Package 'BRIC'

June 14, 2022

Title Bootstrap and Refine Iterative Clustering

Version 1.0.0

Maintainer Adrien Brilhault <adrien.brilhault@gmail.com>

Description The BRIC algorithm searches for the different clusters through three principal steps:
- BOOTSTRAP: a recursive depth (or convex body minimizers) trimming to locate a first central location estimate.
- REFINE: a two-pass outliers filtering, the first relying on euclidean distances to the first estimate and unimodality tests, the second on robust distances and multinormality tests.
- ITERATE: after removing the samples selected in the REFINE step from the global distribution, the same process is reapplied to search for additional clusters.
For more details, consult the publication <https://rdcu.be/cI9Pf>.

Depends R (>= 3.5.0)

Imports ddalpha,
depth,
OjaNP,
MASS,
diptest,
robustbase,
stats,
utils,
nortest,
ICS,
graphics,
grDevices,
CompQuadForm

Remotes cran/OjaNP

Suggests mvtnorm

License GPL-3 | file LICENSE

URL https://adrienbrilhault.github.io/BRIC/, https://github.com/adrienbrilhault/BRIC/

BugReports https://github.com/adrienbrilhault/BRIC/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

R topics documented:

`clustBRIC` *Bootstrap and Refine Iterative Clustering*

Description

Robust clustering algorithm based on depth measures and convex body minimizers

Usage

```
clustBRIC(
  data,
  maxIterations = 0,
  minUnassigned = 0.1,
  nsamp = "best",
  method = "Projection",
  alpha = 0.5,
  testUnimodal = "DIP",
  threshUnimodal = 0.05,
  distUnimodal = "Euclidean",
  testNormal = "Mardia",
  threshNormal = 0.05,
  distNormal = "MCD",
  trimmedPerFilteringIteration = 1,
  debug = FALSE,
  warnings = FALSE
)
```

Arguments

`data`	Matrix or Data-Frame of numerical values containing the observations (rows correspond to observations, columns to variables)
`maxIterations`	Maximum number of iterations performed by the algorithm (i.e. max number of potential clusters encountered). Set to NULL or 0 for unlimited number (default)

minUnassigned	Numerical value between 0 and 1 (default: 0.1), providing the proportion of unassigned samples from data under which the algorithm will terminate the search
nsamp	Number of samples randomly selected from data for subsampling calculations, or "best", "exact" or "sample". If "sample" is chosen, the subset will include up to 2000 observations; with "best" up to 4000 (default); with "exact" (or 0), exhaustive search will be attempted on the complete dataset (computation in this case might take a long time). When subsampling, the remaining observations will be assigned to the cluster of their closest neighbor.
method	Method to use. Valid options are "MCD" and "MVE" for convex body minimizers, or "L2", "Lui", "Mahalanobis", "Oja", "Projection" (default), "Spatial" and "Tukey" for depth functions
alpha	Proportion of samples trimmed at each iteration of the recursive median estimate (numerical value between 0 and 1, default: 0.5), see median_rec()
testUnimodal	Statistical test used for unimodality. Valid options are "DIP" or a user-defined function (see filter_outliers())
threshUnimodal	Threshold of significance for the unimodality test (numerical value between 0 and 1, default: 0.05)
distUnimodal	Distance metric used for ordering the samples in the unimodal filtering. Valid options are "Euclidean" (default), or "MCD", "MVE", and "OGK" for robust distances. "Euclidean" is strongly advised for unimodality tests.
testNormal	Statistical test used for normality. Valid options are "Mardia" (default), "Kurtosis", "Skewness", "KS", "KS-adj", "Shapiro", "Lillie", "Chisq", or a user-defined function (see filter_outliers())
threshNormal	Threshold of significance for the normality test (numerical value between 0 and 1, default: 0.05)
distNormal	Distance metric used for ordering the samples in the normal filtering. Valid options are "Euclidean", or "MCD" (default), "MVE", and "OGK" for robust distances. Robust distances are strongly advised for normality tests.
trimmedPerFilteringIteration	Number of samples trimmed at each iteration of the unimodality and normality filtering (default: 1), see filter_outliers()
debug	Logical value. TRUE will compute all p.values in the filtering steps (even after they exceed the selection threshold, see plot.BRIC.Filtering())
warnings	Logical value, to display the warnings and errors caught

