults to .Machine$double.eps^0.25.

iterlim

is a positive integer specifying the maximum number of iterations to
be performed before the program is terminated (defaults to 100).

Details

For the density function of the Gamma distribution see GammaDist.

Value

mlgamma returns an object of class univariateML. This is a named numeric vector with maximum
likelihood estimates for shape and rate and the following attributes:

model

The name of the model.

density

The density associated with the estimates.

logLik

The loglikelihood at the maximum.

support

The support of the density.

n

The number of observations.

call

The call as captured my match.call
References


See Also

GammaDist for the Gamma density.

Examples

mlgamma(precip)

Description

Joint maximum likelihood estimation as implemented by fGarch::gedFit.

Usage

mlged(x, na.rm = FALSE, ...)

Arguments

x a (non-empty) numeric vector of data values.
na.rm logical. Should missing values be removed?
... currently affects nothing.

Details

For the density function of the Student t-distribution see ged.

Value

mlged returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for the parameters mean, sd, nu and the following attributes:

model The name of the model.
density The density associated with the estimates.
logLik The loglikelihood at the maximum.
support The support of the density.
n The number of observations.
call The call as captured my match.call
References
Fernandez C., Steel M.F.J. (2000); On Bayesian Modelling of Fat Tails and Skewness, Preprint.

See Also
ged for the Student t-density.

Examples

mlged(precip)

mlgumbel
Gumbel distribution maximum likelihood estimation

Description
Uses Newton-Raphson to estimate the parameters of the Gumbel distribution.

Usage

mlgumbel(x, na.rm = FALSE, ...)

Arguments

x a (non-empty) numeric vector of data values.
na.rm logical. Should missing values be removed?
... sigma0 is an optional starting value defaulting to 1. rel.tol is the relative accuracy requested, defaults to .Machine$double.eps^0.25. iterlim is a positive integer specifying the maximum number of iterations to be performed before the program is terminated (defaults to 100).

Details
For the density function of the Gumbel distribution see Gumbel.

Value

mlgumbel returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for mu and s and the following attributes:

model The name of the model.
density The density associated with the estimates.
logLik The loglikelihood at the maximum.
support The support of the density.
The number of observations.

The call as captured by `match.call`

shape and `sigma`.

References


See Also

`Gumbel` for the Gumbel density.

Examples

```r
mlgumbel(precip)
```

### Description

Transforms the data and uses Newton-Raphson to estimate the parameters of the Gamma distribution.

### Usage

```r
mlinvgamma(x, na.rm = FALSE, ...)
```

### Arguments

- `x`  
  a (non-empty) numeric vector of data values.
- `na.rm`  
  logical. Should missing values be removed?
- `...`  
  passed to `mlgamma`.

### Details

For the density function of the inverse Gamma distribution see `InvGamma`.

### Value

A named numeric vector with maximum likelihood estimates for `alpha` and `beta`. 
References


See Also

InvGamma for the Inverse Gamma density.

Examples

mlinvgamma(precip)

---

mlinvgauss

*Inverse Gaussian (Wald) maximum likelihood estimation*

Description

The maximum likelihood estimate of mean is the empirical mean and the maximum likelihood estimate of 1/shape is the difference between the mean of reciprocals and the reciprocal of the mean.

Usage

mlinvgauss(x, na.rm = FALSE, ...)

Arguments

- **x**
  - a (non-empty) numeric vector of data values.
- **na.rm**
  - logical. Should missing values be removed?
- **...**
  - currently affects nothing.

Details

For the density function of the Inverse Gamma distribution see InverseGaussian.
Value

`mlinvgauss` returns an object of class `univariateML`. This is a named numeric vector with maximum likelihood estimates for mean and shape and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured by `match.call`

References


See Also

`InverseGaussian` for the Inverse Gaussian density.

Examples

```r
mlinvgauss(precip)
```

---

**Description**

The maximum likelihood estimate of shape and rate are calculated by calling `mlweibull` on the transformed data.

**Usage**

```r
mlinvweibull(x, na.rm = FALSE, ...)
```

**Arguments**

- `x`: a (non-empty) numeric vector of data values.
- `na.rm`: logical. Should missing values be removed?
- `...`: passed to `mlweibull`.

