Package ‘outlierMBC’

May 28, 2025

Title Sequential Outlier Identification for Model-Based Clustering

Version 0.0.1

Description Sequential outlier identification for Gaussian mixture models using the distribution of Mahalanobis distances. The optimal number of outliers is chosen based on the dissimilarity between the theoretical and observed distributions of the scaled squared sample Mahalanobis distances. Also includes an extension for Gaussian linear cluster-weighted models using the distribution of studentized residuals. Doherty, McNicholas, and White (2025) <doi:10.48550/arXiv.2505.11668>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports ClusterR, dbscan, flexCWM, ggplot2, mixture, mvtnorm, spatstat.univar, stats

Depends R (>= 4.1.0)

LazyData true

NeedsCompilation no

Author Ultán P. Doherty [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-1791-395X>), Paul D. McNicholas [aut] (ORCID: <https://orcid.org/0000-0002-2482-523X>), Arthur White [aut] (ORCID: <https://orcid.org/0000-0002-7268-5163>)

Maintainer Ultán P. Doherty <dohertyu@tcd.ie>

Repository CRAN

Date/Publication 2025-05-28 15:40:02 UTC

Contents

`backtrack` *Move backwards from the minimum to a more conservative solution.*

Description

Given a vector of dissimilarity values, each corresponding to a different number of outliers, this function first finds the index and value of the minimum dissimilarity, then moves backwards from right to left to a reasonable solution with a lower index (i.e. lower number of outliers). Limits are placed on the maximum increase in dissimilarity from a single step (`max_step_rise`) and from all steps (`max_total_rise`), where both are defined in proportion to the minimum dissimilarity value.

Usage

```
backtrack(x, max_total_rise = 0.1, max_step_rise = 0.05)
```

Arguments

`x`	Vector of dissimilarity values corresponding to consecutive and increasing numbers of outliers.
`max_total_rise`	Upper limit for the cumulative increase, as a proportion of the global minimum dissimilarity, from all backward steps.
`max_step_rise`	Upper limit for the increase, as a proportion of the global minimum dissimilarity, from each backward step.

Value

`backtrack` returns a list with two elements, `minimum` and `backtrack`:

`minimum` **is a list with the following elements:** `ind` Index of the minimum solution.
`val` Value of the minimum solution.

`backtrack` **is a list with the following elements:** `ind` Index of the backtrack solution.
`val` Value of the backtrack solution.

Examples

```
ombc_gmm_k3n1000o10 <-
  ombc_gmm(gmm_k3n1000o10[, 1:2], comp_num = 3, max_out = 20)

backtrack(ombc_gmm_k3n1000o10$distrib_diff_vec)
```

`backtrack_gmm` *Fit a Gaussian mixture model to the backtrack solution.*

Description

The backtrack function determines the number of outliers for the backtrack solution and plot_backtrack plots this on a dissimilarity curve. `backtrack_gmm` fits the mixture model corresponding to the number of outliers selected by the backtrack solution (or any manually specified number of outliers).

Usage

```
backtrack_gmm(
  x,
  ombc_out,
  max_total_rise = 0.1,
  max_step_rise = 0.05,
  init_model = NULL,
  init_z = NULL,
  manual_outlier_num = NULL,
  verbose = TRUE
)
```

Arguments

x Data.

ombc_out An "outliermbc_gmm" or "outliermbc_lcwm" object, i.e. an output from ombc_gmm or ombc_lcwm.

max_total_rise Upper limit for the cumulative increase, as a proportion of the global minimum dissimilarity, from all backward steps.

max_step_rise Upper limit for the increase, as a proportion of the global minimum dissimilarity, from each backward step.

init_model Initial mixture model (mixture::gpcm best_model).

init_z Initial component assignment probability matrix.

manual_outlier_num User-specified number of outliers.

verbose Whether the iteration count is printed.

Value

backtrack_gmm returns a list with the following elements:

labels Vector of mixture component labels with outliers denoted by 0.

outlier_bool Logical vector indicating if an observation has been classified as an outlier.

outlier_num Number of observations classified as outliers.

mix Output from mixture::gpcm fitted to the non-outlier observations.

call Arguments / parameter values used in this function call.

Examples

```
ombc_gmm_k3n1000o10 <- ombc_gmm(
  gmm_k3n1000o10[, 1:2],
  comp_num = 3, max_out = 20
)

backtrack_gmm(gmm_k3n1000o10[, 1:2], ombc_gmm_k3n1000o10)
```

---

backtrack_lcwm *Fit a linear cluster-weighted model to the backtrack solution.*

Description

The backtrack function determines the number of outliers for the backtrack solution and plot_backtrack plots this on a dissimilarity curve. backtrack_gmm fits the mixture model corresponding to the number of outliers selected by the backtrack solution (or any manually specified number of outliers).

Usage

```
backtrack_lcwm(
  xy,
  x,
  ombc_lcwm_out,
  max_total_rise = 0.1,
  max_step_rise = 0.05,
  init_z = NULL,
  manual_outlier_num = NULL,
  verbose = TRUE
)
```

Arguments

xy	data.frame containing covariates and response.
x	Covariate data only.
ombc_lcwm_out	An "outliermbc_lcwm" object outputted by ombc_lcwm.
max_total_rise	Upper limit for the cumulative increase, as a proportion of the global minimum dissimilarity, from all backward steps.
max_step_rise	Upper limit for the increase, as a proportion of the global minimum dissimilarity, from each backward step.
init_z	Initial component assignment probability matrix.
manual_outlier_num	User-specified number of outliers.
verbose	Whether the iteration count is printed.

