# Package 'ktaucenters'

August 3, 2019

**Type** Package

**Title** Robust Clustering Procedures

**Version** 0.1.0

**Author** Juan Domingo Gonzalez [cre, aut],
Victor J. Yohai [aut],
Ruben H. Zamar [aut]

**Maintainer** Juan Domingo Gonzalez <juanrst@hotmail.com>

**Description** A clustering algorithm similar to K-Means is implemented, it has two main advantages, namely (a) The estimator is resistant to outliers, that means that results of estimator are still correct when
there are atypical values in the sample and (b) The estimator is efficient, roughly speaking, if there are no outliers in the sample, results will be similar than those obtained by a classic algorithm (K-Means).
Clustering procedure is carried out by minimizing the overall robust scale so-called tau scale. (see Gonzalez, Yohai and Zamar (2019) <arxiv:1906.08198>).

**License** GPL (>= 2)

**Encoding** UTF-8

**Depends** R (>= 2.10), MASS, methods, dplyr, dbscan, stats, GSE

**LazyData** true

**RoxygenNote** 6.1.1

**Suggests** jpeg, tclust, knitr

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2019-08-03 08:20:02 UTC

## R topics documented:

---

`denpoints` *denpoints*

---

## Description

`denpoints` Estimates the densities values of a sample.

## Usage

```
denpoints(x, k = 4)
```

## Arguments

| | |
|---|---|
| `x` | A distance matrix calculated on `data` or a matrix |
| `k` | The number of nearest neighbors to calculate local point density |

## Details

For a fixed `y`, density of `y` is defined as the sum of `distance(y,z)` on all `z` that are the k-nearest neighbors of `y`

## Value

| | |
|---|---|
| `dpoints` | A real vector containing the density values for each point |

## Author(s)

Juan Domingo Gonzalez <juanrst@hotmail.com>

## References

Hasan AM, et al. Robust partitional clustering by outlier and density insensitive seeding. Pattern Recognition Letters, 30(11), 994-1002, 2009.

## Examples

```
#generate normal data in dimension 2
X=matrix(rnorm(1000),ncol=2)
a<- denpoints(x=X,k=4)


#### ten most isolated points
most_isolated=order(a)[1:10]

### plotting results: (most isolated points should be shown in green)
plot(X)
points(X[ most_isolated, ], pch=19,col=3)
```

---

`derpsiOpt` *derpsiOpt the derivative of the psi function (see* `psiOpt`*)*

---

## Description

derpsiOpt the derivative of the psi function (see `psiOpt`)

## Usage

```
derpsiOpt(x, cc)
```

## Arguments

| | |
|---|---|
| `x` | A real number. |
| `cc` | a tunning constant. |

## Value

rho'(`x` / `cc`)

## Examples

```
psiOpt(x=0.5,cc=1)
```

`improvedktaucenters` *improvedktaucenters*

## Description

Robust Clustering algorithm for non-spherical data. This function estimate clusters taking into account that clusters may have different size, volume or orientation.

## Usage

```
improvedktaucenters(X, K, cutoff = 0.999, nstart = 5,
  INITcenters = NULL)
```

## Arguments

| | |
|---|---|
| `X` | numeric matrix of size n x p. |
| `K` | the number of cluster. |
| `cutoff` | optional argument for getOutliers - quantiles of chi-square to be used as a threshold for outliers detection, defaults to 0.999 |
| `nstart` | optional the number of trials that the base algorithm ktaucenters_aux is run at the first stage. #' If it is greater than 1 and center is not set as NULL, a random set of (distinct) rows in x is chosen as the initial centres for each try. |
| `INITcenters` | numeric matrix of size K x p indicating the initial centers for that clusters and robust covarianze matrices will be computed, if it is set as NULL the algorithm will compute @param INITcenters from ktaucenters routine. Set to NULL by default. |

## Value

A list including the estimated K centers and clusters labels for the observations

- `centers`: matrix of size K x p, with the estimated K centers.
- `cluster`: array of size n x 1 integers labels between 1 and K.
- `tauPath`: sequence of tau scale values at each iterations.
- `Wni`: numeric array of size n x 1 indicating the weights associated to each observation.
- `emptyClusterFlag`: a boolean value. True means that in some iteration there were clusters totally empty.
- `niter`: number of iterations untill convergence is achived or maximun number of iteration is reached.
- `sigmas`: a list containing the k covariance matrices found by the procedure at its second step.
- `outliers`: indices observation that can be considered as outliers.

## References

Gonzalez, J. D., Yohai, V. J., & Zamar, R. H. (2019). Robust Clustering Using Tau-Scales. arXiv preprint arXiv:1906.08198.