**Details**

For the density function of the log normal distribution see `InverseWeibull`. 
mlkumar

Value

mlinvweibull returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for shape and rate and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured by `match.call`.

References


See Also

InverseWeibull for the Inverse Weibull density.

Examples

```r
mlinvweibull(precip)
```

See Also

InverseWeibull for the Inverse Weibull density.

Description

Uses Newton-Raphson to estimate the parameters of the Kumaraswamy distribution.

Usage

```r
mlkumar(x, na.rm = FALSE, ...)
```

Arguments

- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: `a0` is an optional starting value for the `a` parameter. `rel.tol` is the relative accuracy requested, defaults to `.Machine$double.eps^0.25`. `iterlim` is a positive integer specifying the maximum number of iterations to be performed before the program is terminated (defaults to 100).
Details

For the density function of the Kumaraswamy distribution see Kumaraswamy.

Value

mlkumar returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for a and b and the following attributes:

- model: The name of the model.
- density: The density associated with the estimates.
- logLik: The loglikelihood at the maximum.
- support: The support of the density.
- n: The number of observations.
- call: The call as captured my match.call

References


See Also

Kumaraswamy for the Kumaraswamy density.

Examples

AIC(mlkumar(USArrests$Rape / 100))

mllaplace

Laplace distribution maximum likelihood estimation

Description

The maximum likelihood estimate of mu is the sample median while the maximum likelihood estimate of sigma is mean absolute deviation from the median.

Usage

mllaplace(x, na.rm = FALSE, ...)

Arguments

- x: a (non-empty) numeric vector of data values.
- na.rm: logical. Should missing values be removed?
- ...: currently affects nothing.
Details

For the density function of the Laplace distribution see Laplace.

Value

mllaplace returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for mu and sigma and the following attributes:

- model: The name of the model.
- density: The density associated with the estimates.
- logLik: The loglikelihood at the maximum.
- support: The support of the density.
- n: The number of observations.
- call: The call as captured by match.call

References


See Also

Laplace for the Laplace density.

Examples

mllaplace(precip)

mllgamma

Log-gamma distribution maximum likelihood estimation

Description

The maximum likelihood estimate of shapelog and ratelog are calculated by calling mlgamma() on the transformed data.

Usage

mllgamma(x, na.rm = FALSE, ...)

Arguments

- x: a (non-empty) numeric vector of data values.
- na.rm: logical. Should missing values be removed?
- ...: passed to mlgamma.
Details

For the density function of the log normal distribution see Loggamma.

Value

mllgamma returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for shapelog and ratelog and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured my match.call

References


See Also

Loggamma for the log normal density.

Examples

mllgamma(precip)

---

mlllogis  
*Log-logistic distribution maximum likelihood estimation*

Description

The maximum likelihood estimate of shape and rate are calculated by transforming the data back to the logistic model and applying mlllogis.

Usage

mlllogis(x, na.rm = FALSE, ...)

Arguments

- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: passed to mlllogis.
Details

For the density function of the log-logistic distribution see Loglogistic.

Value

mlllogis returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for shape and rate and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured by match.call.

References


See Also

Loglogistic for the log-logistic density.

Examples

```r
mllnorm(precip)
```

```r
mllnorm
```

---

Description

The maximum likelihood estimate of meanlog is the empirical mean of the log-transformed data and the maximum likelihood estimate of sdlog is the square root of the biased sample variance based on the log-transformed data.

Usage

```r
mllnorm(x, na.rm = FALSE, ...)
```
Arguments

- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: currently affects nothing.

Details

For the density function of the log normal distribution see `Lognormal`.

Value

`mllonorm` returns an object of class `univariateML`. This is a named numeric vector with maximum likelihood estimates for `meanlog` and `sdlog` and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured my `match.call`.

References


See Also

- `Lognormal` for the log normal density.

Examples

```r
mllnorm(precip)
```

---

### mllogis

*Logistic distribution maximum likelihood estimation*

Description

Calculates the estimates using `nlm` with an exponential transform of the location parameter.