Value

The function returns an S3 object of type BRIC containing the following values:

call	Parameters of the call (contains data, maxIterations, minUnassigned, nsamp, method, alpha, testUnimodal, threshUnimodal, distUnimodal, testNormal, threshNormal, distNormal, and trimmedPerFilteringIteration)
iterations	A list with every global iteration of the algorithm, each containing the two filtering procedures performed: filteringUnimodal and filteringNormal (both being S3 object of class BRIC.Filtering, see filter_outliers())
nbClusters	Number of groups encountered
labels	Labels of the groups encountered (corresponding to the number of the iteration they were identified in)

clustersCenters	Matrix containing the coordinates of the centers of each group (row-wise)
clustersSizes	Array with the number of samples in each group
mainCluster	Index of the group identified as main mode

References

Adrien Brilhault, Sergio Neuenschwander, and Ricardo Rios - A New Robust Multivariate Mode Estimator for Eye-tracking Calibration - Behavior Research Methods, 2022 - rdcu.be/cI9Pf

See Also

plot.BRIC(), print.BRIC(), filter_outliers(), median_rec(), median_mv(), depth_values()

Examples

```
# Create a sample distribution and run clustBRIC() function
data <- rbind(
  mvtnorm::rmvnorm(300, c(0, 0), diag(2) * 3 - 1),
  mvtnorm::rmvnorm(100, c(15, 20), diag(2)),
  mvtnorm::rmvnorm(150, c(-10, 15), diag(2) * 2 - 0.5),
  mvtnorm::rmvnorm(200, c(5, 5), diag(2) * 200)
)

res <- clustBRIC(data, debug = TRUE)
print(res)

# Plot the mode and groups encountered
plot(res)

# Plot the different iterations (interactive)
## Not run:
plot(res, contents = "iterations", asp = 1)

## End(Not run)

# See ?plot.BRIC() for other plotting examples
```

---

depth_values *Depth functions wrapper*

Description

Computes the depth values with respect to the distribution provided in data of either all the coordinates given in u, or all observations from data if u is not provided. Depth computations rely on the packages depth and ddalpha.

Usage

```
depth_values(data, u = NULL, method = "Projection", warnings = FALSE)
```

Arguments

data	Matrix or Data-Frame of numerical values containing the observations (rows correspond to observations, columns to variables)
u	Matrix or Data-Frame of numerical values containing the coordinates for which depth values are to be computed (rows correspond to observations, columns to variables). When missing or NULL, depth values will be computed for all observations from data
method	Depth function used. Valid options are "L2", "Lui", "Mahalanobis", "Oja", "Projection" (default), "Spatial", "Tukey"
warnings	Logical value, to display the warnings and errors raised by the underlying depth functions

Value

Array of numerical values containing the depth of each observation from data, or from u if provided (these values are all set to 0 in the occurrence of errors)

See Also

clustBRIC(), filter_outliers(), median_rec(), median_mv()

Examples

```
# Illustrative data
data <- rbind(
  mvtnorm::rmvnorm(300, c(0, 0), diag(2)),
  mvtnorm::rmvnorm(100, c(15, 20), diag(2) * 3 - 1),
  mvtnorm::rmvnorm(150, c(-10, 15), diag(2) * 2 - 0.5),
  mvtnorm::rmvnorm(200, c(5, 5), diag(2) * 200)
)

# Compute depths
D <- depth_values(data, method = "Projection", warnings = TRUE)

# Plot distribution with depth color scale
plotColors <- colorRampPalette(c("maroon4", "steelblue4", "green4", "gold"))(20)
plot(data, pch = 20, asp = 1, col = plotColors[as.numeric(cut(D, breaks = 20))],
     xlab = "X", ylab = "Y")

# Plot depth values
plot(1:nrow(data), D, pch = 20, col = plotColors[as.numeric(cut(D, breaks = 20))],
     xlab = "Index", ylab = "Depth")

# Compute depth for a single point
depth_values(data, c(10, 3), method = "Projection")

# Compute depth for three sets of coordinates
depth_values(data, rbind(c(10, 3), c(65, 8), c(0, 1)), method = "Projection")

## Data in n>2 dimensions

sigma <- matrix(c(4,2,4,2,4,2,4,2,4), ncol=3)
data <- mvtnorm::rmvnorm(n=500, mean=c(1,2,0), sigma=sigma)
depth_values(data, method = "Projection", warnings = TRUE)
```

```
data <- rbind(
  mvtnorm::rmvnorm(300, c(0, 0), diag(2)),
  mvtnorm::rmvnorm(100, c(15, 20), diag(2) * 3 - 1),
  mvtnorm::rmvnorm(150, c(-10, 15), diag(2) * 2 - 0.5),
  mvtnorm::rmvnorm(200, c(5, 5), diag(2) * 200)
)
depth_values(data, method = "Projection", warnings = TRUE)

sigma <- matrix(1:25,5)
sigma[lower.tri(sigma)] = t(sigma)[lower.tri(sigma)]
data <- mvtnorm::rmvnorm(n=500, mean=c(1,2,0,0,0), sigma=sigma)
depth_values(data, method = "Projection", warnings = TRUE)
```