Value

backtrack_gmm returns a list with the following elements:

labels Vector of component labels with outliers denoted by 0.

outlier_bool Logical vector indicating if an observation has been classified as an outlier.

outlier_num Number of observations classified as outliers.

lcwm Output from flexCWM::cwm fitted to the non-outlier observations.

call Arguments / parameter values used in this function call.

Examples

```
gross_lcwm_k3n1000o10 <- find_gross(lcwm_k3n1000o10, 20)

ombc_lcwm_k3n1000o10 <- ombc_lcwm(
  xy = lcwm_k3n1000o10[, c("X1", "Y")],
  x = lcwm_k3n1000o10$X1,
  y_formula = Y ~ X1,
  comp_num = 2,
  max_out = 20,
```

```
  mnames = "V",
  gross_outs = gross_lcwm_k3n1000o10$gross_bool
)

backtrack_lcwm_k3n1000o10 <- backtrack_lcwm(
  xy = lcwm_k3n1000o10[, c("X1", "Y")],
  x = lcwm_k3n1000o10$X1,
  ombc_lcwm_out = ombc_lcwm_k3n1000o10
)
```

`distrib_diff_gmm` *Compute the dissimilarity for a Gaussian mixture model and identify the lowest density observation.*

Description

At each iteration of ombc_gmm, `distrib_diff_gmm` computes the dissimilarity value of the current Gaussian mixture model. It also identifies the observation with the lowest mixture density.

Usage

```
distrib_diff_gmm(x, z, prop, mu, sigma, logdet)
```

Arguments

`x`	Data.
`z`	Component assignment probability matrix.
`prop`	Vector of component proportions.
`mu`	List of component mean vectors.
`sigma`	List of component covariance matrices.
`logdet`	Vector of log-determinants for covariance matrices.

Value

`distrib_diff_gmm` returns a list with the following elements:

`distrib_diff` Aggregated dissimilarity across components.

`distrib_diff_vec` Vector containing dissimilarity value for each component.

`choice_id` Index of observation with lowest mixture density.

`removal_dens` Value of the lowest mixture density.

distrib_diff_lcwm *Compute the dissimilarity for a linear cluster-weighted model and identify the lowest density observation.*

Description

At each iteration of ombc_lcwm, distrib_diff_lcwm computes the dissimilarity value of the current linear cluster-weighted model. It also identifies the observation with the lowest mixture density.

Usage

```
distrib_diff_lcwm(x, z, prop, mu, sigma, mod_list, y_sigma, dd_weight = 0.5)
```

Arguments

x	Covariate data only.
z	Component assignment probability matrix.
prop	Vector of component proportions.
mu	Matrix of component mean vectors.
sigma	Array of component covariance matrices.
mod_list	List of component regression models.
y_sigma	Vector of component regression standard deviations.
dd_weight	A value between 0 and 1 which controls the weighting of the response and covariate dissimilarities when aggregating.

Value

distrib_diff_lcwm_lcwm returns a list with the following elements:

distrib_diff Aggregated dissimilarity across components.

distrib_diff_vec Vector containing dissimilarity value for each component.

choice_id Index of observation with lowest mixture density.

removal_dens Value of the lowest mixture density.

distrib_diff_mat Two-column matrix containing response and covariate dissimilarities across components.

distrib_diff_lcwm_g *Compute the dissimilarity for a single component of a Linear CWM.*

Description

Computes the covariate dissimilarity value, the response dissimilarity value, and their aggregated dissimilarity value. It also obtains the covariate, response, and joint densities for every observation.

Usage

```
distrib_diff_lcwm_g(x, z_g, mu_g, sigma_g, mod_g, y_sigma_g, dd_weight = 0.5)
```

Arguments

x	Covariate data only.
z_g	Component assignment probability vector.
mu_g	Component mean vector for the covariates.
sigma_g	Component covariance matrix for the covariates.
mod_g	Component regression model.
y_sigma_g	Component regression standard deviation for the response.
dd_weight	A value between 0 and 1 which controls the weighting of the response and covariate dissimilarities when aggregating.

Value

distrib_diff_lcwm_lcwm_g returns a list with the following elements:

diff Aggregated dissimilarity value for this component.

dens Joint (covariate & response) density of all observations for this component.

diff_x Covariate dissimilarity value for this component.

diff_y Response dissimilarity value for this component.

dens_x Covariate density of all observations for this component.

dens_y Response density of all observations for this component.

distrib_diff_mahalanobis

Compute the dissimilarity for a single multivariate Gaussian distribution.

Description

Compute the dissimilarity value and observation densities for a single multivariate Gaussian distribution. This could be a whole component in a Gaussian mixture model or the covariate part of a component in a Linear CWM.

Usage

```
distrib_diff_mahalanobis(x, z_g, mu_g, sigma_g, logdet_g)
```

Arguments

x	Data.
z_g	Assignment probability vector for component g.
mu_g	Mean vector for component g.
sigma_g	Covariance matrix for component g.
logdet_g	Log-determinants of covariance matrix for component g.

Value

`distrib_diff_mahalanobis` returns a list with the following elements:

`diff` Dissimilarity value for this component.

`dens` Gaussian density of all observations for this component.

`mahalas` Scaled squared sample Mahalanobis distances for all observations with respect to this component.

distrib_diff_residual

Compute the response dissimilarity for a single component of a Linear CWM.

Description

Computes the response dissimilarity value and the response density for every observation.

