Package ‘modeest’

October 13, 2022

Type Package
Title Mode Estimation
Version 2.4.0
Description Provides estimators of the mode of univariate
data or univariate distributions.
License GPL-3
LazyData TRUE
Depends R (>= 3.2)
Imports fBasics, stable, stabledist, stats, statip (>= 0.2.3)
Suggests evd, knitr, mvtnorm, testthat, VGAM
URL https://github.com/paulponcet/modeest
BugReports https://github.com/paulponcet/modeest/issues
RoxygenNote 7.0.0
NeedsCompilation no
Author Paul Poncet [aut, cre]
Maintainer Paul Poncet <paulponcet@yahoo.fr>
Repository CRAN
Date/Publication 2019-11-18 15:30:05 UTC

R topics documented:

asselin ................................................................. 2
distrMode ............................................................ 3
grenander .............................................................. 8
hrm ................................................................. 10
hsm ................................................................. 11
lientz .............................................................. 13
meanshift ........................................................... 15
mlv ................................................................. 16
modeest ............................................................. 18
The Asselin de Beauville mode estimator

Description

This mode estimator is based on the algorithm described in Asselin de Beauville (1978).

Usage

asselin(x, bw = NULL, ...)

Arguments

x

numeric. Vector of observations.

bw

numeric. A number in (0, 1]. If bw = 1, the selected 'modal chain' may be too long.

... further arguments to be passed to the quantile function.

Value

A numeric value is returned, the mode estimate.

Note

The user may call asselin through mlv(x, method = "asselin", ...).

References


See Also

mlv for general mode estimation.
Examples

```r
ox <- rbeta(1000, shape1 = 2, shape2 = 5)

## True mode:
betaMode(shape1 = 2, shape2 = 5)

## Estimation:
asselin(x, bw = 1)
asselin(x, bw = 1/2)
mlv(x, method = "asselin")
```

---

distrMode  

*Mode of some continuous and discrete distributions*

Description

These functions return the mode of the main probability distributions implemented in R.

Usage

distrMode(x, ...)

betaMode(shape1, shape2, ncp = 0)

cauhyMode(location = 0, ...)

chisqMode(df, ncp = 0)

dagumMode(scale = 1, shape1.a, shape2.p)

expMode(...)

fMode(df1, df2)

fiskMode(scale = 1, shape1.a)

frechetMode(location = 0, scale = 1, shape = 1, ...)

gammaMode(shape, rate = 1, scale = 1/rate)

normMode(mean = 0, ...)

gemMode(location = 0, scale = 1, shape = 0, ...)

ghMode(alpha = 1, beta = 0, delta = 1, mu = 0, lambda = -1/2)
ghtMode(beta = 0.1, delta = 1, mu = 0, nu = 10)
gldMode(lambda1 = 0, lambda2 = -1, lambda3 = -1/8, lambda4 = -1/8)
gompertzMode(scale = 1, shape)
gpdMode(location = 0, scale = 1, shape = 0)
gumbelMode(location = 0, ...)
hypMode(alpha = 1, beta = 0, delta = 1, mu = 0, pm = c(1, 2, 3, 4))
koenkerMode(location = 0, ...)
kumarMode(shape1, shape2)
laplaceMode(location = 0, ...)
logisMode(location = 0, ...)
lnormMode(meanlog = 0, sdlog = 1)
lomaxMode(...)
maxwellMode(rate)
mvnormMode(mean, ...)
nakaMode(scale = 1, shape)
nigMode(alpha = 1, beta = 0, delta = 1, mu = 0)
paralogisticMode(scale = 1, shape1.a)
paretoMode(scale = 1, ...)
rayleighMode(scale = 1)
stableMode(alpha, beta, gamma = 1, delta = 0, pm = 0, ...)
stableMode2(loc, disp, skew, tail)
tMode(df, ncp)
umifMode(min = 0, max = 1)
weibullMode(shape, scale = 1)
yulesMode(...)  
bernMode(prob)  
binomMode(size, prob)  
geomMode(...)  
hyperMode(m, n, k, ...)  
nbinomMode(size, prob, mu)  
poisMode(lambda)

