

# Package ‘edci’

February 14, 2012

**Version** 1.1-0

**Date** 2010-02-05

**Title** Edge Detection and Clustering in Images

**Author** Tim Garlipp <TimGarlipp@gmx.de>

**Description** Detection of edgepoints 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.

**Maintainer** Tim Garlipp <TimGarlipp@gmx.de>

**License** GPL-2

**Repository** CRAN

**Date/Publication** 2010-02-08 09:03:14

## R topics documented:

bestMclust . . . . .	2
circMclust . . . . .	3
deldupMclust . . . . .	5
edgecluster . . . . .	6
edgepoints . . . . .	8
eplist . . . . .	10
oregMclust . . . . .	11

<b>Index</b>	<b>14</b>
--------------	-----------

---

bestMclust	<i>Choose 'best' clusters</i>
------------	-------------------------------

---

**Description**

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 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 relativ 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 relativ to the radius of the particular circle.

**Value**

Both functions return a matrix of clusters.

**Author(s)**

Tim Garlipp, <TimGarlipp@gmx.de>

## References

Mueller, C.H., Garlipp, T. (2003) Simple consistent cluster methods based on redescending M-estimators with an application to edge identification in images, to appear in *JMVA*.

---

circMclust

*Circular Clustering*

---

## Description

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

## Usage

```
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)
```

```
## 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,  $nx \cdot ny$  equidistant midpoints within the range of the observations with `nr` different radiuses are used as starting circles. The domain of the midpoints and radiuses 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 `prec`. In each iteration `nc` times a circle through three randomly chosen observations is used as starting value. With the parameters `minsd` and `maxsd` the minimal and maximal distance of these observations could be limited.

<code>nx, ny</code>	optional positive integer. Number of starting midpoints, if <code>method="const"</code> is chosen.
<code>nr</code>	optional positive integer. Number of starting radiuses, if <code>method="const"</code> is chosen.
<code>prec</code>	optional positive integer. Tuning parameter for distinguishing different clusters, which is passed to <a href="#">deldupMclust</a> .
<code>minsx, maxx, minsy, maxsy, minsr</code>	optional numbers determining the domain of starting midpoints and the range of radiuses, if <code>method="const"</code> is chosen.
<code>maxsr</code>	optional number determining the maximum radius used as starting value. Note, that this is valid for all methods while <code>minsx, maxx, minsy, maxsy, and minsr</code> are only used by <code>method="const"</code> .
<code>nsc</code>	optional positive integer. Number of starting circles in each iteration, if <code>method="prob"</code> is chosen.
<code>nc</code>	optional positive integer. Number of clusters to search, if <code>method="prob"</code> is chosen. Note that, if <code>nc</code> is too large, e.g. <code>nc</code> clusters cannot be found, the function does not terminate. Attention! Using Windows, the routine cannot even be broken manually in this case!
<code>minsd, maxsd</code>	optional positive numbers. Minimal and maximal distance of starting points which are used, if <code>method="prob"</code> is chosen.
<code>brminx, brmaxx, brminy, brmaxy, brminr, brmaxr</code>	optional numbers. The maximization is stopped, if the midpoint leaves the domain <code>[brminx,brmaxx]x[brminy,brmaxy]</code> or if the radius leaves <code>[brminr,brmaxr]</code> .
<code>brmaxit</code>	optional positive integer. Since the maximization could be very slow in some cases depending on the starting value, the maximization is stopped after <code>brmaxit</code> iterations.
<code>x</code>	object returned by <code>circMclust</code>
<code>ccol, clty, clwd</code>	optional graphic parameters used for plotting the circles.
<code>...</code>	Additional parameters passed to <code>plot</code> .

## Details

`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 [bestMclust](#), [projMclust](#), and [envMclust](#) for choosing the 'real' clusters out of the found.

**Value**

Return value is a numerical matrix containing one row for every found cluster circle. The columns "cx" and "cy" are their midpoints and "r" are the radiuses.

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**

Mueller, C.H., Garlipp, T. (2003) Simple consistent cluster methods based on redescending M-estimators with an application to edge identification in images, to appear in *JMVA*.

**See Also**

[bestMclust](#), [projMclust](#), [envMclust](#), [deldupMclust](#)

**Examples**

```
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**

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

**Arguments**

clust	numerical matrix, which columns contain the parameters of the clusters.
prec	optional positive integer. Number of decimal places for rounding.
ncol	Number of columns describing the clusters.
dz	optional boolean. With dz=TRUE clusters are deleted, for which the objective function has value 0.