`filter_outliers` *Recursive outlier filtering based on unimodality and multinormality tests*

Description

Recursive outlier filtering based on unimodality and multinormality tests

Usage

```
filter_outliers(
  data,
  center,
  test = "Mardia",
  threshold = 0.05,
  distType,
  trimmedPerIteration = 1,
  debug = FALSE,
  warnings = FALSE
)
```

Arguments

`data`	Matrix or Data-Frame of numerical values containing the observations (rows correspond to observations, columns to variables)
`center`	Coordinates used to computes the distances of the samples and order them (array of numerical two values, for X and Y)
`test`	Statistical test to use. Valid options are "DIP" for unimodality test, "Mardia", "Kurtosis", "Skewness", "KS", "KS-adj", "Shapiro", "Lillie", and "Chisq" for multivariate normality test, or a or a user-defined function (see details below)
`threshold`	Threshold of significance for the statistical test (between 0 and 1, default: 0.05)
`distType`	Distance metric used to order the samples. Valid options are "Euclidean", "MCD", "MVE", and "OGK". If empty or NULL, "Euclidean" will be automatically selected for unimodality tests, and "MCD" for normality tests
`trimmedPerIteration`	Number of samples trimmed at each iteration (positive integer, default: 1)

debug	Logical value. TRUE will compute all p.values, even after exceeding the threshold, for plotting purpose (see plot.BRIC.Filtering())
warnings	Logical value, to display the warnings and errors caught

Details

For unimodality tests parameter distType should be set to "Euclidean" (as the distribution might contains a large amount of outliers). For normality tests robust distances are preferable, using a robust estimate estimates of location and scatter ("MCD","MVE", or "OGK")

For user-defined functions, the function should output the p.value of the test (between 0 and 1), and receive the 3 following arguments:

- data - the matrix of observations
- center - estimate of the center of the observations
- distances - distances from each observations to the center (based on distType metric)

Value

The function returns an S3 object of type BRIC.Filtering containing the following values:

call	Parameters of the call (contains data, test, testType, center, threshold, trimmedPerIteration and distType)
distances	Distances of each sample from data to the center provided
p.values	P.Values of the test at each iteration
index.p.values	Subset size corresponding to each P.Value, for plotting purpose
selected	Indices of the samples from data selected at the end of the filtering
cutoffDistance	Distance of the furthest inlier selected

References

Adrien Brilhault, Sergio Neuenschwander, and Ricardo Rios - A New Robust Multivariate Mode Estimator for Eye-tracking Calibration - Behavior Research Methods, 2022 - rdcu.be/cI9Pf

See Also

plot.BRIC.Filtering(), print.BRIC.Filtering(), clustBRIC(), median_rec(), median_mv(), depth_values()