**Arguments**

- **x** character. The name of the distribution to consider.
- **...** Additional parameters.
- **shape1** non-negative parameters of the Beta distribution.
- **shape2** non-negative parameters of the Beta distribution.
- **ncp** non-centrality parameter.
- **location** location and scale parameters.
- **df** degrees of freedom (non-negative, but can be non-integer).
- **scale** location and scale parameters.
- **shape1.a** shape parameters.
- **shape2.p** shape parameters.
- **df1** degrees of freedom. Inf is allowed.
- **df2** degrees of freedom. Inf is allowed.
- **shape** the location parameter $\alpha$, scale parameter $\beta$, and shape parameter $\gamma$.
- **rate** vector of rates.
- **mean** vector of means.
- **alpha** shape parameter $\alpha$; skewness parameter $\beta$, $\text{abs}(\beta)$ is in the range $(0, \alpha)$; scale parameter $\delta$, $\delta$ must be zero or positive; location parameter $\mu$, by default 0. These is the meaning of the parameters in the first parameterization $\text{pm}=1$ which is the default parameterization selection. In the second parameterization, $\text{pm}=2$ $\alpha$ and $\beta$ take the meaning of the shape parameters (usually named) $\zeta$ and $\rho$. In the third parameterization, $\text{pm}=3$ $\alpha$ and $\beta$ take the meaning of the shape parameters (usually named) $\xi$ and $\chi$. In the fourth parameterization, $\text{pm}=4$ $\alpha$ and $\beta$ take the meaning of the shape parameters (usually named) $\bar{a}$ and $\bar{b}$.
- **beta** shape parameter $\alpha$; skewness parameter $\beta$, $\text{abs}(\beta)$ is in the range $(0, \alpha)$; scale parameter $\delta$, $\delta$ must be zero or positive; location parameter $\mu$, by default 0. These is the meaning of the parameters in the first parameterization $\text{pm}=1$ which is the default parameterization selection. In the
second parameterization, \( pm=2 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \zeta \) and \( \rho \). In the third parameterization, \( pm=3 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \xi \) and \( \chi \). In the fourth parameterization, \( pm=4 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \bar{a} \) and \( \bar{b} \).

\section*{delta}
shape parameter \( \alpha \); skewness parameter \( \beta \), \( \text{abs}(\beta) \) is in the range \((0, \alpha)\); scale parameter \( \delta \); \( \delta \) must be zero or positive; location parameter \( \mu \), by default 0. These are the meaning of the parameters in the first parameterization \( pm=1 \) which is the default parameterization selection. In the second parameterization, \( pm=2 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \zeta \) and \( \rho \). In the third parameterization, \( pm=3 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \xi \) and \( \chi \). In the fourth parameterization, \( pm=4 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \bar{a} \) and \( \bar{b} \).

\section*{mu}
shape parameter \( \alpha \); skewness parameter \( \beta \), \( \text{abs}(\beta) \) is in the range \((0, \alpha)\); scale parameter \( \delta \); \( \delta \) must be zero or positive; location parameter \( \mu \), by default 0. These are the meaning of the parameters in the first parameterization \( pm=1 \) which is the default parameterization selection. In the second parameterization, \( pm=2 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \zeta \) and \( \rho \). In the third parameterization, \( pm=3 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \xi \) and \( \chi \). In the fourth parameterization, \( pm=4 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \bar{a} \) and \( \bar{b} \).

\section*{lambda}
shape parameter \( \alpha \); skewness parameter \( \beta \), \( \text{abs}(\beta) \) is in the range \((0, \alpha)\); scale parameter \( \delta \); \( \delta \) must be zero or positive; location parameter \( \mu \), by default 0. These are the meaning of the parameters in the first parameterization \( pm=1 \) which is the default parameterization selection. In the second parameterization, \( pm=2 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \zeta \) and \( \rho \). In the third parameterization, \( pm=3 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \xi \) and \( \chi \). In the fourth parameterization, \( pm=4 \) \( \alpha \) and \( \beta \) take the meaning of the shape parameters (usually named) \( \bar{a} \) and \( \bar{b} \).

\section*{nu}
a numeric value, the number of degrees of freedom. Note, \( \alpha \) takes the limit of \( \text{abs}(\beta) \), and \( \lambda=\frac{-\nu}{2} \).

\section*{lambda1}
are numeric values where \( \lambda_1 \) is the location parameter, \( \lambda_2 \) is the location parameter, \( \lambda_3 \) is the first shape parameter, and \( \lambda_4 \) is the second shape parameter.

\section*{lambda2}
are numeric values where \( \lambda_1 \) is the location parameter, \( \lambda_2 \) is the location parameter, \( \lambda_3 \) is the first shape parameter, and \( \lambda_4 \) is the second shape parameter.

\section*{lambda3}
are numeric values where \( \lambda_1 \) is the location parameter, \( \lambda_2 \) is the location parameter, \( \lambda_3 \) is the first shape parameter, and \( \lambda_4 \) is the second shape parameter.