Usage

```r
mllogis(x, na.rm = FALSE, ...)
```
**mllogitnorm**

### Arguments
- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: currently affects nothing.

### Details
For the density function of the logistic distribution see **Logistic**.

### Value
*mllogis* returns an object of class `univariateML`. This is a named numeric vector with maximum likelihood estimates for location and scale and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured by `match.call`.

### References

### See Also
- **Logistic** for the Logistic density, *nlm* for the optimizer this function uses.

### Examples
```r
mllogis(precip)
```

### Description
The maximum likelihood estimate of *mu* is the empirical mean of the logit transformed data and the maximum likelihood estimate of *sigma* is the square root of the logit transformed biased sample variance.

### Usage
```r
mllogitnorm(x, na.rm = FALSE, ...)
```
Arguments

- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: currently affects nothing.

Details

For the density function of the logit-normal distribution see `dlogitnorm`.

Value

`mllogitnorm` returns an object of class `univariateML`. This is a named numeric vector with maximum likelihood estimates for mu and sigma and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured my `match.call`.

References


See Also

- `link[dlogitnorm]dlogitnorm` for the normal density.

Examples

```r
AIC(mllogitnorm(USArrests$Rape / 100))
```

---

### mllomax

**Lomax distribution maximum likelihood estimation**

Description

Uses Newton-Raphson to estimate the parameters of the Lomax distribution.

Usage

```r
mllomax(x, na.rm = FALSE, ...)
```
Arguments

- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: lambda0 an optional starting value for the lambda parameter. Defaults to \( \text{median}(x) \).
- **rel.tol**: the relative accuracy requested, defaults to \( \text{.Machine}\$\text{double.eps}^{0.25} \).
- **iterlim**: a positive integer specifying the maximum number of iterations to be performed before the program is terminated (defaults to 100).

Details

For the density function of the Lomax distribution see `Lomax`. The maximum likelihood estimate will frequently fail to exist. This is due to the parameterization of the function which does not take into account that the density converges to an exponential along certain values of the parameters, see vignette("Distribution Details",package = "univariateML").

Value

`mllomax` returns an object of class `univariateML`. This is a named numeric vector with maximum likelihood estimates for lambda and kappa and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured by `match.call`.

References


See Also

`Lomax` for the Lomax density.

Examples

```r
set.seed(3)
mllomax(extraDistr::rlomax(100, 2, 4))
```
mlnaka

Nakagami distribution maximum likelihood estimation

Description

The maximum likelihood estimates of shape and scale are calculated by calling m1gamma on the transformed data.

Usage

mlnaka(x, na.rm = FALSE, ...)

Arguments

  x          a (non-empty) numeric vector of data values.
  na.rm      logical. Should missing values be removed?
  ...        passed to m1gamma.

Details

For the density function of the Nakagami distribution see Nakagami.

Value

m1gamma returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for shape and rate and the following attributes:

  model       The name of the model.
  density     The density associated with the estimates.
  logLik      The loglikelihood at the maximum.
  support     The support of the density.
  n           The number of observations.
  call        The call as captured my match.call

References


See Also

Nakagami for the Nakagami distribution. GammaDist for the closely related Gamma density. See m1gamma for the machinery underlying this function.
Examples

mlgamma(precip)

Description

The maximum likelihood estimate of mean is the empirical mean and the maximum likelihood estimate of sd is the square root of the biased sample variance.

Usage

mlnorm(x, na.rm = FALSE, ...)

Arguments

x
a (non-empty) numeric vector of data values.

na.rm
logical. Should missing values be removed?

... currently affects nothing.

Details

For the density function of the normal distribution see Normal.

Value

mlnorm returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for mean and sd and the following attributes:

model The name of the model.

density The density associated with the estimates.

logLik The loglikelihood at the maximum.

support The support of the density.

n The number of observations.

call The call as captured my match.call

References


See Also

Normal for the normal density.
Examples

mlnorm(precip)

mlpareto

Pareto distribution maximum likelihood estimation

Description

The maximum likelihood estimate of \( b \) is the minimum of \( x \) and the maximum likelihood estimate of \( a \) is \( 1/(\text{mean}(\log(x)) - \log(b)) \).

