Package 'PHM'

September 24, 2025

Title The Distinguishability Criterion

Version 1.0.0

Description Implements the Distinguishability Criterion and PHM algorithms.

License GNU General Public License

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Imports mvtnorm,
abind,
mclust,
cubature,
parallel,
dplyr,
tidyr,
ggplot2,
RColorBrewer,
ggtext,
magrittr,
ggnewscale

Depends R (>= 3.5)

LazyData true

Contents

`addPosteriorMatrix` *Compute Posterior Matrix based on PHM merging*

Description

Compute Posterior Matrix based on PHM merging

Usage

```
addPosteriorMatrix(phm, data, initK = length(phm))
```

Arguments

`phm`	Output from `PHM()`
`data`	$N \times D$ matrix of observations
`initK`	Number of clusters from which to start computing the posterior matrix

Details

TODO: Fill me in

Value

List of the same structure as from `PHM()` with `posterior_matrix` and `labels` fields calculated for the specified elements.

computeDeltaPmcMatrix ΔP_mc *Matrix computation*

Description

Compute the ΔP_{mc} matrix for a set of clusters, where the ij^{th} element is $\Delta P_{\mathrm{mc}}^{(i,j)}$.

Usage

```
computeDeltaPmcMatrix(paramsList, integralControl = list())
```

Arguments

paramsList	List containing lists with each component GMM parameters. See generateDistbnFunc for format of components.
integralControl	List specifying arguments to pass to cubature::cubintegrate(). See details.

Details

#' Each step of the PHM algorithm reduces the overall P_{mc} by the ΔP_{mc} value of the merged clusters. For each pair of clusters j, k, ΔP_{mc} is

$$\Delta P_{\mathrm{mc}}^{(j,k)} = \int \pi_j(x)\, \pi_k(x) P(x) dx$$

Where the relationship $P_{\mathrm{mc}} = \sum_{i<j} 2\Delta P_{\mathrm{mc}}^{(i,j)}$ See computePmc for description of integralControl parameters.

Value

$K \times K$ matrix with each pair of clusters' contribution to P_{mc}

Examples

```
set.seed(1)
dat <- matrix(c(rnorm(200), rnorm(200, 3), rnorm(200, -3)), ncol=2, byrow=T)
partition <- c(rep(1, 100), rep(2, 100), rep(3, 100))
params <- constructPmcParamsPartition(partition, dat, G=1:5)
computeDeltaPmcMatrix(params)
```

`computeMonteCarloDeltaPmcMatrix`
Monte Carlo ΔP_mc Matrix computation

Description

Compute the ΔP_{mc} matrix for a set of clusters, where the ij^{th} element is $\Delta P_{\mathrm{mc}}^{(i,j)}$.

Usage

```
computeMonteCarloDeltaPmcMatrix(
  paramsList,
  mcSamples = 1e+06,
  batchSize = mcSamples,
  numCores = 1,
  verbose = F
)
```

Arguments

`paramsList`	List containing lists with each component GMM parameters. See constructPmcParamsMclust for format of components.
`mcSamples`	Numeric for number of MC samples to use to approximate the integral.
`batchSize`	Numeric for the observations to assign to each core. Helps with memory concerns. Default `mcSamples`.
`numCores`	Number of cores to use in parallel::mclapply call. Default is 1.
`verbose`	Boolean whether to print output messages

Details

Each step of the PHM algorithm reduces the overall P_{mc} by the ΔP_{mc} value of the merged clusters. For each pair of clusters j, k we estimate their $\Delta P_{\mathrm{mc}}^{(j,k)}$ value.

$$\Delta \hat{P}_{\mathrm{mc}}^{(j,k)} = \frac{1}{M} \sum_{i=1}^{M} \pi_j(x_i)\, \pi_k(x_i)$$

Where the M observations are sampled from the overall data density $P(x)$. Note that $\Delta \hat{P}_{\mathrm{mc}}^{(j,k)} = \Delta \hat{P}_{\mathrm{mc}}^{(k,j)}$

Value

$K \times K$ matrix with each pair of clusters' ΔP_{mc} value.

