Package: hclust1d (via r-universe)

October 22, 2024

Title Hierarchical Clustering of Univariate (1d) Data

Version 0.1.1.9000

Description Univariate agglomerative hierarchical clustering with a comprehensive list of choices of a linkage function in O(n*log n) time. The better algorithmic time complexity is paired with an efficient 'C++' implementation.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

URL `https://github.com/SzymonNowakowski/hclust1d`

BugReports `https://github.com/SzymonNowakowski/hclust1d/issues`

RoxygenNote 7.2.3

LinkingTo Rcpp

Imports Rcpp, utils

Config/testthat/edition 3

VignetteBuilder knitr

Repository https://szymonnowakowski.r-universe.dev

RemoteUrl https://github.com/szymonnowakowski/hclust1d

RemoteRef HEAD

RemoteSha 809df7559f2e37c34b61b4d21e15afcea5491ff6

Contents

hclust1d-package *hclust1d-package*

Description

Univariate agglomerative hierarchical clustering with a comprehensive list of choices of a linkage function in O(n*log n) time. The better algorithmic time complexity is paired with an efficient 'C++' implementation.

All `hclust1d` Functions

Similar in use to `stats::hclust`. It consists of the following functions:

`hclust1d` - univariate agglomerative hierarchical clustering routine.

`supported_methods` - lists all currently supported linkage methods.

`supported_dist.methods` - lists all currently supported distance methods.

For more information see a friendly "Getting started" vignette:

Examples

```
## Not run:
vignette("getting-started", package="hclust1d")

## End(Not run)
```

hclust1d *Hierarchical Clustering for 1D*

Description

Univariate hierarchical agglomerative clustering routine with a few possible choices of a linkage function.

Usage

```
hclust1d(x, distance = FALSE, squared = FALSE, method = "complete")
```

Arguments

`x` a vector of 1D points to be clustered, or a distance structure as produced by `dist`.

`distance` a logical value indicating, whether `x` is a vector of 1D points to be clustered (`distance = FALSE`, the default), or a distance structure (`distance = TRUE`).

`squared` a logical value indicating, whether `distance` is squared (`squared = TRUE`) or not (`squared = FALSE`, the default). Its value is irrelevant for `distance = FALSE` setting.

`method` linkage method, with `"complete"` as a default. See `supported_methods` for the complete list.

Details

If `x` is a distance matrix, the first step of the algorithm is computing a conforming vector of 1D points (with arbitrary shift and sign choices).

Univariate hierarchical clustering is performed for the provided or calculated vector of points: initially, each point is assigned its own *singleton* cluster, and then the clusters get merged with their nearest neighbors, two at a time.

For `method = "single"`, there is no need to recompute distances, since the original inter-point distances are also the inter-cluster distances, so the algorithm requires only sorting the original points and then sorting the distances.

For other linkage methods, two distances (between the merged cluster and the preceding and the following clusters) get recomputed at each merge, and the resulting distance structure gets updated in an efficiently implemented heap providing a priority queue functionality (the access to the current minimum distance) in O(log n) time at each step. The resulting algorithm has O(n*log n) time complexity.

Value

A list object with S3 class `"hclust"`, compatible with a regular `stats::hclust` output:

`merge` a matrix with n-1 rows and 2 columns. Each i-th row of the matrix details merging performed at the i-th step of the algorithm. If the *singleton* cluster was merged at this step, the value of the element is negative and its absolute value equals the index of this point. Otherwise, a positive value, say j, of an element in i-th row, indicates that at the stage i a cluster created at a previous stage j was merged.

`height` a vector with n-1 values, with the i-th value indicating the distance between the two clusters merged at the i-th step of the algorithm.

`order` a permutation of the input points sorting them in an increasing order. Since the sign of points computed from the distance structure can be arbitrarily chosen, in the case of a distance structure input, the order can be increasing or decreasing.

`labels` either point names, or point values, or point indices, in the order of availability.

`call` the call which produced the results.

`method` the linkage method used for clustering.

`dist.method` the distance method used in building the distance matrix; or `"euclidean"`, if `x` is a vector of 1D points

Note

Please note that in stats::hclust, the inter-cluster distances for ward.D, centroid and median linkages (returned as height) are *squared* euclidean distances between the relevant clusters' centroids, although that behavior is not well documented. This behavior is also in odds with other linkage methods, for which *unsquared* euclidean distances are returned. The implementation in hclust1d::hclust1d follows that behavior in full.

Also, stats::hclust expects *squared* euclidean distance structure as input for method="ward.D", method="centroid" and method="median", although the latter is not well documented, either. Squared distance is not a proper distance (a triangle inequality may not be maintained), so it should be considered *dissimilarity* instead.

