Package: modehunt (via r-universe)

September 4, 2024

Type Package

Title Multiscale Analysis for Density Functions

Version 1.0.7

Date 2015-07-03

Author Kaspar Rufibach <kaspar.rufibach@gmail.com> and Guenther Walther <gwalther@stanford.edu>

Maintainer Kaspar Rufibach <kaspar.rufibach@gmail.com>

Imports stats, utils

Description Given independent and identically distributed observations X(1), ..., X(n) from a density f, provides five methods to perform a multiscale analysis about f as well as the necessary critical values. The first method, introduced in Duembgen and Walther (2008), provides simultaneous confidence statements for the existence and location of local increases (or decreases) of f, based on all intervals I(all) spanned by any two observations X(j), X(k). The second method approximates the latter approach by using only a subset of I(all) and is therefore computationally much more efficient, but asymptotically equivalent. Omitting the additive correction term Gamma in either method offers another two approaches which are more powerful on small scales and less powerful on large scales, however, not asymptotically minimax optimal anymore. Finally, the block procedure is a compromise between adding Gamma or not, having intermediate power properties. The latter is again asymptotically equivalent to the first and was introduced in Rufibach and Walther (2010).

License GPL (>= 2)

URL http://www.kasparrufibach.ch,
http://www-stat.stanford.edu/~gwalther

NeedsCompilation no

Date/Publication 2015-07-03 08:47:50

Repository https://numbersman77.r-universe.dev

RemoteUrl https://github.com/cran/modehunt

RemoteRef HEAD

RemoteSha f555a7e6d38c9860f3d495fef4ecb257c7fe517a

Contents

`modehunt-package` *Multiscale Analysis for Density Functions*

Description

Provides five methods and corresponding critical values to perform mode hunting, i.e. to compute multiscale test statistics based on local order statistics and spacings that provide simultaneous confidence statements for the existence and location of local increases and decreases of a density.

Details

Package:	modehunt
Type:	Package
Version:	1.0.7
Date:	2015-07-03
License:	GPL (>=2)

In Duembgen and Walther (2008) a multiscale test statistic based on spacings was introduced. This method provides simultaneous confidence statements for the existence and location of local increases and decreases of a density. The procedure guarantees finite–sample significance levels and possesses certain asymptotic optimality and adaptivity properties. However, since the local test statistics are computed on all $O(n^2)$ intervals in the set

$$\mathcal{I}_{all} = \Big\{(j,\ k)\ :\ 0 \le j < k \le n+1,\ k-j > 1\Big\},$$

this latter procedure is computationally very expensive. Furthermore, the correction term Γ employed by Duembgen and Walther (2008) to prevent the global test statistic to be dominated by the values of the local test statistics on small scales needs in principle to be re–derived for any new local test statistic, a non–trivial task in general. In Rufibach and Walther (2010), two new procedures are proposed: One that within the original framework of Duembgen and Walther (2008) approximates the set $\mathcal{I}_{all}$ by a specific subset of intervals $\mathcal{I}_{app}$ that only contains $O(n \log n)$ intervals. It is shown that considering $\mathcal{I}_{app}$ yields a procedure that is in terms of power asymptotically equivalent to that based on $\mathcal{I}_{all}$, however, computationally much more efficient.

Finally, Rufibach and Walther (2010) propose a *block procedure*. Here, all intervals under consideration are grouped into blocks, where each interval in a block contains approximately the same number of original observations. Critical values are then computed per block. Again, this procedure is basically asymptotically equivalent to the standard approach proposed in Duembgen and Walther (2008), but again computationally much faster. It further offers a (finite–sample) tradeoff between employing or omitting the additive correction Γ.

The initial procedure by Duembgen and Walther (2008) is implemented as the function `modeHunting`. The help file to the latter function also contains some more description of the mathematical details. `criticalValuesAll` can be used to compute critical values for this approach and `cvModeAll` contains a table of critical values (with and without correction term) for some n and $alpha$.

The corresonding functions and p-values for the approximation are made available as `modeHuntingApprox`, `criticalValuesApprox`, and `cvModeApprox` and for the block method as `modeHuntingBlock`, `criticalValuesBlock`, and `cvModeBlock`.

Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

Guenther Walther, <gwalther@stanford.edu>,
www-stat.stanford.edu/~gwalther

Kaspar Rufibach acknowledges support by the Swiss National Science Foundation SNF, http://www.snf.ch.

