Package 'logcondens.mode'

February 20, 2015

Type Package

Title Compute MLE of Log-Concave Density on R with Fixed Mode, and Perform Inference for the Mode.

Version 1.0.1

Date 2013-07-10

Author Charles Doss

Maintainer Charles Doss <cdoss@stat.washington.edu>

Description Computes maximum likelihood estimate of a log-concave density with fixed and known location of the mode. Performs inference about the mode via a likelihood ratio test. Extension of the logcondens package.

License GPL (>= 2)

Depends logcondens

Imports distr

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2013-09-12 18:31:44

R topics documented:

`logcondens.mode-package`

Computation of Log-Concave Densities on R with fixed mode and Inference for the Mode.

Description

Extension of the logcondens package. Computes maximum likelihood estimate of a log-concave density with fixed and known location of the mode. Performs inference about the mode via a likelihood ratio test comparing the unconstrained log-concave estimator to the constrained one.

Details

Package:	logcondens.mode
Type:	Package
Version:	1.0
Date:	2013-05-24
License:	GPL (>= 2)
Depends:	logcondens
Imports:	distr
Suggests:	distr
LazyLoad:	yes
Packaged:	2013-07-10 07:59:40 UTC; cdoss
Built:	R 2.15.2; ; 2013-07-10 07:59:41 UTC; unix

Index:

intF	Computes the Integral of the estimated CDF at Arbitrary Real Numbers in s
intFfn	Computes the Integral of a log-concave CDF at Arbitrary Real Numbers
logConDens	Compute log-concave density estimator and related quantities

The main functions of this package are LCLRCImode, LRmodeTest, and activeSetLogCon.mode. The latter computes a log-concave density estimate with known and fixed location of the mode. In addition to being of interest on its own, this estimator is of interest for likelihood ratio tests for the mode. LRmodeTest runs this test by using activeSetLogCon and activeSetLogCon.mode to compute the likelihood ratio statistic and LCTLLRdistn to compute the quantiles. LCLRCImode inverts the test to form confidence intervals.

Author(s)

Charles Doss

Maintainer: Charles R. Doss, <cdoss@stat.washington.edu>,
http://www.stat.washington.edu/people/cdoss/

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log-concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Doss, C. R. (2013). Shape-Constrained Inference for Concave-Transformed Densities and their Modes. PhD thesis, Department of Statistics, University of Washington, in preparation.

Doss, C. R. and Wellner, J. A. (2013). Inference for the mode of a log-concave density. Technical Report, University of Washington, in preparation.

See Also

See the package logcondens, from which this package derives much of its code.

Examples

```
nn <- 200
myxx <- rnorm(nn) ## no need to sort
TRUEMODE <- 0
```

```
res.MC <- activeSetLogCon.mode(myxx, mode=TRUEMODE)
LRmodeTest(mode=TRUEMODE, x=myxx, xgrid=.05, alpha=.05)
CI <- LCLRCImode(x=myxx, alpha=0.05)
print(CI[1] <= TRUEMODE && TRUEMODE <= CI[2]) ## approx 95% coverage probability
```

`activeSetLogCon` *Computes a Log-Concave Probability Density Estimate via an Active Set Algorithm*

Description

Given a vector of observations $\boldsymbol{x}_n = (x_1, \ldots, x_n)$ with not necessarily equal entries, `activeSetLogCon` first computes vectors $\boldsymbol{x}_m = (x_1, \ldots, x_m)$ and $\boldsymbol{w} = (w_1, \ldots, w_m)$ where w_i is the weight of each x_i s.t. $\sum_{i=1}^m w_i = 1$. Then, `activeSetLogCon` computes a concave, piecewise linear function $\widehat{\phi}_m$ on $[x_1, x_m]$ with knots only in $\{x_1, \ldots, x_m\}$ such that

$$L(\phi) = \sum_{i=1}^{m} w_i \phi(x_i) - \int_{-\infty}^{\infty} \exp(\phi(t)) dt$$

is maximal. To accomplish this, an active set algorithm is used.

This function is as it is in the logcondens package except we've added the 'prec' variable as an argument and modified the the values returned as output, to be in line with the `activeSetLogCon.mode` function.

Usage

```
activeSetLogCon(x, xgrid = NULL, print = FALSE, w = NA,
prec=10^-10)
```

Arguments

`x`	Vector of independent and identically distributed numbers, not necessarily unique.
`xgrid`	Governs the generation of weights for observations. See `preProcess` for details.
`print`	`print = TRUE` outputs the log-likelihood in every loop, `print = FALSE` does not. Make sure to tell `R` to output (press CTRL+W).
`w`	Optional vector of weights. If weights are provided, i.e., if `w != NA`, then `xgrid` is ignored.
`prec`	Governs precision of various subfunctions, e.g. the Newton-Raphson procedure.