Usage

```
distrib_diff_residual(x, z_g, mod_g, y_sigma_g)
```

Arguments

x	Covariate data only.
z_g	Component assignment probability vector.
mod_g	Component regression model.
y_sigma_g	Component regression standard deviation for the response.

Value

distrib_diff_lcwm_residual returns a list with the following elements:

diff Response dissimilarity value for this component.

dens Response density of all observations for this component.

---

find_gross *Find gross outliers.*

Description

The distance of each observation to its k^{th} nearest neighbour is computed. We assume that the largest max_out kNN distances correspond to potential outliers. We select the next largest kNN distance, outside of the top max_out, as a benchmark value. We multiply this benchmark kNN distance by multiplier to get the minimum threshold for our gross outliers. In other words, a gross outlier must have a kNN distance at least multiplier times greater than all of the observations which we do not consider to be potential outliers.

Usage

```
find_gross(
  x,
  max_out,
  multiplier = 3,
  k_neighbours = floor(nrow(x)/100),
  manual_threshold = NULL,
  scale = TRUE
)
```

Arguments

x	Data.
max_out	Maximum number of outliers.
multiplier	Multiplicative factor used to get gross outlier threshold.
k_neighbours	Number of neighbours for dbscan::kNNdist.
manual_threshold	Optional preset threshold.
scale	Logical value controlling whether we apply scale to x.

Value

`find_gross` returns a list with the following elements:

- `gross_choice` A numeric value indicating the elbow's location.
- `gross_bool` A logical vector identifying the gross outliers.
- `gross_curve` ggplot of the highest `2 * max_out` kNN distances in decreasing order.
- `gross_scatter` ggplot of all kNN distances in index order.

`get_init_z` *Obtain an initial clustering as a component assignment matrix.*

Description

Implement the specified initial clustering, either hierarchical clustering or k-means++, and return a binary component assignment matrix.

Usage

```
get_init_z(
  comp_num,
  dist_mat = NULL,
  x = NULL,
  init_method = c("hc", "kmpp"),
  kmpp_seed = NULL
)
```

Arguments

`comp_num`	Number of mixture components.
`dist_mat`	Euclidean distance matrix.
`x`	Data.
`init_method`	Method used to initialise each mixture model.
`kmpp_seed`	Optional seed for k-means++ initialisation.

Value

A component assignment matrix for initialisation.

gmm_k3n1000o10 *Simulated data set consisting of 1000 observations from 3 Gaussian components and 10 outliers.*

Description

This data set was simulated using `simulate_gmm`. There are 500 observations in Component 1, 250 observations in Component 2, and 250 observations in Component 3

Usage

```
gmm_k3n1000o10
```

Format

`gmm_k3n1000o10`:

A data frame with 1010 rows and 3 columns:

X1, X2 Continuous variables.

G Component label: 0 for outliers; 1, 2, or 3 for true points.

Source

For simulation code, see `gmm_k3n1000o10.R` in `data-raw` folder at https://github.com/UltanPDoherty/outlierMBC.

gmm_k3n2000o20 *Simulated data set consisting of 2000 observations from 3 Gaussian components and 20 outliers.*

Description

This data set was simulated using `simulate_gmm`. There are 1000 observations in Component 1, 500 observations in Component 2, and 500 observations in Component 3.

Usage

```
gmm_k3n2000o20
```

Format

`gmm_k3n2000o20`:

A data frame with 2020 rows and 3 columns:

X1, X2 Continuous variables.

G Component label: 0 for outliers; 1, 2, or 3 for true points.

Source

For simulation code, see `gmm_k3n2000o20.R` in `data-raw` folder at https://github.com/UltanPDoherty/outlierMBC.

`gmm_k3n4000o40` *Simulated data set consisting of 4000 observations from 3 Gaussian components and 40 outliers.*

Description

This data set was simulated using `simulate_gmm`. There are 2000 observations in Component 1, 1000 observations in Component 2, and 1000 observations in Component 3.

Usage

```
gmm_k3n4000o40
```

Format

`gmm_k3n4000o40`**:** A data frame with 4040 rows and 3 columns:

X1, X2 Continuous variables.

G Component label: 0 for outliers; 1, 2, or 3 for true points.

Source

For simulation code, see `gmm_k3n4000o40.R` in `data-raw` folder at https://github.com/UltanPDoherty/outlierMBC.

`lcwm_k3n1000o10` *Simulated data set consisting of 1000 observations from 3 Gaussian components and 10 outliers.*

Description

This data set was simulated using `simulate_lcwm`. There are 300 observations in Component 1, 300 observations in Component 2, and 400 observations in Component 3

Usage

```
lcwm_k3n1000o10
```

Format

`lcwm_k3n1000o10`:

A data frame with 1010 rows and 3 columns:

X1 Continuous explanatory variable.

Y Continuous response variable.

G Component label: 0 for outliers; 1, 2, or 3 for true points.

Source

For simulation code, see `lcwm_k3n1000o10.R` in `data-raw` folder at https://github.com/UltanPDoherty/outlierMBC.

`lcwm_k3n2000o20` *Simulated data set consisting of 2000 observations from 3 Gaussian components and 20 outliers.*

Description

This data set was simulated using `simulate_lcwm`. There are 600 observations in Component 1, 600 observations in Component 2, and 800 observations in Component 3.

Usage

```
lcwm_k3n2000o20
```

Format

`lcwm_k3n2000o20`:

A data frame with 2020 rows and 3 columns:

X1 Continuous explanatory variable.

Y Continuous response variable.

