# Package 'otrimle'

June 27, 2017

**Type** Package

**Title** Robust Model-Based Clustering

**Description** Performs robust cluster analysis allowing for outliers and noise that cannot be fitted by any cluster. The data are modelled by a mixture of Gaussian distributions and a noise component, which is an improper uniform distribution covering the whole Euclidean space. Parameters are estimated by (pseudo) maximum likelihood. This is fitted by a EM-type algorithm. See Coretto and Hennig (2016) <doi:10.1080/01621459.2015.1100996>, and Coretto and Hennig (2017) <arXiv:1309.6895>.

**Version** 1.1

**Date** 2017-06-27

**Author** Pietro Coretto [aut, cre], Christian Hennig [aut]

**Maintainer** Pietro Coretto <pcoretto@unisa.it>

**Imports** stats, utils, graphics, grDevices, mclust, parallel, foreach, doParallel

**License** GPL (>= 2)

**LazyData** TRUE

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2017-06-27 16:15:33 UTC

## R topics documented:

## banknote *Swiss Banknotes Data*

### Description

Data from Tables 1.1 and 1.2 (pp. 5-8) of Flury and Riedwyl (1988). There are six measurements made on 200 Swiss banknotes (the old-Swiss 1000-franc). The banknotes belong to two classes of equal size: *genuine* and *counterfeit*.

### Usage

```
data(banknote)
```

### Format

A `data.frame` of dimension `200x7` with the following variables:

**Class** a `factor` with classes: `genuine`, `counterfeit`

**Length** Length of bill (mm)

**Left** Width of left edge (mm)

**Right** Width of right edge (mm)

**Bottom** Bottom margin width (mm)

**Top** Top margin width (mm)

**Diagonal** Length of diagonal (mm)

### Source

Flury, B. and Riedwyl, H. (1988). *Multivariate Statistics: A practical approach.* London: Chapman & Hall.

## InitClust *Robust Initialization for Model-based Clustering Methods*

### Description

Computes the initial cluster assignment based on a combination of nearest neighbor based noise detection, and agglomerative hierarchical clustering based on maximum likelihood criteria for Gaussian mixture models.

### Usage

```
InitClust(data , G , k = 3 , knnd.trim = 0.5 , modelName='VVV')
```

## Arguments

| | |
|---|---|
| data | A numeric vector, matrix, or data frame of observations. Rows correspond to observations and columns correspond to variables. Categorical variables and NA values are not allowed. |
| G | An integer specifying the number of clusters. |
| k | An integer specifying the number of considered nearest neighbors per point used for the denoising step (see *Details*). |
| knnd.trim | A number in (0,1) which defines the proportion of points initialized as noise. Tipically knnd.trim<=0.5 (see *Details*). |
| modelName | A character string indicating the covariance model to be used. Possible models are:<br>"E": equal variance (one-dimensional)<br>"V" : spherical, variable variance (one-dimensional)<br>"EII": spherical, equal volume<br>"VII": spherical, unequal volume<br>"EEE": ellipsoidal, equal volume, shape, and orientation<br>"VVV": ellipsoidal, varying volume, shape, and orientation (default).<br>See *Details*. |

## Details

The initialization is discussed in details in Coretto and Hennig (2016). Two steps are performed:

*Denoising step:* for each data point compute its kth-nearest neighbors distance (k-NND). All points with k-NND larger than the (1-knnd.trim)-quantile of the k-NND are initialized as noise. Intepretaion of k is that: (k-1), but not k, points close together may still be interpreted as noise or outliers

*Clustering step:* perform the model-based hierarchical clustering (MBHC) proposed in Fraley (1998). This step is performed using hc. The input argument modelName is passed to hc. See *Details* of hc for more details.

## Value

An integer vector specifying the initial cluster assignment with 0 denoting noise/outliers.

## References

Fraley, C. (1998). Algorithms for model-based Gaussian hierarchical clustering. *SIAM Journal on Scientific Computing* 20:270-281.