Examples

```
set.seed(1)
dat <- matrix(c(rnorm(200), rnorm(200, 3), rnorm(200, -3)), ncol=2, byrow=T)
partition <- c(rep(1, 100), rep(2, 100), rep(3, 100))
params <- constructPmcParamsPartition(partition, dat, G=1:5)
computeMonteCarloDeltaPmcMatrix(params, 1e5, verbose=T)
```

---

computeMonteCarloPmc *Monte Carlo P_mc computation*

Description

Compute Monte Carlo estimate of P_{mc} for a given cluster configuration based on estimated GMM densities.

Usage

```
computeMonteCarloPmc(
  paramsList,
  mcSamples = 1e+05,
  batchSize = mcSamples,
  numCores = 1,
  verbose = F
)
```

Arguments

`paramsList`	List containing lists with each component GMM parameters. See constructPmcParamsMclust for format of components.
`mcSamples`	Numeric for number of MC samples to use to approximate the integral.
`batchSize`	Numeric for the observations to assign to each core. Helps with memory concerns. Default `mcSamples`.
`numCores`	Number of cores to use in parallel::mclapply call. Default is 1.
`verbose`	Boolean whether to print output messages

Details

P_{mc} can be difficult to evaluate as standard cubature methods tend to perform poorly in higher dimensions. We can approximate it for a K-cluster configuration using a Monte Carlo integral of the form

$$\hat{P}_{\mathrm{mc}} = \frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{K} 2 \left(1 - \pi_j(x_i)\right) \pi_j(x_i)$$

Where the M observations are sampled from the overall data density $P(x)$

Value

Monte Carlo estimate of P_{mc}

Examples

```
set.seed(1)
dat <- matrix(c(rnorm(200), rnorm(200, 3), rnorm(200, -3)), ncol=2, byrow=T)
partition <- c(rep(1, 100), rep(2, 100), rep(3, 100))
params <- constructPmcParamsPartition(partition, dat, G=1:5)
computeMonteCarloPmc(params, 1e5, verbose=T)
```

`computePairwisePmcMatrix`

Pairwise P_mc Matrix computation

Description

TODO: FILL ME IN

Usage

```
computePairwisePmcMatrix(paramsList, mc = T, ...)
```

Arguments

`paramsList`	List containing lists with each component GMM parameters. See `generateDistbnFunc` for format of components.
`mc`	Boolean whether to compute P_{mc} with Monte Carlo integration or cubature integration
`...`	Additional parameters passed to `computePmc()` or `computeMonteCarloPmc()`

Value

$K \times K$ matrix with Pairwise P_{mc} values for each pair of clusters

export

computePmc *Cubature P_mc computation*

Description

Compute Pmc for a given Gaussian mixture distribution based on cubature package

Usage

```
computePmc(paramsList, integralControl = list())
```

Arguments

paramsList	List containing lists with each component GMM parameters. See generateDistbnFunc for format of components.
integralControl	List specifying arguments to pass to cubature::cubintegrate(). See details.

Details

For a given cluster configuration, the overall misclassification probability P_{mc} can be evaluated as

$$P_{\mathrm{mc}} = \sum_{j=1}^{K} \int 2\left(1 - \pi_j(x)\right)\, \pi_j(x) P(x) dx$$

This integral is implemented using the cubature function. the integralControl variable accepts arguments to the cubature::cubintegrate() function The defaults for this function are:

- method: Which integration method is to be used. Default is hcubature
- lowerLimit and upperLimit: The bounds of integration. Default is $\pm\infty$.
- maxEval: Sets the maximum number of integral evaluations. Default is 1e6
- relTol: Sets the convergence tolerance for the integration. Default is 1e-5

Value

Output from the cubature::cubintegrate() function.

Examples

```
set.seed(1)
dat <- matrix(c(rnorm(200), rnorm(200, 3), rnorm(200, -3)), ncol=2, byrow=T)
partition <- c(rep(1, 100), rep(2, 100), rep(3, 100))
params <- constructPmcParamsPartition(partition, dat, G=1:5)
computePmc(params)
```

constructPmcParamsMclust

Construct P_mc parameter list from Mclust *output*

Description

Take the output of Mclust() and format it for consumption by compute*Pmc and PHM() functions

Usage

```
constructPmcParamsMclust(mclustObj, singleElement = F)
```

Arguments

mclustObj	Output of Mclust() function call
singleElement	Boolean, whether to combine into a single list element

Details

This function takes the parameters object from the output of Mclust() and transforms it to define a GMM for use in the compute*Pmc and PHM functions. The output is a list of lists, where each sublist corresponds ot the parameters of a single cluster distribution (default is a single Gaussian component, K = 1).