Examples

```
## Example 1

# Illustrative data
data <- rbind(
  mvtnorm::rmvnorm(300, c(0, 0), diag(2) * 3 - 1),
  mvtnorm::rmvnorm(100, c(15, 20), diag(2)),
  mvtnorm::rmvnorm(150, c(-10, 15), diag(2) * 2 - 0.5),
  mvtnorm::rmvnorm(200, c(5, 5), diag(2) * 200)
)

# Compute an estimate for the center
center <- median_rec(data)$median
```

```
# Remove non unimodal outliers from this location
filtering <- filter_outliers(data, center, test = "DIP", debug = TRUE)
print(filtering, maxDisplayed = 200)
plot(filtering)

## Example 2

# Illustrative data
data <- rbind(
  mvtnorm::rmvnorm(300, c(0, 0), diag(2) * 4 - 1.5),
  mvtnorm::rmvnorm(150, c(5, 5), diag(2) * 400)
)

# Compute an estimate for the center
center <- median_rec(data)$median

# Remove non normal outliers from this location
filtering <- filter_outliers(data, center, test = "Chisq", distType = "MVE", debug = TRUE)
print(filtering)
plot(filtering, asp = 1)

## Examples of user-defined tests
## Not run:

customTest1 <- function(data, center, distances) {
  return(diptest::dip.test(distances)$p.value)
}
filter_outliers(data, center, test = customTest1,
                distType = "Euclidean", debug = TRUE, warnings = TRUE)

customTest2 <- function(data, center, ...) {
  return(stats::ks.test(stats::mahalanobis(data, center, stats::cov(data), tol = 1e-8),
                        "pchisq", df = ncol(data))$p.value)
}
filtering <- filter_outliers(data, center, distType = "Euclidean",
                             test = customTest2, debug = TRUE, warnings = TRUE)

customTest3 <- function(data, ...) {
  return(ICS::mvnorm.skew.test(data)$p.value)
}
filtering <- filter_outliers(data, center, distType = "Euclidean",
                             test = customTest3, threshold = 0.1, debug = TRUE)


## End(Not run)
```

`median_mv` *Multivariate Median*

Description

Computes the Multivariate Median of the distribution provided in `data` (depth computations rely on the packages `depth`, `OjaNP` and `ddalpha`, while convex body minizers "MCD" and "MVE" use the package `MASS`).

Usage

```
median_mv(data, method = "Projection", sampleMedian = TRUE, warnings = FALSE)
```

Arguments

data	Matrix or Data-Frame of numerical values containing the observations (rows correspond to observations, columns to variables)
method	Method to use. Valid options are "CW", "MCD", "MVE", "L2", "Lui", "Mahalanobis", "Oja", "Projection" (default), "Spatial" and "Tukey"
sampleMedian	Logical value. If TRUE (default), the function will return the Sample Median (observation from data with the highest depth). If FALSE, it will return the classic Multivariate Median (point in space with the highest depth), when applicable (methods "Oja", "Turkey" and "Spatial")
warnings	Logical value, to display the warnings and error caught by the underlying functions

Value

Coordinates of the Multivariate Median

See Also

median_rec(), depth_values(), clustBRIC()

Examples

```
# Illustrative data
data <- rbind(
  mvtnorm::rmvnorm(200, c(0, 0), diag(2)),
  mvtnorm::rmvnorm(100, c(15, 20), diag(2) * 3 - 1),
  mvtnorm::rmvnorm(150, c(-10, 15), diag(2) * 2 - 0.5),
  mvtnorm::rmvnorm(100, c(5, 5), diag(2) * 200)
)

# Compute median
m <- median_mv(data, method = "Projection", warnings = TRUE)

# Plot results
plot(data, asp = 1, xlab = "X", ylab = "Y")
points(m[1], m[2], col = "red", pch = 3, cex = 1.5, lwd = 3)

## Others examples of medians
## Not run:
median_mv(data, method = "Oja")
median_mv(data, method = "Tukey", sampleMedian = TRUE)
median_mv(data, method = "Tukey", sampleMedian = FALSE)

## End(Not run)
```

median_rec *Recursive estimate of central location*

Description

Recursive estimate of central location based on depth measures (from the packages depth and ddalpha) or convex body minimizers (package MASS).

Usage

```
median_rec(
  data,
  method = "Projection",
  alpha = 0.5,
  maxIterations = NULL,
  warnings = FALSE
)
```

Arguments

data	Matrix or Data-Frame of numerical values containing the observations (rows correspond to observations, columns to variables)
method	Method to use. Valid options are "MCD" and "MVE" for convex body minimizers, or "L2", "Lui", "Mahalanobis", "Oja", "Projection" (default), "Spatial" and "Tukey" for depth functions
alpha	Proportion of samples trimmed at each iteration (numerical value between 0 and 1, default: 0.5)
maxIterations	Set to a positive integer to limit the number of iterations, to NULL or 0 (default) for no limits
warnings	Logical value, to display the warnings and error raised by the underlying functions