Coretto, P. and C. Hennig (2017). Consistency, breakdown robustness, and algorithms for robust improper maximum likelihood clustering. arXiv preprint available at arXiv:1309.6895.

## See Also

hc

## Examples

```
## Load  Swiss banknotes data
data(banknote)
x <- banknote[,-1]

## Initial clusters with default arguments
init <- InitClust(data = x, G = 2)
print(init)

## Perform otrimle
a <- otrimle(data = x, G = 2, initial = init,
             logicd = c(-Inf, -50, -10), ncores = 1)
plot(a, what="clustering", data=x)
```

---

`otrimle` *Optimally Tuned Robust Improper Maximum Likelihood Clustering*

---

## Description

`otrimle` searches for `G` approximately Gaussian-shaped clusters with/without noise/outliers. The method's tuning controlling the noise level is adaptively chosen based on the data.

## Usage

```
otrimle(data, G, initial = NULL, logicd = NULL, npr.max = 0.5, erc = 50,
        iter.max = 500, tol = 1e-06, ncores = NULL, monitor = TRUE)
```

## Arguments

| | |
|---|---|
| `data` | A numeric vector, matrix, or data frame of observations. Rows correspond to observations and columns correspond to variables. Categorical variables and `NA` values are not allowed. |
| `G` | An integer specifying the number of clusters. |
| `initial` | An integer vector specifying the initial cluster assignment with `0` denoting noise/outliers. If `NULL` (default) initialization is performed using `InitClust`. |
| `logicd` | A vector defining a grid of *log(icd)* values, where *icd* denotes the improper constant density. If `logicd=NULL` a default grid is considered. A pure Gaussian Mixture Model fit (obtained when `log(icd)=-Inf`) is included in the default search path. |
| `npr.max` | A number in `(0,1)` specifying the maximum proportion of noise/outliers. This defines the *noise proportion constraint*. |
| `erc` | A number `>=1` specifying the maximum allowed ratio between within-cluster covariance matrix eigenvalues. This defines the *eigenratio constraint*. `erc=1` enforces spherical clusters with equal covariance matrices. A large `erc` allows for large between-cluster covariance discrepancies. In order to facilitate the setting of `erc`, it is suggested to scale the columns of `data` (see `scale`) whenever measurement units of the different variables are grossly incompatible. |

| | |
|---|---|
| iter.max | An integer value specifying the maximum number of iterations allowed in the underlying ECM-algorithm. |
| tol | Stopping criterion for the underlying ECM-algorithm. An ECM iteration stops if two successive improper log-likelihood values are within tol. |
| ncores | an integer value defining the number of cores used for parallel computing. When ncores=NULL (default), the number r of available cores is detected, and (r-1) of them are used (See Details). |
| monitor | logical. If TRUE progress messages are printed on screen. |

## Details

The otrimle function computes the OTRIMLE solution based on the ECM-algorithm (expectation conditional maximization algorithm) proposed in Coretto and Hennig (2017).

The otrimle criterion is minimized over the logicd grid of log(icd) values using parallel computing based on the foreach. Note that, depending on the BLAS/LAPACK setting, increasing ncores may not produce the desired reduction in computing time. The latter happens when optimized linear algebra routines are in use (e.g. OpenBLAS, Intel Math Kernel Library (MKL), etc.). These optimized shared libraries already implement multithreading. Therefore, in this case increasing ncores may only reduce the computing time marginally.

Occasionally, there may be datasets for which the function does not provide a solution based on default arguments. This corresponds to code=0 and flag=1 or flag=2 in the output (see *Value*-section below). This usually happens when some (or all) of the following circumstances occur: (i) erc is too large; (ii) npr.max is too large; (iii) choice of the initial partition. Regarding (i) and (ii) it is not possible to give numeric references because whether these numbers are too large/small strongly depends on the sample size and the dimensionality of the data. References given below explain the relationship between these quantities.