Usage

mlpareto(x, na.rm = FALSE, ...)

Arguments

- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: currently affects nothing.

Details

For the density function of the Pareto distribution see Pareto.

Value

mlpareto returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for \( a \) and \( b \) and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured by match.call

References


See Also

Pareto for the Pareto density.
Examples

mlpareto(precip)

mlpower

**Power distribution maximum likelihood estimation**

Description

The maximum likelihood estimate of \( \alpha \) is the maximum of \( x + \epsilon \) (see the details) and the maximum likelihood estimate of \( \beta \) is \( 1/(\log(\alpha) - \text{mean}(\log(x))) \).

Usage

mlpower(x, na.rm = FALSE, ...)

Arguments

- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: \( \epsilon \) is a positive number added to \( \max(x) \) as an to the maximum likelihood. Defaults to \( \text{.Machine}\$\text{double.eps}^0.5 \).

Details

For the density function of the power distribution see \texttt{PowerDist}. The maximum likelihood estimator of \( \alpha \) does not exist, strictly speaking. This is because \( x \) is supported \( c(0, \alpha) \) with an open endpoint on \( \alpha \) in the \texttt{extraDistr} implementation of \texttt{dpower}. If the endpoint was closed, \( \max(x) \) would have been the maximum likelihood estimator. To overcome this problem, we add a possibly user specified \( \epsilon \) to \( \max(x) \).

Value

mlpower returns an object of \texttt{class} \texttt{univariateML}. This is a named numeric vector with maximum likelihood estimates for \( \alpha \) and \( \beta \) and the following attributes:

- \texttt{model}:
  - The name of the model.
- \texttt{density}:
  - The density associated with the estimates.
- \texttt{logLik}:
  - The loglikelihood at the maximum.
- \texttt{support}:
  - The support of the density.
- \texttt{n}:
  - The number of observations.
- \texttt{call}:
  - The call as captured \texttt{by match.call}

References

See Also

PowerDist for the power density. Pareto for the closely related Pareto distribution.

Examples

mlpower(precip)

Description

Calculates the \( \sigma \) parameter as the square root of half the empirical second moment.

Usage

mlrayleigh(x, na.rm = FALSE, ...)

Arguments

- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: currently affects nothing.

Details

For the density function of the Rayleigh distribution see Rayleigh.

Value

mlrayleigh returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for \( \sigma \) and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured my match.call

References

mlsged

See Also
        Rayleigh for the Rayleigh density.

Examples
        mlrayleigh(precip)

---

mlsged  Skew Generalized Error distribution maximum likelihood estimation

Description
        Joint maximum likelihood estimation as implemented by fGarch::sgedFit.

Usage
        mlsged(x, na.rm = FALSE, ...)

Arguments
        x  a (non-empty) numeric vector of data values.
        na.rm  logical. Should missing values be removed?
        ...  currently affects nothing.

Details
        For the density function of the Student t-distribution see sged.

Value
        mlsged returns an object of class univariateML. This is a named numeric vector with maximum
        likelihood estimates for the parameters mean, sd, nu, xi, and the following attributes:
        model  The name of the model.
        density  The density associated with the estimates.
        logLik  The loglikelihood at the maximum.
        support  The support of the density.
        n  The number of observations.
        call  The call as captured my match.call

References
        Nelson D.B. (1991); Conditional Heteroscedasticity in Asset Returns: A New Approach, Econo-
        Fernandez C., Steel M.F.J. (2000); On Bayesian Modelling of Fat Tails and Skewness, Preprint.
See Also

sged for the Student t-density.

Examples

mlsged(precip)

mlsnorm

Skew Normal distribution maximum likelihood estimation

Description

Joint maximum likelihood estimation as implemented by fGarch::snormFit.

Usage

mlsnorm(x, na.rm = FALSE, ...)

Arguments

x a (non-empty) numeric vector of data values.
na.rm logical. Should missing values be removed?
... currently affects nothing.

Details

For the density function of the Student t distribution see dsnorm.

Value

mlsnorm returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for the parameters mean, sd, xi and the following attributes:

model The name of the model.
density The density associated with the estimates.
logLik The loglikelihood at the maximum.
support The support of the density.
n The number of observations.
call The call as captured my match.call

References

Fernandez C., Steel M.F.J. (2000); On Bayesian Modelling of Fat Tails and Skewness, Preprint.

