Package 'episplineDensity'

February 19, 2015

Version 0.0-1

Date 2014-11-08

Title Density Estimation with Soft Information by Exponential Epi-splines

Author Sam Buttrey, Johannes Royset, and Roger Wets, based on the Matlab code of Royset and Wets

Maintainer Sam Buttrey <buttrey@nps.edu>

Depends nloptr, pracma

Description Produce one-dimensional density estimates using exponential epi-splines. The user may incorporate soft information, by imposing constraints that (i) require unimodality; (ii) require that the density be monotone non-increase or non-decreasing; (iii) put upper bounds on first or second moments; (iv) bound the density's values at mesh points; (v) require that the estimate be continuous or continuously differentiable; and more.

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2014-11-10 18:33:30

R topics documented:

expepi *Fit density estimate by exponential epi-splines.*

Description

This code produces one-dimensional density estimates satisfying "soft" conditions like unimodality. A number of possible soft conditions are permitted. This version calls the nloptr suite of optimizers. Convergence is not particularly fast.

Usage

```
expepi(data, lower = NULL, upper = NULL, N = 10, M = 5,
 order = 2, softinfo, opt.args, opt.local.args, postproc.controls)
```

Arguments

`data`	Numeric vector of data for density estimate.
`lower`	Lower bound for density support. If missing, use default as set in `preprocess.data`
`upper`	Upper bound for density support. If missing, use default as set in `preprocess.data`
`N`	Integer: number of segments. Default 10.
`M`	Integer: number of points within each segment to consider. Default 5.
`order`	Integer: order of polynomials used in spline fits. Currently this must be 2.
`softinfo`	List of "soft" conditions to be imposed on the density estimate. See `setup.softinfo` for possibilities.
`opt.args`	List of arguments to be passed to global optimizer. See `setup.optargs` for defaults and more information. Set `print_level = 1` to show each iteration in the global optimizer, which might help convince you that something is happening.
`opt.local.args`	List of arguments to be passed to local optimizer. See `setup.optargs` for defaults and more information.
`postproc.controls`	List of arguments for post-processing. See `postproc.control` for defaults and more information.

Details

This function produces a density estimator for data `data`, imposing constraints in `softinfo`. The density is in the form of an exponential epi-spline. An epi-spline is like a spline estimator in that in consists of polynomials between knots. However, the polynomials are not automatically constrained to meet at knots. The density estimate is an exponential epi-spline, which is `exp(- s)` where `s` is the epi-spline value.

Value

A list of class `c("episplineDensity", "nloptr")` with the output from nloptr, plus additional items:

`softinfo`	The softinfo as passed to the optimizer, consisting of what was passed into this function plus some defaults
`epiparameters`	Epiparameters, as generated by `preprocess.data`
`caseinfo`	A list with the sample size, as `sampleisze`, and notion else.
`x`	Copy of the data
`c.out`	Coefficients associated with this set of data.
`opts`	Copy of opts. See `setup.optargs`.
`orig.integral`	If the postprocessing option `normalize.to.1` is supplied, this item is present and gives the value of the integral of the density before normalization. It should be near 1.
`integral`	If the postprocessing option `normalize.to.1` is supplied, this item is present and gives the value of the integral of the density after normalization. It should exactly 1.

Author(s)

Sam Buttrey, after Matlab code from Royset and Wets.

References

Royset and Wets, Nonparametric Density Estimation with Soft Information Using Exponential Epi-Splines, in press.

See Also

`nloptr`

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

n10 <- c(-0.795173769, -0.268865287, -0.032803042, -0.361751212,
0.699170399, -0.909275685, 0.452956532, 1.501356616, 1.669061521,
-0.524919503)
#
# Generate a unimodal estimate. Plot automatically.
#
## Not run: soft <- setup.softinfo (10, unimodal = TRUE)

## Not run: expepi (n10, softinfo = soft)
#
# Generate a unimodal estimate, but constrain the second non-central
```

```
# moment to be <= 0.4. Plot automatically. This command will require
# a couple of minutes to run.
#
## Not run: soft$upperbound2moment <- 0.4
## Not run: expepi (n10, softinfo = soft)
#
# Generate a nondecreasing estimate without plotting.
#
## Not run: soft <- setup.softinfo (10, monotone="nondecreasing")
## Not run: n10.out <- expepi (n10, softinfo = soft, postproc.controls =
 postproc.control (pic.types = NULL))
## End(Not run)
#
# Now plot.
#
## Not run: plot (n10.out)
```

`plot.episplineDensity` *Plot an exponential epi-spline density esimate*

Description

Plot a density estimate, plus the original data

Usage

```
## S3 method for class 'episplineDensity'
 plot(x, ...)
```

Arguments

`x`	Output from a call to expepi
`...`	Other arguments, currently ignored

Details

This plots the `x.pts` and `y.est` items from the `x` object, and adds red circles for the original observations.

Value

None.

Author(s)

Sam Buttrey

See Also

`expepi`

`postproc.control` *Set options for post-processing of expepi output*

Description

Generate a list of options for post-processing expepi output.

Usage

```
postproc.control(numevalpts = 10000, pic.types = c("1"), normalize.to.1 = TRUE)
```

Arguments

`numevalpts`	Integer, giving number of equally-spaced points at which to compute density estimate. Default 10,000.
`pic.types`	Character vector, naming the types of pictures produced. Default "1", which produces a graph of the density with points in red circles.
`normalize.to.1`	The density should integrate to exactly 1, but sometimes the numeric value is a little different from 1. If this is TRUE, the density's values are scaled so that the integral is exactly 1.

Value

A list of preprossing options with the values of the arguments.

Note

Currently only one picture is supported.

Author(s)

Sam Buttrey

See Also

`expepi`, ~~~

`preprocess.data` *Preprocess data to construct epiparameters.*

Description

The epispline parameters are the data plus lower and upper bounds on the support of the estimated density.

Usage