If singleElement = TRUE, then all components will be combined into a single list.

The parameters in each sublist are

- mean should be a DxK matrix where each column corresponds to a component mean
- var should be a DxDxK array where each slice corresponds to a covariance matrix
- prob should be a K dimensional vector for the proportions within the parent mixture

Value

List of lists where each sublist contains the parameters for the mixture component or a single list if singleEleemnt is TRUE.

Examples

```
set.seed(1)
dat <- matrix(c(rnorm(200), rnorm(200, 3), rnorm(200, -3)), ncol=2, byrow=T)
mcl <- Mclust(dat)
constructPmcParamsMclust(mcl)
```

constructPmcParamsPartition
Construct P_mc parameter list from a partition

Description

Estimates the mixture model density for a partition of the data using `Mclust`

Usage

```
constructPmcParamsPartition(partition, data, ...)
```

Arguments

`partition`	Vector of labels for observations
`data`	Numeric matrix
`...`	Parameters passed to `mclust::Mclust()`

Details

Performs a naive density estimation, fitting a GMM to each partition separately. Only observations in a cluster are considered to fit the GMM.

See `constructPmcParamsMclust()` for a description of the output.

Value

List of lists where each sublist contains the proportion, mean, covariance matrix estimates for each cluster.

Examples

```
set.seed(1)
dat <- matrix(c(rnorm(200), rnorm(200, 3), rnorm(200, -3)), ncol=2, byrow=T)
partition <- c(rep(1, 100), rep(2, 100), rep(3, 100))
params <- constructPmcParamsPartition(partition, dat, G=1:5)
```

`constructPmcParamsWeightedPartition` *Construct P_mc parameter list from a partition with weights*

Description

Estimates the weighted mixture model density for a partition of the data using `Mclust`.

Usage

```
constructPmcParamsWeightedPartition(
  partition,
  data,
  weights = NULL,
  threshold = 1e-04,
  linkFunc = min,
  verbose = F,
  ...
)
```

Arguments

`partition`	Vector of labels for observations
`data`	Numeric matrix for data ($N \times p$)
`weights`	Optional precomputed weight matrix ($N \times K$)
`threshold`	Threshold past which to not include weights (for computational efficiency)
`linkFunc`	Function to use to combine all distances within a cluster for weight calculation. Default is `min`
`verbose`	Whether to print out debug messages. Default is `FALSE`
`...`	Parameters passed to `mclust::Mclust()`

Details

This procedure attempts to account for clustering uncertainty when estimating the cluster densities. For a given observation x_i and cluster C_j we estimate its cluster-specific weight based on an observation-cluster distance $d(x_i, C_j)$. By default we take the observation-cluster distance to be the smallest Euclidean distance to any member in that cluster, and compute the weight for cluster j as:

$$w_{ij} = \frac{e^{-d(x_i, C_j)}}{\sum_{k=1}^{K} e^{-d(x_i, C_k)}}$$

`constructPmcParamsPartition()` is a special case of this procedure with weights of 1 if an observation is in a cluster and 0 otherwise. The weights are then passed to a weighted EM procedure to estimate the cluster-specific density via GMM.

See `constructPmcParamsMclust()` for a description of the output.

Value

List of lists where each sublist contains the proportion, mean, covariance matrix estimates for each cluster.

Examples

```
set.seed(1)
dat <- matrix(c(rnorm(200), rnorm(200, 3), rnorm(200, -3)), ncol=2, byrow=T)
partition <- c(rep(1, 100), rep(2, 100), rep(3, 100))
params_w <- constructPmcParamsWeightedPartition(partition, dat, G=1:5)
```

PHM *PHM Algorithm*

Description

Implements the PHM algorithm which constructs a clustering hierarchy by successively merging clusters with the largest ΔP_{mc} values.