Guenther Walther acknowledges support by NSF grants DMS-9875598, DMS-0505682, and NIH grant 5R33HL068522.

References

Duembgen, L. and Walther, G. (2008). Multiscale Inference about a density. *Ann. Statist.*, **36**, 1758–1785.

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

Examples

```
## generate random sample
set.seed(1977)
```

```
n <- 200; a <- 0; b <- 0.5; s <- 2 / (b - a)
X.raw <- rlin(n, a, b, s)

## input critical values
alpha <- 0.05
data(cvModeAll); data(cvModeApprox); data(cvModeBlock)
cv.all <- cvModeAll[cvModeAll$alpha == alpha & cvModeAll$n == n, 3:4]
cv.approx <- cvModeApprox[cvModeApprox$alpha == alpha & cvModeApprox$n == n, 3:4]
cv.block <- cvModeBlock[cvModeBlock$alpha == alpha & cvModeBlock$n == n, 3:11]

## standard procedure from Duembgen and Walther (2008)
mod1 <- modeHunting(X.raw, lower = 0, upper = 1, cv.all, min.int = TRUE)

## procedure from Rufibach and Walther (2010) based on I_app
mod2 <- modeHuntingApprox(X.raw, lower = 0, upper = 1,
                          crit.vals = cv.approx, min.int = TRUE)

## block procedure from Rufibach and Walther (2010)
mod3 <- modeHuntingBlock(X.raw, lower = 0, upper = 1,
                         crit.vals = cv.block, min.int = TRUE)

## display
mod1; mod2; mod3
```

blocks *Computes number of observations for each block*

Description

In Rufibach and Walther (2010) a new multiscale mode hunting procedure is presented that compares the local test statistics with critical values given by blocks. Blocks are collection of intervals on a given grid that contain roughly the same number of original observations.

Usage

```
blocks(n, m0 = 10, fm = 2)
```

Arguments

n	Number of observations.
m0	Initial parameter that determines the number of observations in one block.
fm	Factor by which m is increased from block to block.

Details

In our block procedure, we only consider a subset $\mathcal{I}_{app}$ of all possible intervals $\mathcal{I}_{all}$ where

$$\mathcal{I}_{all} = \Big\{(j,\ k)\ :\ 0 \le j < k \le n+1,\ k-j > 1\Big\}.$$

This subset $\mathcal{I}_{app}$ is computed as follows:

Set $d_1, m_1, f_m > 1$. Then:

```
$for\ \ r = 1, \ldots, \#blocks$
$d_r := round(d_1 f_m^{(r-1)/2}),\ m_r := m_1 f_m^{r-1}$.
Include $(j,k)$ in $\mathcal{I}_{app}$ if
(a) $j,k \in \{1 + id_r,\ i = 0,1,\ldots\}$ \\ (we only consider every $d$–th observation) and
(b) $m_r \le k - j - 1 \le 2m_r - 1$ \\ ($\mathcal{I}_{jk}$ contains between $m_r$ and $2m_r - 1$ observations)
$end\ \ for$
```

Value

$b \times 2$–matrix, where b is the number of blocks and the columns contain the lower and the upper number of observations that form each block.

Note

The asymptotic results in Rufibach and Walther (2010) are only derived for $f_m = 2$.

Author(s)

Kaspar Rufibach, <`kaspar.rufibach@gmail.com`>,
`http://www.kasparrufibach.ch`

Guenther Walther, <`gwalther@stanford.edu`>,
`www-stat.stanford.edu/~gwalther`

References

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

See Also

This function is called by `modeHuntingBlock`.

`criticalValuesAll` *Compute critical values based on the set of all intervals*

Description

This function computes critical values that are needed to perform the multiscale analysis about a density using the function `modeHunting`.

Usage

```
criticalValuesAll(n, alpha, M, display, path)
```

Arguments

n	Number of observations.
alpha	Significance level, real number in $(0, 1)$.
M	Number of runs to perform.
display	If display == 1, every 100–th step is indicated in the output window, else not.
path	If path != NA, the current number of performed simulations is saved in this location.

Details

For more details see the function modeHunting and the data set cvModeAll.

Value

A 2-dimensional vector containing the critical value for the test statistic with or without additive correction Γ.

