Package ‘delt’

June 3, 2015

Version 0.8.2

Date 2015-06-02

Title Estimation of Multivariate Densities Using Adaptive Partitions

Author Jussi Klemela [aut, cre],
Sauli Herrala [ctb]

Maintainer Jussi Klemela <jussi.klemela@gmail.com>

Imports denpro (>= 0.9.2)

Description We implement methods for estimating multivariate densities.
We include a discretized kernel estimator,
an adaptive histogram (a greedy histogram and a CART-histogram),
stagewise minimization, and bootstrap aggregation.

License GPL (>= 2)

URL http://jussiklemela.com/delt/

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-06-03 12:12:23

R topics documented:

`delt-package` *Estimation of Multivariate Densities Using Adaptive Partitions*

Description

We implement methods for estimating multivariate densities. We include a discretized kernel estimator, an adaptive histogram (a greedy histogram and a CART-histogram), stagewise minimization, and bootstrap aggregation.

Details

Package:	delt
Version:	0.8.2
Date:	2015-05-17
Depends:	R
License:	GPL version 2 or newer
URL:	http://www.jussiklemela.com/delt
Packaged:	Mon Jun 15 16:34:17 2015; jsk
Built:	R 3.2.0; i486-pc-linux-gnu; 2015-05-15 16:34:35; unix

Greedy histograms: eval.greedy, lstseq.greedy.

CART-histograms: eval.cart, lstseq.cart.

CART histograms step by step: densplit, prune, eval.pick.

Bootstrap aggregation of histograms: eval.bagg, lstseq.bagg.

Stagewise minimization: eval.stage, eval.stage.gauss.

Other utilities: partition, plotparti.

Tree transformation: lefrig2par, makebina.

Miscallenous: intpcf, supp.

Index:

```
densplit                Calculation of an overfitting histogram
eval.bagg               Returns a bootstrap aggregation of adaptive
```

	histograms
eval.cart	Calculates a CART histogram
eval.greedy	Returns a greedy histogram
eval.pick	Returns a subtree of an evaluation tree
eval.stage	Returns a stagewise minimization estimate
eval.stage.gauss	Returns a 1D Gaussian mixture density estimate
intpcf	Calculates the integral of a piecewise constant function
lefrig2par	Transforms an evaluation tree so that it can be plotted with the "plottree" function of package "denpro"
lstseq.bagg	Calculates a scale of bootstrap aggregated histograms
lstseq.cart	Calculates a scale of CART histograms
lstseq.greedy	Calculates a scale of greedy histograms
makebina	Tranforms and evaluation tree to the tree object of R
partition	Finds the partition generated by an evaluation tree
plotparti	Draws a partition
prune	Prepares for pruning an overfitting evaluation tree
scaspa	Finds the number of modes of histograms which are obtained by pruning an overfitting histogram
supp	Returns the bounding box of observations

Author(s)

Jussi Klemela <jussi.klemela@gmail.com>

Maintainer: Jussi Klemela <jussi.klemela@gmail.com>

Examples

```
library(denpro)

# Generate the data

dendat<-sim.data(n=500,seed=5,type="mulmodII")

# Calculate the estimates

eva<-eval.greedy(dendat,leaf=16)

eva<-eval.cart(dendat,leaf=16)

eva<-eval.bagg(dendat,B=3,leaf=12,prune="on")

eva<-eval.stage(dendat,leaf=10,M=3)
```

```
# Draw the estimates

lst<-leafsfirst(eva)
plotvolu(lst)

dp<-draw.pcf(eva,pnum=c(60,60))
persp(dp$x,dp$y,dp$z,theta=-20,phi=30)
```

---

cluster.lst *Assigns labels to data points*

Description

Assigns labels to data points according to cluster membership, when the clusters are defined as high density regions

Usage

```
cluster.lst(dendat, h, N = NULL, cut = NULL, lambda = NULL, complete = FALSE,
type = "grid", labels = "number", nodes = NULL, minobs = 1)
```