Usage

```
PHM(
  mclustObj = NULL,
  paramsList = NULL,
  partition = NULL,
  data = NULL,
  verbose = T,
  computePosterior = T,
  partitionWeightedDensity = T,
  partitionModel = "VVI",
  partitionMaxComponents = 10,
  mc = T,
  ...
)
```

Arguments

mclustObj	Output from mclust::Mclust()
paramsList	A list generated from constructPmcParamsMclust(), constructPmcParamsPartition(), constructPmcParamsPartition() providing the initial cluster parameter estimates
partition	A vector providing obseration partition memberships for the initial state
data	An $N \times D$ matrix of observations
verbose	Boolean whether to suppress debug statements

computePosterior
: Boolean whether to compute the Posterior matrix for the merges

partitionModel
: If no partition provided, the covariance structure to estimate the density for each partition using `constructPmcParamsPartition()`

partitionMaxComponents
: If specifying `partition`, the maximum number of components to estimate the density for each partition

mc
: Boolean whether to use Monte Carlo integration to evaluate the ΔP_{mc} matrix

...
: Parameters pased to either `computeDeltaPmcMatrix()` or `computeMonteCarloDeltaPmcMatrix()` to evaluate the ΔP_{mc} matrix

Value

A list of lists for each step of the PHM algorithm. Each sublist contains

- `clusters`: Number of clusters K at this merge
- `posterior_matrix`: $N \times K$ matrix of posterior cluster probabilities
- `labels`: Partition of the observations
- `pmc_change`: Value of ΔP_{mc} leading to this value of K
- `params`: Cluster-specific densities
- `pmc_components`: Number of original clusters involved in this merge
- `pmc_accum`: Accumulated ΔP_{mc} in this subtree (unused)
- `min_merge_pmc`: Minimum value of ΔP_{mc} for all merges in this subtree
- `merge_components`: Index of components merged in this step
- `pmc`: Overall P_{mc} remaining in the cluster configuration
- `pmc_matrix`: ΔP_{mc} matrix for the remaining clusters

Examples

```
set.seed(1)
dat <- matrix(c(rnorm(200), rnorm(200, 3), rnorm(200, -3)), ncol=2, byrow=T)
partition <- c(rep(1, 100), rep(2, 100), rep(3, 100))
params <- constructPmcParamsPartition(partition, dat, G=1:5)
phm <- PHM(paramsList=params, data=dat, partition=partition)
```

---

plotPHMDendrogram *Visualize PHM merging procedure via Dendrogram*

Description

Visualize the PHM merging procedure using a dendrogram. Visualization options, such as displaying the merge ΔP_{mc} value and tracking group membership across merges is included.

Usage

```
plotPHMDendrogram(
  phm,
  colors = NULL,
  scaleHeights = c("log10", "unscaled", "pmcdist"),
  heightValue = c("merge", "min"),
  threshold = 0,
  suppressLabels = F,
  mergeLabels = c("delta", "pmc", "percent"),
  mergeLabelsSize = 2,
  mergeLabelsBorderSize = 0.15,
  mergeLabelsPadding = 0.15,
  mergeLabelsR = 0.1,
  displayAxis = c("box", "label", "index", "none"),
  displayAxisSize = NULL,
  colorAxis = NULL,
  groupProbs = NULL,
  groupColorMax = "black",
  groupColorMin = "lightgray"
)
```

Arguments

phm	Output from PHM()
colors	Vector of K hex codes to color the leaf node labels
scaleHeights	String specifying how to set the heights in the dendrogram. See Details for more information.
heightValue	Whether to use the ΔP_{mc} or $\min \Delta P_{\text{mc}}$ value to determine branch height for a merge. See Details for more information.
threshold	Error threshold for the integral past which to represent the merges as dashed lines
suppressLabels	Boolean whether or not to display P_{mc} reduction labels on the dendrogram or not
mergeLabels	String indicating what value to display in the labels on the dendrogram
mergeLabelsSize	Text size for the numeric value in the labels on the dendrogram
mergeLabelsPadding	Padding for the merge label text
mergeLabelsR	Radius for the rounded edges of the merge label box
displayAxis	String indicating what label to place on the leaf nodes of the dendrogram. See Details for more information.
displayAxisSize	Text size for the leaf node labels
colorAxis	Whether or not to color the labels on the leaf nodes
groupProbs	Vector of class probability conditional on base group membership

`groupColorMax`	Color of lines corresponding to high group probability
`groupColorMin`	Color of lines corresponding to low group probability

Details

There are two options for the value for the P_{mc} height for merging subtrees $\mathcal{S}_1, \mathcal{S}_2$ ($m = |\mathcal{S}_1| + |\mathcal{S}_2|$):

- `"merge"`: $\binom{m}{2}^{-1} \Delta P_{\text{mc}}^{(\mathcal{S}_1, \mathcal{S}2)}$
- `"min"`: $\min_{i \in \mathcal{S}_1,\, j \in \mathcal{S}_2} \Delta P_{\text{mc}}^{(i,j)}$

Once the value for the tree height is obtained, there are three options to scale.