Author(s)

Kaspar Rufibach, <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

Guenther Walther, <gwalther@stanford.edu>,
www-stat.stanford.edu/~gwalther

References

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

See Also

The resulting critical values can be used by the function modeHunting. Critical values for some combinations of n and α are available in cvModeAll.

Examples

```
## compute critical values and compare to those in cvModeAll
## (to see output in R, press CTRL + W)
cv1 <- criticalValuesAll(n = 200, alpha = 0.05, M = 10 ^ 2, display = 1, path = NA)
data(cvModeAll)
cv2 <- cvModeAll[cvModeAll$alpha == 0.05 & cvModeAll$n == 200, 3:4]
rbind(cv1, cv2)
```

`criticalValuesApprox` *Compute critical values for (1) the original test statistic with or without additive correction, based on the aprroximating set of intervals and (2) for the block procedure*

Description

This function computes critical values that can be used to perform the multiscale analysis about a density with the functions `modeHuntingApprox` and `modeHuntingBlock`.

Usage

```
criticalValuesApprox(n, d0 = 2, m0 = 10, fm = 2, alpha = 0.05,
        gam = 2, tail = 10, M = 10 ^ 5, display = 0, path = NA)
```

Arguments

`n`	Number of observations.
`d0`	Initial parameter for the grid resolution.
`m0`	Initial parameter for the number of observations in one block.
`fm`	Factor by which m is increased from block to block.
`alpha`	Significance level, real number in $(0, 1)$.
`gam`	Weighting exponent for level in each block.
`tail`	Offset, determines together with `gam` the decrease of the level from one block to another.
`M`	Number of runs to perform.
`display`	If `display` == 1, every 100–th step is indicated in the output window, else not.
`path`	If `path` != `NA`, the current number of performed simulations is saved in this location.

Details

For details see the function `modeHuntingApprox` and the data set `cvModeApprox`.

Value

`approx`	A 2-dimensional vector containing the critical value for the test statistic with or without additive correction Γ.
`block`	A vector containing the critical value for each block.

Note

The asymptotic results in Rufibach and Walther (2010) are only derived for $f_m = 2$.

Author(s)

Kaspar Rufibach, <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

Guenther Walther, <gwalther@stanford.edu>,
www-stat.stanford.edu/~gwalther

References

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

See Also

The resulting critical values are used by the functions `modeHuntingApprox` and `modeHuntingBlock`. Critical values for some combinations of n and α are available in `cvModeApprox` and `cvModeBlock`.

Examples

```
## compute critical values and compare to those in cvModeAll and cvModeBlock
## (to see output in R, press CTRL + W)
cv <- criticalValuesApprox(n = 200, d0 = 2, m0 = 10, fm = 2,
     alpha = 0.05, gam = 2, tail = 10, M = 10 ^ 2, display = 1, path = NA)
cv1 <- cv$approx; cv2 <- cv$block

data(cvModeApprox); data(cvModeBlock)
cv3 <- cvModeApprox[cvModeApprox$alpha == 0.05 & cvModeApprox$n == 200, 3:4]
cv4 <- cvModeBlock[cvModeBlock$alpha == 0.05 & cvModeBlock$n == 200, 3:6]
rbind(cv1, cv3)
rbind(cv2, cv4)
```

---

cvModeAll *Critical values for test statistic based on all intervals*

Description

This dataset contains critical values for some n and α for the test statistic based on all intervals, with or without additive correction term Γ.

Usage

```
data(cvModeAll)
```

Format

A data frame providing 15 different combinations of n and α and the following columns:

`alpha`	The levels at which critical values were simulated.
`n`	The number of observations for which critical values were simulated.
`withadd`	Critical values based on $T_n^+(\mathbf{U})$ and the set of all intervals $\mathcal{I}_{all}$.
`noadd`	Critical values based on $T_n(\mathbf{U})$ and the set of all intervals $\mathcal{I}_{all}$.

Details

For details on the above test statistics see `modeHunting`. Critical values are based on $M = 100'000$ simulations of i.i.d. random vectors

$$\mathbf{U} = (U_1, \ldots, U_n)$$

where U_i is a uniformly on $[0, 1]$ distributed random variable, $i = 1, \ldots, M$.

