Package ‘edci’

May 16, 2018

Version 1.1-3

Date 2018-05-16

Title Edge Detection and Clustering in Images

Author Tim Garlipp <timgarlipp@gmx.de>

Maintainer Dominik Kirchhoff <dominik.kirchhoff@tu-dortmund.de>

Description Detection of edge points in images based on the difference
of two asymmetric M-kernel estimators. Linear and circular
regression clustering based on redescending M-estimators.
Detection of linear edges in images.

License GPL-2

Date/Publication 2018-05-16 21:49:37 UTC

NeedsCompilation yes

Repository CRAN

R topics documented:

bestMclust ......................................................... 2
circMclust ......................................................... 3
deldupMclust ....................................................... 5
dedgecluster ......................................................... 6
dedgepoints ......................................................... 8
delist ............................................................... 10
deregMclust ......................................................... 11

Index 14
Choose 'best' clusters

Descripción

Chooses the 'best' regression cluster(s), if the number of true clusters is known.

Usage

bestMclust(clust, nc = 1, crit = "value")
projMclust(clust, x, y)
envMclust(clust, x, y, dist = 0)

Arguments

clust  Cluster object returned by oregMclust or circMclust.
nc     Number of 'best' clusters.
crit   Name of the column that should be used to determine the best clusters.
x, y   Original observations.
dist Maximal distance of observation from cluster center.

Details

oregMclust and circMclust return a matrix containing not only the parameters of the found clusters but the value of the heights of the corresponding local maxima as well as how often each cluster is found. Both are reasonable criteria for choosing 'best' clusters, which can be done by bestMclust. Additional criteria could be the number of observations projected to each cluster or the number of observations lying in a certain neighbourhood of the cluster center point.

projMclust adds a column proj to clust which contains the number of points belonging to each cluster in the sense that each observation belongs to the cluster with shortest orthogonal distance. If clust is coming from circMclust, a second column projrel is added which contains this number relative to the radius of the particular circle.

envMclust adds a column env to clust which contains the number of observations lying in a dist-neighbourhood of each cluster center. If clust is coming from circMclust a second column envrel is added which contains this number relative to the radius of the particular circle.

Value

A matrix of clusters.

Author(s)

Tim Garlipp, <TimGarlipp@gmx.de>


**References**


---

**circMclust**

*Circular Clustering*

**Description**

Computation of cluster center points for circular regression data. A cluster method based on redescending M-estimators is used.

**Usage**

```r
circmclust(datax, datay, bw, 
method = "const", prec = 4, 
minsx = min(datax), maxsx = max(datax), nx = 10, 
minsy = min(datay), maxsy = max(datay), ny = 10, 
minsr = 0.01 * max(datax, datay), 
maxsr = (max(datax, datay) - min(datax, datay)), 
nr = 10, nsc = 5, nc = NULL, 
minsd = NULL, maxsd = NULL, 
brminx = minsx, brmaxx = maxsx, 
brminy = minsy, brmaxy = maxsy, 
brminr = minsr, brmaxr = maxsr, 
brmaxit = 1000)
```

```r
## S3 method for class 'circMclust'
plot(x, datax, datay, ccol="black", clty=1, clwd=3, ...) 
## S3 method for class 'circMclust'
print(x, ...) 
```

**Arguments**

- `datax`, `datay`: numerical vectors of coordinates of the observations.
- `bw`: positive number. Bandwidth for the cluster method.
- `method`: optional string. Method of choosing starting values for maximization. Possible values are:
  - "const": a constant number of circles is used. By default, `nx*ny` equidistant midpoints within the range of the observations with `nr` different radii are used as starting circles. The domain of the midpoints and radii can optionally be given by `[minsx, maxsx]`, `[minsy, maxsy]`, and `[minsr, maxsr]`.
  - "all": every circle through any three observations is used.
• "prob": Clusters are searched iteratively with randomly chosen starting circles until either no new clusters are found (default), or until \( nc \) clusters are found. The precision of distinguishing the clusters can be tuned with the parameter \( \text{prec} \). In each iteration \( nc \) times a circle through three randomly chosen observations is used as starting value. With the parameters \( \text{minsd} \) and \( \text{maxsd} \) the minimal and maximal distance of these observations could be limited.