Arguments

`dendat`	n*d matrix of real numbers; the data matrix.
`h`	positive real number; smoothing parameter of a kernel density estimator
`N`	d vector of positive integers; a kernel estimate is evaluated on a regular grid which is such that in direction i there are N[i] points; N is needed only when type="grid".
`cut`	real number between 0 and 1; this parameter is used to determine the level "lambda" of the level set whose disconnected components determine the clusters.
`lambda`	positive real number between; "lambda" is the level of the level set whose disconnected components determine the clusters.
`complete`	TRUE or FALSE; if complete=FALSE, then partial clustering is performed, otherwise complete clustering is performed.
`type`	either "grid" or "adaptive"; if type="grid", then the density is estimated using a discretized kernel estimator with a regular grid; otherwise the density is estimated using a discretized kernel estimator with an adaptive grid.
`labels`	if labels="number", then the cluster labels are integers 1,2,..., otherwise the cluster labels are colors.
`nodes`	a vector of positive integers; contains pointers to the nodes of a level set tree; the nodes indicate which disconnected components of level sets define the clusters.
`minobs`	a positive integer; this is a parameter of function "pcf.greedy.kernel".

Value

a vector of cluster labels; the vector has length equal to the number of rows of the data matrix "dendat". The cluster labels are either numbers or names of colors.

Author(s)

Jussi Klemela

See Also

pcf.greedy.kernel

Examples

```
library(denpro)
# generate data
seed<-1
n<-50
d<-2
l<-3; D<-4; c<-D/sqrt(2)
M<-matrix(0,l,d); M[2,]<-c; M[3,]<--c
sig<-matrix(1,l,d)
p<-rep(1/l,l)
dendat<-sim.data(type="mixt",n=n,M=M,sig=sig,p=p,seed=seed)

# partial clustering with a fixed level
h<-(4/(d+2))^(1/(d+4))*n^(-1/(d+4))*apply(dendat,2,sd)
N<-rep(20,d)
cl<-cluster.lst(dendat,h,N=N,labels="colors",type="grid",lambda=0.02)
#plot(dendat,col=cl)

# complete clustering with a fixed level
cl<-cluster.lst(dendat,h,N=N,complete=TRUE,labels="colors",type="grid",lambda=0.02)
#plot(dendat,col=cl)

# complete clustering with locally changing levels
N<-rep(20,d)
pcf<-pcf.kern(dendat,h,N)
lst<-leafsfirst(pcf)
nodes<-findbnodes(lst,modenum=3)
cl<-cluster.lst(dendat,h,N,nodes=nodes,complete=TRUE,labels="colors")
#plot(dendat,col=cl)
```

---

densplit *Calculation of an overfitting histogram*

Description

The function returns an overfitting histogram when a data matrix is given as an input. The output is an evaluation tree which is grown with greedy growing. The evaluation tree defines a partition of the sample space. The evaluation tree may be pruned to get a density estimate.

Usage

```
densplit(dendat, minobs=NULL, leaf=0, method="loglik",
splitscan=0, seedf=1, suppo=NULL)
```

Arguments

`dendat`	n*d data matrix
`minobs`	non-negative integer; splitting of a bin will be continued if the bin containes "minobs" or more observations
`leaf`	internal (maximal number of leafs in the evaluation tree)
`method`	"loglik" or "projec"; the contrast function
`splitscan`	internal (random selection of splits)
`seedf`	internal
`suppo`	2*d vector of real numbers; the rectangle to be splitted; the rectangle has to contain the data

Value

Returns an evaluation tree as a list of vectors.

`direc`	integer in 1,...,d; variable which is splitted
`split`	real number; splitting point
`mean`	nonnegative number; value of the histogram on the rectangle corresponding to the node
`nelem`	nonnegative integer; number of observations in the rectangle corresponding to the node
`ssr`	real number; value of the likelihood criterion
`volume`	non-negative number; volume of the rectangle corresponding to the node
`left`	non-negative integer; link to the left child, 0 if terminal node
`right`	non-negative integer; link to the right child, 0 if terminal node
`low`	the lower vertice of the rectangles
`upp`	the upper vertice of the rectangles
`N`	the number of grid points at each direction
`support`	the support of the histogram

Author(s)

Jussi Klemela

See Also

prune, eval.pick

Examples

```
library(denpro)

dendat<-sim.data(n=200,seed=5,type="mulmodII")
et<-densplit(dendat)

treeseq<-prune(et)
treeseq$leafs
len<-length(treeseq$leafs)

leaf<-treeseq$leafs[len-10]
leaf
etsub<-eval.pick(treeseq,leaf=leaf)

dp<-draw.pcf(etsub)
persp(dp$x,dp$y,dp$z,phi=25,theta=-120)
```

---

eval.bagg *Returns a bootstrap aggregation of adaptive histograms*

Description

Returns a bootstrap aggregation of CART-histograms or greedy histograms.