- `"unscaled"`: The height can be left unscaled
- `"log10"`: A $\log_{10}$ scaling can be applied to the height to better reveal different clustering resolutions.
- `"pmcdist"`: A spline-based scaling where we map the value to a linear distance between two Gaussian clusters.

The `displayAxis` parameter controls what to display on the axes of the heatmap. `box` displays a standard box, which is most useful for color-coded axes. `label` displays the cluster label, as specified in the `phm` parameters. `index` displays the numeric index of each cluster in the `phm` parameter list. `none` suppresses the cluster label entirely, and is mose useful for when there are a large number of clusters.

Value

A ggplot object

---

plotPHMDistruct *Generate the distruct plot from the posterior matrix*

Description

Visualize a distruct plot based on either a partition or the posterior cluster probabilities.

Usage

```
plotPHMDistruct(
  phm,
  K = length(phm),
  colors = NULL,
  labels = NULL,
  axisTextSize = 6,
  partition = F
)
```

Arguments

phm	Output from the PHM() function
K	Number of clusters for which to generate the distruct plot
colors	Optinal vector with colors for the mixture components
labels	Ground truth class labels for the observations (ordered factor vector)
axisTextSize	Size for axis labels
partition	Whether to visualize from the posterior matrix or partition labels

Details

In the case of visualizing for a partition, the posterior probabilities are set to 1 if it is the cluster the obesrvation is assigned to and 0 otherwise.

Value

A ggplot object

---

plotPHMMatrix *Visualize PHM dendrogram structure with a heatmap*

Description

For a pair of clusters i, j, the heatmap position i, j is the value of ΔP_{mc} where the clusters are first merged. This allows for a more straightforward visualization of the multi-resolution structure of heatmaps when combined with the dendrogram.

Usage

```
plotPHMMatrix(
  phm,
  colors = NULL,
  displayAxis = c("box", "label", "index", "none"),
  displayAxisSize = NULL,
  colorAxis = NULL,
  gridColor = "black",
  fillLimits = NULL,
  fillScale = c("log10", "pmcdist"),
  legendPosition = "none"
)
```

Arguments

phm	Output from PHM()
colors	Vector of K hex codes to color the leaf node labels
displayAxis	String indicating what label to place along the axis (corresponding to clusters). See Details for more information.
displayAxisSize	Text size for the axis labels
colorAxis	Whether or not to color the axis labels
gridColor	What color to make the heatmap grid
fillLimits	Optional vector to manually set limits of the fill scaling. Default is to use the min and max ΔP_{mc} values from phm
fillScale	Whether to use $\log_{10} \Delta P_{\text{mc}}$ or the spline scaling for the heatmap color
legendPosition	Where to put the legend for the heatmap colors. Default is to suppress.

Details

Consider a pair of initial clusters i, j, and let $\mathcal{S}$ be the smallest subtree containing both i, j ($\mathcal{S}_i$ contains cluster i and $\mathcal{S}_j$ contains cluster j). The pairwise value for i, j, $\rho(i, j)$ is based on $\Delta P_{\text{mc}}^{(\mathcal{S}(i),\mathcal{S}(j))}$, which is the ΔP_{mc} value at which i, j are merged into a single cluster.

As with the dendrogram heights (plotPHMDendrogram()) there are multiple options for scaling $\rho(i, j)$. The first is log10 scaling, where $\rho(i, j) = \log_{10} \Delta P_{\text{mc}}$. The second is a spline-based scaling, pmcdist, where we map the ΔP_{mc} value to a linear distance between two Gaussian clusters.

The displayAxis parameter controls what to display on the axes of the heatmap. box displays a standard box, which is most useful for color-coded axes. label displays the cluster label, as specified in the phm parameters. index displays the numeric index of each cluster in the phm parameter list. none suppresses the cluster label entirely, and is mose useful for when there are a large number of clusters.