\[\text{nx}, \text{ny}\]
optional positive integer. Number of starting midpoints for method "const"

\[\text{nr}\]
optional positive integer. Number of starting radiiuses for method "const"

\[\text{prec}\]
optional positive integer. Tuning parameter for distinguishing different clusters, which is passed to \text{deldupMclust}.

\[\text{minsx}, \text{maxsx}, \text{minsy}, \text{maxsy}, \text{minsr}\]
optional numbers determining the domain of starting midpoints and the range of radii for method "const"

\[\text{maxsr}\]
optional number determining the maximum radius used as starting value. Note that this is valid for all methods while \( \text{minsx}, \text{maxsx}, \text{minsy}, \text{maxsy}, \text{minsr} \) are only used for method "const".

\[\text{nsc}\]
optional positive integer. Number of starting circles in each iteration for method "prob".

\[\text{nc}\]
optional positive integer. Number of clusters to search if method "const" is chosen. Note that if \( nc \) is too large, i.e., \( nc \) clusters cannot be found, the function does not terminate. Attention! Using Windows, it is impossible to interrupt the routine manually in this case!

\[\text{minsd}, \text{maxsd}\]
optional positive numbers. Minimal and maximal distance of starting points which are used for method "const".

\[\text{brminx}, \text{brmaxx}, \text{brminy}, \text{brmaxy}, \text{brminr}, \text{brmaxr}\]
optional numbers. The maximization is stopped if the midpoint leaves the domain \([\text{brminx}, \text{brmaxx}] \times [\text{brminy}, \text{brmaxy}]\) or if the radius leaves \([\text{brminr}, \text{brmaxr}]\).

\[\text{brmaxit}\]
optional positive integer. Since the maximization could be very slow in some cases, depending on the starting value, the maximization is stopped after \( \text{brmaxit} \) iterations.

\[\text{x}\]
object returned by \text{circMclust}

\[\text{ccol, clty, clwd}\]
optional graphic parameters used for plotting the circles.

\[\ldots\]
additional parameters passed to \text{plot}.

Details

\text{circMclust} implements a cluster method using local maxima of redescending M-estimators for the case of circular regression. This method is based on a method introduced by Mueller and Garlipp in 2003 (see references).

See also \text{bestMclust}, \text{projMclust}, and \text{envMclust} for choosing the 'best' clusters out of all found clusters.
**Value**

Numerical matrix containing one row for every found cluster circle. The columns "cx" and "cy" are their midpoints and "r" are the radii.

The columns "value" and "count" give the value of the objective function and the number how often each cluster is found.

**Author(s)**

Tim Garlipp, <timgarlipp@gmx.de>

**References**


**See Also**

bestMclust, projMclust, envMclust, deldupMclust

**Examples**

```r
z = (1:100 * pi)/50
x = c(sin(z) * 10 + 20, sin(z) * 30 + 80) + rnorm(200,0,2)
y = c(cos(z) * 10 + 20, cos(z) * 30 + 80) + rnorm(200,0,2)

circ = circMclust(x, y, 5, method = "prob",
prec = 1, nsc = 20, minsd = 10, maxsd = 40)
bestMclust(circ, 2)
plot(bestMclust(circ, 2), x, y)
```

---

**deldupMclust**

*Delete duplicate found clusters*

**Description**

Delete clusters differing only by rounding errors or having maximization value zero.

**Usage**

```r
deldupMclust(clust, prec = NULL,
ncol = NULL, dz = TRUE)
```
edgecluster

Arguments

clust  
numerical matrix whose columns contain the parameters of the clusters.
prec  
optional positive integer. Number of decimal places for rounding.
ncol  
number of columns describing the clusters. See details.
dz  
optional boolean. With \( \text{dz} = \text{TRUE} \), those clusters for which the objective function has value 0 are deleted.

Details

Since clusters found by \text{oregMclust} or \text{circMclust} often differ only by rounding errors, the function \text{deldupMclust} can be used for rounding and deleting duplicates. If \text{clust} has a column named "count", its values are summed appropriately. Otherwise such a column is added.

For parameter \text{clust} the object returned from \text{oregMclust} or \text{circMclust} can be used. Alternatively, an arbitrary matrix can be given, of which the first \text{ncol} columns describe the clusters. The parameter \text{prec} is the number of decimal places for rounding; the default is no rounding. With \text{ncol}, the number of columns that describe the clusters can be given. This is not needed, if \text{clust} is an object returned from \text{oregMclust} or \text{circMclust}.