## Examples

```
# Generate Sintetic data (three normal cluster in two dimension)
# clusters have different shapes and orentation.
# The data is contaminated uniformly (level 20%).

################################################
#### Start data generating process ############
################################################

# generates base clusters
set.seed(1)
Z1 <- c(rnorm(100,0),rnorm(100,0),rnorm(100,0))
Z2 <- rnorm(300);
X <-  matrix(0, ncol=2,nrow=300);
X[,1]=Z1;X[,2]=Z2
true.cluster= c(rep(1,100),rep(2,100),rep(3,100))

# rotate, expand and translate base clusters
theta=pi/3;
aux1=matrix(c(cos(theta),-sin(theta),sin(theta),cos(theta)),nrow=2)
aux2=sqrt(4)*diag(c(1,1/4))
B=aux1%*%aux2%*%t(aux1)
X[true.cluster==3,]=X[true.cluster==3,]%*%aux2%*%aux1 + matrix(c(5,2),byrow = TRUE,nrow=100,ncol=2)
X[true.cluster==2,2]=X[true.cluster==2,2]*5
X[true.cluster==1,2]=X[true.cluster==1,2]*0.1
X[true.cluster==1, ]=X[true.cluster==1,]+ matrix(c(-5,-1),byrow = TRUE,nrow=100,ncol=2)
### Generate 60 sintetic outliers (contamination level 20%)

outliers=sample(1:300,60)
X[outliers, ] <- matrix(runif( 40, 2 * min(X), 2 * max(X) ),
                                ncol = 2, nrow = 60)
################################################
#### END data generating process ############
################################################

################################################
### Applying the algortihm #################
################################################
ret=improvedktaucenters(X,K=3,cutoff=0.999)

################################################
### plotting results #######################
################################################
par(mfrow=c(2,1))
#' plot(X,main="actual clusters")
for (j in 1:3){
 points(X[true.cluster==j,],pch=19, col=j+1)
}
points(X[outliers,],pch=19,col=1)
plot(X,main="clusters estimation")
for (j in 1:3){
```

```
 points(X[ret$cluster==j,],pch=19, col=j+1)
}
points(X[ret$outliers,],pch=19)
```

---

ktaucenters *ktaucenters*

---

## Description

Robust Clustering algorithm based on centers, a robust and efficient version of KMeans.

## Usage

```
ktaucenters(X, K, centers = NULL, tolmin = 1e-06, NiterMax = 100,
  nstart = 1, startWithKmeans = TRUE, startWithROBINPD = TRUE,
  cutoff = 0.999)
```

## Arguments

| | |
|---|---|
| X | numeric matrix of size n x p. |
| K | the number of cluster. |
| centers | a matrix of size K x p containing the K initial centers, one at each matrix-row. If centers is NULL a random set of (distinct) rows in X are chosen as the initial centres. |
| tolmin | a tolerance parameter used for the algorithm stopping rule |
| NiterMax | a maximun number of iterations used for the algorithm stopping rule |
| nstart | the number of trials that the base algorithm ktaucenters_aux is run. If it is greater than 1 and center is not set as NULL, a random set of (distinct) rows in X will be chosen as the initial centres. |
| startWithKmeans | TRUE if kmean centers values is included as starting point. |
| startWithROBINPD | TRUE if ROBINDEN estimator is included as starting point |
| cutoff | optional argument for outliers detection - quantiles of chi-square to be used as a threshold for outliers detection, defaults to 0.999 |

## Value

A list including the estimated K centers and labels for the observations

- centers: matrix of size K x p, with the estimated K centers.
- cluster: array of size n x 1 integers labels between 1 and K.
- tauPath: sequence of tau scale values at each iterations.
- Wni: numeric array of size n x 1 indicating the weights associated to each observation.

- `emptyClusterFlag`: a boolean value. True means that in some iteration there were clusters totally empty
- `niter`: number of iterations untill convergence is achived or maximun number of iteration is reached
- `di`: distance of each observation to its assigned cluster-center
- `outliers`: indices observation that can be considered as outliers

## References

Gonzalez, J. D., Yohai, V. J., & Zamar, R. H. (2019). Robust Clustering Using Tau-Scales. arXiv preprint arXiv:1906.08198.

