Package ‘adeba’
March 4, 2018

Encoding  UTF-8
Type   Package
Title  Adaptive Density Estimation by Bayesian Averaging
Version  1.1.2
Date  2018-03-02
Author  Christofer Backlin, Mats Gustafsson
Maintainer  Christofer Backlin <adeba@christofer.backlin.se>
Description  Univariate and multivariate non-parametric kernel density
              estimation with adaptive bandwidth using a Bayesian approach to Abramson's
              square root law.
Imports  graphics, mixtools, pdist, Rcpp (>= 0.11.6), stats
LinkingTo  Rcpp
Suggests  parallel, testthat
URL  https://github.com/backlin/adeba
BugReports  https://github.com/backlin/adeba/issues
License  GPL (>= 2)
LazyLoad  yes
RoxygenNote  6.0.1
NeedsCompilation  yes
Repository  CRAN
Date/Publication  2018-03-03 23:16:25 UTC

R topics documented:

adeba ............................................. 2
dimension ........................................ 3
is.rendered ..................................... 4
iterate .......................................... 5
make.adeba ...................................... 5
adeba

Description

Make ADEBA density estimate

Usage

adeba(data, adaptive = TRUE, beta = 0.5, parallel, na.rm = FALSE, ...)

Arguments

data

Dataset with examples as rows and dimensions as columns, supplied as a matrix
or something that can be converted to a matrix, like a vector, data.frame or
data.table.

adaptive

Whether to used fixed bandwidths identical for all kernels, or adaptive band-
widths unique to each kernel. Can also be specified as a positive integer to
iterate beyond the adaptive estimate (see the original publication).

beta

The level of adaptiveness. The default value of 0.5 corresponds to Silverman’s
square root law, which is suitable for the normal distribution. Can also be speci-
fied as multiple equispaced values, automatically weighted by the Bayesian ma-
chinery (the equispace requirement is for the integral approximation to hold).

parallel

Whether to use multiple CPU cores for calculation, using mclapply and mcMap
of the parallel package. To control the number of cores please use the mc.cores
option. If unset, all cores will be used.

na.rm

Whether to let missing values break execution (FALSE) or to remove them from
calculations (TRUE).

...

Sent to the internal function make.adeba.

Value

A density estimate.

Author(s)

Christofer Bäcklin
dimension

Retrieve the dimension or dimensions of the dataset used to make the estimate

Description

Retrieve the dimension or dimensions of the dataset used to make the estimate

See Also

predict.adeba, render, radeba

Examples

# Univariate
x.train <- c(rnorm(20, 0, .5), rnorm(20, 2, 2))
f <- adeba(x.train, beta = 0:2/2)
curve((dnorm(x, 0, .5) + dnorm(x, 2, 2))/2, -2, 7, lwd=5, col="grey85")
plot(f, type="both", add=TRUE)
points(f)

# Bi-variate
x.train <- 3 + sweep(matrix(rnorm(60), 30), 2, 1:2, "*") %*% matrix(c(1, .4, .4, 1), 2)
x.test <- 3 + sweep(matrix(rnorm(40), 20), 2, 1:2, "*") %*% matrix(c(1, .4, .4, 1), 2)
f <- adeba(x.train, adaptive=FALSE)
f <- render(f)
plot(f, type="both")
points(x.test[,1], x.test[,2], cex=10*predict(f, x.test), pch=19)
legend("topleft", c("Training", "Test"), pch=c(1,19), bg="white")

# Draw random sample from the estimated density
x.new <- radeba(400, f)
plot(f)
points(x.new[,1], x.new[,2])

# Slice the distribution
plot(0, 0, type="n", xlim=c(-4, 12), ylim=c(0, 0.15))
for(i in 1:6:16){
  f <- render(f, list(seq(-4, 12, length.out=200), i))
  plot(f, col= hsv(h=(i+6)/30), add=TRUE)
}
plot(f, type="data")

# See package ‘adebaExtra’ for how to plot in full 3d
Usage

dimension(x)

## S3 method for class 'adeba'
dim(x)

Arguments

x Density estimate.

Value

Integer scalar.
Integer vector.

Author(s)

Christofer Bäcklin

---

is.rendered

Determines if an estimate is rendered

Description

Determines if an estimate is rendered

Usage

is.rendered(object)

Arguments

object Density estimate.

Value

Logical.

Author(s)

Christofer Bäcklin

See Also

render
iterate

**Fit parameters**

**Description**

This function finds appropriate values for alpha, using a fixed beta=0.5, and calculates the estimate.

**Usage**

iterate(object, ...)

**Arguments**

- `object`: Density estimate.
- `...`: Sent to the internal function `get_log_likelihood`.

**Details**

To identify a suitable range for alpha containing the bulk of its posterior probability, a rough logarithmically spaced grid is searched. The returned interval is searched again to yield more accurate endpoints, and finally a fine linearly spaced grid is used for the final density estimate.

**Value**

An ADEBA estimate with increased number of iterations.