\section*{lambda4}
are numeric values where \( \lambda_1 \) is the location parameter, \( \lambda_2 \) is the location parameter, \( \lambda_3 \) is the first shape parameter, and \( \lambda_4 \) is the second shape parameter.
distrMode

- **pm**: an integer value between 1 and 4 for the selection of the parameterization. The default takes the first parameterization.
- **meanlog**: mean and standard deviation of the distribution on the log scale with default values of 0 and 1 respectively.
- **sdlog**: mean and standard deviation of the distribution on the log scale with default values of 0 and 1 respectively.
- **gamma**: value of the index parameter alpha in the interval= (0, 2]; skewness parameter beta, in the range [-1, 1]; scale parameter gamma; and location (or 'shift') parameter delta.
- **loc**: vector of (real) location parameters.
- **disp**: vector of (positive) dispersion parameters.
- **skew**: vector of skewness parameters (in [-1,1]).
- **tail**: vector of parameters (in [1,2]) related to the tail thickness.
- **min**: lower and upper limits of the distribution. Must be finite.
- **max**: lower and upper limits of the distribution. Must be finite.
- **prob**: Probability of success on each trial.
- **size**: number of trials (zero or more).
- **m**: the number of white balls in the urn.
- **n**: number of observations. If length(n) > 1, the length is taken to be the number required.
- **k**: the number of balls drawn from the urn.

**Value**

A numeric value is returned, the (true) mode of the distribution.

**Note**

Some functions like normMode or cauchyMode, which relate to symmetric distributions, are trivial, but are implemented for the sake of exhaustivity.

**Author(s)**

ghMode and ghtMode are from package fBasics; hypMode was written by David Scott; gldMode, nigMode and stableMode were written by Diethelm Wuertz.

**See Also**

mlv for the estimation of the mode; the documentation of the related distributions Beta, GammaDist, etc.
Examples

```r
## Beta distribution
curve(dbeta(x, shape1 = 2, shape2 = 3.1),
     xlim = c(0,1), ylab = "Beta density")
M <- betaMode(shape1 = 2, shape2 = 3.1)
abline(v = M, col = 2)
mlv("beta", shape1 = 2, shape2 = 3.1)

## Lognormal distribution
curve(stats::dlnorm(x, meanlog = 3, sdlog = 1.1),
      xlim = c(0, 10), ylab = "Lognormal density")
M <- lnormMode(meanlog = 3, sdlog = 1.1)
abline(v = M, col = 2)
mlv("lnorm", meanlog = 3, sdlog = 1.1)

curve(VGAM::dpareto(x, scale = 1, shape = 1), xlim = c(0, 10))
abline(v = paretoMode(scale = 1), col = 2)

## Poisson distribution
poisMode(lambda = 6)
poisMode(lambda = 6.1)
mlv("poisson", lambda = 6.1)
```

grenander

The Grenander mode estimator

description

This function computes the Grenander mode estimator.

Usage

```r
grenander(x, bw = NULL, k, p, ...)
```

Arguments

- `x` numeric. Vector of observations.
- `bw` numeric. The bandwidth to be used. Should belong to (0, 1].
- `k` numeric. Parameter 'k' in Grenander's mode estimate, see below.
- `p` numeric. Parameter 'p' in Grenander's mode estimate, see below. If `p = Inf`, the function `venter` is used.
- `...` Additional arguments to be passed to `venter`.

```r
grenander(x, bw = NULL, k, p, ...) 
```
Details

The Grenander estimate is defined by

\[
\frac{\sum_{j=1}^{n-k} \frac{(x_{j+k}+x_j)}{2(x_{j+k}-x_j)^p}}{\sum_{j=1}^{n-k} \frac{1}{(x_{j+k}-x_j)^p}}
\]

If \( p \) tends to infinity, this estimate tends to the Venter mode estimate; this justifies to call \texttt{venter} if \( p = \text{Inf} \).

The user should either give the bandwidth \( bw \) or the argument \( k \), \( k \) being taken equal to \texttt{ceiling}(\( bw \times n \)) \(- 1 \) if missing.

Value

A numeric value is returned, the mode estimate. If \( p = \text{Inf} \), the \texttt{venter} mode estimator is returned.

Note

The user may call \texttt{grenander} through \texttt{mlv(x, method = "grenander", bw, k, p, \ldots)}.

Author(s)

D.R. Bickel for the original code, P. Poncet for the slight modifications introduced.

References


See Also

\texttt{mlv} for general mode estimation; \texttt{venter} for the Venter mode estimate.

Examples

```r
# Unimodal distribution
x <- rnorm(1000, mean = 23, sd = 0.5)

## True mode
normMode(mean = 23, sd = 0.5) # (!)

## Parameter 'k'
k <- 5
```
## Many values of parameter 'p'
ps <- seq(0.1, 4, 0.01)

## Estimate of the mode with these parameters
M <- sapply(ps, function(p) grenander(x, p = p, k = k))

## Distribution obtained
plot(density(M), xlim = c(22.5, 23.5))

---

**hrm**

*Bickel’s half-range mode estimator*

**Description**

SINCE THIS FUNCTION USED TO DEPEND ON THE BIOCONDUCTOR PACKAGE 'GENE-FILTER', IT IS CURRENTLY DEFUNCT.

This function computes Bickel’s half range mode estimator described in Bickel (2002). It is a wrapper around the function `half.range.mode` from package `genefilter`.

**Usage**

hrm(x, bw = NULL, ...)