Value

An object of the same type as \text{clust}.

Author(s)

Tim Garlipp, \text{<Timgarlipp@gmx.de>}

References


See Also

\text{oregMclust}, \text{circMclust}

edgecluster  
Edge detection in noisy images

Description

\text{edgecluster} is a simple combination of \text{edgepoints} and \text{oregMclust}. It just passes the results of \text{edgepoints} to \text{oregMclust}. 
edgecluster

Usage

edgecluster(data, h1n, h2n, maxval, 
bw = max(h1n, h2n)/qnorm(0.975),
asteps = 4, estimator = "M_median",
kernel = "gauss", score = "gauss",
sigma = 1, kernelfunc = NULL)

Arguments

data See description of edgepoints.
h1n, h2n See description of edgepoints.
asteps See description of edgepoints.
estimator See description of edgepoints.
kernel See description of edgepoints.
score See description of edgepoints.
sigma See description of edgepoints.
kernelfunc See description of edgepoints.
maxval See description of eplist.
bw See description of oregmclust.

Value

A list of two numerical matrices. The first matrix contains the results of oregmclust, which are the 'edgeclusters'. The second matrix contains the result of edgepoints.

Author(s)

Tim Garlipp, <TimGarlipp@gmx.de>

See Also

dgepoints, oregmclust, eplist

Examples

# generate a 60x60 zero matrix
y = matrix(rep(0, 60 * 60), nrow = 60)
# set a square-shaped set of elements to 1
y[21:40, 21:40] = 1
# add some noise
y = y + matrix(rnorm(60 * 60, 0, 0.2), nrow = 60)
# plot it
image(y, col = gray(seq(0, 1, 1/255)))

# find edge points of the square-shaped object
ec = edgecluster(y, 0.05, 0.05, 0.7,
estimator = "M_median", kernel = "gauss")
plot(bestMclust(ec[[1]], 4), ec[[2]], xlim = c(0, 1), ylim = c(0, 1))
edgepoints  

Edge detection in noisy images

Description

Detection of edge points by the difference of two rotated and asymmetric Kernel- or M-Kernel-Estimators.

Usage

```r
edgepoints(data, h1n, h2n, asteps = 4,
            estimator = "kernel", kernel = "mean",
            score = "gauss", sigma = 1,
            kernelfunc = NULL, margin = FALSE)
```

Arguments

data  numerical matrix representation of the (noisy) image.

h1n, h2n  positive numbers. Bandwidth for the kernels.

asteps  optional positive integer. Number of different angles used.

estimator  optional string. Estimator used within the windows. Possible values are:

- "kernel": Kernel-Estimators. The used kernel function can be selected by means of kernel.
- "M_mean": M-Kernel-Estimators with mean as starting value. The used kernel function can be selected by means of kernel, the score function can be chosen with score.
- "M_median": M-Kernel-Estimators with median as starting value. The used kernel function can be selected by means of kernel, the score function can be chosen with score.
- "median": Median, what is a special M-Kernel-Estimator.
- "test_mean": Multiple Test for equal means in both windows for every angle.
- "test_median": Multiple Test for equal means in both windows for every angle.

kernel  optional string. Kernel function for estimator = "kernel", estimator = "M_mean", or estimator = "M_median". Possible values are:

- "mean": Rectangular kernel. With estimator = "kernel", this gives an ordinary mean estimator. With estimator = "M_mean" or estimator = "M_median", this gives an M-Estimator.
- "linear": Linear kernel function. The distance of the observations to the common midpoint of both windows is linearly measured.
- "linear2": Linear kernel function. The distance of the observations to the midpoint of the window they belong to is linearly measured.
edgepoints 9

- "gauss": Density of the normal distribution with sd = 0.5 and zero outside [-1,1]x[-1,1].
- "func": Arbitrary kernel function given by kernelfunc.

score optional string. Score function for M-Kernel-Estimators if estimator = "M_mean" or estimator = "M_median". Possible values are:
- "gauss": negative density of the normal distribution. The deviation can be given by means of parameter sigma.
- "huber": The Huber score function is the absolute value (median) within an interval [-c, c] and the square function (mean) outside this interval. The value of c can be given by means of the parameter sigma.