See Also

dsnorm for the Student-t density.
**mlsstd**

**Examples**

```r
mlsnorm(precip)
```

---

**mlsstd**  
*Skew Student t-distribution maximum likelihood estimation*

**Description**

Joint maximum likelihood estimation as implemented by `fGarch::sstdFit`.

**Usage**

```r
mlsstd(x, na.rm = FALSE, ...)
```

**Arguments**

- `x`: a (non-empty) numeric vector of data values.
- `na.rm`: logical. Should missing values be removed?
- `...`: currently affects nothing.

**Details**

For the density function of the skew Student t-distribution see `sstd`.

**Value**

`mlsstd` returns an object of class `univariateML`. This is a named numeric vector with maximum likelihood estimates for the parameters `mean`, `sd`, `nu`, `xi` and the following attributes:

- `model`: The name of the model.
- `density`: The density associated with the estimates.
- `logLik`: The loglikelihood at the maximum.
- `support`: The support of the density.
- `n`: The number of observations.
- `call`: The call as captured by `match.call`

**References**

Fernandez C., Steel M.F.J. (2000); On Bayesian Modelling of Fat Tails and Skewness, Preprint.

**See Also**

- `sstd` for the Skew Student t-density.

**Examples**

```r
mlsstd(precip)
```
mlstd

Student-t distribution maximum likelihood estimation

Description

Joint maximum likelihood estimation as implemented by fGarch::stdFit.

Usage

mlstd(x, na.rm = FALSE, ...)

Arguments

x
  a (non-empty) numeric vector of data values.
na.rm
  logical. Should missing values be removed?
...
  currently affects nothing.

Details

For the density function of the Student t-distribution see std.

Value

mlstd returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for the parameters mean, sd, nu and the following attributes:

model
  The name of the model.
density
  The density associated with the estimates.
logLik
  The loglikelihood at the maximum.
support
  The support of the density.
n
  The number of observations.
call
  The call as captured my match.call

References


See Also

std for the Student-t density.

Examples

mlstd(precip)
mlunif

Uniform distribution maximum likelihood estimation

Description

The estimates are \( \min(x) \) and \( \max(x) \).

Usage

\[
\text{mlunif}(x, \text{na.rm} = \text{FALSE}, \ldots)
\]

Arguments

- \( x \): a (non-empty) numeric vector of data values.
- \( \text{na.rm} \): logical. Should missing values be removed?
- \( \ldots \): currently affects nothing.

Details

For the density function of the logistic distribution see \texttt{Uniform}.

Value

\texttt{mlunif} returns an object of \texttt{class} \texttt{univariateML}. This is a named numeric vector with maximum likelihood estimates for \( \min \) and \( \max \) and the following attributes:

- \texttt{model}: The name of the model.
- \texttt{density}: The density associated with the estimates.
- \texttt{logLik}: The loglikelihood at the maximum.
- \texttt{support}: The support of the density.
- \texttt{n}: The number of observations.
- \texttt{call}: The call as captured my \texttt{match.call}.

References


See Also

- \texttt{Uniform} for the uniform density.

Examples

\[
\text{mlunif}(\text{precip})
\]
Description

Uses Newton-Raphson to estimate the parameters of the Weibull distribution.

Usage

```r
mlweibull(x, na.rm = FALSE, ...)
```

Arguments

- **x**: a (non-empty) numeric vector of data values.
- **na.rm**: logical. Should missing values be removed?
- **...**: `shape0` is an optional starting value for the `shape` parameter. `rel.tol` is the relative accuracy requested, defaults to `Machine$double.eps^0.25`. `iterlim` is a positive integer specifying the maximum number of iterations to be performed before the program is terminated (defaults to 100).

Details

For the density function of the Weibull distribution see `Weibull`.

Value

`mlweibull` returns an object of class `univariateMl`. This is a named numeric vector with maximum likelihood estimates for `shape` and `scale` and the following attributes:

- **model**: The name of the model.
- **density**: The density associated with the estimates.
- **logLik**: The loglikelihood at the maximum.
- **support**: The support of the density.
- **n**: The number of observations.
- **call**: The call as captured by `match.call`.