**Arguments**

- **x** numeric. Vector of observations.
- **bw** numeric. The bandwidth to be used. Should belong to (0, 1]. This gives the fraction of the observations to consider at each step of the iterative algorithm.
- **...** Additional arguments.

**Details**

The mode estimator is computed by iteratively identifying densest half ranges. A densest half range is an interval whose width equals half the current range, and which contains the maximal number of observations. The subset of observations falling in the selected densest half range is then used to compute a new range, and the procedure is iterated.

**Value**

A numeric value is returned, the mode estimate.

**Note**

The user may call hrm through mlv(x, method = "hrm", bw, ...).
Author(s)

The C and R code are due to Richard Bourgon <bourgon@stat.berkeley.edu>, see package gene-filter. The algorithm is described in Bickel (2002).

References


See Also

*mlv()* for general mode estimation; *hsm()* for the half sample mode; *venter()* for the Venter mode estimate.

Examples

```r
## Not run:
# Unimodal distribution
x <- rgamma(1000, shape = 31.9)
# True mode
gammaMode(shape = 31.9)

## Estimate of the mode
hrm(x, bw = 0.4)
mlv(x, method = "hrm", bw = 0.4)

## End(Not run)
```

---

**hsm**

*Half sample mode estimator*

Description

This function computes the Robertson-Cryer mode estimator described in Robertson and Cryer (1974), also called half sample mode (if bw = 1/2) or fraction sample mode (for some other bw) by Bickel (2006).

Usage

```
hsm(x, bw = NULL, k, tie.action = "mean", tie.limit = 0.05, ...)
```
Arguments

- **x** numeric. Vector of observations.
- **bw** numeric or function. The bandwidth to be used. Should belong to (0, 1].
- **k** numeric. See 'Details'.
- **tie.action** character. The action to take if a tie is encountered.
- **tie.limit** numeric. A limit deciding whether or not a warning is given when a tie is encountered.
- **...** Additional arguments.

Details

The modal interval, i.e. the shortest interval among intervals containing \( k+1 \) observations, is computed iteratively, until only one value is found, the mode estimate. At each step \( i \), one takes \( k = \text{ceiling}(bw*n) - 1 \), where \( n \) is the length of the modal interval computed at step \( i-1 \). If \( bw \) is of class "function", then \( k = \text{ceiling}(bw(n)) - 1 \) instead.

Value

A numeric value is returned, the mode estimate.

Note

The user may call `hsm` through `mlv(x, method = "hsm", ...)`.

Author(s)

D.R. Bickel for the original code, P. Poncet for the slight modifications introduced.

References


See Also

`mlv` for general mode estimation; `venter` for the Venter mode estimate.

Examples

```r
# Unimodal distribution
x <- rweibull(10000, shape = 3, scale = 0.9)

## True mode
weibullMode(shape = 3, scale = 0.9)
```
## Estimate of the mode

```r
bandwidth <- function(n, alpha) {1/n^alpha}
hsm(x, bw = bandwidth, alpha = 2)
mlv(x, method = "hsm", bw = bandwidth, alpha = 2)
```

---

### lientz

The empirical Lientz function and the Lientz mode estimator

---

### Description

The Lientz mode estimator is nothing but the value minimizing the empirical Lientz function. A 'plot' and a 'print' methods are provided.

### Usage

```r
lientz(x, bw = NULL)
```

- **S3 method for class 'lientz'**

```r
plot(x, zoom = FALSE, ...)
```

- **S3 method for class 'lientz'**

```r
print(x, digits = NULL, ...)
```

- **S3 method for class 'lientz'**

```r
mlv(x, bw = NULL, abc = FALSE, par = shorth(x), optim.method = "BFGS", ...)
```

### Arguments

- **x** numeric (vector of observations) or an object of class "lientz".
- **bw** numeric. The smoothing bandwidth to be used. Should belong to (0, 1). Parameter 'beta' in Lientz (1970) function.
- **zoom** logical. If TRUE, one can zoom on the graph created.
- **...** if abc = FALSE, further arguments to be passed to optim, or further arguments to be passed to plot.
- **digits** numeric. Number of digits to be printed.
- **abc** logical. If FALSE (the default), the Lientz empirical function is minimised using optim.
- **par** numeric. The initial value used in optim.
- **optim.method** character. If abc = FALSE, the method used in optim.

### Details

The Lientz function is the smallest non-negative quantity \( S(x, \beta) \), where \( \beta = bw \), such that

\[
F(x + S(x, \beta)) - F(x - S(x, \beta)) \geq \beta.
\]

Lientz (1970) provided a way to estimate \( S(x, \beta) \); this estimate is what we call the empirical Lientz function.
Value

lientz returns an object of class c("lientz", "function"); this is a function with additional attributes:

• x the x argument
• bw the bw argument
• call the call which produced the result

mlv.lientz returns a numeric value, the mode estimate. If abc = TRUE, the x value minimizing the Lientz empirical function is returned. Otherwise, the optim method is used to perform minimization, and the attributes: 'value', 'counts', 'convergence' and 'message', coming from the optim method, are added to the result.