G Component label: 0 for outliers; 1, 2, or 3 for true points.

Source

For simulation code, see `lcwm_k3n2000o20.R` in `data-raw` folder at https://github.com/UltanPDoherty/outlierMBC.

`lcwm_k3n4000o40` *Simulated data set consisting of 4000 observations from 3 Gaussian components and 40 outliers.*

Description

This data set was simulated using `simulate_lcwm`. There are 1200 observations in Component 1, 1200 observations in Component 2, and 1600 observations in Component 3.

Usage

```
lcwm_k3n4000o40
```

Format

`lcwm_k3n4000o40`:

A data frame with 4040 rows and 3 columns:

X1 Continuous explanatory variable.

Y Continuous response variable.

G Component label: 0 for outliers; 1, 2, or 3 for true points.

Source

For simulation code, see `lcwm_k3n4000o40.R` in `data-raw` folder at https://github.com/UltanPDoherty/outlierMBC.

`new_outliermbc_gmm` *Constructor for `"outliermbc_gmm"` S3 class.*

Description

Constructor for `"outliermbc_gmm"` S3 class.

Usage

```
new_outliermbc_gmm(x = list())
```

Arguments

`x`	List.

Value

"outliermbc_gmm" S3 object.

new_outliermbc_lcwm — *Constructor for "outliermbc_lcwm" S3 object.*

Description

Constructor for "outliermbc_lcwm" S3 object.

Usage

```
new_outliermbc_lcwm(x = list())
```

Arguments

`x`	List.

Value

"outliermbc_lcwm" S3 object.

ombc_gmm — *Sequentially identify outliers while fitting a Gaussian mixture model.*

Description

This function performs model-based clustering and outlier identification. It does so by iteratively fitting a Gaussian mixture model and removing the observation that is least likely under the model. Its procedure is summarised below:

1. Fit a Gaussian mixture model to the data.
2. Compute a dissimilarity between the theoretical and observed distributions of the scaled squared sample Mahalanobis distances for each mixture component.
3. Aggregate across the components to obtain a single dissimilarity value.
4. Remove the observation with the lowest mixture density.
5. Repeat Steps 1-4 until `max_out` observations have been removed.
6. Identify the number of outliers which minimised the aggregated dissimilarity, remove only those observations, and fit a Gaussian mixture model to the remaining data.

Usage

```
ombc_gmm(
  x,
  comp_num,
  max_out,
  gross_outs = rep(FALSE, nrow(x)),
  init_scheme = c("update", "reinit", "reuse"),
  mnames = "VVV",
  nmax = 1000,
  atol = 1e-08,
  init_z = NULL,
  init_model = NULL,
  init_method = c("hc", "kmpp"),
  init_scaling = FALSE,
  kmpp_seed = 123,
  fixed_labels = NULL,
  verbose = TRUE
)
```

Arguments

`x`	Data.
`comp_num`	Number of mixture components.
`max_out`	Maximum number of outliers.
`gross_outs`	Logical vector identifying gross outliers.
`init_scheme`	Which initialisation scheme to use.
`mnames`	Model names for mixture::gpcm.
`nmax`	Maximum number of iterations for `mixture::gpcm`.
`atol`	EM convergence tolerance threshold for `mixture::gpcm`.
`init_z`	Initial component assignment probability matrix.
`init_model`	Initial mixture model (`mixture::gpcm` `best_model`).
`init_method`	Method used to initialise each mixture model.
`init_scaling`	Logical value controlling whether the data should be scaled for initialisation.
`kmpp_seed`	Optional seed for k-means++ initialisation.
`fixed_labels`	Cluster labels that are known a prior. See `label` argument in `mixture::gpcm`.
`verbose`	Whether the iteration count is printed.

Value

ombc_gmm returns an object of class "outliermbc_gmm", which is essentially a list with the following elements:

`labels` Vector of mixture component labels with outliers denoted by 0.

`outlier_bool` Logical vector indicating if an observation has been classified as an outlier.

`outlier_num` Number of observations classified as outliers.

`outlier_rank` Order in which observations are removed from the data set. Observations which were provisionally removed, including those that were eventually not classified as outliers, are ranked from `1` to `max_out`. All gross outliers have rank `1`. If there are `gross_num` gross outliers, then the observations removed during the main algorithm itself will be numbered from `gross_num + 1` to `max_out`. Observations that were ever removed have rank `0`.

`gross_outs` Logical vector identifying the gross outliers. This is identical to the `gross_outs` vector passed to this function as an argument / input.

`mix` Output from `mixture::gpcm` fitted to the non-outlier observations.

`loglike` Vector of log-likelihood values for each iteration.

`removal_dens` Vector of mixture densities for the removed observations. These are the lowest mixture densities at each iteration.

`distrib_diff_vec` Vector of aggregated cross-component dissimilarity values for each iteration.

`distrib_diff_mat` Matrix of component-specific dissimilarity values for each iteration.

`call` Arguments / parameter values used in this function call.

`version` Version of `outlierMBC` used in this function call.

`conv_status` Logical vector indicating which iterations' mixture models reached convergence during model-fitting.