Value

`xn`	Vector with initial observations $x_1, \ldots, x_n$.
`x`	Vector of observations $x_1, \ldots, x_m$ that was used to estimate the density, i.e.\ points that include all possible knots of the estimate. Note that this x is not identical to the x passed in (`xn` is identical).
`w`	The vector of weights that had been used. Depends on the chosen setting for `xgrid`. Of the same length as x.
`L`	The value $L(\widehat{\phi}_m)$ of the log-likelihood-function L at the maximum $\widehat{\phi}_m$.
`IsKnot`	Vector with entries $\text{IsKnot}_i = 1\{\widehat{\phi}_m \text{ has a kink at } x_i\}$.
`knots`	`knots` equals x[IsKnot>0], gives the values of the points that are knots.
`phi`	Vector with entries $\widehat{\phi}_m(x_i)$, $i = 1, \ldots, m$. Named "phi" not "phihat" for backwards compatibility.
`fhat`	Vector with entries $\widehat{f}_m(x_i) = e^{\widehat{\phi}_m(x_i)}$, $i = 1, \ldots, m$.
`Fhat`	A vector $(\widehat{F}_{m,i})_{i=1}^m$ of the same size as $\boldsymbol{x}$ with entries $$\widehat{F}_{m,i} = \int_{x_1}^{x_i} \exp(\widehat{\phi}_m(t))dt.$$
`H`	Numeric vector $(H_1, \ldots, H_m)'$ where H_i is the derivative of $$t \to L(\phi + t\Delta_i)$$ at zero and $\Delta_i(x) = \min(x - x_i, 0)$
`n`	Number of initial observations.
`m`	Number of points used to compute the estimator, either unique observations or output from `preProcess`.
`mode`	Mode of the estimated density $\hat{f}_m$. This is redundant with `dlcMode`, but is included for backwards compatibility with the logcondens package.
`dlcMode`	A list, of class "dlc.mode", with components `$val`, `$idx`, and `$isx`. `dlcMode$val` gives the mode estimate value, `dlcMode$idx` gives the corresponding index in `x`. `dlcMode$isx` is always `TRUE`. (`dlMode$isx` is sometimes `FALSE` when a `dlc.mode` object is output from `activeSetLogCon.mode`.)
`sig`	The standard deviation of the initial sample $x_1, \ldots, x_n$.
`phi.f`	All outputs named "name.f" are functions corresponding to name. So, `phi.f(x)` equals $\widehat{\phi}_m(x)$.
`fhat.f`	Is a function such that `fhat.f(x)` equals $\widehat{f}_m(x)$.
`Fhat.f`	Is a function such that `Fhat.f(x)` equals $\widehat{F}_m(x)$.
`E.f`	`E.f(l,u)` $= \int_l^u \widehat{F}_m(t)dt$
`phiPL`	Numeric vector of length m with values $\widehat{\phi}'_m(x_i-)$
`phiPR`	Numeric vector of length m with values $\widehat{\phi}'_m(x_i+)$
`phiPL.f`	Is a function such that `phiPL.f(x)` equals $\widehat{\phi}'_m(x-)$.
`phiPR.f`	Is a function such that `phiPR.f(x)` equals $\widehat{\phi}'_m(x+)$.

Author(s)

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

Lutz Duembgen, <duembgen@stat.unibe.ch>,
http://www.staff.unibe.ch/duembgen

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log–concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

See Also

activeSetLogCon can be used to estimate a log-concave density. However, to generate an object of class dlc that allows application of summary and plot we recommend to use logConDens.

The following functions are used by activeSetLogCon:

J00, J10, J11, J20, Local_LL, Local_LL_all, LocalCoarsen, LocalConvexity, LocalExtend, LocalF, LocalMLE, LocalNormalize, MLE

Log concave density estimation via an iterative convex minorant algorithm can be performed using icmaLogCon.

Examples

```
## estimate gamma density
set.seed(1977)
n <- 200
x <- rgamma(n, 2, 1)
res <- activeSetLogCon(x, w = rep(1 / n, n), print = FALSE)

## plot resulting functions
par(mfrow = c(2, 2), mar = c(3, 2, 1, 2))
plot(res$x, exp(res$phi), type = 'l'); rug(x)
plot(res$x, res$phi, type = 'l'); rug(x)
plot(res$x, res$Fhat, type = 'l'); rug(x)
plot(res$x, res$H, type = 'l'); rug(x)

## compute and plot function values at an arbitrary point
x0 <- (res$x[100] + res$x[101]) / 2
Fx0 <- evaluateLogConDens(x0, res, which = 3)[, "CDF"]
plot(res$x, res$Fhat, type = 'l'); rug(res$x)
abline(v = x0, lty = 3); abline(h = Fx0, lty = 3)
```

```
## compute and plot 0.9-quantile of Fhat
q <- quantilesLogConDens(0.9, res)[2]
plot(res$x, res$Fhat, type = 'l'); rug(res$x)
abline(h = 0.9, lty = 3); abline(v = q, lty = 3)
```

`activeSetLogCon.mode` *Computes the Modally-Constrained Log-Concave Probability Density Maximum Likelihood Estimate via an Active Set Algorithm.*

Description

This is an adapted version of activeSetLogCon from the logcondens package for computing the MLE of a log-concave density with known location of mode.

Given a vector of observations $\boldsymbol{x}_n = (x_1, \ldots, x_n)$ with potentially distinct or nondistinct entries, `activeSetLogCon.mode` first computes vectors $\boldsymbol{x}_m = (x_1, \ldots, x_m)$ and $\boldsymbol{w} = (w_1, \ldots, w_m)$ where w_i is the weight of each x_i s.t. $\sum_{i=1}^{m} w_i = 1$. The vector $\boldsymbol{x}_m$ contains the fixed location of the mode, `mode`. Then, `activeSetLogCon.mode` computes a concave, piecewise linear function $\widehat{\phi}_m^0$ on $[x_1, x_m]$ with p knots only in $\{x_1, \ldots, x_m\}$ and with mode value, `mode`, such that

$$L(\phi) = \sum_{i=1}^{m} w_i \phi(x_i) - \int_{-\infty}^{\infty} \exp(\phi(t)) dt$$

is maximal. To accomplish this, an active set algorithm is used.

Usage

```
activeSetLogCon.mode(x, xgrid = NULL, mode=x[1], print = FALSE, w
= NA, prec=10^-10)
```

Arguments

`x`	Vector of independent and identically distributed numbers, not necessarily unique.
`xgrid`	Governs the generation of weights for observations. See `preProcess` for details.
`mode`	This is the constrained value for the location of the mode.
`print`	`print = TRUE` outputs the log-likelihood in every loop, `print = FALSE` does not. Make sure to tell `R` to output (press CTRL+W).
`w`	Optional vector of weights. If weights are provided, i.e., if `w != NA`, then `xgrid` is ignored.
`prec`	Governs precision of various subfunctions, e.g., the Newton-Raphson procedure.

Value

`xn` Vector with initial observations $x_1, \ldots, x_n$.

`x` Vector of observations $x_1, \ldots, x_m$ that was used to estimate the density, i.e., points that include all possible knots of the estimate. Note that $\boldsymbol{x}$ always includes the mode value `mode`, since that point is a possible knot! Note also that this x is not identical to the x passed in (`xn` is identical). This vector is referred to as 'z' in Doss (2013).

`w` The vector of weights that had been used. Depends on the chosen setting for `xgrid`. Of the same length as x. The weight corresponding to the mode will be 0 if the mode is not a data point, and otherwise will be nonzero.

`L` The value $L(\widehat{\boldsymbol{\phi}}_m^0)$ of the log-likelihood-function L at the maximum $\widehat{\boldsymbol{\phi}}_m^0$.