sigma optional positiv number. Parameter for the score function "gauss" or "huber".

cornelfunc optional function taking two numbers as arguments and returning a positive number. Used as kernel function given kernel = "func". Note that the function should be zero outside [-1,1]x[-1,1] and that only one function must be handed over for both windows. The 'lower' part of the domain, e.g., [-1,1][-1,0], is used within one window while the 'upper' part is used within the other.

margin Optional value. Results near the margin are in general not very reasonable. Setting margin = TRUE, they are calculated nevertheless. With margin = FALSE, the returned matrices have the same dimension as data but the jump heights at the margin are set to zero. Setting margin = "cut", the returned matrices are cut down by the margins. The default is margin = FALSE.

Details

edgepoints implements several versions of the RDKE method, introduced by Qiu in 1997. The original method, which uses kernel estimates, is a generalized version which uses M-Kernel-Estimators and two test procedures. The test procedures are multiple tests for different angles for the hypothesis of equal means (or medians) in both windows. All methods apply rotating and scaling in the correct order (see Garlipp, 2004).

Value

A list of two numerical matrices. The first matrix contains the maximal jump height for every pixel if the chosen estimator is not a test procedure, and p-values otherwise. The second matrix contains the angle which leads to the maximal jump height or minimal p-value.

Author(s)

Tim Garlipp, <timgarlipp@gmx.de>

References


See Also
eplist

Examples

```r
## produce a matrix representation of a simple
## noisy image showing a black rectangle
y = matrix(rep(0L, 60 * 60), nrow = 60)
y[21:40, 21:40] = 1
y = y + matrix(rnorm(60 * 60, 0, 0.2), nrow = 60)
image(y, col = gray(seq(0, 1, 1/255)))

## find the rectangle's edge points
ye = edgepoints(y, 0.05, 0.05, estimator = "M_median", kernel = "gauss")
image(ye[[1]] > 0.7, col = gray(c(1, 0)))
```

Description

The matrices returned by `edgepoints` are converted into a list of edge points and a list of corresponding angles. This is useful for processing the results of `edgepoints` by `oregMclust`.

Usage

eplist(data, maxval, test = FALSE, xc = NULL, yc = NULL)

Arguments

data list object returned from `edgepoints`.
maxval positive numbers. Critical value for deciding whether a pixel belongs to an edge or not.
test optional boolean. Must be set to `TRUE` if `edgepoints` was used with estimator = "test_mean" or estimator = "test_median". Then, `maxval` is the level of the test.
xc, yc optional numerical vectors defining the coordinates of the edge points. A pixel with jump height `data[[1]][i,j]` gets the coordinates `(xc[i], yc[j])`. By default, the coordinates are assumed as equidistant within `[0,1]`, e.g., for an `(n x m)` matrix of jump heights, the pixel at position `(i,j)` gets the coordinates `(i/n, j/m)`.

Value

A numerical matrix. The first two columns contain the coordinates of the pixels for which the detected jump height is larger than `maxval` (or smaller than `maxval` if `test = TRUE`). The third column contains the corresponding angles.
**oregMclust**

**Description**

Computation of center points for regression data by means of orthogonal regression. A cluster method based on redescending M-estimators is used.

**Usage**

```r
oregMclust(datax, datay, bw, method = "const",
          xrange = range(datax), yrange = range(datay),
          prec = 4, na = 1, sa = NULL, nl = 10, nc = NULL,
          brmaxit = 1000)
```

```r
regparm(reg)
```

```r
## S3 method for class 'oregMclust'
plot(x, datax, datay, prec = 3, rcol = "black",
     rltty = 1, rlwd = 3, ...)
```

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

**Arguments**

- `datax`, `datay`: numerical vectors of coordinates of the observations. Alternatively, a matrix with two or three columns can be given. Then, the first two columns are interpreted as coordinates of the observations and, if available, the third is passed to parameter `sa`.
- `bw`: positive number. Bandwidth for the cluster method.
- `method`: optional string. Method of choosing starting values for maximization. Possible values are:
  - "const": a constant number of angles for every observation is used. By default, one horizontal line through any observation is used as starting value. If a value for parameter `na` is passed, `na` lines through any observation are used. Alternatively, with the parameter `sa` a proper starting angle for every observation can be specified. In this case, `na` is ignored. The length of `sa` must be the number of observations.
• "all": every line through any two observations is used.
• "prob": Clusters are searched iteratively with randomly chosen starting values until either no new clusters are found (default), or until nc clusters are found. The precision of distinguishing the clusters can be tuned with the parameter prec. In each iteration, nl times a line through two randomly chosen observations is used as starting value.