Examples

```
ombc_gmm_k3n1000o10 <- ombc_gmm(
  gmm_k3n1000o10[, 1:2],
  comp_num = 3, max_out = 20
)

plot_curve(ombc_gmm_k3n1000o10)
```

`ombc_lcwm` *Sequentially identify outliers while fitting a linear cluster-weighted model.*

Description

This function performs model-based clustering, clusterwise regression, and outlier identification. It does so by iteratively fitting a linear cluster-weighted model and removing the observation that is least likely under the model. Its procedure is summarised below:

1. Fit a linear cluster-weighted model to the data.
2. Compute a dissimilarity between the theoretical and observed distributions of the scaled squared sample Mahalanobis distances for each mixture component.
3. Compute a dissimilarity between the theoretical and observed distributions of the scaled squared studentised residuals for each mixture component.
4. Aggregate these two dissimilarities to obtain one dissimilarity value for each component.

5. Aggregate across the components to obtain a single dissimilarity value.
6. Remove the observation with the lowest mixture density.
7. Repeat Steps 1-6 until `max_out` observations have been removed.
8. Identify the number of outliers which minimised the aggregated dissimilarity, remove only those observations, and fit a linear cluster-weighted model to the remaining data.

Usage

```
ombc_lcwm(
  xy,
  x,
  y_formula,
  comp_num,
  max_out,
  gross_outs = rep(FALSE, nrow(x)),
  init_scheme = c("update", "reinit", "reuse"),
  mnames = "VVV",
  nmax = 1000,
  atol = 1e-08,
  init_z = NULL,
  init_method = c("hc", "kmpp"),
  init_scaling = TRUE,
  kmpp_seed = 123,
  verbose = TRUE,
  dd_weight = 0.5
)
```

Arguments

`xy`	`data.frame` containing covariates and response.
`x`	Covariate data only.
`y_formula`	Regression formula.
`comp_num`	Number of mixture components.
`max_out`	Maximum number of outliers.
`gross_outs`	Logical vector identifying gross outliers.
`init_scheme`	Which initialisation scheme to use.
`mnames`	Model names for mixture::gpcm.
`nmax`	Maximum number of iterations for `flexCWM::cwm`.
`atol`	EM convergence threshold for `flexCWM::cwm`.
`init_z`	Initial component assignment probability matrix.
`init_method`	Method used to initialise each mixture model.
`init_scaling`	Logical value controlling whether the data should be scaled for initialisation.
`kmpp_seed`	Optional seed for k-means++ initialisation.
`verbose`	Whether the iteration count is printed.
`dd_weight`	A value between `0` and `1` which controls the weighting of the response and covariate dissimilarities when aggregating.

Value

ombc_lcwm returns an object of class "outliermbc_lcwm", which is essentially a list with the following elements:

labels Vector of mixture component labels with outliers denoted by 0.

outlier_bool Logical vector indicating if an observation has been classified as an outlier.

outlier_num Number of observations classified as outliers.

outlier_rank Order in which observations are removed from the data set. Observations which were provisionally removed, including those that were eventually not classified as outliers, are ranked from 1 to max_out. All gross outliers have rank 1. If there are gross_num gross outliers, then the observations removed during the main algorithm itself will be numbered from gross_num + 1 to max_out. Observations that were ever removed have rank 0.

gross_outs Logical vector identifying the gross outliers. This is identical to the gross_outs vector passed to this function as an argument / input.

lcwm Output from flexCWM::cwm fitted to the non-outlier observations.

loglike Vector of log-likelihood values for each iteration.

removal_dens Vector of mixture densities for the removed observations. These are the lowest mixture densities at each iteration.

distrib_diff_vec Vector of aggregated cross-component dissimilarity values for each iteration.

distrib_diff_mat Matrix of component-specific dissimilarity values for each iteration.

distrib_diff_arr Array of component-specific response and covariate dissimilarity values for each iteration.

call Arguments / parameter values used in this function call.

version Version of outlierMBC used in this function call.

conv_status Logical vector indicating which iterations' mixture models reached convergence during model-fitting.

Examples

```
gross_lcwm_k3n1000o10 <- find_gross(lcwm_k3n1000o10, 20)

ombc_lcwm_k3n1000o10 <- ombc_lcwm(
  xy = lcwm_k3n1000o10[, c("X1", "Y")],
  x = lcwm_k3n1000o10$X1,
  y_formula = Y ~ X1,
  comp_num = 3,
  max_out = 20,
  mnames = "V",
  gross_outs = gross_lcwm_k3n1000o10$gross_bool
)
```

plot.outliermbc_gmm — *plot method for* "outliermbc_gmm" *S3 class.*

Description

plot method for "outliermbc_gmm" S3 class.

Usage

```
## S3 method for class 'outliermbc_gmm'
plot(x, backtrack = FALSE, ...)
```

Arguments

x	List
backtrack	Logical
...	Other arguments

Value

A ggplot

plot.outliermbc_lcwm — *plot method for* "outliermbc_lcwm" *S3 class.*

Description

plot method for "outliermbc_lcwm" S3 class.

Usage

```
## S3 method for class 'outliermbc_lcwm'
plot(x, backtrack = FALSE, ...)
```

Arguments

x	List
backtrack	Logical
...	Other arguments

Value

A ggplot

plot_backtrack *Plot the dissimilarity curve showing the backtrack solution.*

Description

Plots a rescaled dissimilarity curve where the dissimilarity values (y axis) have been divided by their minimum so that the rescaled minimum is at 1. Vertical lines mark the minimum and backtrack solutions.

Usage

```
plot_backtrack(ombc_out, max_total_rise = 0.1, max_step_rise = 0.05)
```

Arguments

ombc_out	An "outliermbc_gmm" or "outliermbc_lcwm" object, i.e. an output from ombc_gmm or ombc_lcwm.
max_total_rise	Upper limit for the cumulative increase, as a proportion of the global minimum dissimilarity, from all backward steps.
max_step_rise	Upper limit for the increase, as a proportion of the global minimum dissimilarity, from each backward step.