## Examples

```
# Generate Sintetic data (three cluster well separated)
Z <- rnorm(600);
mues <- rep(c(-3, 0, 3), 200)
X <-  matrix(Z + mues, ncol=2)

# Generate 60 sintetic outliers (contamination level 20%)
X[sample(1:300,60), ] <- matrix(runif( 40, 3 * min(X), 3 * max(X) ),
                                ncol = 2, nrow = 60)

### Applying the algortihm ####
sal <- ktaucenters(
     X, K=3, centers=X[sample(1:300,3), ],
     tolmin=1e-3, NiterMax=100)

#### plotting  the clusters####
par(mfrow = c(1,1));

par(mfrow = c(1,2))
plot(X,type = "n", main = "ktaucenters (Robust) \n outliers: solid black dots")
points(X[sal$cluster==1,],col=2);
points(X[sal$cluster==2,],col=3);
points(X[sal$cluster==3,],col=4)
points(X[sal$outliers,1], X[sal$outliers,2], pch=19)

### Applying a classical (non Robust) algortihm ###
sal <- kmeans(X, centers=3,nstart=100)

### plotting the clusters ###
plot(X, type ="n", main = "kmeans (Classical)")
points(X[sal$cluster==1,],col=2);
points(X[sal$cluster==2,],col=3);
points(X[sal$cluster==3,],col=4)
```

ktaucenters_aux *ktaucenters_aux*

## Description

Robust Clustering algorithm based on centers, a robust and efficient version of KMeans.

## Usage

```
ktaucenters_aux(X, K, centers, tolmin, NiterMax)
```

## Arguments

| | |
|---|---|
| X | A matrix of size n x p. |
| K | The number of clusters. |
| centers | matrix of size K x p containing the K initial centers, one at each matrix-row. |
| tolmin | tolerance parameter used for the algorithm stopping rule |
| NiterMax | a maximun number of iterations used for the algorithm stopping rule |

## Value

A list including the estimated K centers and labels for the observations

- centers: matrix of size K x p, with the estimated K centers.
- cluster: array of size n x 1 integers labels between 1 and K.
- tauPath: sequence of tau scale values at each iterations.
- Wni: numeric array of size n x 1 indicating the weights associated to each observation.
- emptyClusterFlag: a boolean value. True means that in some iteration there were clusters totally empty
- niter: number of iterations untill convergence is achived or maximun number of iteration is reached
- didistance of each observation to its assigned cluster-center

## Note

Some times, if the initial centers are wrong, the algorithm converges to a non-optimal (local) solution. To avoid that, the algorithm must be run several times. This task is carried out by ktaucenters

## References

Gonzalez, J. D., Yohai, V. J., & Zamar, R. H. (2019). Robust Clustering Using Tau-Scales. arXiv preprint arXiv:1906.08198.

## See Also

ktaucenters

## Examples

```
# Generate Sintetic data (three cluster well separated)
Z=rnorm(600);
mues=rep(c(0,10,20),200)
X= matrix(Z+mues,ncol=2)

# Applying the algortihm
sal = ktaucenters_aux(
X, K=3, centers=X[sample(1:300,3), ],
tolmin=1e-3, NiterMax=100)

#plot the results
plot(X,type="n")
points(X[sal$cluster==1,],col=1);
points(X[sal$cluster==2,],col=2);
points(X[sal$cluster==3,],col=3);
```

---

`mars_screw` *Intensity and saturation values of a picture from mars.*

---

## Description

A dataset containing the Intensity and Saturation values of a picture from Mars taken from Rover Curiosity.

## Usage

```
mars_screw
```

## Format

A list containing information about pixels of a picture form mars mainly containing red sand and metal form Rover itself. List include

- SI_matrix: A matrix with 5063 rows and 128 columns. Elements 1 to 64 of each row indicate the Saturation values of pixels in a square cell 8 x 8 whereas elements 65 to 128 of each row indicate the cell's Intensity values.
- geographic_matrix: An integer matrix of dimension 5063 x 2, each row indicates each square cell's locations (x-axis y-axis) at the picture.
- screw_index: the index corresponding to the screw observation (screw_index=4180)

## Source

https://www.nasa.gov/mission_pages/msl/multimedia/pia16225.html

## Examples

```
# upload matrix
A <- mars_screw$SI_matrix;
B <- mars_screw$geographic_matrix
screw_index <- mars_screw$screw_index
## looking for the brightest cells
maximun_at_each_cell<- apply(A[ ,1:64], 1, max)
ten_brightest_cells <-order(maximun_at_each_cell, decreasing=TRUE)[1:10]

## plot locations where the ten brightest cells are.
plot(B, pch=19 )
points(B[ten_brightest_cells, ], pch=19,col="yellow" )

## plot locations where the screw are.
points(B[screw_index, ], pch=19,col="blue" )
```

---

`Mscale` *Mscale the M scale of an univariate sample (see reference below)*

---

## Description

Mscale the M scale of an univariate sample (see reference below)

## Usage

```
Mscale(u, b = 0.5, c)
```

## Arguments

| | |
|---|---|
| u | an univariate sample of size n. |
| b | the desired break down point |
| c | a tuning constant, if consistency to standard normal distribution is desired use `normal_consistency_constants` |

## Value

the Mscale value

## References

Maronna, R. A., Martin, R. D., Yohai, V. J., & Salibian-Barrera, M. (2018). Robust statistics: theory and methods (with R). Wiley.