It is suggested to try the following whenever a code=0 non-solution occurs. Set the logicd range wide enough (e.g. logicd=seq(-500,-5, length=50)), choose erc=1, and a low choice of npr.max (e.g. npr.max=0.02). Monitor the solution with the criterion profiling plot (plot.otrimle). According to the criterion profiling plot change logicd, and increase erc and npr.max up to the point when a "clear" minimum in the criterion profiling plot is obtained. If this strategy does not work it is suggested to experiment with a different initial partitions (see initial above).

Note that an earlier approximate version of the algorithm was originally proposed in Coretto and Hennig (2016). Software for the original version of the algorithm can be found in the supplementary materials of Coretto and Hennig (2016).

## Value

An S3 object of class 'otrimle' providing the optimal (according to the OTRIMLE criterion) clustering. Output components are as follows:

| | |
|---|---|
| code | An integer indicator for the convergence. code=0 if no solution is found (see *Details*); code=1 if at the optimal icd value the corresponding EM-algorithm did not converge within em.iter.max; code=2 convergence is fully achieved. |

| | |
|---|---|
| `flag` | A character string containing one or more flags related to the EM iteration at the optimal icd. `flag=1` if it was not possible to prevent the numerical degeneracy of improper posterior probabilities (`tau` value below). `flag=2` if enforcement of the *noise proportion constraint* failed for numerical reasons. `flag=3` if the *noise proportion constraint* has been successfully applied at least once. `flag=4` if the *eigenratio constraint* has been successfully applied at least once. |
| `iter` | Number of iterations performed in the underlying EM-algorithm at the optimal `icd`. |
| `logicd` | Resulting value of the optimal `log(icd)`. |
| `iloglik` | Resulting value of the improper likelihood. |
| `criterion` | Resulting value of the OTRIMLE criterion. |
| `npr` | Estimated expected noise proportion. |
| `cpr` | Vector of estimated expected cluster proportions (notice that `sum(cpr)=1-npr`). |
| `mean` | A matrix of dimension `ncol(data) x G` containing the mean parameters of each cluster (column-wise). |
| `cov` | An array of size `ncol(data) x ncol(data) x G` containing the covariance matrices of each cluster. |
| `tau` | A matrix of dimension `nrow(data) x {1+G}` where `tau[i, 1+j]` is the estimated (improper) posterior probability that the *i*th observation belongs to the *j*th cluster. `tau[i,1]` is the estimated (improper) posterior probability that *i*th observation belongs to the noise component. |
| `smd` | A matrix of dimension `nrow(data) x G` where `smd[i,j]` is the squared Mahalanobis distance of `data[i,]` from `mean[,j]` according to `cov[,,j]`. |
| `cluster` | A vector of integers denoting cluster assignments for each observation. It's `0` for observations assigned to noise/outliers. |
| `size` | A vector of integers with sizes (counts) of each cluster. |
| `optimization` | A data.frame with the OTRIMLE optimization profiling. For each value of `log(icd)` explored by the algorithm the data.frame stores `criterion, iloglik, code` and `flag`. |

## References

Coretto, P. and C. Hennig (2016). Robust improper maximum likelihood: tuning, computation, and a comparison with other methods for robust Gaussian clustering. *Journal of the American Statistical Association*, Vol. 111(516), pp. 1648-1659. doi: 10.1080/01621459.2015.1100996

Coretto, P. and C. Hennig (2017). Consistency, breakdown robustness, and algorithms for robust improper maximum likelihood clustering. arXiv preprint available at arXiv:1309.6895.