`MI` Numeric vector of length 2 giving the endpoints of the modal interval.

`IsKnot` Vector with entries IsKnot$_i = 1\{\widehat{\phi}_m^0$ has a kink at $x_i\}$.

`IsMIC` Analogous to IsKnot; stands for "Is Modally Inactive Constraint," i.e., denotes whether the modal constraints are active or inactive. It is a numeric vector of length 2, corresponding to whether the mode is a left-knot or a right-knot. Just as with IsKnot, a 1 denotes an inactive constraint and a 0 denotes an active one. Thus a 0 indicates that the constraint that the estimate be equal in value at the mode and the nearest knot to the left or to the right, respectively, is active. Note also that if max(IsMIC)==1 then the corresponding index in IsKnot is a 1 (i.e., IsKnot[dlcMode$idx] ==1).

`constr` knots[constr] is equal to MI; that is, constr is a numeric (integral) vector of length two with values in $1, \ldots, p$ indicating which of the p knots are the left and right of the modal interval.

`knots` `knots` equals x[IsKnot>0], gives the values of the points that are knots.

`phi` Vector with entries $\widehat{\phi}_m(x_i)$, $i = 1, \ldots, m$. Named "phi" not "phihat" for backwards compatibility.

`fhat` Vector with entries $\widehat{f}_m^0(x_i) = e^{\widehat{\phi}_m^0(x_i)}$, $i = 1, \ldots, m$.

`Fhat` A vector $(\widehat{F}_{m,i}^0)_{i=1}^m$ of the same size as $\boldsymbol{x}$ with entries

$$\widehat{F}_{m,i}^0 = \int_{x_1}^{x_i} \exp(\widehat{\phi}_m^0(t))dt.$$

`H` Numeric vector $(H_1, \ldots, H_m)'$ where H_i is the derivative of

$$t \to L(\phi + t\Delta_i)$$

at zero and $\Delta_i(x) = \min(x - x_i, 0)$ if x_i is less than `dlcMode$val` or $\Delta_i(x) = \min(x_i - x, 0)$ if x_i is greater than dlcMode$val. If x_i is the mode (i.e., equals dlcMode$val) H_i, is set to 0. The corresponding values for the mode are accessed via H.m.

Note that in the unconstrained problems the derivatives in the directions $\min(x_i - x, 0)$ and $\min(x - x_i, 0)$ are equal, but in the constrained problem these derivatives are not equal.

`H.m` Vector $(H.m_1, H.m_2)'$ where $H.m_1$ is the derivative of

$$t \to L(\phi + t\Delta_i)$$

at zero and $\Delta_1(x) = \min(x - a, 0)$ and $\Delta_2(x) = \min(a - x, 0)$, where a is the mode.

`n` Number of initial observations, i.e., length of `xn`.

`m1` Number of unique observations. This count excludes the mode if the mode is not a data point (or if `xgrid` is not `NULL` then excludes the mode if it is not in the output of `preProcess`).

`m` Number of points used to compute the estimator, i.e., unique observations as well as the mode, i.e., length of `x`. So is either $m_1 + 1$ or m_1 depending on whether `dlcMode$isx` is FALSE or TRUE, respectively.

`dlcMode` A list, of class "dlc.mode", with components `$val`, `$idx`, and `$isx`. `dlcMode$val` gives the constrained mode value, `dlcMode$idx` gives the corresponding index in `x`, and `dlcMode$isx` is `TRUE` or `FALSE` depending on whether the value is or is not equal to an element of the vector `preProcess(x, xgrid)$x` (where `x` is the argument passed in, not the value returned).

Note, when the mode is not an `x` value, `w[dlcMode$idx] == 0`. This can often be used in place of an explicit check via `$isx` as to whether the mode is or is not an `x` value.

`sig` The standard deviation of the initial sample $x_1, \ldots, x_n$.

`phi.f` All outputs named "name.f" are functions corresponding to name. So, `phi.f(x)` equals $\widehat{\phi}_m^0(x)$.

`fhat.f` Is a function such that `fhat.f(x)` equals $\widehat{f}_m^0(x)$.

`Fhat.f` Is a function such that `Fhat.f(x)` equals $\widehat{F}_m^0(x)$.

`EL.f` `EL.f(l,u)` $= \int_l^u \widehat{F}_m^0(t)dt$

Note that this is not analogous to `H` or `H.m`, which are derivatives of the log likelihood and so have subtracted an integral of the empirical cdf.

`ER.f` `ER.f(l,u)` $= \int_l^u (1 - \widehat{F}_m^0(t))dt$

`E.f` Equals `EL.f`. Included so as to be compatible (i.e., follow inheritance principles) with `activeSetLogCon`, which returns an `E.f` variable.

`phiPL` Numeric vector of length m with values $(\widehat{\phi}_m^0)'(x_i-)$

`phiPR` Numeric vector of length m with values $(\widehat{\phi}_m^0)'(x_i+)$

`phiPL.f` Is a function such that `phiPL.f(x)` equals $(\widehat{\phi}_m^0)'(x-)$.

`phiPR.f` Is a function such that `phiPR.f(x)` equals $(\widehat{\phi}_m^0)'(x+)$.

Note

Adapted from `activeSetLogCon` in the package `logcondens`.

Author(s)

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

Lutz Duembgen, <duembgen@stat.unibe.ch>,
http://www.staff.unibe.ch/duembgen

Charles R. Doss, <cdoss@stat.washington.edu>,
http://www.stat.washington.edu/people/cdoss/

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log-concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Doss, C. R. (2013). Shape-Constrained Inference for Concave-Transformed Densities and their Modes. PhD thesis, Department of Statistics, University of Washington, in preparation.

Doss, C. R. and Wellner, J. A. (2013). Inference for the mode of a log-concave density. Technical Report, University of Washington, in preparation.

See Also

The following functions are used by activeSetLogCon.mode:

J00, J10, J11, J20, Local_LL.mode, LocalLLall.mode, LocalCoarsen.mode, LocalConvexity.mode, LocalExtend, LocalF, LocalMLE.mode, LocalNormalize, MLE.mode

logConDens (or activeSetLogCon) can be used to estimate an unconstrained log-concave density.