Note

The user may call mlv.lientz through mlv(x, method = "lientz", ...).

References


See Also

mlv for general mode estimation; shorth for the shorth estimate of the mode

Examples

# Unimodal distribution
x <- rbeta(1000,23,4)

## True mode
betaMode(23, 4)

## Lientz object
f <- lientz(x, 0.2)
print(f)
plot(f)

## Estimate of the mode
mlv(f) # optim(shorth(x), fn = f)
mlv(f, abc = TRUE) # x[which.min(f(x))]
mlv(x, method = "lientz", bw = 0.2)

# Bimodal distribution
x <- c(rnorm(1000,5,1), rnorm(1500, 22, 3))
meanshift

\[
f \leftarrow \text{lentz}(x, 0.1)
\]
\[
\text{plot}(f)
\]

---

**Description**

The Meanshift mode estimator.

**Usage**

```r
meanshift(
  x,
  bw = NULL,
  kernel = "gaussian",
  par = shorth(x),
  iter = 1000,
  tolerance = sqrt(.Machine$double.eps)
)
```

**Arguments**

- `x` numeric. Vector of observations.
- `bw` numeric. The smoothing bandwidth to be used.
- `kernel` character. The kernel to be used. Available kernels are "biweight", "cosine", "eddy", "epanechnikov", "gaussian", "optcosine", "rectangular", "triangular", "uniform". See `density` for more details on some of these kernels.
- `par` numeric. The initial value used in the meanshift algorithm.
- `iter` numeric. Maximal number of iterations.
- `tolerance` numeric. Stopping criteria.

**Value**

`meanshift` returns a numeric value, the mode estimate, with an attribute "iterations". The number of iterations can be less than `iter` if the stopping criteria specified by `eps` is reached.

**Note**

The user should preferentially call `meanshift` through `mlv(x, method = "meanshift", ...)`.  

**References**

See Also

mlv, tsybakov.

Examples

# Unimodal distribution
x <- rweibull(100, shape = 12, scale = 0.8)

## True mode
weibullMode(shape = 12, scale = 0.8)

## Estimate of the mode
mlv(x, method = "meanshift", par = mean(x))

---

**mlv**

*Estimation of the Mode(s) or Most Likely Value(s)*

Description

mlv is a generic function for estimating the mode of a univariate distribution. Different estimates (or methods) are provided:

- **mfv**, which returns the most frequent value(s) in a given numerical vector,
- the **Lientz** mode estimator, which is the value minimizing the Lientz function estimate,
- the Chernoff mode estimator, also called **naive** mode estimator, which is defined as the center of the interval of given length containing the most observations,
- the **Venter** mode estimator, including the **shorth**, i.e. the midpoint of the modal interval,
- the **Grenander** mode estimator,
- the half sample mode (**HSM**) and the half range mode (**HRM**), which are iterative versions of the Venter mode estimator,
- **Parzen**'s kernel mode estimator, which is the value maximizing the kernel density estimate,
- the **Tsybakov** mode estimator, based on a gradient-like recursive algorithm,
- the **Asselin** de Beauville mode estimator, based on an algorithm detecting chains and holes in the sample,
- the **Vieu** mode estimator,
- the **meanshift** mode estimator.

mlv can also be used to compute the mode of a given distribution, with `mlv.character`. 
### Usage

```r
m.lv(x, ...) # S3 method for class 'character'
m.lv(x, na.rm = FALSE, ...)
```

```r
# S3 method for class 'factor'
m.lv(x, na.rm = FALSE, ...)
```

```r
# S3 method for class 'logical'
m.lv(x, na.rm = FALSE, ...)
```

```r
# S3 method for class 'integer'
m.lv(x, na.rm = FALSE, ...)
```

```r
# Default S3 method:
m.lv(x, bw = NULL, method, na.rm = FALSE, ...)
```

### Arguments

- `x`: numeric (vector of observations), or an object of class "factor", "integer", etc.
- `...`: Further arguments to be passed to the function called for computation.
- `na.rm`: logical. Should missing values be removed?
- `bw`: numeric. The bandwidth to be used. This may have different meanings regarding the method used.
- `method`: character. One of the methods available for computing the mode estimate. See 'Details'.

### Details

For the default method of `m.lv`, available methods are "lientz", "naive", "venter", "grenander", "hsm", "parzen", "tsybakov", "asselin", and "meanshift". See the description above and the associated links.

If `x` is of class "character" (with length > 1), "factor", or "integer", then the most frequent value found in `x` is returned using `mfv` from package `statip`.