## See Also

`plot.otrimle`, `InitClust`, `rimle`,

## Examples

```
## Load  Swiss banknotes data
data(banknote)
x <- banknote[,-1]

## Perform otrimle clustering with default arguments
set.seed(1)
a <- otrimle(data=x, G=2, logicd=c(-Inf, -50, -10), ncores=1)


## Plot clustering
plot(a, data=x, what="clustering")

## Plot OTRIMLE criterion profiling
plot(a, what="criterion")

## Plot Improper log-likelihood profiling
plot(a, what="iloglik")

## P-P plot of the clusterwise empirical weighted squared Mahalanobis
## distances against the target distribution pchisq(, df=ncol(data))
plot(a, what="fit")
plot(a, what="fit", cluster=1)


## Not run:
## Perform the same example using the finer default grid of logicd
## values using multiple cores
##
a <- otrimle(data = x, G = 2)

## Inspect the otrimle criterion-vs-logicd
plot(a, what = 'criterion')

## The minimum occurs at  a$logicd=-9, and exploring a$optimization it
## cane be seen that the interval [-12.5, -4] brackets the optimal
## solution. We search with a finer grid located around the minimum
##
b <- otrimle(data = x, G = 2, logicd = seq(-12.5, -4, length.out = 25))

## Inspect the otrimle criterion-vs-logicd
plot(b, what = 'criterion')

## Check the difference between the two clusterings
table(A = a$cluster, B = b$cluster)

## Check differences in estimated parameters
##
colSums(abs(a$mean - b$mean))               ## L1 distance for mean vectors
apply({a$cov-b$cov}, 3, norm, type = "F")   ## Frobenius distance for covariances
c(Noise=abs(a$npr-b$npr), abs(a$cpr-b$cpr)) ## Absolute difference for proportions
```

```
## End(Not run)
```

---

`plot.otrimle` *Plot Methods for OTRIMLE Objects*

---

## Description

Plot robust model-based clustering results: scatter plot with clustering information, optimization profiling, and cluster fit.

## Usage

```
## S3 method for class 'otrimle'
plot(x, what=c("criterion","iloglik", "fit", "clustering"),
     data=NULL, margins=NULL, cluster=NULL, ...)
```

## Arguments

| | |
|---|---|
| `x` | Output from `otrimle` |
| `what` | The type of graph. It can be one of the following: `"criterion"` (default), `"iloglik"`, `"fit"`, `"clustering"`. See *Details*. |
| `data` | The data vector, matrix or data.frame (or some transformation of them), used for obtaining the `'otrimle'` object. This is only relevant if `what="clustering"`. |
| `margins` | A vector of integers denoting the variables (numbers of columns of `data`) to be used for a `pairs`-plot if `what="clustering"`. When `margins=NULL` it is set to `1:ncol(data)` (default). |
| `cluster` | An integer denoting the cluster for which the *fit* plot is returned. This is only relevant if `what="fit"`. |
| `...` | further arguments passed to or from other methods. |

## Value

**If** `what="criterion"` A plot with the profiling of the OTRIMLE criterion optimization. Criterion at `log(icd)=-Inf` is always represented.

**If** `what="iloglik"` A plot with the profiling of the improper log-likelihood function along the search path for the OTRIMLE optimization.

**If** `what="fit"` The P-P plot (probability-probability plot) of the weighted empirical distribution function of the Mahalanobis distances of observations from clusters' centers against the target distribution. The target distribution is the Chi-square distribution with degrees of freedom equal to `ncol(data)`. The weights are given by the improper posterior probabilities. If `cluster=NULL` P-P plots are produced for all clusters, otherwise `cluster` selects a single P-P plot at times.

**If** `what="clustering"` A pairwise scatterplot with cluster memberships. Points assigned to the noise/outliers component are denoted by `'+'`.

## References

Coretto, P. and C. Hennig (2016). Robust improper maximum likelihood: tuning, computation, and a comparison with other methods for robust Gaussian clustering. *Journal of the American Statistical Association*, Vol. 111(516), pp. 1648-1659. doi: 10.1080/01621459.2015.1100996

Coretto, P. and C. Hennig (2017). Consistency, breakdown robustness, and algorithms for robust improper maximum likelihood clustering. arXiv preprint available at arXiv:1309.6895.