Examples

```
## estimate gamma density

set.seed(1977)
n <- 200
x <- rgamma(n, 2, 1)
TRUEMODE <- 1; ## (2-1)*1
res <- activeSetLogCon.mode(x, mode=TRUEMODE, w = rep(1 / n, n), print = FALSE)

## plot resulting functions
par(mfrow = c(2, 2), mar = c(3, 2, 1, 2))
plot(res$x, res$fhat, type = 'l'); rug(res$xn)
plot(res$x, res$phi, type = 'l'); rug(res$xn)
plot(res$x, res$Fhat, type = 'l'); rug(res$xn)
plot(res$x, res$H, type = 'l'); rug(res$xn)
```

```
## Or can use the ".f" functions
xpts <- seq(from=0, to=9, by=.01)
par(mfrow = c(2, 2), mar = c(3, 2, 1, 2))
plot(xpts, res$fhat.f(xpts), type = 'l'); rug(res$xn)
plot(xpts, res$phi.f(xpts), type = 'l'); rug(res$xn)
## these are not analogous to res$H.
plot(xpts, res$EL.f(upper=xpts), type = 'l'); rug(res$xn)
plot(xpts, res$ER.f(lower=xpts), type = 'l'); rug(res$xn)


## compute and plot function values at an arbitrary point
x0 <- (res$x[100] + res$x[101]) / 2
Fx0 <- evaluateLogConDens(x0, res, which = 3)[, "CDF"]
plot(res$x, res$Fhat, type = 'l'); rug(res$x)
abline(v = x0, lty = 3); abline(h = Fx0, lty = 3)

## compute and plot 0.9-quantile of Fhat
alpha <- .1
q <- quantilesLogConDens(1-alpha, res)[2]
plot(res$x, res$Fhat, type = 'l'); rug(res$x)
abline(h = 1-alpha, lty = 3); abline(v = q, lty = 3)
```

`activeSetRoutines.mode.Rd`

Auxiliary Numerical Routine for the Function activeSetLogCon.mode

---

Description

Function used by activeSetLogCon.mode.

Usage

```
LocalExtend(x, IsKnot, x2, phi2, constr=NULL)
```

Arguments

`x`	Vector of independent and identically distributed numbers, with strictly increasing entries.
`IsKnot`	Vector with entries $\text{IsKnot}_i = 1\{\phi \text{ has a kink at } x_i\}$.
`x2`	Vector of same type as $\boldsymbol{x}$.
`phi2`	Vector of same type as $\boldsymbol{\phi}$.
`constr`	If doing unconstrained optimization then constr can be of length less or equal to 1 (e.g. `NULL`) or have two identical entries. If modally constrained optimization, then `constr` is a numeric vector of length two containing two integer indices for x2 corresponding to the knots on either side of the mode.

Author(s)

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

Lutz Duembgen, <duembgen@stat.unibe.ch>,
http://www.staff.unibe.ch/duembgen

Charles Doss, <cdoss@stat.washington.edu>,
www.stat.washington.edu/people/cdoss

See Also

Used by activeSetLogCon.mode to estimate a log-concave probability density with known location of mode.

---

dir.exists *Utility for checking existence of a directory.*

Description

Utility using a system interface to check whether a directory exists.

Usage

```
dir.exists(path)
```

Arguments

path	Character string.

Details

Uses system function to test if the directory specified by path exists.

Value

Returns TRUE or FALSE.

Author(s)

Charles Doss <cdoss@stat.washington.edu>,
http://www.stat.washington.edu/people/cdoss/

Examples

```
dir.exists("/") ## true (on *nix)
dir.exists("~/") ## probably true
```

`estimateLRdistn` *Estimate "the" limiting distribution of the likelihood ratio statistic for location of mode.*

Description

Sampling from a given distribution, we estimate via Monte Carlo the limiting distribution of 2-log-likelihood-ratio of the modally-constrained log-concave MLE to the (unconstrained) log-concave MLE.

Usage

```
estimateLRdistn(rdist = rnorm, mode = 0, N.MC = 1e2, n.SS = 10000,
xgrid=NULL, prec = 10^-10, seedVal = NULL, debugging = NULL)
```

Arguments

`rdist`	A function taking an integer argument `n` and returning `n` values simulated from a distribution. The distribution is generally log-concave (otherwise we are in a misspecified setting).
`mode`	fixed/known location of mode for constrained estimator.
`N.MC`	Number of Monte Carlo simulations to do for the limiting distribution.
`n.SS`	Sample Size used for each Monte Carlo. (Each MC simulates `n.SS` values from rdist and computes constrained and unconstrained MLE).
`xgrid`	Governs the generation of weights for observations. `NULL` then data are used as they are. Otherwise can be a single numeric of a numeric vector of length `n.SS`. Please see `preProcess` for details.
`prec`	Precision variable
`seedVal`	An optional seed value
`debugging`	Turns off/on debugging. Any non-character value turns debugging off. If debugging is a character string, then this string gives the name of an output file to which `myxx` (the simulated data from rdist), `myxx.uniq` (the corresponding unique values), and `rdist` are saved. If the code crashes, this can be examined. If debugging is on (i.e., is a character) then if `TLLRs[i]` is less than 0, the value of `myxx` will be saved to a file with name given by paste(debugging, "tmpxxs", i, ".rsav", sep=""), along with corresponding weights `myww` and the `mode` passed in.

Details

Computes an estimate of the asymptotic distribution of the likelihood ratio statistic $2(\log \hat{f}_n - \log \hat{f}_n^0)$ under the assumption that the true log-concave density f_0 satisfies $f_0''(m) < 0$ where m is the true

mode of f_0. The estimate is computed based on a sample of size `n.SS` from `rdist` via `N.MC` Monte Carlo iterations.

Note: the object `LCTLLRdistn` was created by output from this function with `n.SS` set to 1.2e3 and `N.MC` set to 1e4. Thus, `estimateLRdistn` is _NOT_ needed to simply compute fairly accurate quantiles of the limit distribution of the likelihood ratio statistic. `estimateLRdistn` is more useful for research purposes. For instance, by passing to `mode` values that are not the true mode of `myr`, the statistic can be studied under the alternative hypothesis.