Value

plot_backtrack returns a ggplot of the rescaled dissimilarity curve showing the minimum solution and the backtrack solutions.

plot_comparison *Plot multiple dissimilarity curves.*

Description

Given a range of ombc_gmm outputs, each arising from a different model, this function is designed to produce a graphical aid for selecting the best model. It displays the dissimilarity curves from each of these models on the same plot.

Usage

```
plot_comparison(ombc_list)
```

Arguments

ombc_list	A list of outputs from ombc_gmm.

Value

`plot_comparison` returns a ggplot object consisting of multiple dissimilarity curves overlaid on the same plot.

`plot_comparison_bic` *Plot multiple dissimilarity curves.*

Description

Given a range of ombc_gmm outputs, each arising from a different model, this function is designed to produce a graphical aid for selecting the best model. It displays the dissimilarity curves from each of these models on the same plot.

Usage

```
plot_comparison_bic(ombc_list)
```

Arguments

`ombc_list`	A list of outputs from `ombc_gmm`.

Value

`plot_comparison` returns a ggplot object consisting of multiple dissimilarity curves overlaid on the same plot.

`plot_curve` *Plot the dissimilarity curve.*

Description

Given the output from ombc_gmm or ombc_lcwm, this function extracts the dissimilarity value associated with each outlier number and plots them as a curve. It also draws a vertical line at the outlier number which minimised the dissimilarity.

Usage

```
plot_curve(ombc_out)
```

Arguments

`ombc_out`	An `"outliermbc_gmm"` or `"outliermbc_lcwm"` object, i.e. an output from `ombc_gmm` or `ombc_lcwm`.

Value

`plot_curve` returns a ggplot object showing the dissimilarity values as a curve and marking the minimum solution with a vertical line.

plot_selection *Plot dissimilarity values for multiple solutions.*

Description

Given a range of ombc_gmm outputs, each arising from a different model, this function is designed to produce a graphical aid for selecting the best model. It plots the dissimilarity values of the models' minimum and backtrack solutions against their number of components (`x_axis = "comp_num"`), number of outliers (`x_axis = "outlier_num"`), or number of parameters (`x_axis = "param_num"`).

Usage

```
plot_selection(ombc_list, x_axis = c("comp_num", "outlier_num", "param_num"))
```

Arguments

`ombc_list`	A list of outputs from `ombc_gmm`.
`x_axis`	The quantity to be plotted on the x axis.

Value

`plot_selection` return a ggplot object plotting the minimum dissimilarity and backtrack solutions from a number of outputs from `ombc_gmm` versus their number of components, outliers, or parameters.

print.outliermbc_gmm *print method for* `"outliermbc_gmm"` *S3 class.*

Description

print method for `"outliermbc_gmm"` S3 class.

Usage

```
## S3 method for class 'outliermbc_gmm'
print(x, backtrack = FALSE, max_total_rise = 0.1, max_step_rise = 0.05, ...)
```

Arguments

`x`	List
`backtrack`	Logical
`max_total_rise`	Upper limit for the cumulative increase, as a proportion of the global minimum dissimilarity, from all backward steps.
`max_step_rise`	Upper limit for the increase, as a proportion of the global minimum dissimilarity, from each backward step.
`...`	Other arguments

Value

A ggplot

`print.outliermbc_lcwm` *print method for* "`outliermbc_lcwm`" *S3 class.*

Description

print method for "`outliermbc_lcwm`" S3 class.

Usage

```
## S3 method for class 'outliermbc_lcwm'
print(x, backtrack = FALSE, max_total_rise = 0.1, max_step_rise = 0.05, ...)
```

Arguments

`x`	List
`backtrack`	Logical
`max_total_rise`	Upper limit for the cumulative increase, as a proportion of the global minimum dissimilarity, from all backward steps.
`max_step_rise`	Upper limit for the increase, as a proportion of the global minimum dissimilarity, from each backward step.
`...`	Other arguments

Value

A ggplot

`simulate_gmm` *Simulate data from a Gaussian mixture model with outliers.*

Description

Simulates data from a Gaussian mixture model, then simulates outliers from a hyper-rectangle, with a rejection step to ensure that the outliers are sufficiently unlikely under the model.

Usage

```
simulate_gmm(
  n,
  mu,
  sigma,
  outlier_num,
  seed = NULL,
  crit_val = 0.9999,
  range_multiplier = 1.5,
  verbose = TRUE,
  max_rejection = 1e+06
)
```

Arguments

`n`	Vector of component sizes.
`mu`	List of component mean vectors.
`sigma`	List of component covariance matrices.
`outlier_num`	Desired number of outliers.
`seed`	Seed.
`crit_val`	Critical value for uniform sample rejection.
`range_multiplier`	How much greater should the range of the Uniform samples be than the range of the Normal samples?
`verbose`	Whether a message should be printed if a high number of outliers are being simulated. This suggests that many simulated outliers are being rejected and the other arguments may need to be adjusted.
`max_rejection`	Maximum number of simulated outliers to be rejected.

Details

The simulated outliers are sampled from a Uniform distribution over a hyper-rectangle. For each dimension, the hyper-rectangle is centred at the midpoint between the maximum and minimum values for that variable from all of the Gaussian observations. Its width in that dimension is the distance between the minimum and maximum values for that variable multiplied by the value of `range_multiplier`. If `range_multiplier = 1`, then this hyper-rectangle is the axis-aligned minimum bounding box for all of the Gaussian data points in this data set.