Usage

```
eval.bagg(dendat, B, leaf, minobs = NULL, seed = 1, sample = "bagg",
prune = "off", splitscan = 0, seedf = 1, scatter = 0, src = "c",
method = "loglik")
```

Arguments

dendat	n*d data matrix
B	positive integer; the number of aggregated histograms
leaf	the cardinality of the partitions of the aggregated histograms
minobs	non-negative integer; a property of aggregated histograms; splitting of a bin will be continued if the bin containes "minobs" or more observations
seed	the seed for the random number generation of the random selection of the bootstrap sample
sample	"bagg" or "worpl"; the bootstrapping method; "worpl" for the n/2-out-of-n without replacement; "bagg" for n-out-of-n with replacement

`prune`	"on" or "off"; if "on", then CART-histograms will be aggregated; if "off", then greedy histograms will be aggregated
`splitscan`	internal (how many splits will be used for random split selection)
`seedf`	internal (seed for random split selection)
`scatter`	internal (random perturbation of observations)
`src`	internal ("c" or "R" code)
`method`	"loglik" or "projec"; the empirical risk is either the log-likelihood or the L2 empirical risk

Value

An evaluation tree

Author(s)

Jussi Klemela

See Also

`lstseq.bagg`, `eval.cart`, `eval.greedy`

Examples

```
library(denpro)
dendat<-sim.data(n=600,seed=5,type="mulmodII")

leaf<-7     # number of leaves in the histograms
seed<-1     # seed for choosing bootstrap samples
sample="worpl" # without-replacement bootstrap
prune="on"  # we use CART-histograms
B<-5        # the number of histograms in the average

eva<-eval.bagg(dendat,B,leaf,seed=seed,sample=sample,prune=prune)

dp<-draw.pcf(eva,pnum=c(60,60))
persp(dp$x,dp$y,dp$z,theta=-20,phi=30)
```

`eval.cart` *Calculates a CART histogram*

Description

Calculates a CART histogram. The estimate is represented as an evaluation tree. An CART histogram is a multivariate adaptive histogram which is obtained by pruning an evaluation tree of an overfitting histogram.

Usage

```
eval.cart(dendat, leaf, minobs = NULL)
```

Arguments

`dendat`	n*d data matrix
`leaf`	positive integer; the cardinality of the partition of the histogram
`minobs`	non-negative integer; splitting of a bin of the overfitting histogram will be continued if the bin containes "minobs" or more observations

Details

The partition of the histogram may not contain exactly "leaf" rectangles: the cardinality of the partition is as close as possible to "leaf"

Value

An evaluation tree

Author(s)

Jussi Klemela

See Also

`lstseq.cart`, `densplit`

Examples

```
library(denpro)
dendat<-sim.data(n=600,seed=5,type="mulmodII")
eva<-eval.cart(dendat,16)

dp<-draw.pcf(eva,pnum=c(60,60))
persp(dp$x,dp$y,dp$z,theta=-20,phi=30)
```

`eval.greedy` *Returns a greedy histogram*

Description

Returns a greedy histogram. A greedy histogram is grown stagewise by minimizing an empirical risk functional.