Value

A `list(LRs,TLLRs)`, i.e., "likelihood ratio" and "two log likelihood ratios". Both are numeric vectors of length `N.MC`.

Note that theoretically all elements of `LRs` should be nonnegative, but in practice some rounding errors can occur when `n.SS` is very large.

Author(s)

Charles Doss <cdoss@stat.washington.edu>,
http://www.stat.washington.edu/people/cdoss/

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log-concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Doss, C. R. (2013). Shape-Constrained Inference for Concave-Transformed Densities and their Modes. PhD thesis, Department of Statistics, University of Washington, in preparation.

Doss, C. R. and Wellner, J. A. (2013). Inference for the mode of a log-concave density. Technical Report, University of Washington, in preparation.

See Also

See `activeSetLogCon` and `activeSetLogCon.mode`, which compute the unconstrained and constrained MLEs, which form the likelihood ratio. The object `LCTLLRdistn` was created by output from this function.

Examples

```
myseed <- 561

{if(require(distr)){
  mydistn <- Norm() ##demonstrate use of distr package
```

```
  myr <- mydistn@r
}
else {
  myr <- rnorm
}}

hypothesis.mode <- 0
N.MC <- 100 ## should increase these values for better estimate
n.SS <- 50

LRres <- estimateLRdistn(rdist=myr, mode=hypothesis.mode, N.MC=N.MC, prec=10^-10,
               n.SS=n.SS, seedVal=myseed,
               debugging=FALSE)
TLLRs <- sort(LRres$TLLRs) ##sort is unnecessary, just for examining data
negIdcs <- TLLRs<=0; ## rounding errors
Nneg <- sum(negIdcs)
print(Nneg)
TLLRs[negIdcs] <- 0

cdf.empirical.f <- ecdf(TLLRs)
xlims <- c(min(TLLRs), max(TLLRs))
xpts <- seq(from=xlims[1], to=xlims[2], by=.001)
plot(xpts, cdf.empirical.f(xpts), type="l",
     xlab="TLLRs", ylab="Probability")


#### LCTLLRdistn used 1e4 Monte Carlos with 1.2e3 samples each Monte
####Carlo.
##lines(xpts, LCTLLRdistn@p(xpts), col="blue") ## "object
##'C_R_approxfun' not found" error on winbuilder
```

`intECDFfn` *Gives the Integrated Empirical Distribution Function*

Description

Like `intECDF`, except returns a function $\bar{I}$ instead of a value. The function $\bar{I}(l, r)$ is given by

$$\bar{I}(l, r) = \int_l^r \bar{F}(u) d\, u$$

where $\bar{F}$ is the empirical distribution function of $x_1, \ldots, x_m$. Note that l and r must lie in $[x_1, x_m]$. For an exact formula related to $\bar{I}$, see `intECDF`.

Usage

```
intECDFfn(x)
```

Arguments

x	Vector $\boldsymbol{x} = (x_1, \ldots, x_m)$ of original observations, which are used to define the empirical CDF, $\bar{F}$.

Value

The function $\bar{I}$.

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log-concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Doss, C. R. (2013). Shape-Constrained Inference for Concave-Transformed Densities and their Modes. PhD thesis, Department of Statistics, University of Washington, in preparation.

Doss, C. R. and Wellner, J. A. (2013). Inference for the mode of a log-concave density. Technical Report, University of Washington, in preparation.

See Also

See intECDF which returns values instead of a function.

Examples

```
set.seed(100)
xx <- runif(50) ## min .056, max .88

myIntECDF <- intECDFfn(xx);
evalpts <- c(.3, .5) ## lie within [ min(xx) , max(xx) ]
myIntECDF(evalpts)
## equal to
intECDF(evalpts, xx)
```

intF — *Computes the Integral of the estimated CDF at Arbitrary Real Numbers in s*

Description

Based on the output of the function activeSetLogCon, this gives values of

$$\widehat{I}(t) = \int_{x_1}^{t} \widehat{F}(r)dr$$

at all numbers in $\boldsymbol{s}$. Note that t (so all elements in $\boldsymbol{s}$) must lie in $[x_1, x_m]$. The exact formula for $\widehat{I}(t)$ is

$$\widehat{I}(t) = \Big(\sum_{i=1}^{i_0} \widehat{I}_i(x_{i+1})\Big) + \widehat{I}_{i_0}(t)$$

where $i_0 = \min\{m-1\,,\ \{i\ :\ x_i \le t\}\}$ and

$$I_j(x) = \int_{x_j}^{x} \widehat{F}(r)dr = (x-x_j)\widehat{F}(x_j) + \Delta x_{j+1}\Big(\frac{\Delta x_{j+1}}{\Delta\widehat{\varphi}_{j+1}} J\Big(\widehat{\varphi}_j, \widehat{\varphi}_{j+1}, \frac{x-x_j}{\Delta x_{j+1}}\Big) - \frac{\widehat{f}(x_j)(x-x_j)}{\Delta\widehat{\varphi}_{j+1}}\Big)$$

for $x \in [x_j, x_{j+1}]$, $j = 1, \ldots, m-1$, $\Delta v_{i+1} = v_{i+1} - v_i$ for any vector $\boldsymbol{v}$ and the function J introduced in Jfunctions.

Note that this version of intF is similar to that in the logcondens package, versions 1.3.5 and earlier. Newer versions of that package have modified arguments. Here, we have also added the argument 'prec'.

Usage

```
intF(s, x, phi, Fhat, prec=1e-10)
```

Arguments

s	Vector of real numbers where the functions should be evaluated at.
x	Vector $\boldsymbol{x} = (x_1, \ldots, x_m)$ of original observations (sorted).
phi	Vector $(\widehat{\varphi}_m(x_i))_{i=1}^m$, as computed by activeSetLogCon.
Fhat	Vector $(\widehat{F}_{m,i})_{i=1}^m$ with entries $\widehat{F}_{m,i} = \int_{x_1}^{x_i} \exp(\widehat{\varphi}_m(t))dt,$ as computed by activeSetLogCon.
prec	Governs cutoff at which an approximation for the needed integral is used.

Value

Vector of the same length as s, containing the values of $\widehat{I}$ at the elements of s.