Value

The function returns an S3 object of type BRIC.MedianRec, containing the following values:

median	Coordinate of the recursive median
max	Coordinate of the sample with the highest depth (or the center of the first iteration in the case of convex body minimizers)
iterations	List containing the indices from the samples of data selected at each iteration

References

Adrien Brilhault, Sergio Neuenschwander, and Ricardo Rios - A New Robust Multivariate Mode Estimator for Eye-tracking Calibration - Behavior Research Methods, 2022 - rdcu.be/cI9Pf

See Also

plot.BRIC.MedianRec(), print.BRIC.MedianRec(), median_mv(), depth_values(), clustBRIC()

Examples

```
# Illustrative data
data <- rbind(
  mvtnorm::rmvnorm(300, c(0, 0), diag(2)),
  mvtnorm::rmvnorm(100, c(15, 20), diag(2) * 3 - 1),
  mvtnorm::rmvnorm(150, c(-10, 15), diag(2) * 2 - 0.5),
  mvtnorm::rmvnorm(200, c(5, 5), diag(2) * 200)
)

# Compute the recursive median
res <- median_rec(data)

print(res)
plot(res)
```

`plot.BRIC` *Plot method for* `BRIC` *objects*

Description

Plot method for `BRIC` objects

Usage

```
## S3 method for class 'BRIC'
plot(
  x,
  contents = "plot",
  showCenters = FALSE,
  col,
  colCenters,
  colClusters = NULL,
  iterationsIndices = NULL,
  iterationsOptions = NULL,
  ...
)
```

Arguments

`x`	An object of class `BRIC` (see `clustBRIC()`)
`contents`	Contents to be displayed, options are "scatterplot", or "iterations" (only one option possible)
`showCenters`	Logical value used when `contents = "scatterplot"`, to show or not the clusters' center
`col`	Default color of samples
`colCenters`	Color of the center(s) when `contents = "scatterplot"`
`colClusters`	List (or array) of colors for each of the clusters/iterations (length must be at least equal to the number of groups identified by the function `clustBRIC()`, i.e. `x$nbClusters`)

`iterationsIndices`
: Numerical value or array of numerical values, used when `contents = "iterations"`, which provides the indices of the iterations to be plotted. If more than one iteration is requested, an interactive menu in the console will be used for the selection. 0 or NULL (default) will include all the iterations. Values that are negative or superior to the number of iterations performed by the execution `clustBRIC()` will be ignored

`iterationsOptions`
: List of additional parameters to be passed to the `plot.BRIC.Filtering()` function when `contents = "iterations"` is selected (see `plot.BRIC.Filtering()` for details). Example: `iterationsOptions = list(xlab = NA,ylab = NA,contents = c("p.values","scatterplot"),asp = 1)`

`...`
: Other arguments passed to or from other methods (such as pch for the symbols, main and sub for title and subtitle, xlab, xmin, ...)

See Also

`clustBRIC()`, `print.BRIC()`, `filter_outliers()`, `print.BRIC.Filtering()`

Examples

```
# Create a sample distribution and run clustBRIC() function
data <- rbind(
  mvtnorm::rmvnorm(300, c(0, 0), diag(2) * 3 - 1),
  mvtnorm::rmvnorm(100, c(15, 20), diag(2)),
  mvtnorm::rmvnorm(150, c(-10, 15), diag(2) * 2 - 0.5),
  mvtnorm::rmvnorm(200, c(5, 5), diag(2) * 200)
)
res <- clustBRIC(data, debug = TRUE)


# Plot the clusters
plot(res)

# Plot the clusters with extra graphic options
plot(res, showCenters = TRUE, main = "Multivariate Mode Estimate",
     col = "gray", colCenters = "black", colClusters = c("yellow","cyan",
     "purple","red"), asp = 1, pch = 3 )

# Plot the second iteration
plot(res, contents = "iteration", iterationsIndices = 2)

# Plot the second iteration (with arguments to plot.BRIC.filtering())
plot(res, contents = "iteration", iterationsIndices = 2,
     iterationsOptions = list(
       contents = c("scatterplot", "p.values"),
       colSelection = "blue", mfrow = c(2,1), asp = 1))

## Not run:
# Plot all iterations (interactive mode)
plot(res, contents = "iterations")

# Plot the 3 first iterations with options (interactive mode)
plot(res, contents = "iterations", iterationsIndices = c(1:3),
     iterationsOptions = list(
       contents = c("scatterplot"),
```

```
       xlim  = c(-50,50),  ylim  = c(-30,30), asp = 1))

## End(Not run)
```