**Author(s)**

Christofer Bäcklin

---

make.nadeba

**Initialize a density estimate**

**Description**

Initialize a density estimate

**Usage**

make.nadeba(data, range = c(Inf, Inf), alpha = NULL, beta = 0.5, pilot, transform = TRUE, na.rm = FALSE, parallel = FALSE, log_prior = uniform_log_prior)
Arguments

data Domain of each variable (column) of the dataset. The idea is to allow for bounded kernels in the future, but at the moment this argument has no effect.
range Alpha values controlling the global bandwidth scaling. This should be set to NULL, implying automatic estimation from the data, unless you know what you are doing.
alpha Pilot function values for the first iteration. Leave unspecified to use a constant pilot.
transform If estimating multivariate densities, this argument controls whether the dataset should be pre-rotated and scaled before estimation. If the covariance matrix of data is large this is a very good idea, but it takes a little extra time to do.
na.rm Whether to remove observations with missing values (TRUE) or throw an error (FALSE).
parallel A function that calculates a log prior from a data frame with parameters. See log_prior for options and details.
log_prior A function that calculates a log prior from a data frame with parameters. See log_prior for options and details.

NOTE: Unless you know ADEBA's internals well you should probably not touch this argument. It was only exposed to facilitate analyses on the prior's importance, or rather lack of it, which were added as a supplement to the original publication.

Value

An unfitted density estimate that it to be passed to iterate. The estimate consist of the following components:

iterations Number of iterations calculated.
parameters All the parameters and posterior values.
bandwidths Bandwidths corresponding to the parameters. These are pre-calculated to make the results easier for the user to digest and manipulate, and since are often needed multiple times.
distance Distance matrix of the original data set.
constant Logical vector marking columns in the data set without any variation. These are excluded from the analysis.
transform To be able to use spherical kernels on multivariate data sets with variables of very different variances or high linear dependence, a PCA-based transformation is applied to the data prior to computation. This element contains that function.
retransform Function for converting transformed data back to the original domain. Needed by radeba.

Author(s)

Christofer Bäcklin
**plot.adeba**  
*Plot density estimate*

**Description**
Plot density estimate

**Usage**
```r
## S3 method for class 'adeba'
plot(x, y, type = c("estimate", "data", "both"), ..., 
     add = FALSE)

## S3 method for class 'adeba'
lines(x, ...)

## S3 method for class 'adeba'
points(x, ...)

## S3 method for class 'adeba'
contour(x, ...)
```

**Arguments**
- `x`: Density estimate.
- `y`: Ignored, kept for S3 consistency.
- `type`: What type of plot to draw.
- `...`: Sent to `plot`, `image` or some other base plotting function, depending on `type` and dimension of the estimate.
- `add`: Whether to start a new plot (FALSE) or add to an existing (TRUE).

**Author(s)**
Christofer Bäcklin

---

**predict.adeba**  
*Predict PDF at selected points*

**Description**
Predict PDF at selected points
Usage

```R
## S3 method for class 'adeba'
predict(object, newx, ...)
```

Arguments

- `object`: Density estimate.
- `newx`: New data points to predict. Can either be a dataset of the same dimension as was used to make the estimate, or a point grid in list form (see `render` for details).
- `...`: Ignored, kept for S3 consistency.

Value

A vector if `newx` is a matrix or array, or an array if `newx`.

Author(s)

Christofer Bäcklin

---

**radeba**

**ADEBA estimates**

Description

Density, distribution, and random generation for ADEBA estimates. Works just like `rnorm`, `runif`, etc.

Usage

```R
radeba(n, object)
dadeba(x, object)
padeba(q, object, lower.tail = TRUE)
```

Arguments

- `n`: Number of examples to sample.
- `object`: Density estimate.
- `x`, `q`: Vector of quantiles.
- `lower.tail`: Logical; If TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Author(s)

Christofer Bäcklin
Examples

data(faithful)
f <- adeba(faithful$eruptions, adaptive=FALSE)
f.eruptions <- radeba(1e5, f)
hist(f.eruptions, breaks=100, col="skyblue", probability=TRUE)
lines(f, lwd=3)
rug(faithful$eruptions)

render  

Render and store the estimate

Description

Render and store the estimate

Usage

 render(object, grid)

Arguments

 object  Density estimate.
 grid  A list of numerical vectors corresponding to the dimensions of the estimate.

Value

Nothing, the object is modified in place.

Author(s)

Christofer Bäcklin

See Also

is.rendered
uniform_log_prior

---

**Description**

Functions designed to be plugged in as log_prior to `make.adeba`.

**Usage**

```r
uniform_log_prior(parameters)
jeffreys_log_prior(parameters)
empirical_gaussian_log_prior(parameters)
```

**Arguments**

- `parameters`: A data frame containing `alpha` and `beta` values, `alpha.range` that specifies sampling density, and `log.likelihood` that contains the unnormalized log likelihood for each \((\alpha, \beta)\) pair.

**Value**

A vector of log prior values.

**Author(s)**

Christofer Bäcklin

**References**

A demo explaining how `empirical_gaussian_log_prior` was implemented can be found in this Rmarkdown notebook.
Index

adeba, 2, 6
contour.adeba(plot.adeba), 7
dadeba(radeba), 8
dim.adeba(dimension), 3
dimension, 3

empirical_gaussian_log_prior
  (uniform_log_prior), 10

image, 7
is.rendered, 4
iterate, 5, 6

jeffreys_log_prior (uniform_log_prior),
  10

lines.adeba(plot.adeba), 7
log_prior, 6
log_prior (uniform_log_prior), 10

make.adeba, 5, 10
mclapply, 2
mcMap, 2

padeba(radeba), 8
plot, 7
plot.adeba, 7
points.adeba(plot.adeba), 7
predict.adeba, 3, 7

radeba, 3, 6, 8
render, 3, 8, 9
rnorm, 8
runif, 8

uniform_log_prior, 10