Remember

n is the number of *interior observations*, i.e. if you are analyzing a sample of size m, then you need critical values corresponding to

`n = m-2`	If no additional information on a and b is available.
`n = m-1`	If either a or b is known to be a certain finite number.
`n = m`	If both a and b are known to be certain finite numbers,

where $[a, b] = \{x\ :\ f(x) > 0\}$ is the support of f.

Source

These critical values were generated using the function `criticalValuesAll`. Critical values for other combinations for α and n can be computed using this latter function.

References

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

Examples

```
## extract critical values for alpha = 0.05, n = 200
data(cvModeAll)
cv <- cvModeAll[cvModeAll$alpha == 0.05 & cvModeAll$n == 200, 3:4]
cv
```

cvModeApprox — *Critical values for test statistic based on the approximating set of intervals*

Description

This dataset contains critical values for some n and α for the test statistic based on the approximating set of intervals, with or without additive correction term Γ.

Usage

```
data(cvModeApprox)
```

Format

A data frame providing 15 different combinations of n and α and the following columns:

`alpha`	The levels at which critical values were simulated.
`n`	The number of observations for which critical values were simulated.
`withadd`	Critical values based on $T_n^+(\mathbf{U})$ and the approximating set of intervals $\mathcal{I}_{app}$.
`noadd`	Critical values based on $T_n(\mathbf{U})$ and the approximating set of intervals $\mathcal{I}_{app}$.

Details

For details see `modeHunting`. Critical values are based on $M = 100'000$ simulations of i.i.d. random vectors

$$\mathbf{U} = (U_1, \ldots, U_n)$$

where U_i is a uniformly on $[0,1]$ distributed random variable, $i = 1, \ldots, M$.

Remember

n is the number of *interior observations*, i.e. if you are analyzing a sample of size m, then you need critical values corresponding to

`n = m-2`	If no additional information on a and b is available.
`n = m-1`	If either a or b is known to be a certain finite number.
`n = m`	If both a and b are known to be certain finite numbers,

where $[a, b] = \{x \,:\, f(x) > 0\}$ is the support of f.

Source

These critical values were generated using the function `criticalValuesApprox`. Critical values for other combinations for α and n can be computed using this latter function.

References

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

Examples

```
## extract critical values for alpha = 0.05, n = 200
data(cvModeApprox)
cv <- cvModeApprox[cvModeApprox$alpha == 0.05 & cvModeApprox$n == 200, 3:4]
cv
```

`cvModeBlock` *Critical values for test statistic based on the block procedure*

Description

This dataset contains critical values for some n and α for the block procedure.

Usage

```
data(cvModeBlock)
```

Format

A data frame providing 15 different combinations of n and α and the following columns:

`alpha`	The levels at which critical values were simulated.
`n`	The number of observations for which critical values were simulated.
`block 1 - 9`	Critical values for the respective blocks.

Details

For details see `modeHunting`. Critical values are based on $M = 100'000$ simulations of i.i.d. random vectors

$$\mathbf{U} = (U_1, \ldots, U_n)$$

where U_i is a uniformly on $[0,1]$ distributed random variable, $i = 1, \ldots, M$.

Remember

n is the number of *interior observations*, i.e. if you are analyzing a sample of size m, then you need critical values corresponding to

`n = m-2`	If no additional information on a and b is available.
`n = m-1`	If either a or b is known to be a certain finite number.
`n = m`	If both a and b are known to be certain finite numbers,

where $[a,b] = \{x \ : \ f(x) > 0\}$ is the support of f.

Source

These critical values were generated using the function `criticalValuesBlock`. Critical values for other combinations for α and n can be computed using this latter function.

References

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

Examples

```
## extract critical values for alpha = 0.05, n = 200
data(cvModeBlock)
cv <- cvModeBlock[cvModeBlock$alpha == 0.05 & cvModeBlock$n == 200, 3:11]
cv
```

`lin` *Perturbed Uniform Distribution*

Description

Density function, distribution function, quantile function and random generation for the perturbed uniform distribution having a linear increase of slope s on an interval $[a, b] \in [0, 1]$.

Usage

```
dlin(x, a, b, s)
plin(q, a, b, s)
qlin(p, a, b, s)
rlin(n, a, b, s)
```

Arguments

x, q	Vector of quantiles.
p	Vector of probabilities.
n	Number of observations.
a	Left interval endpoint, real number in $[0, 1)$.
b	Right interval endpoint, real number in $(0, 1]$.
s	Slope parameter, real number such that $\|s\| \leq 2/(b-a)$.