The `crit_val` ensures that it would have been sufficiently unlikely for a simulated outlier to have been sampled from any of the Gaussian components. The Mahalanobis distances of a proposed outlier from each component's mean vector with respect to that component's covariance matrix are computed. If any of these Mahalanobis distances are smaller than the critical value of the appropriate Chi-squared distribution, then the proposed outlier is rejected. In summary, for a Uniform sample to be accepted, it must be sufficiently far from each component in terms of Mahalanobis distance.

Value

simulate_gmm return a data.frame with continuous variables X1, X2, ..., followed by a mixture component label vector G with outliers denoted by 0.

Examples

```
gmm_k3n1000o10 <- simulate_gmm(
  n = c(500, 250, 250),
  mu = list(c(-1, 0), c(+1, -1), c(+1, +1)),
  sigma = list(diag(c(0.2, 4 * 0.2)), diag(c(0.2, 0.2)), diag(c(0.2, 0.2))),
  outlier_num = 10,
  seed = 123,
  crit_val = 0.9999,
  range_multiplier = 1.5
)

plot(
  gmm_k3n1000o10[, c("X1", "X2")],
  col = gmm_k3n1000o10$G + 1, pch = gmm_k3n1000o10$G + 1
)
```

---

simulate_lcwm *Simulate data from a linear cluster-weighted model with outliers.*

Description

Simulates data from a linear cluster-weighted model, then simulates outliers from a region around each mixture component, with a rejection step to control how unlikely the outliers are under the model.

Usage

```
simulate_lcwm(
  n,
  mu,
  sigma,
  beta,
  error_sd,
  outlier_num,
  outlier_type = c("x_and_y", "x_only", "y_only"),
  seed = NULL,
  prob_range = c(1e-08, 1e-06),
  range_multipliers = c(3, 3),
  more_extreme = FALSE
)
```

Arguments

n	Vector of component sizes.
mu	List of component mean vectors.
sigma	List of component covariance matrices.
beta	List of component regression coefficient vectors.
error_sd	Vector of component regression error standard deivations.
outlier_num	Desired number of outliers.
outlier_type	Character string governing whether the outliers are outlying with respect to the explanatory variable only ("x_only"), the response variable only ("y_only"), or both ("x_and_y"). "x_and_y" is the default value.
seed	Seed.
prob_range	Values for uniform sample rejection.
range_multipliers	For every explanatory variable, the sampling region The sampling region for the Uniform distribution used to simulate proposed outliers is controlled by multiplying the component widths by these values.
more_extreme	Whether to return a column in the data frame consisting of the probabilities of sampling more extreme true observations than the simulated outliers.

Details

simulate_lcwm samples a user-defined number of outliers for each component. However, even though an outlier may be associated with one component, it must be outlying with respect to every component.

The covariate values of the simulated outliers for a given component g are sampled from a Uniform distribution over a hyper-rectangle which is specific to that component. For each covariate dimension, the hyper-rectangle is centred at the midpoint between the maximum and minimum values for that variable from all of the Gaussian observations from component g. Its width in that dimension is the distance between the minimum and maximum values for that variable multiplied by the value of range_multiplier[1].

The response values of the simulated outliers for a given component g are obtained by sampling random errors from a Uniform distribution over a univariate interval, simulating covariate values as discussed above, computing the mean response value for those covariate values, then adding this simulated error to the response. The error sampling interval is centred at the midpoint between the maximum and minimum errors for that variable from all of the Gaussian observations from component g. Its width is the distance between the minimum and maximum errors multiplied by the value of range_multiplier[2].

A proposed outlier for component g is rejected if the probability of sampling a more extreme point from any of the components is greater than prob_range[2] or if the probability of sampling a less extreme point from component g is less than prob_range[1]. This can be visualised as a pair of inner and outer envelopes around each component. To be accepted, a proposed outlier must lie inside the outer envelope for its component and outside the inner envelopes of all components. Setting prob_range[1] = 0 will eliminate the outer envelope, while setting prob_range[2] = 0 will eliminate the inner envelope.

By setting outlier_type = "x_only" and giving arbitrary values to error_sd (e.g. a zero vector) and beta (e.g. a list of zero vectors), then ignoring the simulated Y variable, simulate_lcwm can be used to simulate a Gaussian mixture model. Since simulate_lcwm simulates component-specific outliers from sampling regions around each component, rather than a single sampling region around all of the components, this will not be equivalent to simulate_gmm. simulate_lcwm also allows the user to set an upper bound on how unlikely an outlier is, as well as a lower bound, whereas simulate_gmm only sets a lower bound.

Value

simulate_lcwm returns a data.frame with continuous variables X1, X2, ..., followed by a continuous response variable, Y, and a mixture component label vector G with outliers denoted by 0. The optional variable more_extreme may be included, if specified by the corresponding argument.

Examples

```
lcwm_k3n1000o10 <- simulate_lcwm(
  n = c(300, 300, 400),
  mu = list(c(3), c(6), c(3)),
  sigma = list(as.matrix(1), as.matrix(0.1), as.matrix(1)),
  beta = list(c(0, 0), c(-75, 15), c(0, 5)),
  error_sd = c(1, 1, 1),
  outlier_num = c(3, 3, 4),
  outlier_type = "x_and_y",
  seed = 123,
  prob_range = c(1e-8, 1e-6),
  range_multipliers = c(1, 2)
)

plot(
  lcwm_k3n1000o10[, c("X1", "Y")],
  col = lcwm_k3n1000o10$G + 1,
  pch = lcwm_k3n1000o10$G + 1
)
```

---

test_outlier_ombc *Check if a new sample satisfies the outlier criteria.*

Description

This function checks whether a given sample is an acceptable outlier with respect to prob_range and also computes the probability of sampling a more extreme point from component g.