**xrange, yrange**
optional numerical intervals describing the domains of the observations. This is only used for normalization of the data. Note that both intervals should have approximately the same length or should be transformed otherwise. This is not done automatically, since this transformation affects the choice of the bandwidth.

**prec**
optional positive integer. Tuning parameter for distinguishing different clusters, which is passed to deldupMclust.

**na**
optional positive integer. Number of angles per observation used as starting values for method = "const" (default).

**sa**
optional numerical vector. Angles (within \([0, \pi]\)) used as starting values for method = "const" (default).

**nl**
optional positive integer. Number of starting lines in each iteration for method = "prob".

**nc**
optional positive integer. Number of clusters to search if method "const" is chosen. Note that if nc is too large, i.e., nc clusters cannot be found, the function does not terminate. Attention! Using Windows, it is impossible to interrupt the routine manually in this case!

**brmaxit**
optional positive integer. Since the maximization could be very slow in some cases depending on the starting value, the maximization is stopped after brmaxit iterations.

**reg, x**
object returned from oregMclust.

**rcol, rlt, rlwd**
optional graphic parameters used for plotting regression lines.

**...**
additional parameters passed to plot.

**Details**

oregMclust implements a cluster method based on redescending M-estimators for the case of orthogonal regression. This method is introduced by Mueller and Garlipp in 2003 (see references). regparm transforms the columns "alpha" and "beta" to "intercept" and "slope".

See also bestMclust, projMclust, and envMclust for choosing the 'best' clusters out of all found clusters.

**Value**

A numerical matrix containing one row for every found regression center line. The columns "alpha" and "beta" are their parameters in the representation \((\cos(\alpha), \sin(\alpha)) \cdot (x, y) = \beta\), where alpha is within \([0, \pi]\). For the alternative representation \(y = mx + b\), the return value can be passed to regparm.

The columns "value" and "count" give the value of the objective function and the number how often they are found.
Author(s)

Tim Garlipp, <TimGarlipp@gmx.de>

References


See Also

bestMclust, projMclust, envMclust, deldupMclust

Examples

```r
x = c(rnorm(100, 0, 3), rnorm(100, 5, 3))
y = c(-2 * x[1:100] - 5, 0.5 * x[101:200] + 30)/2
x = x + rnorm(200, 0, 0.5)
y = y + rnorm(200, 0, 0.5)

reg = oregMclust(x, y, 1, method = "prob")
reg = projMclust(reg, x, y)
reg
plot(bestMclust(reg, 2, crit = "proj"), x, y)
```
Index

*Topic **cluster**
  - bestMclust, 2
  - circMclust, 3
  - deldupMclust, 5
  - oregMclust, 11

*Topic **multivariate**
  - edgecluster, 6
  - edgepoints, 8
  - eplist, 10

*Topic **nonparametric**
  - edgecluster, 6
  - edgepoints, 8
  - eplist, 10

*Topic **regression**
  - bestMclust, 2
  - circMclust, 3
  - deldupMclust, 5
  - oregMclust, 11

*Topic **robust**
  - bestMclust, 2
  - circMclust, 3
  - deldupMclust, 5
  - edgecluster, 6
  - edgepoints, 8
  - eplist, 10
  - oregMclust, 11

bestMclust, 2, 4, 5, 12, 13

circMclust, 3, 6

deldupMclust, 4, 5, 5, 12, 13

edgecluster, 6
edgepoints, 6, 7, 8, 10, 11
envMclust, 4, 5, 12, 13
envMclust (bestMclust), 2
eplist, 7, 10, 10

oregMclust, 6, 7, 10, 11

plot.circMclust (circMclust), 3
plot.oregMclust (oregMclust), 11
print.circMclust (circMclust), 3
print.oregMclust (oregMclust), 11
projMclust, 4, 5, 12, 13
projMclust (bestMclust), 2
regparm (oregMclust), 11