**Details**

Since clusters found by `oregMclust` or `circMclust` often differ only by rounding errors, the function `deldupMclust` can be used for rounding and deleting duplicates. If `clust` has a column named `count` their values are summed appropriate. Otherwise such a column is added.

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

**Value**

Return value is an object of the same type as `clust`.

**Author(s)**

Tim Garlipp, <TimGarlipp@gmx.de>

**References**

Mueller, C.H., Garlipp, T. (2003) Simple consistent cluster methods based on redescending M-estimators with an application to edge identification in images, to appear in *JMVA*.

**See Also**

[oregMclust](#), [circMclust](#)

---

edgecluster

*Edge detection in noisy images*

---

**Description**

`edgecluster` is a simple combination of [edgepoints](#) and [oregMclust](#). It just passes the results of [edgepoints](#) to [oregMclust](#).

**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 <a href="#">edgepoints</a> .
h1n, h2n	See description of <a href="#">edgepoints</a> .
asteps	See description of <a href="#">edgepoints</a> .
estimator	See description of <a href="#">edgepoints</a> .
kernel	See description of <a href="#">edgepoints</a> .
score	See description of <a href="#">edgepoints</a> .
sigma	See description of <a href="#">edgepoints</a> .
kernelfunc	See description of <a href="#">edgepoints</a> .
maxval	See description of <a href="#">eplist</a> .
bw	See description of <a href="#">oregMclust</a> . See default gives the observations within the 'edgecluster-window' 95% of the weight.

**Value**

Return value is 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**

[edgepoints](#), [oregMclust](#), [eplist](#)

**Examples**

```
y <- matrix(rep(0,60*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)))

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 edgepoints by the difference of two rotated and asymmetric Kernel- or M-Kernel-Estimators.

### Usage

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

### Arguments

data	numerical matrix describing 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: <ul style="list-style-type: none"> <li>• "kernel": Kernel-Estimators. The used kernel function can be chosen by kernel.</li> <li>• "M_mean": M-Kernel-Estimators with mean as starting value. The used kernel function can be chosen by kernel, the score function can be chosen by score.</li> <li>• "M_median": M-Kernel-Estimators with median as starting value. The used kernel function can be chosen by kernel, the score function can be chosen by score.</li> <li>• "median": Median, what is a special M-Kernel-Estimator.</li> <li>• "test_mean": Multiple Test for equal means in both windows for every angle.</li> <li>• "test_median": Multiple Test for equal means in both windows for every angle.</li> </ul>
kernel	optional string. Kernel function if estimator="kernel", estimator="M_mean", or estimator="M_median". Possible values are: <ul style="list-style-type: none"> <li>• "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.</li> </ul>

	<ul style="list-style-type: none"> <li>• "linear": Linear kernel function. The distance of the observations to the common midpoint of both windows is linearly measured.</li> <li>• "linear2": Linear kernel function. The distance of the observations to the midpoint of the window they belong to is linearly measured.</li> <li>• "gauss": Density of the normal distribution with <math>sd=0.5</math> and zero outside <math>[-1,1] \times [-1,1]</math>.</li> <li>• "func": Arbitrary kernel function given by <code>kernelfunc</code>.</li> </ul>
score	optional string. Score function for M-Kernel-Estimators if <code>estimator="M_mean"</code> or <code>estimator="M_median"</code> . Possible values are: <ul style="list-style-type: none"> <li>• "gauss": negativ density of the normal distribution. The deviation can be given by parameter <code>sigma</code>.</li> <li>• "huber": The Huber score function is the absolute value (median) within an interval <math>[-c,c]</math> and the square function (mean) outside this interval. The value of <code>c</code> can be given by the parameter <code>sigma</code>.</li> </ul>
sigma	optional positiv number. Parameter for the score function "gauss" or "huber".
kernelfunc	optional function taking two numbers as arguments and returning a positive number. Used as kernelfunction if <code>kernel="func"</code> is given. Note that the function should be zero outside $[-1,1] \times [-1,1]$ and that only one function is given for both windows. The 'lower' part of the domain, e.g. $[-1,1] \times [-1,0]$ , 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 <code>margin=TRUE</code> they are calculated nevertheless. With <code>margin=FALSE</code> the returned matrices have the same dimension as data but the jump heights at the margin are set to zero. Setting <code>margin="cut"</code> the returned matrices are cutted down by the margins. The default is <code>margin=FALSE</code> .

## 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, that there are equal means (or medians) in both windows. All methods apply rotating and scaling in the correct order (see Garlipp, 2004).

## Value

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

## Author(s)

Tim Garlipp, <TimGarlipp@gmx.de>

## References

- Garlipp, T. (2004), On Robust Jump Detection in Regression Surface with Applications to Image Analysis, *Carl-von-Ossietzky-Universitaet Oldenburg, Dissertation*
- Qiu, P. (1997), Nonparametric Estimation of Jump Surface, *The Indian Journal of Statistics*, 59A, No.2, 268-294.

## See Also

[eplist](#)

## Examples