Details

The what we call perturbed uniform distribution (PUD) with perturbation on an interval $[a, b] \in [0, 1]$ with slope parameter s such that $|s| \le 2/(b - a)$ has density function

$$f_{a,b,s}(x) = \Big(sx - s\frac{a+b}{2}\Big)1\{x \in [a, b)\} + 1\{[0, a) \cup [b, 1]\},$$

distribution function

$$F_{a,b,s}(q) = \Big(q + \frac{s}{2}(q^2 - a^2 + (a - x)(a + b))\Big)1\{q \in [a, b)\} + q\{[0, a) \cup [b, 1]\},$$

and quantile function

$$F^{-1}_{a,b,s}(p) = \Big(-s^{-1} + \frac{a+b}{2} + \frac{s\sqrt{(a-b)^2 + \frac{4}{s}(\frac{1}{s} - (a + b) + 2p)}}{2|s|}\Big)\,1\{p \in [a, b)\} + p\{[0, a) \cup [b, 1]\}.$$

This function was used to carry out the simulations to compute the power curves given in Rufibach and Walther (2010).

Value

`dlin` gives the values of the density function, `plin` those of the distribution function, and `qlin` those of the quantile function of the PUD at x, q, and p, respectively. `rlin` generates n random numbers, returned as an ordered vector.

Author(s)

Kaspar Rufibach, <`kaspar.rufibach@gmail.com`>,
`http://www.kasparrufibach.ch`

Guenther Walther, <`gwalther@stanford.edu`>,
`www-stat.stanford.edu/~gwalther`

References

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

`minimalIntervals` *Compute set of minimal intervals*

Description

In general, all intervals that have a test statistic bigger than the respective critical value are output. For a given set of intervals $\mathcal{K}$, all intervals J such that $\mathcal{K}$ does not contain a proper subset of J are called *minimal*. Given $\mathcal{K}$, this function computes the set of minimal intervals.

Usage

```
minimalIntervals(ints)
```

Arguments

`ints`	Either one of the sets $\mathcal{D}^+$ or $\mathcal{D}^-$ as output by one of the functions `modeHunting`, `modeHuntingApprox`, or `modeHuntingBlock`.

Value

Returns the set of minimal elements $\mathbf{D}^{\pm}$, corresponding to the set of input intervals $\mathcal{D}^{\pm}$.

Note

Depending on the value of $min.int$, this function is called by `modeHunting`, `modeHuntingApprox`, and `modeHuntingBlock`.

Author(s)

Kaspar Rufibach, <`kaspar.rufibach@gmail.com`>,
`http://www.kasparrufibach.ch`

Guenther Walther, <`gwalther@stanford.edu`>,
`www-stat.stanford.edu/~gwalther`

References

Minimal intervals were first introduced (although for a different multiscale procedure) on p. 517 in

Lutz Dümbgen (2002). Application of Local Rank Tests to Nonparametric Regression. *Journal of Nonparametric Statistics*, **14**, 511–537.

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

`modeHunting`	*Multiscale analysis of a density on all possible intervals*

Description

Simultanous confidence statements for the existence and location of local increases and decreases of a density f, computed on all intervals spanned by two observations.

Usage

```
modeHunting(X.raw, lower = -Inf, upper = Inf, crit.vals, min.int = FALSE)
```

Arguments

`X.raw`	Vector of observations.
`lower`	Lower support point of f, if known.
`upper`	Upper support point of f, if known.
`crit.vals`	2-dimensional vector giving the critical values for the desired level.
`min.int`	If `min.int = TRUE`, the set of minimal intervals is output, otherwise all intervals with a test statistic above the critical value are given.

Details

In general, the methods `modeHunting`, `modeHuntingApprox`, and `modeHuntingBlock` compute for a given level $\alpha \in (0,1)$ and the corresponding critical value $c_{jk}(\alpha)$ two sets of intervals

$$\mathcal{D}^{\pm}(\alpha) = \Big\{\mathcal{I}_{jk} \ : \ \pm T_{jk}(\mathbf{X}) > c_{jk}(\alpha)\Big\}$$

where $\mathcal{I}_{jk} := (X_{(j)}, X_{(k)})$ for $0 \le j < k \le n+1, k-j > 1$ and c_{jk} are appropriate critical values.