To retain compatibility, hlust1d::hclust1d accepts x in a form of a squared euclidean distance structure between points as well (indicated by both distance and squared arguments set to TRUE). Also, note that hlust1d::hclust1d returns the same heights for unsquared proper distances in x (with distance=TRUE setting and the default squared=FALSE argument) and for x in a form of a vector of 1D points (with the default distance=FALSE argument). Please consult the Examples section below for further reference on that behavior.

See Also

supported_methods for listing of all currently supported linkage methods, supported_dist.methods for listing of all currently supported distance methods.

Examples

```
# Faster replacements for
# stats::hclust(dist(rnorm(100))) with a default complete linkage
dendrogram <- hclust1d(rnorm(100))
dendrogram <- hclust1d(dist(rnorm(100)), distance = TRUE)

# Faster replacements for
# stats::hclust(dist(rnorm(100)), method = "average")
dendrogram <- hclust1d(rnorm(100), method = "average")
dendrogram <- hclust1d(dist(rnorm(100)), distance = TRUE, method = "average")

# Faster replacements for
# stats::hclust(dist(rnorm(100))^2, method = "centroid")
# Note that stats::hclust expects squared euclidean distance input for centroid linkage
# While in case of hclust1d, 3 below calls result in the equivalent output
dendrogram <- hclust1d(rnorm(100), method = "centroid")
dendrogram <- hclust1d(dist(rnorm(100)), distance = TRUE, method = "centroid")
dendrogram <- hclust1d(dist(rnorm(100))^2, distance = TRUE, squared = TRUE, method = "centroid")

# Faster replacements for
# stats::hclust(dist(rnorm(100))^2, method = "median")
# Note that stats::hclust expects squared euclidean distance input for median linkage
# While in case of hclust1d, 3 below calls result in the equivalent output
dendrogram <- hclust1d(rnorm(100), method = "median")
dendrogram <- hclust1d(dist(rnorm(100)), distance = TRUE, method = "median")
dendrogram <- hclust1d(dist(rnorm(100))^2, distance = TRUE, squared = TRUE, method = "median")
```

```
# Faster replacements for
# stats::hclust(dist(rnorm(100)), method = "mcquitty")
dendrogram <- hclust1d(rnorm(100), method = "mcquitty")
dendrogram <- hclust1d(dist(rnorm(100)), distance = TRUE, method = "mcquitty")

# Faster replacements for
# stats::hclust(dist(rnorm(100))^2, method = "ward.D")
# Note that stats::hclust expects squared euclidean distance input for ward.D linkage
# While in case of hclust1d, 3 below calls result in the equivalent output
dendrogram <- hclust1d(rnorm(100), method = "ward.D")
dendrogram <- hclust1d(dist(rnorm(100)), distance = TRUE, method = "ward.D")
dendrogram <- hclust1d(dist(rnorm(100))^2, distance = TRUE, squared = TRUE, method = "ward.D")

# Faster replacements for
# stats::hclust(dist(rnorm(100)), method = "ward.D2")
dendrogram <- hclust1d(rnorm(100), method = "ward.D2")
dendrogram <- hclust1d(dist(rnorm(100)), distance = TRUE, method = "ward.D2")

# Faster replacements for
# stats::hclust(dist(rnorm(100)), method = "single")
dendrogram <- hclust1d(rnorm(100), method = "single")
dendrogram <- hclust1d(dist(rnorm(100)), distance = TRUE, method = "single")

# A 1D-specific true median linkage
dendrogram <- hclust1d(rnorm(100), method = "true_median")

# Plotting the resulting dendrogram
plot(dendrogram)
```

supported_dist.methods *Supported Distance Methods*

Description

Lists all choices of a distance method currently supported in hclust1d via stats::dist call.

Usage

```
supported_dist.methods()
```

Value

A character vector with currently supported distance methods.

Examples

```
if ("minkowski" %in% supported_dist.methods()) {   # the condition under if evaluates to TRUE
   dendrogram <- hclust1d(dist(rnorm(100), method = "minkowski", p = 3))
   plot(dendrogram)
} else {
   stop("Error: minkowski distance method not supported in hclust1d")
}
```

---

supported_methods *Supported Linkage Methods*

Description

Lists all choices of a linkage method currently supported in hclust1d.

Usage

```
supported_methods()
```

Value

A character vector with currently supported linkage methods.

Examples

```
if ("median" %in% supported_methods()) {    # the condition under if evaluates to TRUE
   dendrogram <- hclust1d(rnorm(100), method = "median")
   plot(dendrogram)
} else {
   stop("Error: median linkage method not supported in hclust1d")
}
```

Index