Usage

```
eval.greedy(dendat, leaf, method = "loglik", minobs = NULL, bound = 0,
suppo = NULL)
```

Arguments

`dendat`	n*d data matrix
`leaf`	the (maximal) number of rectangles in the partition of the histogram
`method`	"loglik" or "projec"; the empirical risk is either the log-likelihood or the L2 empirical risk
`minobs`	non-negative integer; splitting of a bin will be continued if the bin containes "minobs" or more observations
`bound`	internal
`suppo`	2*d vector of real numbers; the rectangle to be splitted; the rectangle has to contain the data

Value

An evaluation tree

Author(s)

Jussi Klemela

See Also

`lstseq.greedy`, `partition`

Examples

```
library(denpro)
dendat<-sim.data(n=200,seed=5,type="mulmodII")
eva<-eval.greedy(dendat,leaf=15)

dp<-draw.pcf(eva,pnum=c(60,60))
persp(dp$x,dp$y,dp$z,theta=-20,phi=30)
```

`eval.pick` *Returns a subtree of an evaluation tree*

Description

Returns a subtree of an evaluation tree. The subtree has a specified number of leafs. The evaluation tree is calculated by "densplit" function. To find out the possible values for the number of leaves we use function "prune".

Usage

```
eval.pick(treeseq, leaf)
```

Arguments

`treeseq`	an overfitting evaluation tree with information on the possible pruning nodes; output of function "prune"
`leaf`	positive integer; number of leaves in the subtree to be returned

Value

Returns an evaluation tree, see the documentation of function "eval.cart"

Author(s)

Jussi Klemela

See Also

`densplit`, `prune`

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=5,type="mulmodII")
et<-densplit(dendat)

treeseq<-prune(et)
treeseq$leafs
len<-length(treeseq$leafs)

leaf<-treeseq$leafs[len-10]
leaf
etsub<-eval.pick(treeseq,leaf=leaf)

dp<-draw.pcf(etsub)
persp(dp$x,dp$y,dp$z,phi=25,theta=-120)
```

`eval.stage` *Returns a stagewise minimization estimate*

Description

Returns a stagewise minimization estimate. A stagewise minimization estimator is a convex combination of greedy histograms. The convex combination is constructed by a stagewise minimization of an empirical risk functional.

Usage

```
eval.stage(dendat, leaf, M, pis = NULL, mcn = dim(dendat)[1],
minobs = NULL, seedi = 1, method = "projec", bound = 0)
```

Arguments

dendat	n*d data matrix
leaf	the (maximal) number of rectangles in the partition of the greedy histograms
M	the number of histograms in the convex combination
pis	the vector of weights of the convex combination
mcn	the size of the Monte Carlo sample used in the numerical integration in calculating the empirical risk functional
minobs	non-negative integer; splitting of a bin of a greedy histogram will be continued if the bin containes "minobs" or more observations
seedi	the seed for the generation of the Monte Carlo sample
method	"loglik" or "projec"; the empirical risk is either the log-likelihood or the L2 empirical risk
bound	internal

Value

An evaluation tree

Author(s)

Jussi Klemela

See Also

eval.greedy, eval.stage.gauss

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=5,type="mulmodII")
leaf<-13  # the number of leafs of the greedy histograms
M<-5      # the number of greedy histograms

pcf<-eval.stage(dendat,leaf=leaf,M=M)

dp<-draw.pcf(pcf,pnum=c(120,120))
persp(dp$x,dp$y,dp$z,ticktype="detailed",phi=25,theta=-120)
```

eval.stage.gauss *Returns a 1D Gaussian mixture density estimate*

Description

Estimates a 1D density with a mixture of Gaussians. The mixture is found by minimizing the L2 empirical risk in a stagewise manner.

Usage

```
eval.stage.gauss(dendat, M, mugrid, siggrid = 1, sigeka = TRUE, src = "c",
sampstart=FALSE, boost=FALSE, N=60)
```

Arguments

`dendat`	n-vector of 1D observations
`M`	integer >= 1; the number of mixture components in the estimate
`mugrid`	a vector of real numbers; the range for the means of mixture components
`siggrid`	a vector of real numbers; the range of possible standard deviations in the mixture components
`sigeka`	TRUE or FALSE; if TRUE, then the standard deviation of the first mixture component is equal to 1, otherwise the standard deviation of the first mixture component is found by minimization
`src`	"R" or "c"; if "R", then the R-code is used, otherwise the c-code is used
`sampstart`	internal
`boost`	internal
`N`	postive integer; the number of evaluation points

Value

A piecewise constant function with the additional components:

`muut`	vector of real numbers; the means of the mixture components
`sigit`	vector of positive real numbers; the standard deviations of the mixture components
`curmix`	a probability vector; the weights of the mixture components

Author(s)

Jussi Klemela

References

Jussi Klemela (2005). Density Estimation with Stagewise Optimization of the Empirical Risk

See Also

eval.stage

Examples

```
library(denpro)
dendat<-sim.data(n=100,type="1d2modal",seed=1)

mugrid<-seq(-1,5,0.3)    # grid of mu-values
siggrid<-seq(0.2,2,0.2)  # grid of sigma-values
M<-17                     # number of mixture components
pcf<-eval.stage.gauss(dendat,M,mugrid,siggrid)

dp<-draw.pcf(pcf)
plot(dp$x,dp$y,type="l")

# draw the estimate with the help of package "denpro"
#N<-100
#pcf2<-pcf.func("mixt",N,sig=pcf$sigit,M=pcf$muut,p=pcf$curmix)
#pnum<-100
#dm<-draw.pcf(pcf2,pnum=pnum)
#plot(dm$x,dm$y,type="l")
```

---

intpcf *Calculates the integral of a piecewise constant function*

Description

Calculates the integral of a piecewise constant function.

Usage

```
intpcf(pcf)
```

Arguments

pcf	piecewise constant function

Value

Real number; the integral of the piecewise cosntant function

Author(s)

Jussi Klemela

Examples

```
library(denpro)
dendat<-sim.data(n=50,seed=5,type="mulmodII")
eva<-eval.greedy(dendat,leaf=5)
intpcf(eva)
```

lefrig2par *Transforms an evaluation tree so that it can be plotted with the "plottree" function of package "denpro"*

Description

Evaluation trees are trees which are implemented with "left" and "right" pointers. We transform this tree representation to the representation with "parent" pointers (level set tree), so that it can be plotted with the "plottree" function of package "denpro". For example, functions "densplit" and "eval.pick" return evaluation trees.

Usage

```
lefrig2par(et)
```

Arguments

`et`	evaluation tree; result of "densplit", "eval.pick", "eval.cart", ...; see the documentation of "eval.cart"

Value

Returns a level set tree:

`parent`	parent links
`level`	height of the node
`center`	determines the ordering of the nodes
`volume`	determines the horizontal positioning of the nodes

Author(s)

Jussi Klemela

See Also

densplit, eval.pick

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=5,type="mulmodII")
et<-densplit(dendat)

lst<-lefrig2par(et)

plottree(lst)
```

---

lstseq.bagg *Calculates a scale of bootstrap aggregated histograms*

Description

Calculates a scale of bootstrap aggregated histograms. The estimates in the sequence are calculated with function "eval.bagg".

Usage

```
lstseq.bagg(dendat, B, lstree=NULL, level = NULL,
maxleaf = NULL, leafseq = NULL,
minobs = NULL, seed = 1, sample = "bagg", prune = "off",
splitscan = 0, seedf = 1, scatter = 0, src = "c", method = "loglik")
```

Arguments

`dendat`	n*d data matrix
`B`	positive integer; the number of aggregated histograms
`maxleaf`	the maximal cardinality of the partitions of the histograms in the sequence
`lstree`	if NULL, then level set trees are not calculated
`level`	if NULL, then shape trees are not calculated; if positive number, then it is the level of the level sets for which the shape trees are calculated
`leafseq`	a vector giving the cardinalities of the partitions of the aggregated histograms
`minobs`	non-negative integer; a property of aggregated histograms; splitting of a bin will be continued if the bin containes "minobs" or more observations
`seed`	the seed for the random number generation of the random selection of the bootstrap sample
`sample`	"bagg" or "worpl"; the bootstrapping method; "worpl" for the n/2-out-of-n without replacement; "bagg" for n-out-of-n with replacement
`prune`	"on" or "off"; if "on", then CART-histograms will be aggregated; if "off", then greedy histograms will be aggregated
`splitscan`	internal (how many splits will be used for random split selection)
`seedf`	internal (seed for random split selection)

`scatter`	internal (random perturbation of observations)
`src`	internal ("c" or "R" code)
`method`	"loglik" or "projec"; the empirical risk is either the log-likelihood or the L2 empirical risk

Value

A list with components

`lstseq`	a list of level set trees
`pcfseq`	a list of piecewise constant functions
`stseq`	a list of shape trees
`hseq`	a vector of smoothing parameters corresponding to the estimates in the sequence; the smoothing parameter is the cardinality of the partitions of the aggregated histograms

Author(s)

Jussi Klemela

See Also

`eval.bagg`

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=1,type="mulmodII")

seed<-1        # seed for choosing bootstrap samples
sample="worpl" # without-replacement bootstrap
prune="on"     # we use CART-histograms
B<-2           # the number of histograms in the average

estiseq<-lstseq.bagg(dendat,B,maxleaf=10,lstree=TRUE,
         seed=seed,sample=sample,prune=prune)

mt<-modegraph(estiseq)

plotmodet(mt)

#scaletable(estiseq)
```