Usage

```
test_outlier_ombc(
  outlier_type,
  mu,
  sigma,
```

```
    beta,
    error_sd,
    x_sample,
    y_sample,
    prob_range,
    g
  )
```

Arguments

outlier_type	Character string governing whether the outliers are outlying with respect to the explanatory variable only ("x_only"), the response variable only ("y_only"), or both ("x_and_y"). "x_and_y" is the default value.
mu	List of component mean vectors.
sigma	List of component covariance matrices.
beta	List of component regression coefficient vectors.
error_sd	Vector of component regression error standard deivations.
x_sample	New covariate sample.
y_sample	New response sample.
prob_range	Values for uniform sample rejection.
g	Component number.

Value

test_outlier_ombc returns a vector consisting of a logical value indicating whether the new sample satisfies the outlier checks, and a numeric value giving the probability of sampling a more extreme point from component g.

---

try_mixture_gpcm — *Run* mixture::gpcm *and try alternative covariance structures or initialisations if necessary.*

Description

If mixture::gpcm returns an error, this function first tries the other covariance structures, and then tries a k-means initialisation.

Usage

```
try_mixture_gpcm(x, comp_num, mnames, z, nmax, atol, fixed_labels)
```

Arguments

x	Data.
comp_num	Number of mixture components.
mnames	Model names for mixture::gpcm.
z	Component assignment probability matrix for initialisation.
nmax	Maximum number of iterations for `mixture::gpcm`.
atol	EM convergence tolerance threshold for `mixture::gpcm`.
fixed_labels	Cluster labels that are known a prior. See `label` argument in `mixture::gpcm`.

Value

Object of class "gpcm" outputted by `mixture::gpcm`.

---

uniform_outlier_ombc *Produce a single sample that passes the outlier checks.*

Description

This function calls uniform_sample_lcwm to sample a proposed outlier and then calls test_outlier_ombc to check if it satisfies the required criteria.

Usage

```
uniform_outlier_ombc(
  outlier_type,
  mu,
  sigma,
  beta,
  error_sd,
  g,
  uniform_spans,
  prob_range
)
```

Arguments

outlier_type	Character string governing whether the outliers are outlying with respect to the explanatory variable only ("x_only"), the response variable only ("y_only"), or both ("x_and_y"). "x_and_y" is the default value.
mu	List of component mean vectors.
sigma	List of component covariance matrices.
beta	List of component regression coefficient vectors.
error_sd	Vector of component regression error standard deivations.
g	Component index.
uniform_spans	Covariate and response error spans.
prob_range	Values for uniform sample rejection.

Value

uniform_outlier_ombc returns a simulated outlier as a vector containing its covariate values, response value, and its component label 0. This vector's final element is the probability of sampling a more extreme Gaussian point from this outlier's associated component.

---

uniform_sample_lcwm *Sample a potential outlier.*

Description

If outlier_type = "x_and_y", then both the covariate values and response error of the outlier proposed by this function will be Uniformly distributed. If outlier_type = "x_only", then the covariate values will be Uniformly distributed but the response error will be Normally distributed. If outlier_type = "y_only", then the response error will be Uniformly distributed but the covariate values will be Normally distributed.

Usage

```
uniform_sample_lcwm(
  outlier_type,
  mu_g,
  sigma_g,
  beta_g,
  error_sd_g,
  uniform_spans_g
)
```

Arguments

outlier_type	Character string governing whether the outliers are outlying with respect to the explanatory variable only ("x_only"), the response variable only ("y_only"), or both ("x_and_y"). "x_and_y" is the default value.
mu_g	Covariate mean vector for component g.
sigma_g	Covariate covariance matrix for component g.
beta_g	Regression coefficient vector for component g.
error_sd_g	Regression error standard deviation for component g.
uniform_spans_g	Covariate and response error ranges for component g.

Value

uniform_sample_lcwm returns a list with the following elements:

x Vector of covariate values.

y Response value.

uniform_spans_lcwm *Obtain the span of the observations for each component.*

Description

Determine the minimum and maximum values for each covariate / explanatory variable and for the response errors from all Gaussian observations.

Usage

```
uniform_spans_lcwm(range_multipliers, covariates_g, errors_g)
```

Arguments

`range_multipliers`
: For every explanatory variable, the sampling region The sampling region for the Uniform distribution used to simulate proposed outliers is controlled by multiplying the component widths by these values.

`covariates_g`
: Covariate values of the sampled observations.

`errors_g`
: Response errors of the sampled observations.

Value

`uniform_spans_lcwm` returns a 2-column matrix. The final row contains the minimum and maximum values of the response errors, while the previous rows contain the minimum and maximum values for each covariate.

validate_outliermbc_gmm *Validator for* `"outliermbc_gmm"` *S3 class.*

Description

Validator for `"outliermbc_gmm"` S3 class.

Usage

```
validate_outliermbc_gmm(x)
```

Arguments

`x`
: List.

validate_outliermbc_lcwm
Validator for "outliermbc_lcwm" S3 class.

Description

Validator for "outliermbc_lcwm" S3 class.

Usage

```
validate_outliermbc_lcwm(x)
```

Arguments

x	List.

Index