Author(s)

Kaspar Rufibach, <kaspar.rufibach@ifspm.uzh.ch>,
http://www.biostat.uzh.ch/aboutus/people/rufibach.html

Lutz Duembgen, <duembgen@stat.unibe.ch>,
http://www.staff.unibe.ch/duembgen

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log-concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Doss, C. R. (2013). Shape-Constrained Inference for Concave-Transformed Densities and their Modes. PhD thesis, Department of Statistics, University of Washington, in preparation.

Doss, C. R. and Wellner, J. A. (2013). Inference for the mode of a log-concave density. Technical Report, University of Washington, in preparation.

See Also

This function uses the output of activeSetLogCon. The function intECDF is similar, but based on the empirical distribution function.

Examples

```
## estimate gamma density
set.seed(1977)
x <- sort(rgamma(200, 2, 1))
res <- activeSetLogCon(x, w = NA, print = FALSE)

## compute and plot the process D(t) in Duembgen and Rufibach (2009)
s <- seq(min(x), max(x), by = 10 ^ -3)
D1 <- intF(s, x, res$phi, res$Fhat)
D2 <- intECDF(s, x)
par(mfrow = c(2, 1))
plot(x, res$phi, type = 'l'); rug(x)
plot(s, D1 - D2, type = 'l'); abline(h = 0, lty = 2)
```

intFfn *Computes the Integral of a log-concave CDF at Arbitrary Real Numbers*

Description

Based on output from the function logConDens, activeSetLogCon, or activeSetLogCon.mode, this function gives a function $\widehat{I}$ given by

$$\widehat{I}(l,r) = \int_l^r \widehat{F}(u)du$$

or by

$$\widehat{I}(l,r) = \int_l^r (1-\widehat{F}(u))du$$

Note that l and r must lie in $[x_1, x_m]$. For exact formulas related to these integrals, see the intF function.

Usage

```
intFfn(x, phi, Fhat, prec = 1e-10, side = "left")
```

Arguments

x	Vector of (unique) observations from which the (modally-constrained or -unconstrained) log-concave density is estimated. This corresponds to output of preProcess, potentially with a constrained mode value inserted. Weights associated with x are not passed in since we pass in the estimated values phi and Fhat.
phi	Numeric vector of same length as x that gives the log-concave estimate's values at x.
Fhat	Numeric vector of same length as x that gives the log-concave estimate CDF's values at x.
prec	Precision argument for the intF function.
side	String taking values "left" or "right". If "left" then returns the first integral given in the description (integral of $\widehat{F}$). If "right" then returns the second integral given in the description (integral of $1-\widehat{F}$).

Value

Returns a function H. If side is "left" then the return is of type

```
function(upper, lower=rep(x[1],length(upper))).
```

If side is "right" then the return is of type

`function(lower, upper=rep(x[length(x)],length(lower)))`.

Note that the order of the arguments are changed, so that passing an unnamed numeric value or vector has a default behavior of integrating "from the outside-in".

Author(s)

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

Lutz Duembgen, <duembgen@stat.unibe.ch>,
http://www.staff.unibe.ch/duembgen

Charles Doss, <cdoss@stat.washington.edu>,
http://www.stat.washington.edu/people/cdoss/

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log-concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Doss, C. R. (2013). Shape-Constrained Inference for Concave-Transformed Densities and their Modes. PhD thesis, Department of Statistics, University of Washington, in preparation.

Doss, C. R. and Wellner, J. A. (2013). Inference for the mode of a log-concave density. Technical Report, University of Washington, in preparation.

See Also

This function uses the output of `activeSetLogCon` or `activeSetLogCon.mode`. The function `intECDFfn` is similar, but based on the empirical distribution function. The function `intF` behaves similarly but returns a vector instead of a function.

Examples

```
## estimate gamma density
set.seed(1977)
x <- rgamma(200, 2, 1)
res <- activeSetLogCon.mode(x, mode=1)

##  res$x is not equal to x
myIntF <- intFfn( res$x,  res$phi, res$Fhat, side="left")


s <- seq(min(res$x), max(res$x), by = 10 ^ -3)
```

```
D1 <- myIntF(s)
```

LCLRCImode *Compute Log-Concave Likelihood-Ratio Confidence interval for the mode.*

Description

Compute the confidence interval (CI) for the mode of a log-concave density by "inverting" the likelihood ratio statistic, i.e. the $1-\alpha$ CI is composed of mode values at which the likelihood ratio test does not reject at the α-level.

Usage

```
LCLRCImode(x, xgrid = NULL, w = NA, nn = length(x), alpha = 0.05, prec =
1e-10, CIprec = 1e-04, print = F)
```

Arguments

x	Points at which to compute the unconstrained and constrained estimators. Either iid data observations (from a log-concave density) or, such data binned. If x is binned, there should be corresponding weights w. Binning is usually handled by passing in a non-NULL value for xgrid.
xgrid	Governs binning of x and generation of corresponding weights w. See logcondens::preProcess. If w is not NA then xgrid should be NULL.
w	Numeric vector of length length(x) or NA. Weights corresponding to x. Can be NA (regardless of the value of xgrid) which indicates the weights are uniform (equal to 1/length(x)). If w is not NA then xgrid should be NULL. If nn is not equal to length(x) then w should be given a non-NA value. If w is not NA, then we assume that x has no duplicate entries.
nn	The number of data points initially observed. Numeric of length 1. Usually equal to length(x). If some sort of preProcessing is done in advance, may be not equal to length(x). To pass in a non-default value for nn (i.e. something other than length(x)), w must also be passed in a (numeric vector) value, and xgrid must be NULL.
alpha	Numeric value in [0,1], the coverage probability for the confidence interval (i.e., the level for the corresponding test).
prec	Numeric value, giving the precision passed to activeSetLogCon and to activeSetLogCon.mode.
CIprec	Numeric value giving precision for the endpoints of the confidence interval.
print	TRUE or FALSE, depending on whether debugging information should be printed or not, respectively.

Details

The confidence set is given by the values of the mode that LRmodeTest does not reject. See the details of that function.

Value

Returns a numeric vector of length 2, giving the asymptotic confidence interval for the mode location.

Author(s)

Charles R. Doss, <cdoss@stat.washington.edu>,
http://www.stat.washington.edu/people/cdoss/

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log-concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Doss, C. R. (2013). Shape-Constrained Inference for Concave-Transformed Densities and their Modes. PhD thesis, Department of Statistics, University of Washington, in preparation.