`lstseq.cart` *Calculates a scale of CART histograms*

Description

Calculates a scale of CART histograms. The histograms in the scale have partitions of growing cardinality. Returns a sequence of estimates as piecewise constant functions and optionally as level set trees. Optionally a shape tree for a level set of each estimate is calculated.

Usage

```
lstseq.cart(treeseq, maxleaf=NULL, lstree=NULL, level = NULL,
indvec = NULL)
```

Arguments

`treeseq`	output of function "prune"
`maxleaf`	the maximal cardinality of the partitions of the histograms in the sequence
`lstree`	if NULL, then level set trees are not calculated
`level`	if NULL, then shape trees are not calculated; if positive number, then it is the level of the level sets for which the shape trees are calculated
`indvec`	a vector of indeces; chooses a subset of the complete sequence of subhistograms of the overfitting histogram

Value

A list with components

`lstseq`	a list of level set trees
`pcfseq`	a list of piecewise constant functions
`stseq`	a list of shape trees
`hseq`	a vector of smoothing parameters corresponding to the members in the sequences; the smoothing parameter is the cardinality of the partition

Author(s)

Jussi Klemela

See Also

`densplit`, `prune`

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=1,type="mulmodII")
et<-densplit(dendat)
treeseq<-prune(et)

estiseq<-lstseq.cart(treeseq,maxleaf=20,lstree=TRUE)

mt<-modegraph(estiseq)

plotmodet(mt)

#scaletable(estiseq)
```

`lstseq.greedy` *Calculates a scale of greedy histograms*

Description

Calculates a scale of greedy histograms. The histograms in the scale have a partition of growing cardinality. Returns a sequence of estimates as piecewise constant functions and optionally as level set trees. Optionally a shape tree for a level set of each estimate is calculated.

Usage

```
lstseq.greedy(dendat, maxleaf, lstree = NULL, level = NULL)
```

Arguments

`dendat`	n*d data matrix
`maxleaf`	integer>1; the scale consists of histograms whose partitions have cardinality 1,...,maxleaf
`lstree`	if NULL, then level set trees are not calculated
`level`	if NULL, then shape trees are not calculated; if positive number, then it is the level of the level sets for which the shape trees are calculated

Value

A list with components

`lstseq`	a list of level set trees
`pcfseq`	a list of piecewise constant functions
`stseq`	a list of shape trees
`hseq`	a vector of smoothing parameters corresponding to the members in the sequences; the smoothing parameter is the cardinality of the partition

Author(s)

Jussi Klemela

See Also

eval.greedy,

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=1,type="mulmodII")
estiseq<-lstseq.greedy(dendat,maxleaf=20,lstree=TRUE)

mt<-modegraph(estiseq)

plotmodet(mt)

#scaletable(estiseq)
```

---

makebina *Tranforms and evaluation tree to the tree object of R*

Description

Evaluation trees are such trees that are implemented with "left" and "right" pointers. We transform this tree representation to the tree object of the package "tree", so that it can be plotted by "plot.tree" function from package "tree" or by "draw.tree" function from package "maptree".

Usage

```
makebina(et)
```

Arguments

et	evaluation tree; result of "densplit", "eval.pick", "eval.cart", ...; see the documentation of "eval.cart"

Value

Returns an object of class tree.