Specifically, the function `modeHunting` computes $\mathcal{D}^{\pm}(\alpha)$ based on the two test statistics

$$T_n^{+}(\mathbf{X}, \mathcal{I}) = \max_{(j,k)\in\mathcal{I}} \Big(|T_{jk}(\mathbf{X})|/\sigma_{jk} - \Gamma\Big(\frac{k-j}{n+2}\Big)\Big)$$

and

$$T_n(\mathbf{X}, \mathcal{I}) = \max_{(j,k)\in\mathcal{I}} (|T_{jk}(\mathbf{X})|/\sigma_{jk}),$$

using the set $\mathcal{I} := \mathcal{I}_{all}$ of all intervals spanned by two observations $(X_{(j)}, X_{(k)})$:

$$\mathcal{I}_{all} = \Big\{(j,\ k) \ : \ 0 \le j < k \le n+1,\ k-j > 1\Big\}.$$

We introduced the local test statistics

$$T_{jk}(\mathbf{X}) := \sum_{i=j+1}^{k-1} (2X_{(i;j,k)} - 1)1\{X_{(i;j,k)} \in (0,1)\},$$

for local order statistics

$$X_{(i;j,k)} := \frac{X_{(i)} - X_{(j)}}{X_{(k)} - X_{(j)}},$$

the standard deviation $\sigma_{jk} := \sqrt{(k-j-1)/3}$ and the additive correction term $\Gamma(\delta) := \sqrt{2\log(e/\delta)}$ for $\delta > 0$.

If `min.int` = `TRUE`, the set $\mathcal{D}^{\pm}(\alpha)$ is replaced by the set $\mathbf{D}^{\pm}(\alpha)$ of its *minimal elements*. An interval $J \in \mathcal{D}^{\pm}(\alpha)$ is called *minimal* if $\mathcal{D}^{\pm}(\alpha)$ contains no proper subset of J. This *minimization* post-processing step typically massively reduces the number of intervals. If we are mainly interested in locating the ranges of increases and decreases of f as precisely as possible, the intervals in $\mathcal{D}^{\pm}(\alpha) \setminus \mathbf{D}^{\pm}(\alpha)$ do not contain relevant information.

Value

`Dp`	The set $\mathcal{D}^{+}(\alpha)$ (or $\mathbf{D}^{+}(\alpha)$), based on the test statistic with additive correction Γ.
`Dm`	The set $\mathcal{D}^{-}(\alpha)$ (or $\mathbf{D}^{-}(\alpha)$), based on the test statistic with Γ.
`Dp.noadd`	The set $\mathcal{D}^{+}(\alpha)$ (or $\mathbf{D}^{+}(\alpha)$), based on the test statistic without Γ.
`Dm.noadd`	The set $\mathcal{D}^{+}(\alpha)$ (or $\mathbf{D}^{-}(\alpha)$), based on the test statistic without Γ.

Note

Critical values for `modeHunting` and some combinations of n and α are provided in the data set `cvModeAll`. Critical values for other values of n and α can be generated using `criticalValuesAll`.

Parts of this function were derived from MatLab code provided on Lutz Duembgen's webpage, `http://www.staff.unibe.ch/duembgen`.

Author(s)

Kaspar Rufibach, <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

Guenther Walther, <gwalther@stanford.edu>,
www-stat.stanford.edu/~gwalther

References

Duembgen, L. and Walther, G. (2008). Multiscale Inference about a density. *Ann. Statist.*, **36**, 1758–1785.

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

See Also

modeHuntingApprox, modeHuntingBlock, and cvModeAll.

Examples

```
## for examples type
help("mode hunting")
## and check the examples there
```

---

modeHuntingApprox — *Multiscale analysis of a density on the approximating set of intervals*

Description

Simultanous confidence statements for the existence and location of local increases and decreases of a density f, computed on the approximating set of intervals.

Usage

```
modeHuntingApprox(X.raw, lower = -Inf, upper = Inf,
    d0 = 2, m0 = 10, fm = 2, crit.vals, min.int = FALSE)
```

Arguments

`X.raw`	Vector of observations.
`lower`	Lower support point of f, if known.
`upper`	Upper support point of f, if known.
`d0`	Initial parameter for the grid resolution.
`m0`	Initial parameter for the number of observations in one block.
`fm`	Factor by which m is increased from block to block.
`crit.vals`	2-dimensional vector giving the critical values for the desired level.
`min.int`	If `min.int = TRUE`, the set of minimal intervals is output, otherwise all intervals with a test statistic above the critical value are given.