Value

A ggplot object

---

plotPmc2D *Visualize regions contributing to P_mc in a 2D plot*

Description

Visualize the point-specific P_{mc} over a grid of points to visually inspect cluster contributions to P_{mc}

Usage

```
plotPmc2D(
  paramsList,
  data,
  partition,
  colors = RColorBrewer::brewer.pal(length(paramsList), "Paired"),
  xlim = NULL,
  ylim = NULL,
  suppressPmc = F,
  numPmcPatches = 200,
  logPmcThreshold = -4,
  logPmcMidpoint = 0.6 * logPmcThreshold,
  PmcColor = "#222",
  suppressDensity = F,
  densityLevels = c(0.01, 0.1),
  densityLevelWidth = 0.3,
  suppressObservations = F,
  pointSize = 0.5,
  labelSize = 2,
  textSize = 9,
  legendPosition = "none"
)
```

Arguments

paramsList	A list generated from constructPmcParamsMclust(), constructPmcParamsPartition(), constructPmcParamsPartition() providing the initial cluster parameter estimates. Parameters should be 2D
data	A $N \times 2$ matrix containing observations
partition	A vector containing class labels
colors	Vector of K hex codes to color the leaf node labels
xlim	Vector with min/max values for the x-axis NULL sets this based on the maximum and minimum x-values for observations
ylim	Vector with min/max values for the y-axis NULL sets this based on the maximum and minimum y-values for observations
suppressPmc	Flag whether to display the regions contributing to P_{mc} in the plot
numPmcPatches	Number of points along each axis in the P_{mc} grid to evaluate. Higher values gives a smoother visualization
logPmcThreshold	Threshold for P_{mc} values to display. All coordinates with $\log_{10} P_{\text{mc}}$ below this value will be ignored.
logPmcMidpoint	Midpoint in the P_{mc} color gradient
PmcColor	Hex value for the maximum value in the P_{mc} color gradient
suppressDensity	Flag whether to overlay the cluster-specific densities

densityLevels	Values of the density at which to display the level curves
densityLevelWidth	Line width for the density curves
suppressObservations	Flag whether to display the observations from data
pointSize	Size of the data scatterplot points
labelSize	Text size for the cluster labels
textSize	Text size for the plots
legendPosition	Where to put the legend for the heatmap colors. Default is to suppress

Details

For an arbitrary point $\mathbf{x}$, using the Monte Carlo evaluation of P_{mc} its point-specific P_{mc} can be calculated as

$$\sum_{k=1}^{K} \pi_k(\mathbf{x})(1 - \pi_k(\mathbf{x})) \times f(x = \mathbf{x})$$

Where $f(\mathbf{x})$ is the overall data density evaluated at a point. Note that for a sample of M points, the average of these point-specific P_{mc} values produce the Monte Carlo estimate of P_{mc} described in Turfah and Wen (2025).

The point-specific P_{mc} of each point in a grid is evaluated and visualized. The cluster-specific density level sets and/or the partitioned observations can be overlaid over this to better understand how the P_{mc} value was obtained.

Value

A ggplot object

---

plotPmcMatrix *Plot* ΔP_mc *matrix*

Description

Visualize the matrix of ΔP_{mc} values

Usage

```
plotPmcMatrix(
  phm,
  K = length(phm),
  colors = NULL,
  displayAxis = c("box", "label", "index", "none"),
  displayAxisSize = NULL,
  colorAxis = NULL,
  visScale = c("absolute", "percent"),
  visSize = 2,
```

```
  visThreshold = 0.001,
  visDigits = 3
)
```

Arguments

phm	Output from PHM()
K	Number of clusters for which to visualize the heatmap
colors	Vector of K hex codes to color the leaf node labels
displayAxis	String indicating what label to place along the axis (corresponding to clusters)
displayAxisSize	Text size for the axis labels
colorAxis	Whether or not to color the axis labels
visScale	Whether to display the raw ΔP_{mc} values or scale them to be percent of total P_{mc}
visSize	Text size for the values inside the heatmap
visThreshold	At what value suppress the value and show "< (visThreshold)"
visDigits	Number of digits to round the displayed values

Value

A ggplot object

---

thresholdPHM *Find a clustering for a given P_mc threshold*

Description

Each step of the PHM algorithm reduces P_{mc}. This gives the results from the PHM algorithm terminated when P_{mc} falls below some specified threshold.

Usage

```
thresholdPHM(phm, threshold = 0.01)
```

Arguments

phm	Output of PHM()
threshold	P_{mc} threshold, default is 0.01

Value

Result of the PHM merging procedure terminated when the P_{mc} threshold is satisfied export

Index