Author(s)

Jussi Klemela

See Also

densplit, eval.pick

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=5,type="mulmodII")
et<-densplit(dendat)
mb<-makebina(et)

set.seed(1)
dendat<-matrix(rnorm(20),10)
et<-densplit(dendat,minobs=2)
mb<-makebina(et)

#library(tree)
#plot.tree(mb)

#library(maptree)
#draw.tree(mb)
```

`partition` *Finds the partition generated by an evaluation tree*

Description

Finds the partition generated by an evaluation tree. An evaluation tree makes a recursive partition of a rectangle. Functions "eval.cart", "densplit",... return evaluation trees.

Usage

```
partition(et, grid=TRUE, zerorecs=FALSE)
```

Arguments

`et`	an evaluation tree; output of "eval.cart", "densplit", ...
`grid`	TRUE or FALSE; whether the true coordinates or relative coordinates are used
`zerorecs`	TRUE or FALSE; whether the rectangles where the density vanishes are included; (evaluation trees are used by package "delt" to represent density functions)

Value

Returns a list with the following elements.

`values`	vector whose lenght is equal to the number of rectangles in the partition; value of the function on the corresponding rectangle
`recs`	recnum*(2*d) matrix; recnum is the number of rectangles in the partition and d is the dimension of the observations. The rows of "recs" describe the rectangles. Column (2*j-1) gives the lower value for the j:th interval and (2*j):th column gives upper value for the j:th interval, j=1,...,d.
`support`	the rectangle which is partitioned

Author(s)

Jussi Klemela

See Also

plotparti,

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=5,type="mulmodII")
et<-densplit(dendat)

pa<-partition(et)

plotparti(pa)
```

---

pcf.greedy.kernel *Computes a discretized kernel estimator with an adaptive partition*

Description

Computes a discretized kernel estimator with an adaptive partition and the output is a piecewise constant function object.

Usage

```
pcf.greedy.kernel(dendat, h, leaf=round(dim(dendat)[1]/2), minobs=NULL,
type="greedy")
```

Arguments

`dendat`	n*d matrix of real numbers; the data matrix
`h`	d vector of positive real numbers; vector of smoothing parameters
`leaf`	positive integer
`minobs`	positive integer smaller than n; the partition is such that there are at most "minobs" observation in each member
`type`	a character string; "greedy" (partition is generated by binary splits using maximum likelihood), "cpp" (just like "greedy" but uses C++ code, which is not a part of the package but has to loaded separately, see home page of delt), "dyadic" (only splits at the midpoints are made, which leads to a loss of accuracy), "prune" (using CART pruning), "old" (not recommended).

Value

a piecewise constant function object with an adaptive partition, see the web site

Author(s)

Jussi Klemela

See Also

densplit

Examples

```
library(denpro)
# generate data
seed<-1
n<-50
d<-2
l<-3; D<-4; c<-D/sqrt(2)
M<-matrix(0,l,d); M[2,]<-c; M[3,]<--c
sig<-matrix(1,l,d)
p<-rep(1/l,l)
dendat<-sim.data(type="mixt",n=n,M=M,sig=sig,p=p,seed=seed)

# colored volume function
h<-(4/(d+2))^(1/(d+4))*n^(-1/(d+4))*apply(dendat,2,sd)
minobs<-1
pcf<-pcf.greedy.kernel(dendat,h,minobs=minobs,type="greedy")
#lst<-leafsfirst.adagrid(pcf)
#plotvolu(lst,colo=TRUE)

#dp<-draw.pcf(pcf)
#contour(dp$x,dp$y,dp$z,drawlabels=FALSE)
```

`plotparti` *Draws a partition*

Description

Draws the partition calculated with function "partition".