Details

See `blocks` for details how $\mathcal{I}_{app}$ is generated and `modeHunting` for a proper introduction to the notation used here. The function `modeHuntingApprox` computes $\mathcal{D}^{\pm}(\alpha)$ based on the two test statistics $T_n^+(\mathbf{X}, \mathcal{I}_{app})$ and $T_n(\mathbf{X}, \mathcal{I}_{app})$.

If `min.int = TRUE`, the set $\mathcal{D}^{\pm}(\alpha)$ is replaced by the set $\mathbf{D}^{\pm}(\alpha)$ of its *minimal elements*. An interval $J \in \mathcal{D}^{\pm}(\alpha)$ is called *minimal* if $\mathcal{D}^{\pm}(\alpha)$ contains no proper subset of J. This *minimization* post-processing step typically massively reduces the number of intervals. If we are mainly interested in locating the ranges of increases and decreases of f as precisely as possible, the intervals in $\mathcal{D}^{\pm}(\alpha) \setminus \mathbf{D}^{\pm}(\alpha)$ do not contain relevant information.

Value

`Dp`	The set $\mathcal{D}^+(\alpha)$ (or $\mathbf{D}^+(\alpha)$), based on the test statistic with additive correction Γ.
`Dm`	The set $\mathcal{D}^-(\alpha)$ (or $\mathbf{D}^-(\alpha)$), based on the test statistic with Γ.
`Dp.noadd`	The set $\mathcal{D}^+(\alpha)$ (or $\mathbf{D}^+(\alpha)$), based on the test statistic without Γ.
`Dm.noadd`	The set $\mathcal{D}^+(\alpha)$ (or $\mathbf{D}^-(\alpha)$), based on the test statistic without Γ.

Note

Critical values for `modeHuntingApprox` and some combinations of n and α are provided in the data set `cvModeApprox`. Critical values for other values of n and α can be generated using `criticalValuesApprox`.

Author(s)

Kaspar Rufibach, <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

Guenther Walther, <gwalther@stanford.edu>,
www-stat.stanford.edu/~gwalther

References

Duembgen, L. and Walther, G. (2008). Multiscale Inference about a density. *Ann. Statist.*, **36**, 1758–1785.

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

See Also

`modeHunting`, `modeHuntingBlock`, and `cvModeApprox`.

Examples

```
## for examples type
help("mode hunting")
## and check the examples there
```

`modeHuntingBlock` *Multiscale analysis of a density via block procedure*

Description

Simultanous confidence statements for the existence and location of local increases and decreases of a density f, computed via the block procedure.

Usage

```
modeHuntingBlock(X.raw, lower = -Inf, upper = Inf, d0 = 2,
    m0 = 10, fm = 2, crit.vals, min.int = FALSE)
```

Arguments

`X.raw`	Vector of observations.
`lower`	Lower support point of f, if known.
`upper`	Upper support point of f, if known.
`d0`	Initial parameter for the grid resolution.
`m0`	Initial parameter for the number of observations in one block.
`fm`	Factor by which m is increased from block to block.
`crit.vals`	2-dimensional vector giving the critical values for the desired level.
`min.int`	If `min.int = TRUE`, the set of minimal intervals is output, otherwise all intervals with a test statistic above the critical value (in their respective block) are given.

Details

See `blocks` for details how $\mathcal{I}_{app}$ is generated and `modeHunting` for a proper introduction to the notation used here. The function `modeHuntingBlock` uses the test statistic $T_n^+(\mathbf{X}, \mathcal{B}_r)$, where $\mathcal{B}_r$ contains all intervals of Block r, $r = 1, \ldots, \#blocks$. Critical values for each block individually are received via finding an $\tilde{\alpha}$ such that

$$P(B_n(\mathbf{X}) > q_{r,\tilde{\alpha}/(r+tail)^\gamma}\ for\ at\ least\ one\ r) \leq \alpha,$$

where $q_{r,\alpha}$ is the $(1-\alpha)$–quantile of the distribution of $T_n^+(\mathbf{X}, \mathcal{B}_r)$. We then define the sets $\mathcal{D}^\pm(\alpha)$ as

$$\mathcal{D}^\pm(\alpha) := \Big\{\mathcal{I}_{jk}\ :\ \pm T_{jk}(\mathbf{X}) > q_{r,\tilde{\alpha}/(r+tail)^\gamma}\ ,\ r = 1, \ldots \#blocks\Big\}.$$

Note that γ and $tail$ are automatically determined by $crit.vals$.