## See Also

plot.otrimle

## Examples

```
## Load  Swiss banknotes data
data(banknote)
x <- banknote[,-1]

## Perform otrimle clustering on a small grid of logicd values
a <- otrimle(data = x, G = 2, logicd = c(-Inf, -50, -10), ncores = 1)
print(a)

## Plot clustering
plot(a, data = x, what = "clustering")

## Plot clustering on selected margins
plot(a, data = x, what = "clustering", margins = 4:6)

## Plot clustering on the first two principal components
z <- scale(x) %*%   eigen(cor(x), symmetric = TRUE)$vectors
colnames(z) <- paste("PC", 1:ncol(z), sep = "")
plot(a, data = z, what = "clustering", margins = 1:2)

## Plot OTRIMLE criterion profiling
plot(a, what = "criterion")

## Plot Improper log-likelihood profiling
plot(a, what = "iloglik")

## Fit plot for all clusters
plot(a, what = "fit")

## Fit plot for cluster 1
plot(a, what = "fit", cluster = 1)



## Not run:
## Perform the same example using the finer default grid of logicd
## values using multiple cores
##
a <- otrimle(data = x, G = 2)
```

```
## Inspect the otrimle criterion-vs-logicd
plot(a, what = 'criterion')

## The minimum occurs at  a$logicd=-9, and exploring a$optimization it
## cane be seen that the interval [-12.5, -4] brackets the optimal
## solution. We search with a finer grid located around the minimum
##
b <- otrimle(data = x, G = 2, logicd = seq(-12.5, -4, length.out = 25))

## Inspect the otrimle criterion-vs-logicd
plot(b, what = 'criterion')

## Check the difference between the two clusterings
table(A = a$cluster, B = b$cluster)

## Check differences in estimated parameters
##
colSums(abs(a$mean - b$mean))               ## L1 distance for mean vectors
apply({a$cov-b$cov}, 3, norm, type = "F")   ## Frobenius distance for covariances
c(Noise=abs(a$npr-b$npr), abs(a$cpr-b$cpr)) ## Absolute difference for proportions

## End(Not run)
```

---

`plot.rimle` *Plot Methods for RIMLE Objects*

---

## Description

Plot robust model-based clustering results: scatter plot with clustering information and cluster fit.

## Usage

```
## S3 method for class 'rimle'
plot(x, what=c("fit", "clustering"),
     data=NULL, margins=NULL, cluster=NULL, ...)
```

## Arguments

| | |
|---|---|
| `x` | Output from `rimle` |
| `what` | The type of graph. It can be one of the following: `"fit"` (default), `"clustering"`. See *Details*. |
| `data` | The data vector, matrix or data.frame (or some transformation of them), used for obtaining the `'rimle'` object. This is only relevant if `what="clustering"`. |
| `margins` | A vector of integers denoting the variables (numbers of columns of `data`) to be used for a `pairs`-plot if `what="clustering"`. When `margins=NULL` it is set to `1:ncol(data)` (default). |

`cluster` An integer denoting the cluster for which the *fit* plot is returned. This is only relevant if `what="fit"`.

`...` further arguments passed to or from other methods.

## Value

**If** `what="fit"` The P-P plot (probability-probability plot) of the weighted empirical distribution function of the Mahalanobis distances of observations from clusters' centers against the target distribution. The target distribution is the Chi-square distribution with degrees of freedom equal to `ncol(data)`. The weights are given by the improper posterior probabilities. If `cluster=NULL` P-P plots are produced for all clusters, otherwise `cluster` selects a single P-P plot at times.

**If** `what="clustering"` A pairwise scatterplot with cluster memberships. Points assigned to the noise/outliers component are denoted by `'+'`.

## References

Coretto, P. and C. Hennig (2016). Robust improper maximum likelihood: tuning, computation, and a comparison with other methods for robust Gaussian clustering. *Journal of the American Statistical Association*, Vol. 111(516), pp. 1648-1659. doi: 10.1080/01621459.2015.1100996

Coretto, P. and C. Hennig (2017). Consistency, breakdown robustness, and algorithms for robust improper maximum likelihood clustering. arXiv preprint available at arXiv:1309.6895.