If `x` is of class "character" (with length 1), `x` should be one of "beta", "cauchy", "gev", etc. i.e. a character for which a function `*Mode` exists (for instance `betaMode`, `cauchyMode`, etc.). See `distrMod` for the available functions. The mode of the corresponding distribution is returned.

If `x` is of class `mlvlientz`, see `Lientz` for more details.

### Value

A vector of the same type as `x`. Be aware that the length of this vector can be > 1.
References

See the references on mode estimation on the modeest-package’s page.

See Also

mfv, parzen, venter, meanshift, grenander, hsm, lientz, naive, tsybakov, skewness

Examples

# Unimodal distribution
x <- rbeta(1000, 23, 4)

## True mode
betaMode(23, 4)
# or
mlv("beta", shape1 = 23, shape2 = 4)

## Be aware of this behaviour:
mlv("normal") # returns 0, the mode of the standard normal distribution
mlv("normal") # returns 0 again, since "normal" is matched with "norm"
mlv("abnormal") # returns "abnormal", since the input vector "abnormal"
# is not recognized as a distribution name, hence is taken as a character
# vector from which the most frequent value is requested.

## Estimate of the mode
mlv(x, method = "lientz", bw = 0.2)
mlv(x, method = "naive", bw = 1/3)
mlv(x, method = "venter", type = "shorth")
mlv(x, method = "grenander", p = 4)
mlv(x, method = "hsm")
mlv(x, method = "parzen", kernel = "gaussian")
mlv(x, method = "tsybakov", kernel = "gaussian")
mlv(x, method = "asselin", bw = 2/3)
mlv(x, method = "vieu")
mlv(x, method = "meanshift")

Description

This package provides estimators of the mode of univariate unimodal (and sometimes multimodal) data, and values of the modes of usual probability distributions.

For a complete list of functions, use library(help = "modeest") or help.start().
References

The Chernoff or 'naive' mode estimator

Description
This estimator, also called the *naive* mode estimator, is defined as the center of the interval of given length containing the most observations. It is identical to Parzen’s kernel mode estimator, when the kernel is chosen to be the uniform kernel.

Usage
naive(x, bw = 1/2)

Arguments
- **x** numeric. Vector of observations.
- **bw** numeric. The smoothing bandwidth to be used. Should belong to (0, 1). See below.

Value
A numeric vector is returned, the mode estimate, which is the center of the interval of length 2*bw containing the most observations.

Note
The user may call naive through mlv(x, method = "naive", bw).

References

See Also
mlv for general mode estimation; parzen for Parzen’s kernel mode estimation.
Examples

# Unimodal distribution
x <- rf(10000, df1 = 40, df2 = 30)

## True mode
fMode(df1 = 40, df2 = 30)

## Estimate of the mode
mean(naive(x, bw = 1/4))
mlv(x, method = "naive", bw = 1/4)

parzen

Parzen’s Kernel mode estimator

Description

Parzen’s kernel mode estimator is the value maximizing the kernel density estimate.

Usage

parzen(
  x,
  bw = NULL,
  kernel = "gaussian",
  abc = FALSE,
  tolerance = .Machine$double.eps^0.25,
  ...
)

Arguments

x numeric. Vector of observations.
bw numeric. The smoothing bandwidth to be used.
kernel character. The kernel to be used. For available kernels see densityfun in package statip.
abc logical. If FALSE (the default), the kernel density estimate is maximised using optim.
tolerance numeric. Desired accuracy in the optimize function.
... If abc = FALSE, further arguments to be passed to optim.

Details

If kernel = "uniform", the naive mode estimate is returned.
Value

parzen returns a numeric value, the mode estimate. If abc = TRUE, the x value maximizing the density estimate is returned. Otherwise, the optim method is used to perform maximization, and the attributes: 'value', 'counts', 'convergence' and 'message', coming from the optim method, are added to the result.

Note

The user may call parzen through mlv(x, method = "kernel", ...) or mlv(x, method = "parzen", ...).

Presently, parzen is quite slow.

References


See Also

mlv, naive

Examples

# Unimodal distribution
x <- rlnorm(10000, meanlog = 3.4, sdlog = 0.2)

### True mode
lnormMode(meanlog = 3.4, sdlog = 0.2)

### Estimate of the mode
mlv(x, method = "kernel", kernel = "gaussian", bw = 0.3, par = shorth(x))
Description

This function encodes different methods to calculate the skewness from a vector of observations.

Usage

skewness(x, na.rm = FALSE, method = c("moment", "fisher", "bickel"), M, ...)

Arguments

x numeric. Vector of observations.
na.rm logical. Should missing values be removed?
method character. Specifies the method of computation. These are either "moment", "fisher" or "bickel". The "moment" method is based on the definition of skewness for distributions; this form should be used when resampling (bootstrap or jackknife). The "fisher" method corresponds to the usual "unbiased" definition of sample variance, although in the case of skewness exact unbiasedness is not possible.
M numeric. (An estimate of) the mode of the observations x. Default value is shorth(x).
... Additional arguments.