`plot.BRIC.Filtering` *Plot method for* `BRIC.Filtering` *objects*

Description

Plot method for `BRIC.Filtering` objects

Usage

```
## S3 method for class 'BRIC.Filtering'
plot(
  x,
  contents = c("p.values", "scatterplot", "dist", "hist"),
  showCenter = TRUE,
  showSelection = TRUE,
  col = "black",
  colSelection = "red",
  colCenter = "orange",
  mtextTitles = TRUE,
  mfrow,
  ...
)
```

Arguments

`x`	An object of class `BRIC.Filtering` (see `filter_outliers()`)
`contents`	Contents to be displayed, options are "p.values", "scatterplot", "dist", "hist" and "all"
`showCenter`	Logical value, to show the center used in the filtering
`showSelection`	Logical value, to highlight the samples selected by the filtering process
`col`	Default color for non-selected samples (default: "black")
`colSelection`	Color of the selected samples (default: "red")
`colCenter`	Color for the center in "scatterplot" (default: "orange")
`mtextTitles`	Logical value, `TRUE` to set smaller titles/subtitles on top, `FALSE` to use the default plot title options.
`mfrow`	Number of rows and columns of the figure (example: c(4,1))
`...`	Other arguments passed to or from other methods (such as pch for the symbols, main and sub for title and subtitle, xlab, xmin, ...)

Details

Red intercept lines correspond to the selection based on the p.values exceeding the given threshold. To display all the p.values, rerun the function `filter_outliers()` *with the parameter* debug = TRUE

`contents` options:

- **"p.values"** provides a plot of the test p.values (in function of the subset size)
- **"scatterplot"** displays the data in cartesian coordinates. Selected samples are displayed in red, and the center used to compute distances as an orange cross
- **"dist"** shows the distances of each sample to the center provided in `filter_outliers()` (in function of sample index)
- **"hist"** draws an histogram of the distances of the samples to the center provided in `filter_outliers()`
- **"all"** displays a figure with all of the options above

See Also

`filter_outliers()`, `print.BRIC.Filtering()`, `clustBRIC()`, `median_rec()`, `median_mv()`, `depth_values()`

Examples

```
# Illustrative data
data <- rbind(
  mvtnorm::rmvnorm(300, c(0, 0), diag(2) * 3 - 1),
  mvtnorm::rmvnorm(100, c(15, 20), diag(2)),
  mvtnorm::rmvnorm(150, c(-10, 15), diag(2) * 2 - 0.5),
  mvtnorm::rmvnorm(200, c(5, 5), diag(2) * 200)
)

# Process the data
filtering <- filter_outliers(data, median_rec(data)$median, test = "DIP", debug = TRUE)

# Plot all default figures
plot(filtering)

# Plot P.Values and Scatterplot only
plot(filtering, contents = c("pvalues", "scatterplot"))

# Change the layout to vertical
plot(filtering, contents = c("pvalues", "scatterplot"), mfrow = c(2, 1))

# Remove title, subtitle, and axis labels
plot(filtering, contents = "scatterplot", main = "", sub = "",
  ylab = "", xlab = "")

# Other graphical options
plot(filtering, contents = "scatterplot", asp = 1,
  xlim = c(-30, 30), ylim = c(-30, 30))

plot(filtering,
  contents = "scatterplot", asp = 1, pch = 4, lwd = 2, col = "blue",
  colSelection = "green", showCenter = FALSE
```

```
)

plot(filtering, contents = "hist", main = "My Histogram",
  showSelection = FALSE, breaks = 50)
```

`plot.BRIC.MedianRec` *Plot method for* `BRIC.MedianRec` *objects*

Description

Plot method for `BRIC.MedianRec` objects

Usage

```
## S3 method for class 'BRIC.MedianRec'
plot(x, nbIterations = 5, showMedian = FALSE, showMax = FALSE, ...)
```