## See Also

`otrimle`

## Examples

```
## Load  Swiss banknotes data
data(banknote)
x <- banknote[,-1]

## Perform rimle clustering with default arguments
set.seed(1)
a <- rimle(data = x, G = 2)
print(a)

## Plot clustering
plot(a, data = x, what = "clustering")

## Plot clustering on selected margins
plot(a, data = x, what = "clustering", margins = 4:6)

## Plot clustering on the first two principal components
z <- scale(x) %*%   eigen(cor(x), symmetric = TRUE)$vectors
colnames(z) <- paste("PC", 1:ncol(z), sep = "")
plot(a, data = z, what = "clustering", margins = 1:2)

## Fit plot for all clusters
```

```
plot(a, what = "fit")

## Fit plot for cluster 1
plot(a, what = "fit", cluster = 1)
```

---

`rimle` *Robust Improper Maximum Likelihood Clustering*

---

## Description

`rimle` searches for `G` approximately Gaussian-shaped clusters with/without noise/outliers. The method's tuning controlling the noise level is fixed and is to be provided by the user or will be guessed by the function in a rather quick and dirty way (`otrimle` performs a more sophisticated data-driven choice).

## Usage

```
rimle(data, G, initial=NULL, logicd=NULL, npr.max=0.5, erc=50, iter.max=500, tol=1e-6)
```

## Arguments

| | |
|---|---|
| `data` | A numeric vector, matrix, or data frame of observations. Rows correspond to observations and columns correspond to variables. Categorical variables and `NA` values are not allowed. |
| `G` | An integer specifying the number of clusters. |
| `initial` | An integer vector specifying the initial cluster assignment with `0` denoting noise/outliers. If `NULL` (default) initialization is performed using `InitClust`. |
| `logicd` | A number `log(icd)`, where `0 <= icd < Inf` is the value of the improper constant density (`icd`). This is the RIMLE's tuning for controlling the size of the noise. If `logicd=NULL` (default), an `icd` value is guessed based on the data. A pure Gaussian Mixture Model fit is obtained with `log(icd)=-Inf`. |
| `npr.max` | A number in `(0,1)` specifying the maximum proportion of noise/outliers. This defines the *noise proportion constraint*. |
| `erc` | A number `>=1` specifying the maximum allowed ratio between within-cluster covariance matrix eigenvalues. This defines the *eigenratio constraint*. `erc=1` enforces spherical clusters with equal covariance matrices. A large `erc` allows for large between-cluster covariance discrepancies. In order to facilitate the setting of `erc`, it is suggested to scale the columns of `data` (see `scale`) whenever measurement units of the different variables are grossly incompatible. |
| `iter.max` | An integer value specifying the maximum number of iterations allowed in the ECM-algorithm (see Details). |
| `tol` | Stopping criterion for the underlying ECM-algorithm. An ECM iteration stops if two successive improper log-likelihood values are within `tol`. |

## Details

The rimle function computes the RIMLE solution using the ECM-algorithm proposed in Coretto and Hennig (2017).

There may be datasets for which the function does not provide a solution based on default arguments. This corresponds to code=0 and flag=1 or flag=2 in the output (see *Value*-section below). This usually happens when some (or all) of the following circumstances occur: (i) log(icd) is too large; (ii) erc is too large; (iii) npr.max is too large; (iv) choice of the initial partition. In these cases it is suggested to find a suitable interval of icd values by using the otrimle function. The *Details* section of otrimle suggests several actions to take whenever a code=0 non-solution occurs.

An earlier approximate version of the algorithm was originally proposed in Coretto and Hennig (2016). Software for the original version of the algorithm can be found in the supplementary materials of Coretto and Hennig (2016).