References


See Also

- `Weibull` for the Weibull density.

Examples

```r
BIC(mlweibull(precip))
```
**ml_input_checker**

*Input Checker for ML functions*

**Description**

Checks that \( x \) in the ML functions is numeric and has only one dimension.

**Usage**

```r
ml_input_checker(x)
```

**Arguments**

- `x` input to a ML*** function.

**Value**

NULL

---

**model_select**

*Fit multiple models and select the best fit*

**Description**

Selects the best model by log-likelihood, AIC, or BIC.

**Usage**

```r
model_select(
  x,
  models = univariateML_models,
  criterion = c("aic", "bic", "loglik"),
  na.rm = FALSE,
  ...
)
```

**Arguments**

- `x` a (non-empty) numeric vector of data values.
- `models` a character vector containing the distribution models to select from; see `print(univariateML_models)`.
- `criterion` the model selection criterion. Must be one of "aic", "bic", and "loglik".
- `na.rm` logical. Should missing values be removed?
- `...` unused.
Value

model_select returns an object of class univariateML. This is a named numeric vector with maximum likelihood estimates for the parameters of the best fitting model and the following attributes:

- `model` The name of the model.
- `density` The density associated with the estimates.
- `logLik` The loglikelihood at the maximum.
- `support` The support of the density.
- `n` The number of observations.
- `call` The call as captured by `match.call`.

See Also


Examples

model_select(precip)

plot.univariateML

Plot, Lines and Points Methods for Maximum Likelihood Estimates

Description

The plot, lines, and points methods for univariateML objects.

Usage

```r
## S3 method for class 'univariateML'
plot(x, range = NULL, ...)  
## S3 method for class 'univariateML'
lines(x, range = NULL, ...)  
## S3 method for class 'univariateML'
points(x, range = NULL, ...)  
```

Arguments

- `x` a univariateML object.
- `range` range of x values to plot, i.e. c(lower, upper).
- `...` parameters passed to plot, lines, or points.
Value

An invisible copy of x.

Examples

```r
plot(mlweibull(datasets::precip), main = "Annual Precipitation in US Cities")
lines(mlgamma(datasets::precip), lty = 2)
rug(datasets::precip)
```

Description

Make quantile-quantile plots and probability-probability plots using maximum likelihood estimation.

Usage

```r
ppmlplot(y, obj, plot.it = TRUE, datax = FALSE, ...) 
ppmlline(...) 
ppmlpoints(y, obj, plot.it = TRUE, datax = TRUE, ...) 
qqmlplot(y, obj, plot.it = TRUE, datax = FALSE, ...) 
qqmlline(y, obj, datax = FALSE, probs = c(0.25, 0.75), qtype = 7, ...) 
qqmlpoints(y, obj, plot.it = TRUE, datax = TRUE, ...) 
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Numeric vector; The data to plot on the y axis when datax is FALSE.</td>
</tr>
<tr>
<td>obj</td>
<td>Either an univariateML object or a function that returns a univariateML object when called with y as its only argument.</td>
</tr>
<tr>
<td>plot.it</td>
<td>Logical; should the result be plotted?</td>
</tr>
<tr>
<td>datax</td>
<td>Logical; should y be plotted on the x-axis? Defaults to FALSE in qqmlplot and ppmlplot but TRUE in qqmlpoints and ppmlpoints.</td>
</tr>
<tr>
<td>...</td>
<td>Graphical parameters.</td>
</tr>
<tr>
<td>probs</td>
<td>Numeric vector of length two, representing probabilities. Corresponding quantile pairs define the line drawn.</td>
</tr>
<tr>
<td>qtype</td>
<td>The type of quantile computation used in quantile.</td>
</tr>
</tbody>
</table>
Details

qqmlplot produces a quantile-quantile plot (Q-Q plot) of the values in \( y \) with respect to the distribution defined by \( obj \), which is either a `univariateML` object or a function returning a `univariateML` object when called with \( y \). `qqmlline` adds a line to a “theoretical” quantile-quantile plot which passes through the `probs` quantiles, by default the first and third quartiles. `qqmlpoints` behaves like `stats::points` and adds a Q-Q plot to an existing plot.

`ppmlplot`, `ppmlline`, and `ppmlpoints` produce probability-probability plots (or P-P plots). They behave similarly to the quantile-quantile plot functions.

This function is modeled after `qqnorm`.

Graphical parameters may be given as arguments to all the functions below.

Value

For `qqmlplot`, `qqmlpoints`, `ppmlplot`, and `ppmlpoints`, a list with components `x` (plotted on the x axis) and `y` (plotted on the y axis). `qqmlline` and `ppmlline` returns nothing.

References


Examples