Value

skewness returns a numeric value. An attribute reports the method used.

Author(s)

Diethelm Wuertz and contributors for the original skewness function from package fBasics.

References


See Also

mlv for general mode estimation; shorth for the shorth estimate of the mode
Example

```r
## Skewness = 0
x <- rnorm(1000)
skewness(x, method = "bickel", M = shorth(x))

## Skewness > 0 (left skewed case)
x <- rbeta(1000, 2, 5)
skewness(x, method = "bickel", M = betaMode(2, 5))

## Skewness < 0 (right skewed case)
x <- rbeta(1000, 7, 2)
skewness(x, method = "bickel", M = hsm(x, bw = 1/3))
```

**tsybakov**

The Tsybakov mode estimator

**Description**

This mode estimator is based on a gradient-like recursive algorithm, more adapted for online estimation. It includes the Mizoguchi-Shimura (1976) mode estimator, based on the window training procedure.

**Usage**

```r
tsyaakov(
  x, 
  bw = NULL, 
  a, 
  alpha = 0.9, 
  kernel = "triangular", 
  dmp = TRUE, 
  par = shorth(x)
)
```

**Arguments**

- **x** numeric. Vector of observations.
- **bw** numeric. Vector of length length(x) giving the sequence of smoothing bandwidths to be used.
- **a** numeric. Vector of length length(x) used in the gradient algorithm
- **alpha** numeric. An alternative way of specifying a. See 'Details'.
- **kernel** character. The kernel to be used. Available kernels are "biweight", "cosine", "eddy", "epanechnikov", "gaussian", "optcosine", "rectangular", "triangular", "uniform". See `density` for more details on some of these kernels.
- **dmp** logical. If TRUE, Djeddour et al. version of the estimate is used.
- **par** numeric. Initial value in the gradient algorithm. Default value is `shorth(x)`. 


Details
If bw or a is missing, a default value advised by Djeddour et al (2003) is used: bw = (1:length(x))^(-1/7) and a = (1:length(x))^(-alpha). (with alpha = 0.9 if alpha is missing).

Value
A numeric value is returned, the mode estimate.

Warning
The Tsybakov mode estimate as it is presently computed does not work very well. The reasons of this inefficiency should be further investigated.

Note
The user may call tsybakov through mlv(x, method = "tsybakov", ...).

References

See Also
mlv for general mode estimation.

Examples
x <- rbeta(1000, shape1 = 2, shape2 = 5)

## True mode:
betaMode(shape1 = 2, shape2 = 5)

## Estimation:
tsybakov(x, kernel = "triangular")
tsybakov(x, kernel = "gaussian", alpha = 0.99)
mlv(x, method = "tsybakov", kernel = "gaussian", alpha = 0.99)
The Venter / Dalenius / LMS mode estimator

Description
This function computes the Venter mode estimator, also called the Dalenius, or LMS (Least Median Square) mode estimator.

Usage
venter(
  x,
  bw = NULL,
  k,
  iter = 1,
  type = 1,
  tie.action = "mean",
  tie.limit = 0.05,
  warn = FALSE
)

Arguments
x numeric. Vector of observations.
bw numeric. The bandwidth to be used. Should belong to (0, 1]. See 'Details'.
k numeric. See 'Details'.
iter numeric. Number of iterations.
type numeric or character. The type of Venter estimate to be computed. See 'Details'.
tie.action character. The action to take if a tie is encountered.
tie.limit numeric. A limit deciding whether or not a warning is given when a tie is encountered.
warn logical. If TRUE, a warning is thrown when a tie is encountered.
...
Further arguments.

Details
The modal interval, i.e. the shortest interval among intervals containing k+1 observations, is first computed. (In dimension > 1, this question is known as a 'k-enclosing problem'.) The user should either give the bandwidth bw or the argument k, k being taken equal to ceiling(bw*n) - 1 if missing, so bw can be seen as the fraction of the observations to be considered for the shortest interval.

If type = 1, the midpoint of the modal interval is returned. If type = 2, the floor((k+1)/2)th element of the modal interval is returned. If type = 3 or type = "dalenius", the median of the
modal interval is returned. If \( \text{type} = 4 \) or \( \text{type} = \text{"shorth"} \), the mean of the modal interval is returned. If \( \text{type} = 5 \) or \( \text{type} = \text{"ekblom"} \), Ekblom’s \( L_{-\infty} \) estimate is returned, see Ekblom (1972). If \( \text{type} = 6 \) or \( \text{type} = \text{"hsm"} \), the half sample mode (hsm) is computed, see \text{hsm}.

**Value**

A numeric value is returned, the mode estimate.

**Note**

The user may call \text{venter} through \text{mlv(x, method = "venter", ...)}. 

**References**


**See Also**

\text{mlv} for general mode estimation, \text{hsm} for the half sample mode.