Doss, C. R. and Wellner, J. A. (2013). Inference for the mode of a log-concave density. Technical Report, University of Washington, in preparation.

See Also

See also LRmodeTest for the corresponding test.

Examples

```
nn <- 200
myxx <- rnorm(nn) ## no need to sort

LCLRCImode(x=myxx,
           xgrid=NULL,
           w=NA,
           ##nn=nn,
           alpha=0.05,
           CIprec=1e-04,
           print=FALSE)

LCLRCImode(x=myxx,
           xgrid=.05,
```

```
w=NA,
##nn=nn,
alpha=0.05,
CIprec=1e-04,
print=FALSE)
```

LCTLLRdistn *Limit Distribution of the Likelihood Ratio Statistic*

Description

The LCTLLRdistn object gives the (estimated) limit distribution of Two times the log likelihood ratio for the location of the mode of a log-concave density f_0, under the assumption that $f_0''(m) < 0$, where m is the mode of f_0.

Usage

```
LCTLLRdistn
```

Format

LCTLLRdistn is an object with formal (S4) class 'distr' and subclass 'DiscreteDistribution' [package "distr"] with 12 slots. It is an estimate of a continuous limit distribution by a discrete one.

@support Gives the (discrete) support, i.e., the simulated values on which the estimate is based.

@img Formal class 'Reals' [package "distr"] with 2 slots

- **@dimension** 1
- **@name** "Real Space"

@param NULL; unused slot.

@r function (n); simulates n values.

@d function (x, log = FALSE); constant 0 function.

@p function (q, lower.tail = TRUE, log.p = FALSE); the cumulative distribution function.

@q function (p, lower.tail = TRUE, log.p = FALSE); the quantile function.

@.withSim logi FALSE; for internal use

@.withArith logi FALSE; for internal use

@.logExact logi FALSE; for internal use

@.lowerExact logi TRUE; for internal use

@Symmetry Formal class 'NoSymmetry' [package "distr"] with 2 slots

- **@type** character "non-symmetric distribution"
- **@SymmCenter** NULL

Details

LCTLLRdistn is an object of class "distr" and subclass "DiscreteDistribution" from the package distr. The main uses are the three functions q (the quantile function), p (the cumulative distribution function) and r (which returns random samples). Note that d always returns 0 since the distribution is estimated discretely.

See the distr package for more details.

Source

Obtained via simulation from a Gamma(3,1) distribution with density proportional to x^2e^{-x} on $(0, \infty)$. We simulated the log likelihood ratio statistic 10^4 times, each time with a sample size of $1.2*10^3$. The statistic was computed via the activeSetLogCon and activeSetLogCon.mode functions.

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log-concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Doss, C. R. (2013). Shape-Constrained Inference for Concave-Transformed Densities and their Modes. PhD thesis, Department of Statistics, University of Washington, in prepa- ration.

Doss, C. R. and Wellner, J. A. (2013). Inference for the mode of a log-concave density. Technical Report, University of Washington, in preparation.

See Also

See the "distr" package. The LRmodeTest and LCLRCImode functions use LCTLLRdistn.

Examples

```
LCTLLRdistn@q(.95); ##~1.06 is the 95% quantile
```

logConDens *Compute log-concave density estimator and related quantities*

Description

Compute the log-concave and smoothed log-concave density estimator.

This function is as it is in the logcondens package except we've added the 'prec' variable as an argument, and modified the values returned as output, to be in line with the `activeSetLogCon.mode` function.

Usage

```
logConDens(x, xgrid = NULL, smoothed = TRUE, print = FALSE,
    gam = NULL, xs = NULL, prec=10^-10)
```

Arguments

x	Vector of independent and identically distributed numbers, not necessarily unique.
xgrid	Governs the generation of weights for observations. See preProcess for details.
smoothed	If TRUE, the smoothed version of the log-concave density estimator is also computed.
print	print = TRUE outputs the log-likelihood in every loop, print = FALSE does not. Make sure to tell R to output (press CTRL+W).
gam	Only necessary if smoothed = TRUE. The standard deviation of the normal kernel. If equal to NULL, gam is chosen such that the variances of the original sample $x_1, \ldots, x_n$ and $\widehat{f}_n^*$ coincide.
xs	Only necessary if smoothed = TRUE. Either provide a vector of support points where the smoothed estimator should be computed at, or leave as NULL. Then, a sufficiently width equidistant grid of points will be used.
prec	Governs precision of various subfunctions, e.g. the Newton-Raphson procedure.

Details

See activeSetLogCon for details on the computations.

Value

logConDens returns an object of class "dlc", a list containing the following components: xn, x, w, L, IsKnot, knots, phi, fhat, Fhat, H, n, m, mode, dlcMode, sig, phi.f, fhat.f, Fhat.f, E.f, phiPL, phiPR, phiPL.f, and phiPR.f, as generated by activeSetLogCon. If smoothed = TRUE, then the returned object additionally contains f.smoothed, F.smoothed, gam, and xs as generated by evaluateLogConDens. Finally, the entry smoothed of type "logical" returnes the value of smoothed.

The methods summary.dlc and plot.dlc are used to obtain a summary and generate plots of the estimated density.

Author(s)