```r
## Make a single probability plot with a line.
obj <- mlgamma(Nile)
qqmlplot(Nile, obj)
qqmlline(Nile, obj)

## Make multiple probability plots. datax = TRUE must be used to make this
## look good.
ppmlplot(airquality$Wind, mlgamma, main = "Many P-P plots")
ppmlpoints(airquality$Wind, mlexp, col = "red")
ppmlpoints(airquality$Wind, mlweibull, col = "purple")
ppmlpoints(airquality$Wind, mlnorm, col = "blue")
```

Description

Implemented models

Usage

`univariateML`
univariateML_models

Format

An object of class character of length 29.

Examples

print(univariateML_models)
Index

* datasets
  abalone, 3
  egypt, 7
  univariateML_models, 42

abalone, 3

Beta, 9, 10
BetaPrime, 10, 11
bootstrapml, 4
bootstrapml(), 6, 7

Cauchy, 11, 12
class, 9–15, 18–38, 40
confint(), 5
confint.univariateML, 6

dlogitnorm, 26
dml (MaximumLikelihoodDistribution), 8
dsnorm, 34

egypt, 7
Exponential, 12, 13

fGarch::gedFit, 14
fGarch::gedFit, 15
fGarch::snormFit, 34
fGarch::snormFit, 35
fGarch::stdFit, 36

GammaDist, 13, 14, 28
ged, 14, 15
Gumbel, 15, 16

InverseGaussian, 17, 18
InverseWeibull, 18, 19
InvGamma, 16, 17

Kumaraswamy, 20

Laplace, 21

lines.univariateML (plot.univariateML), 40
Loggamma, 22
Logistic, 25
Loglogistic, 23
Lognormal, 24
Lomax, 27

MaximumLikelihoodDistribution, 8
ml_input_checker, 39
mlbeta, 9, 10, 11
mlbetapr, 10
mlcauchy, 11
mlexp, 12
mlgamma, 13, 16, 21, 28
mlgamma(), 21
mlged, 14
mlgumbel, 15
mlinvgamma, 16
mlinvgauss, 17
mlinvweibull, 18
mlkumar, 19
mllaplace, 20
mllogis, 21
mllogis, 22
mllogitnorm, 25
mllognorm, 26
mlnaka, 28
mlnorm, 29
mlpareto, 30
mlpower, 31
mlrayleigh, 32
mlsged, 33
mlsnorm, 34
mlsstd, 35
mlstd, 36
mlunif, 37
mlweibull, 18, 38
model_select, 39
Nakagami, 28
nlm, 10–12, 25
Normal, 8, 29
Pareto, 30, 32
plot.univariateML, 40
pml (MaximumLikelihoodDistribution), 8
points.univariateML
  (plot.univariateML), 40
PowerDist, 31, 32
ppmlline (ProbabilityPlots), 41
ppmlplot (ProbabilityPlots), 41
ppmlpoints (ProbabilityPlots), 41
ProbabilityPlots, 41
qml (MaximumLikelihoodDistribution), 8
qqmlline (ProbabilityPlots), 41
qqmlplot (ProbabilityPlots), 41
qqmlpoints (ProbabilityPlots), 41
qqnorm, 42
Rayleigh, 32, 33
rml (MaximumLikelihoodDistribution), 8
sged, 33, 34
sstd, 35
stats::confint(), 7
std, 36
tibble, 3, 7
Uniform, 37
univariateML (univariateML-package), 2
univariateML-package, 2
univariateML_models, 42
Weibull, 38