## Value

An S3 object of class 'rimle'. Output components are as follows:

| | |
|---|---|
| code | An integer indicator for the convergence. code=0 if no solution is found (see *Details*); code=1 if the EM-algorithm did not converge within em.iter.max; code=2 convergence is fully achieved. |
| flag | A character string containing one or more flags related to the EM iteration at the optimal icd. flag=1 if it was not possible to prevent the numerical degeneracy of improper posterior probabilities (tau value below). flag=2 if enforcement of the *noise proportion constraint* failed for numerical reasons. flag=3 if enforcement of the *eigenratio constraint* failed for numerical reasons. flag=4 if the *noise proportion constraint* has been successfully applied at least once. flag=5 if the *eigenratio constraint* has been successfully applied at least once. |
| iter | Number of iterations performed in the underlying EM-algorithm. |
| logicd | Value of the log(icd). |
| iloglik | Value of the improper likelihood. |
| criterion | Value of the OTRIMLE criterion. |
| npr | Estimated expected noise proportion. |
| cpr | Vector of estimated expected cluster proportions (notice that sum(cpr)=1-npr). |
| mean | A matrix of dimension ncol(data) x G containing the mean parameters of each cluster (column-wise). |
| cov | An array of size ncol(data) x ncol(data) x G containing the covariance matrices of each cluster. |
| tau | A matrix of dimension nrow(data) x {1+G} where tau[i, 1+j] is the estimated (improper) posterior probability that the *i*th observation belongs to the *j*th cluster. tau[i,1] is the estimated (improper) posterior probability that *i*th observation belongs to the noise component. |
| smd | A matrix of dimension nrow(data) x G where smd[i,j] is the squared Mahalanobis distance of data[i,] from mean[,j] according to cov[,,j]. |
| cluster | A vector of integers denoting cluster assignments for each observation. It's 0 for observations assigned to noise/outliers. |
| size | A vector of integers with sizes (counts) of each cluster. |

## References

Coretto, P. and C. Hennig (2016). Robust improper maximum likelihood: tuning, computation, and a comparison with other methods for robust Gaussian clustering. *Journal of the American Statistical Association*, Vol. 111(516), pp. 1648-1659. doi: 10.1080/01621459.2015.1100996

Coretto, P. and C. Hennig (2017). Consistency, breakdown robustness, and algorithms for robust improper maximum likelihood clustering. arXiv preprint available at arXiv:1309.6895.

## See Also

plot.rimle, InitClust, otrimle,

## Examples

```
## Load  Swiss banknotes data
data(banknote)
x <- banknote[,-1]

## ------------------------------------------------------------------------------
## EXAMPLE 1:
## Perform RIMLE with default inputs
## ------------------------------------------------------------------------------
set.seed(1)
a <- rimle(data = x, G = 2)
print(a)

## Plot clustering
plot(a, data = x, what = "clustering")

## P-P plot of the clusterwise empirical weighted squared Mahalanobis
## distances against the target distribution pchisq(, df=ncol(data))
plot(a, what = "fit")
plot(a, what = "fit", cluster = 1)



## ------------------------------------------------------------------------------
## EXAMPLE 2:
## Compare solutions for different choices of logicd
## ------------------------------------------------------------------------------
set.seed(1)

## Case 1: noiseless solution, that is fit a pure Gaussian Mixture Model
b1 <- rimle(data = x, G = 2, logicd = -Inf)
plot(b1, data=x, what="clustering")
plot(b1, what="fit")

## Case 2: low noise level
b2 <- rimle(data = x, G = 2, logicd = -100)
plot(b2, data=x, what="clustering")
plot(b2, what="fit")
```

```
## Case 3: medium noise level
b3 <- rimle(data = x, G = 2, logicd = -10)
plot(b3, data=x, what="clustering")
plot(b3, what="fit")

## Case 3: large noise level
b3 <- rimle(data = x, G = 2, logicd = 5)
plot(b3, data=x, what="clustering")
plot(b3, what="fit")
```

# Index