Arguments

`x`	An object of class `BRIC.MedianRec` (see `median_rec()`)
`nbIterations`	Number of iterations to display, or 0 to show all of them (default: 5)
`showMedian`	Logical value, to show the final recursive median (indicated by a "+")
`showMax`	Logical value, to show the overall deepest point, or the center of the first MCD/MVE iteration (indicated by a "x")
`...`	Other arguments passed to or from other methods

See Also

`median_rec()`, `median_mv()`, `clustBRIC()`

Examples

```
# Illustrative data
data <- rbind(
  mvtnorm::rmvnorm(300, c(0, 0), diag(2)),
  mvtnorm::rmvnorm(100, c(15, 20), diag(2) * 3 - 1),
  mvtnorm::rmvnorm(150, c(-10, 15), diag(2) * 2 - 0.5),
  mvtnorm::rmvnorm(200, c(5, 5), diag(2) * 200)
)

# Process the data
res <- median_rec(data)

# Default plot
plot(res)

# Adjust axis
plot(res, asp = 1, xlim = c(-20, 20), ylim = c(-20, 20))

# Change other graphical options
plot(res, showMedian = TRUE, pch = 16, main = "Recursive Median")
```

print.BRIC *Print method for* BRIC *objects*

Description

Print method for BRIC objects

Usage

```
## S3 method for class 'BRIC'
print(x, maxDisplayed = NULL, ...)
```

Arguments

x	An object of class BRIC (see clustBRIC())
maxDisplayed	Number of elements to display in the output. Set to NULL (or 0) to show all values.
...	Other arguments passed to or from other methods

See Also

clustBRIC(), plot.BRIC()

print.BRIC.Filtering *Print method for* BRIC.Filtering *objects*

Description

Print method for BRIC.Filtering objects

Usage

```
## S3 method for class 'BRIC.Filtering'
print(x, maxDisplayed = 200, ...)
```

Arguments

x	An object of class BRIC.Filtering (see filter_outliers())
maxDisplayed	Number of elements to display in the output (default: 200). Set to NULL (or 0) to show all values.
...	Other arguments passed to or from other methods

See Also

filter_outliers(), plot.BRIC.Filtering(), clustBRIC()

print.BRIC.MedianRec *Print method for* BRIC.MedianRec *objects*

Description

Print method for BRIC.MedianRec objects

Usage

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

Arguments

x	An object of class BRIC.MedianRec (see median_rec())
...	Other arguments passed to or from other methods

See Also

median_rec(), plot.BRIC.MedianRec()

test_custom *Custom test of a distribution*

Description

Custom test of a distribution

Usage

```
test_custom(data, test, center = NULL, distances = NULL, warnings = FALSE)
```

Arguments

data	Matrix of numerical values containing the observations (one per row, with columns corresponding to variables)
test	Statistical test used (user-defined function)
center	Center of the observations from data
distances	Unidimensional array of numerical values (distances of the observations to center)
warnings	Logical value, to display the warnings and errors caught

Value

p-value of the test

See Also

filter_outliers()

`test_multinormality` *Test the multivariate normality of a distribution*

Description

Test the multivariate normality of a distribution

Usage

```
test_multinormality(
  values,
  test = "Mardia",
  threshold = 0.05,
  data,
  center,
  warnings = FALSE
)
```

Arguments

`values`	Unidimensional array of numerical values (distances)
`test`	Statistical test used, valid options are "Mardia", "Kurtosis", "Skewness", "KS", "KS-adj", "Shapiro", "Lillie", and "Chisq"
`threshold`	Threshold of significance for the statistical test (default: 0.05)
`data`	Matrix of numerical values containing the observations (one per row, with columns corresponding to variables)
`center`	Center of the observations from `data`
`warnings`	Logical value, to display the warnings and errors caught

Details

Parameter `data` is only required for the tests "Mardia", "Skewness", "Kurtosis" and "Chisq", while parameter `center` is only required for "Chisq"

Value

p-value of the test (lower values suggest non normality)

See Also

`filter_outliers()`

`test_unimodality` *Test the unimodality of a distribution*

Description

Test the unimodality of a distribution

Usage

```
test_unimodality(values, test = "DIP", warnings = FALSE)
```

Arguments

`values`	Unidimensional array of numerical values (distances)
`test`	Statistical test used (for now the only option is "DIP")
`warnings`	Logical value, to display the warnings and errors caught

Value

p-value of the test (lower values suggest multimodality)

See Also

`filter_outliers()`

Index