```
y <- matrix(rep(0,60*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)))

ye <- edgepoints(y,0.05,0.05,estimator="M_median",kernel="gauss")
image(ye[[1]]>0.7,col=gray(c(1,0)))
```

---

eplist

*Conversion of Matrices returned by edgepoints*

---

## Description

The matrices returned by [edgepoints](#) are converted into a list of edgepoints 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 <a href="#">edgepoints</a> .  |
| maxval | positive numbers. Critical value for deciding, if an pixel belongs to an edge or not.   |
| test   | optional boolean. Must be set to TRUE if <a href="#">edgepoints</a> was used with <code>estimator="test_mean"</code> or <code>estimator="test_median"</code> . Then maxval is the level of the test.  |
| xc,yc  | optional numerical vectors defining the coordinates of the edgepoints. A pixel with jump height data[[1]][i,j] gets the coordinates (xc[i],yc[j]). As default, the coordinates are assumed as equidistant within [0,1], e.g. if the matrix of jump heights is a n-m-matrix, the pixel at position i,j gets the coordinates (i/n,j/m). |

**Value**

A numerical matrix is returned. 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.

**Author(s)**

Tim Garlipp, <TimGarlipp@gmx.de>

**See Also**

[edgepoints](#)

---

oregMclust

*Orthogonal Regression Clustering*

---

**Description**

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

**Usage**

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

regparm(reg)

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

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

**Arguments**

<code>datax, datay</code>	numerical vectors of coordinates of the observations. Alternatively, a matrix with two or three columns can be given. The first two columns are interpreted as coordinates of the observations and, if available, the third is passed to parameter <code>sa</code> .
<code>bw</code>	positive number. Bandwidth for the cluster method.
<code>method</code>	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 `na` is given, `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 `n1` times a line through two randomly chosen observations is used as starting value.

<code>xrange, yrange</code>	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.
<code>prec</code>	optional positive integer. Tuning parameter for distinguishing different clusters, which is passed to <a href="#">deldupMclust</a> .
<code>na</code>	optional positive integer. Number of angles per observation used as starting values if <code>method="const"</code> is chosen (default).
<code>sa</code>	optional numerical vector. Angles (within $[0, 2\pi)$ ) used as starting values if <code>method="const"</code> is chosen (default).
<code>n1</code>	optional positive integer. Number of starting lines in each iteration, if <code>method="prob"</code> is chosen.
<code>nc</code>	optional positive integer. Number of clusters to search, if <code>method="prob"</code> is chosen. Note that, if <code>nc</code> is too large, e.g. <code>nc</code> clusters cannot be found, the function does not terminate. Attention! Using Windows, the routine cannot even be broken manually in this case!
<code>brmaxit</code>	optional positive integer. Since the maximization could be very slow in some cases depending on the starting value, the maximization is stopped after <code>brmaxit</code> iterations.
<code>reg, x</code>	Object returned from <code>oregMclust</code> .
<code>rcol, r1ty, r1wd</code>	optional graphic parameters used for plotting regression lines.
<code>...</code>	Additional parameters passed to <code>plot</code> .

## 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 "intersept" and "slope".

See also [bestMclust](#), [projMclust](#), and [envMclust](#) for choosing the 'real' clusters out of the found.

**Value**

Return value is a numerical matrix containing one row for every found regression center line. The Columns "alpha" and "beta" are their parameters in the representation  $(\cos(\alpha), \sin(\alpha))(x, y)' = \beta$ , where alpha is within  $[0, 2\pi)$ . For 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**

Mueller, C.H., Garlipp, T. (2003) Simple consistent cluster methods based on redescending M-estimators with an application to edge identification in images, to appear in *JMVA*.

**See Also**

[bestMclust](#), [projMclust](#), [envMclust](#), [deldupMclust](#)

**Examples**

```
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