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

Lutz Duembgen, <duembgen@stat.unibe.ch>,
http://www.staff.unibe.ch/duembgen

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010). Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009). Maximum likelihood estimation of a log–concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011). logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Examples

```
## ====================================================
## Illustrate on simulated data
## ====================================================

## Set parameters
n <- 50
x <- rnorm(n)

res <- logConDens(x, smoothed = TRUE, print = FALSE, gam = NULL,
    xs = NULL)
summary(res)
plot(res, which = "density", legend.pos = "topright")
plot(res, which = "log-density")
plot(res, which = "CDF")

## Compute slopes and intercepts of the linear functions that
## compose phi
slopes <- diff(res$phi) / diff(res$x)
intercepts <- -slopes * res$x[-n] + res$phi[-n]


## ====================================================
## Illustrate method on reliability data
## Reproduce Fig. 2 in Duembgen & Rufibach (2009)
## ====================================================

## Set parameters
data(reliability)
x <- reliability
n <- length(x)
res <- logConDens(x, smooth = TRUE, print = TRUE)
```

```
phi <- res$phi
f <- exp(phi)

## smoothed log-concave PDF
f.smoothed <- res$f.smoothed
xs <- res$xs

## compute kernel density
sig <- sd(x)
h <- sig / sqrt(n)
f.kernel <- rep(NA, length(xs))
for (i in 1:length(xs)){
    xi <- xs[i]
    f.kernel[i] <- mean(dnorm(xi, mean = x, sd = h))
}

## compute normal density
mu <- mean(x)
f.normal <- dnorm(xs, mean = mu, sd = sig)

## ================================================
## Plot resulting densities, i.e. reproduce Fig. 2
## in Duembgen and Rufibach (2009)
## ================================================
plot(0, 0, type = 'n', xlim = range(xs), ylim = c(0, 6.5 * 10^-3))
rug(res$x)
lines(res$x, f, col = 2)
lines(xs, f.normal, col = 3)
lines(xs, f.kernel, col = 4)
lines(xs, f.smoothed, lwd = 3, col = 5)
legend("topleft", c("log-concave", "normal", "kernel",
    "log-concave smoothed"), lty = 1, col = 2:5, bty = "n")


## ================================================
## Plot log-densities
## ================================================
plot(0, 0, type = 'n', xlim = range(xs), ylim = c(-20, -5))
legend("bottomright", c("log-concave", "normal", "kernel",
    "log-concave smoothed"), lty = 1, col = 2:5, bty = "n")
rug(res$x)
lines(res$x, phi, col = 2)
lines(xs, log(f.normal), col = 3)
lines(xs, log(f.kernel), col = 4)
lines(xs, log(f.smoothed), lwd = 3, col = 5)


## ================================================
## Confidence intervals at a fixed point for the density
## see help file for logConCI()
## ================================================
```

LRmodeTest *Computes an Asymptotic Confidence Interval for the mode of a Log-Concave Density*

Description

A likelihood ratio test to test whether mode is the location of the mode of a (log-concave) density. Uses activeSetLogCon and activeSetLogCon.mode to compute the log-concave MLE and the log-concave MLE where the mode is restricted to be mode, respectively.

Usage

```
LRmodeTest(mode, x, xgrid = NULL, w, nn = length(x), alpha, prec=1e-10,
print=FALSE)
```

Arguments

mode	Numeric value giving the constrained value of the mode location.
x	Points at which to compute the unconstrained and constrained estimators. Either iid data observations (from a log-concave density) or, such data binned. If x is binned, there should be corresponding weights w. Binning is usually handled by passing in a non-NULL value for xgrid.
xgrid	Governs binning of x and generation of corresponding weights w. See logcondens::preProcess. If w is not NA then xgrid should be NULL.
w	Numeric vector of length length(x) or NA. Weights corresponding to x. Can be NA (regardless of the value of xgrid) which indicates the weights are uniform (equal to 1/length(x)). If w is not NA then xgrid should be NULL. If nn is not equal to length(x) then w should be given a non-NA value. If w is not NA, then we assume that x has no duplicate entries.
nn	The number of data points initially observed. Numeric of length 1. Usually equal to length(x). If some sort of preProcessing is done in advance, may be not equal to length(x). To pass in a non-default value for nn (i.e. something other than length(x)), w must also be passed in a (numeric vector) value, and xgrid must be NULL.
alpha	Numeric value in [0,1], the level for the test.
prec	Numeric value, giving the precision passed to activeSetLogCon and activeSetLogCon.mode.
print	TRUE or FALSE, depending on whether debugging information should be printed or not, respectively.

Details

Uses activeSetLogCon and activeSetLogCon.mode to compute the log-concave MLE $\hat{f}_n$ and the log-concave MLE where the mode is restricted to be mode, $\hat{f}_n^0$. The statistic, Two times the Log Likelihood Ratio (TLLR) is then defined to be $2(\log\hat{f}_n - \log\hat{f}_n^0)$.

Our test is based on the assumption that the true log-concave density f_0 is twice differentiable at its true mode m, and f_0 satisfies f_0''(m)<0. Under that condition, Doss (2013) conjectures that the log likelihood ratio statistic is asymptotically pivotal (i.e., its limit distribution does not depend on the true log-concave density).

Using the pivotal nature of TLLR, its limit distribution can be simulated from any given known log-concave density (e.g., a standard normal), and the estimated distribution function of this limit is given by the LCTLLRdistn object. The quantiles of the limit distribution are used to either reject or not reject the test.

Value

Returns TRUE or FALSE for not reject or to reject mode, respectively.

Author(s)

Charles R. Doss, <cdoss@stat.washington.edu>,
http://www.stat.washington.edu/people/cdoss/

References

Duembgen, L, Huesler, A. and Rufibach, K. (2010) Active set and EM algorithms for log-concave densities based on complete and censored data. Technical report 61, IMSV, Univ. of Bern, available at http://arxiv.org/abs/0707.4643.

Duembgen, L. and Rufibach, K. (2009) Maximum likelihood estimation of a log-concave density and its distribution function: basic properties and uniform consistency. *Bernoulli*, **15(1)**, 40–68.

Duembgen, L. and Rufibach, K. (2011) logcondens: Computations Related to Univariate Log-Concave Density Estimation. *Journal of Statistical Software*, **39(6)**, 1–28. http://www.jstatsoft.org/v39/i06

Doss, C. R. (2013). Shape-Constrained Inference for Concave-Transformed Densities and their Modes. PhD thesis, Department of Statistics, University of Washington, in preparation.

Doss, C. R. and Wellner, J. A. (2013). Inference for the mode of a log-concave density. Technical Report, University of Washington, in preparation.

See Also

LCLRCImode uses LRmodeTest to compute asymptotic confidence sets.

Examples

```
nn <- 200
myxx <- rnorm(nn) ## no need to sort

## Under null/true hypothesis with or without grid
```

```
LRmodeTest(mode=0, x=myxx, xgrid=NULL, alpha=.05)
LRmodeTest(mode=0, x=myxx, xgrid=.05, alpha=.05)

## Under alternative/false hypothesis
LRmodeTest(mode=3, x=myxx, xgrid=NULL, alpha=.05)
```

Index