**Examples**

library(evd)

# Unimodal distribution
x <- rgev(1000, loc = 23, scale = 1.5, shape = 0)

## True mode
gevMode(loc = 23, scale = 1.5, shape = 0)

## Estimate of the mode
venter(x, bw = 1/3)
mlv(x, method = "venter", bw = 1/3)
Vieu's mode estimator

Description

Vieu's mode estimator is the value at which the kernel density derivative estimate is null.

Usage

vieu(x, bw = NULL, kernel = "gaussian", abc = FALSE, ...)

Arguments

x numeric. Vector of observations.
bw numeric. The smoothing bandwidth to be used.
kernel character. The kernel to be used. Available kernels are "biweight", "cosine", "eddy", "epanechnikov", "gaussian", "optcosine", "rectangular", "triangular", "uniform". See density for more details on some of these kernels.
abc logical. If FALSE (the default), the root of the density derivate estimate is searched with uniroot.
... If abc = FALSE, further arguments to be passed to uniroot.

Value

vieu returns a numeric value, the mode estimate. If abc = TRUE, the x value at which the density derivative estimate is null is returned. Otherwise, the uniroot method is used.

Note

The user may call vieu through mlv(x, method = "vieu", ...).

Presently, vieu is quite slow.

References


See Also

mlv, parzen.
Examples

# Unimodal distribution
x <- rlnorm(10000, meanlog = 3.4, sdlog = 0.2)

## True mode
lnormMode(meanlog = 3.4, sdlog = 0.2)

## Estimate of the mode
mlv(x, method = "vieu", kernel = "gaussian")
## Index

* distribution
  modeest, 18
* nonparametric
  modeest, 18
* robust
  modeest, 18
* univar
  modeest, 18

Asselin, 16
Asselin (asselin), 2
asselin, 2

bernMode (distrMode), 3
Beta, 7
betaMode (distrMode), 3
binomMode (distrMode), 3

cauchyMode (distrMode), 3
Chernoff (naive), 21
chernoff (naive), 21
chisqMode (distrMode), 3

dagumMode (distrMode), 3
density, 15, 25, 29
densityfun, 22
distrMode, 3, 17

expMode (distrMode), 3
fiskMode (distrMode), 3
fMode (distrMode), 3
frechetMode (distrMode), 3

GammaDist, 7
gammaMode (distrMode), 3
gemnMode (distrMode), 3
gmvMode (distrMode), 3
ghMode, 7
gmMode (distrMode), 3
ghtMode, 7
gldMode, 7
gldMode (distrMode), 3
gompertzMode (distrMode), 3
gpdMode (distrMode), 3
Grenander, 16
Grenander (grenander), 8
grenander, 8, 18
gumbelMode (distrMode), 3

HRM, 16
HRM (hrm), 10
hrm, 10
HSM, 16
HSM (hsm), 11
hsm, 11, 11, 18, 28
hyperMode (distrMode), 3
hypMode, 7
hypMode (distrMode), 3

koenkerMode (distrMode), 3
kumarMode (distrMode), 3

laplaceMode (distrMode), 3
Lientz, 16, 17
Lientz (lientz), 13
lientz, 13, 18
lnormMode (distrMode), 3
logisMode (distrMode), 3
lomaxMode (distrMode), 3

maxwellMode (distrMode), 3
meanshift, 15, 16, 18
mfv, 16–18
mlv, 2, 7, 9, 11, 12, 14, 16, 16, 21, 23, 24, 26, 28, 29
mlv.lientz (lientz), 13
mlv1 (mlv), 16
modeest, 18
modeest-package (modeest), 18
mvnormMode (distrMode), 3
naive, 16, 18, 21, 22, 23
nakaMode (distrMode), 3
nbinomMode (distrMode), 3
nigMode, 7
nigMode (distrMode), 3
normMode (distrMode), 3
optim, 13, 14, 22, 23
optimize, 22
paralogisticMode (distrMode), 3
paretoMode (distrMode), 3
Parzen, 16
Parzen (parzen), 22
parzen, 18, 21, 22, 29
plot, 13
plot.lientz (lientz), 13
poisMode (distrMode), 3
print.lientz (lientz), 13
quantile, 2
rayleighMode (distrMode), 3
shorth, 14, 16, 24, 25
shorth (venter), 27
skewness, 18, 24
stableMode, 7
stableMode (distrMode), 3
stableMode2 (distrMode), 3
tMode (distrMode), 3
Tsybakov, 16
Tsybakov (tsybakov), 25
tsymbakov, 16, 18, 25
unifMode (distrMode), 3
uniroot, 29
Venter, 16
Venter (venter), 27
venter, 8, 9, 11, 12, 18, 27
Vieu, 16
Vieu (vieu), 29
vieu, 29
weibullMode (distrMode), 3
yulesMode (distrMode), 3