## Examples

```
Mscale(u=rnorm(100),c=1)
```

normal_consistency_constants *normal_consistency_constants*

## Description

constants previously computed in order the M scale value is consistent with the standard normal distribution for the optimal rho function considered in `rhoOpt`. (Constant were computed from p=1 till p =400)

## Usage

```
normal_consistency_constants(p)
```

## Arguments

| | |
|---|---|
| p | dimension where observation lives |

## Value

cvalue

## References

[1] Maronna, R. A., Martin, R. D., Yohai, V. J., & Salibián-Barrera, M. (2018). Robust statistics: theory and methods (with R). Wiley. [2] Salibian-Barrera, M., Willems, G., & Zamar, R. (2008). The fast-tau estimator for regression. Journal of Computational and Graphical Statistics, 17(3), 659-682.

## Examples

```
p=5;
n=1000
X=matrix(rnorm(n*p), ncol=p)
dist=apply(X,1,function(t){sqrt(sum(t^2))})
s= Mscale(dist,b=0.5, c=normal_consistency_constants(p))

### variable s should be near from one for all p values between 1 and 400.
```

psiOpt *psiOpt psi function is the derivative of optimal rho function (see rhoOpt)*

## Description

psiOpt psi function is the derivative of optimal rho function (see rhoOpt)

## Usage

```
psiOpt(x, cc)
```

## Arguments

| | |
|---|---|
| x | A real number. |
| cc | a tunning constant. |

## Value

rho"(x/ cc)

## Examples

```
psiOpt(x=0.5,cc=1)
```

rhoOpt *rhoOpt function*

## Description

An implementation of quasi optimal rho functions following reference [1]

## Usage

```
rhoOpt(x, cc)
```

## Arguments

| | |
|---|---|
| x | A real number. |
| cc | a tunning constant. |

## Value

rho(x/cc)

## References

[1] Salibian-Barrera, M., Willems, G., & Zamar, R. (2008). The fast-tau estimator for regression. Journal of Computational and Graphical Statistics, 17(3), 659-682.

## Examples

```
val= rhoOpt(x=0.5,cc=1)
```

---

`ROBINDEN` *ROBINDEN (ROBust INitialization based on inverse DENsity estimator)*

---

## Description

`ROBINDEN` searches for k initial cluster seeds for k-means-based clustering methods.

## Usage

```
ROBINDEN(D, data, k, mp = 10)
```

## Arguments

| | |
|---|---|
| `D` | A distance matrix calculated on `data`. |
| `data` | A data matrix with n observations and p variables. |
| `k` | The number of cluster centers to find. |
| `mp` | The number of the nearest neighbors to find dense regions by LOF, the default is 10. |

## Details

The centers are the observations located in the most dense region and far away from each other at the same time. In order to find the observations in the highly dense region, ROBINPOINTDEN uses point density estimation (instead of Local Outlier Factor, Breunig et al (2000)), see more details.

## Value

| | |
|---|---|
| `centers` | A numeric vector of k initial cluster centers corresponding to the k indices of observations. |
| `idpoints` | A real vector containing the inverse density values of each point (observation). |

## Note

this is a slightly modified version of ROBIN algorithm implementation done by Sarka Brodinova <sarka.brodinova@tuwien.ac.at>.

## Author(s)

Juan Domingo Gonzalez <juanrst@hotmail.com>

## References

Hasan AM, et al. Robust partitional clustering by outlier and density insensitive seeding. Pattern Recognition Letters, 30(11), 994-1002, 2009.

## See Also

lof

## Examples

```
K=5;
nk=100
Z <- rnorm(2 * K * nk);
centers_aux <- -floor(K/2):floor(K/2)
mues <- rep(5*centers_aux,2*nk*K )
X <-  matrix(Z + mues, ncol=2)
# Generate sintetic outliers (contamination level 20%)
X[sample(1:(nk * K),(nk * K) * 0.2), ] <-matrix(runif((nk * K) * 0.2 * 2,
                                        3 * min(X), 3 * max(X)),
                                        ncol = 2, nrow = (nk * K) * 0.2)
res <- ROBINDEN(D =dist(X), data=X, k = K);
# plot the Initial centers found
plot(X)
points(X[res$centers,],pch=19,col=4,cex=2)
```

# Index