Usage

```
plotparti(pa, d1 = NULL, d2 = NULL, dendat = NULL, restri = NULL,
pch = 21, support = pa$support, col = "black", cex.axis = 1)
```

Arguments

pa	partition; output of function "partition"
d1	integer 1,...,d; for the case the partition is a partition of a higher than 2 dimensional rectangle, "d1" is the first direction of the partition

`d2`	integer 1,...,d; for the case the partition is a partition of a higher than 2 dimensional rectangle, "d2" is the second direction of the partition
`dendat`	n*d data matrix; if given as an argument it will also be plotted
`restri`	internal
`pch`	symbol for plotting "dendat"; see function "points"
`support`	the bounds of the partition
`col`	color of the lines of the partition
`cex.axis`	magnification factor for the axis annotation; see "par"

Value

A plot at the graphics window

Author(s)

Jussi Klemela

See Also

`partition`

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=5,type="mulmodII")
et<-densplit(dendat)

pa<-partition(et)

plotparti(pa)
```

`prune` *Prepares for pruning an overfitting evaluation tree*

Description

Finds a sequence of nodes of an overfitting evaluation tree which are candidates to be the pruning nodes. Pruning a tree means removing a branch starting from a node.

Usage

```
prune(et)
```

Arguments

`et`	an evaluation tree; output of "eval.cart", "densplit", ...

Value

A list containing the following components.

`tree`	the original tree which was given as the input
`delnodes`	vector giving a sequence of nodes in the order in which we should prune the branches starting from these nodes
`delend`	vector whose length is the number of subtrees of the original tree. With the help of "delend" we define the subtrees. Elements of "delend" define a sequence of nodes from "delnodes" in the following way: (1:delend[1]) is the first sequence, (delend[1]+1:delend[2]) is the second sequence, and so on. Then, i:th subtree is the result of pruning branches away whose roots are the nodes which are the first delend[i] elements of delnodes.
`leafs`	vector whose length is the number of subtrees of the original tree; number of leafs of the subtrees
`alfa`	vector whose length is the number of subtrees of the original tree; value of the corresponding alfa (complexity parameter) for every subtree
`loglik`	vector whose length is the number of subtrees of the original tree; the value of the likelihood criterion for the subtree

Author(s)

Jussi Klemela

See Also

`densplit`, `eval.pick`

Examples

```
library(denpro)
dendat<-sim.data(n=100,seed=5,type="mulmodII")
et<-densplit(dendat)

treeseq<-prune(et)
treeseq$leafs
len<-length(treeseq$leafs)

leaf<-treeseq$leafs[len-10]
leaf
etsub<-eval.pick(treeseq,leaf=leaf)

dp<-draw.pcf(etsub)
#persp(dp$x,dp$y,dp$z,phi=25,theta=-120)
```

scaspa — *Finds the number of modes of histograms which are obtained by pruning an overfitting histogram*

Description

Function "densplit" returns an overfitting histogram as an evaluation tree. Function "prune" finds the candidate nodes for pruning. Function "scaspa" finds the number of modes (local maxima) in the histograms which are obtained by using these candidate nodes for pruning.

Usage

```
scaspa(treeseq, bind, eind)
```

Arguments

`treeseq`	a list returned by function "prune"
`bind`	integer in 1:subnum, where subnum is the number of elements in field "leafs" of "treeseq". We need bind<eind.
`eind`	integer in 1:subnum, where subnum is the number of elements in field "leafs" of "treeseq". We need bind<eind.

Value

List with the following vectors whose length is (eind-bind+1)

`moodilkm`	number of local maxima for each subtree
`alfas`	value of the smoothing parameter alpha for each subtree
`leafnums`	number of leaves for each subtree

Author(s)

Jussi Klemela

See Also

densplit, prune

Examples

```
set.seed(1)
dendat<-matrix(rnorm(20),10)
minlkm<-2
et<-densplit(dendat,minlkm)
treeseq<-prune(et)
treeseq$leafs

scaspa(treeseq,1,5)
```

supp *Returns the bounding box of observations*

Description

Returns the smallest rectangle containing the observations. The sides of the rectangle are parallel to the coordinate axis.

Usage

```
supp(dendat, epsi=0, blown=FALSE)
```

Arguments

`dendat`	n*d data matrix
`epsi`	positive number: option to return the smallest rectangle such that the epsi-shrinkage of the rectangle containes the observations
`blown`	internal

Value

2*d vector: (2*i-1)-element, i=1,...,d, is the start of the i:th interval and (2*i)-element is the end of the i:th interval

Author(s)

Jussi Klemela

Examples

```
set.seed(1)
dendat<-matrix(rnorm(20),10)
supp(dendat)
```

Index