```
preprocess.data(data, lower, upper)
```

Arguments

`data`	Numeric vector of data to be used in density estimation.
`lower`	Lower bound on density support. Default: if missing or NULL, the lower bound is taken to be min(x) - 2 * sd(x). If -inf, the lower bound is taken to be mean (x) - 10 * sd(x).
`upper`	Upper bound on density support. Default: if missing or NULL, the upper bound is taken to be max(x) + 2 * sd(x). If inf, the upper bound is taken to be mean (x) + 10 * sd(x).

Details

Data outside the bounds is discarded.

Value

List of epiparameters, with entries

`data`	Data as passed, with entries outside the bounds deleted
`m0`	Lower bound
`mN`	Upper bound

Author(s)

Sam Buttrey, from matlab code by Royset and Wets

See Also

`expepi`

print.episplineDensity *Print method for episplineDensity objects.*

Description

This prints the `status` item and nothing else.

Usage

```
## S3 method for class 'episplineDensity'
 print(x, ...)
```

Arguments

`x`	Output from a call to `expepi`.
`...`	Other arguements, currently ignored.

Details

The current intent of this is to keep this whole object from printing to the screen.

Value

None

Author(s)

Sam Buttrey

See Also

`expepi`

setup.optargs *Set up arguments for global and local optimization programs*

Description

The exponential epi-spline scheme uses a global optimization routine from package nloptr that itself calls a local one. This function produces a list of options for either or both.

Usage

```
setup.optargs(param.length, opts, local.opts)
```

Arguments

param.length	Length of parameter vector.
opts	Options to global solver. These will be passed as argument "opts" to nloptr.
local.opts	Options to local solver. These will be passed as a list named "local_opts" attached to the "opts" list above.

Details

Default values for opts are algorithm = "NLOPT_LD_AUGLAG", maxeval = 2500, xtol_rel = 1e-05, xtol_abs = 1e-05, and for local.opts, algorithm = "NLOPT_LD_SLSQP", maxeval = 1000, and xtol_rel = 1e-05.

Value

List with default opts overridden by any that were supplied, plus a list named local_opts with default local_opts, overridden by any that were supplied.

Author(s)

Sam Buttrey

See Also

expepi

---

setup.softinfo *Set up softinfo for exponential epi-splines.*

Description

The softinfo prescribes constraints imposed on the density estimate.

Usage

```
setup.softinfo(N = 10, order = 2, warn = FALSE, ...)
```

Arguments

N	Integer giving number of interior mesh points (knots) for the splines. Default 10.
order	Integer giving the order for the polynomial splines. Default 2, and only 2 is permitted right now.
warn	Logical: emit warnings when contradictory conditions are imposed? Currently ignored. It is easy to generate contradictory conditions and the code only tests for a few combinations.

... A set of named arguments describing the possible values of soft information. The current possibilities are:

M Numeric : number of points in each segment at which Fisher and other constraints are imposed

unimodal Logical: if `TRUE`, require that the density be unimodal.

unimodaluppertail, unimodallowertail Numeric. Impose unimodality only on the lower or upper `floor (N * unimodallowertail)` or `floor (N * unimodaluppertail)` segments.

monotone Character: describes what sort of monotonicty is required. Possible values "nondecreasing" or "nonincreasing".

lowerboundsk, upperboundsk Numeric, length `N+1`. Bounds on epiparameters `s[0]` through `s[N]`. See `expepi` for details. Default: `-1000` for lower, `+1000` for upper.

lowerboundak0, upperboundak0, lowerboundakp, upperboundakp Numeric, length `N`. Lower and upper bounds on the linear coefficients (ak0) and quadratic coefficients (akp) of the splines.

continuous, continuousDiff, lsc, usc Logicals. When `TRUE`, require continuity, continuous differentiability, or that the density be lower semi-continuous (`lsc`) or upper semi-continuous (`usc`).

pointwiseFisherLower, pointwiseFisherUpper Numeric, length 1. Lower and upper bound on the value of slope/value at every point in every segment.

lowerdensityvalue, upperdensityvalue Numeric vectors of length `N` giving lower and upper bounds on the density estimate inside segments.

lowerdensityvalueEndpt, upperdensityvalueEndpt Numeric vectors of length `N + 1` giving lower and upper bounds on the density estimate at segment end points.

lowerdensityvalueSpecific Two-column numeric matrix. Each row has an `x` value and a density value and the density estimate is constrained to be at least `lowerdensityvalueSpecific[j,2]` at `x = lowerdensityvalueSpecific[j,1]` for each row `j`.

KLDivergenceUpper, KLDivergenceLower, KLDensity, KLDensityParams Upper and lower bounds on the KL divergence of the density estimate from the density whose name is given as an R density function in `KLDensity`, e.g. `dnorm`, and whose parameters are given as a list in `DLDensityParams`, e.g. `list (mean = 0, sd = 1)`

upperbound1moment, upperbound2moment Numeric; upper bounds on the first or second (non-central) moment of the estimate

Value

List with any specified values, plus any defaults (notably M = 5).

Author(s)

Sam Buttrey, after Matlab code by Royset and Wets.

See Also

expepi

Index