If `min.int = TRUE`, the set $\mathcal{D}^\pm(\alpha)$ is replaced by the set $\mathbf{D}^\pm(\alpha)$ of its *minimal elements*. An interval $J \in \mathcal{D}^\pm(\alpha)$ is called *minimal* if $\mathcal{D}^\pm(\alpha)$ contains no proper subset of J. This *minimization* post-processing step typically massively reduces the number of intervals. If we are mainly interested in locating the ranges of increases and decreases of f as precisely as possible, the intervals in $\mathcal{D}^\pm(\alpha) \setminus \mathbf{D}^\pm(\alpha)$ do not contain relevant information.

Value

`Dp`	The set $\mathcal{D}^+(\alpha)$ (or $\mathbf{D}^+(\alpha)$).
`Dm`	The set $\mathcal{D}^-(\alpha)$ (or $\mathbf{D}^-(\alpha)$).

Note

Critical values for some combinations of n and α are provided in the data sets `cvModeBlock`. Critical values for other values of n and α can be generated using `criticalValuesApprox`.

Author(s)

Kaspar Rufibach, <`kaspar.rufibach@gmail.com`>,
`http://www.kasparrufibach.ch`

Guenther Walther, <`gwalther@stanford.edu`>,
`www-stat.stanford.edu/~gwalther`

References

Duembgen, L. and Walther, G. (2008). Multiscale Inference about a density. *Ann. Statist.*, **36**, 1758–1785.

Rufibach, K. and Walther, G. (2010). A general criterion for multiscale inference. *J. Comput. Graph. Statist.*, **19**, 175–190.

See Also

`modeHunting`, `modeHuntingApprox`, and `cvModeBlock`.

Examples

```
## for examples type
help("mode hunting")
## and check the examples there
```

`myRound` *Round 5 up to the next higher integer*

Description

The built-in R function `round` rounds a 5 to the even digit. Instead, we preferred the more intuitive rounding meaning that a 5 is always rounded to the next higher digit.

Usage

```
myRound(d)
```

Arguments

`d`	Real number.

Value

The biggest integer not bigger than d if $d - \lfloor d \rfloor < 0.5$ and the smallest integer greater than d if $d - \lfloor d \rfloor \geq 0.5$.

Author(s)

Kaspar Rufibach, <`kaspar.rufibach@gmail.com`>,
`http://www.kasparrufibach.ch`

Guenther Walther, <`gwalther@stanford.edu`>,
`www-stat.stanford.edu/~gwalther`

See Also

The built-in R function `round`.

Examples

```
x <- c(1.5, 2.5)

## built in R function
round(x)
## [1] 2 2

## this function
myRound(x)
## [1] 2 3
```

`preProcessX` *Prepare data vector according to available information on support endpoints of f*

Description

Preprocesses the initial data vector X.raw according to whether the upper and/or lower endpoint of the support of f is known.

Usage

```
preProcessX(X.raw, lower = -Inf, upper = Inf)
```

Arguments

`X.raw`	Vector of observations.
`lower`	Lower support point of f, if known.
`upper`	Upper support point of f, if known.

Details

Depending whether $lower$ and $upper$ are known, the vector of raw observations $X.raw$ is supplemented by $lower$ and/or $upper$ and finally sorted.

Value

Sorted vector of (processed) observations.

Note

This function is called by `modeHunting`, `modeHuntingApprox`, and `modeHuntingBlock`.

This function was derived from MatLab code provided on Lutz Duembgen's webpage, `http://www.staff.unibe.ch/duembgen`.

Author(s)

Kaspar Rufibach, <`kaspar.rufibach@gmail.com`>,
`http://www.kasparrufibach.ch`

Guenther Walther, <`gwalther@stanford.edu`>,
`www-stat.stanford.edu/~gwalther`

Index