Package ‘HCV’

October 12, 2022

**Title**  Hierarchical Clustering from Vertex-Links

**Description**  Hierarchical clustering for spatial data, which requires clustering results not only homogeneous in non-geographical features among samples but also geographically close to each other within a cluster. It modified typically used hierarchical agglomerative clustering algorithms for introducing the spatial homogeneity, by considering geographical locations as vertices and converting spatial adjacency into whether a shared edge exists between a pair of vertices (Tzeng & Hsu, 2022) <arXiv:2201.08302>. The constraints of the vertex links automatically enforce the spatial contiguity property at each step of iterations. In addition, methods to find an appropriate number of clusters and to report cluster members are also provided.

**Version**  1.2.0

**Date**  2022-02-20

**Depends**  R (>= 4.0.0)

**Imports**  BLSM (>= 0.1.0), cluster, geometry (>= 0.4.5), graphics, grDevices, M3C (>= 1.12.0), MASS, Matrix, rgeos (>= 0.5.1), sp (>= 1.4.2)

**Suggests**  alphahull, knitr, fields (>= 11.4)

**Maintainer**  ShengLi Tzeng <slt.cmu@gmail.com>

**License**  LGPL-3

**Date/Publication**  2022-02-22 14:10:01 UTC

**Encoding**  UTF-8

**RoxygenNote**  7.1.1

**Author**  ShengLi Tzeng [cre, aut], Hao-Yun Hsu [aut]

**Repository**  CRAN

**R topics documented:**

getCluster .......................... 2
getCluster  

Determining Appropriate Clusters for HCV Objects

Description

The function provides two methods to determine an appropriate number of clusters for an HCV object, and reports individual cluster members. One of the method is a novel internal index named Spatial Mixture Index (SMI), considering both the within-cluster sum of squared difference of geographical attributes and non-geographical attributes. The other is an M3C-based method taking account of the stability of clusters.

Usage

getCluster(
  HCVobj,
  method = c("SMI", "M3C"),
  Kmax = 10,
  niter = 25,
  criterion = "PAC"
)

Arguments

HCVobj  an object resulting from calling the HCV function.
method  character indicating the method to determine an appropriate number of clusters. Default ‘SMI’ is faster, while ‘M3C’ is more precise but slower.
Kmax  integer for the upper bound of the potential number of clusters to be considered.
niter  integer for the number of resampling, only used in method="M3C".
criterion  character indicating whether to use ‘PAC’ or ‘entropy’ as the objective function. Default is ‘PAC’. Only used in method=‘M3C’. See the reference for details.

Value

A vector giving the cluster ID assigned for each sample.

Author(s)

ShengLi Tzeng and Hao-Yun Hsu.
References


See Also

M3C

Examples

```
set.seed(0)
pcase <- synthetic_data(3,30,0.02,100,2,2)
HCVobj <- HCV(pcase$geo, pcase$feat)
smi <- getCluster(HCVobj,method="SMI")
oldpar <- par(no.readonly = TRUE)
par(mfrow=c(2,2))
labcolor <- (pcase$labels+1)%%3+1
plot(pcase$feat, col = labcolor, pch=19, xlab = 'First attribute',
ylab = 'Second attribute', main = 'Feature domain')
plot(pcase$geo, col = labcolor, pch=19, xlab = 'First attribute',
ylab = 'Second attribute', main = 'Geometry domain')
plot(pcase$feat, col=factor(smi),pch=19, xlab = 'First attribute',
ylab = 'Second attribute',main = 'Feature domain')
plot(pcase$geo, col=factor(smi),pch=19, xlab = 'First attribute',
ylab = 'Second attribute',main = 'Geometry domain')
par(oldpar)
```

Description

This function implements the hierarchical clustering for spatial data. It modified typically used hierarchical agglomerative clustering algorithms for introducing the spatial homogeneity, by considering geographical locations as vertices and converting spatial adjacency into whether a shared edge exists between a pair of vertices.

Usage

```
HCV(  
   geometry_domain,  
   feature_domain,  
   linkage = "ward",  
   diss = "none",  
   adjacency = FALSE,  
   dist_method = "euclidean"  
)
```
Arguments

geometry_domain
one of the three formats: (i) \( n \) by \( d \) matrix (NA not allowed), (ii) a SpatialPolygonsDataFrame object defining polygons, (iii) a matrix with 0-1 value adjacency (with adjacency=TRUE)

feature_domain 
either (i) \( n \) by \( p \) matrix (NA allowed) for \( n \) samples with \( p \) attributes, or (ii) \( n \) by \( n \) matrix (NA not allowed) with dissimilarity between \( n \) samples (with \( \text{diss} = \text{precomputed} \) )

linkage
the agglomeration method to be used, one of "ward", "single", "complete", "average" (= UPGMA), "weight" (= WPGMA), "median" (= WPGMC) or "centroid" (= UPGMC). Default is 'ward'.

diss
character indicating if feature_domain is a dissimilarity matrix: 'none' for not dissimilarity, and 'precomputed' for dissimilarity. Default is 'none'.

adjacency
logical indicating if geometry_domain is a adjacency matrix. Default is FALSE.

dist_method
the distance measure to be used when feature_domain is not a dissimilarity matrix (\( \text{diss} = \text{none} \) ), one of "euclidean", "correlation", "abscor", "maximum", "manhattan", "canberra", "binary" or "minkowski". Default is 'euclidean'.

Details

geometry_domain can be a user-specific adjacency matrix, an \( n \) by \( d \) matrix with geographical coordinates for point-level data, or a SpatialPolygonsDataFrame object defining polygons for areal data. If an adjacency matrix is given, the user should use adjacency=TRUE.

Value

An object of class hclust which describes the tree produced by the clustering process. See the documentation in hclust.

Author(s)

ShengLi Tzeng and Hao-Yun Hsu.

References


See Also

hclust

Examples

calculate seed(0)
pcase <- synthetic_data(3,30,0.02,100,2,2)
HCVobj <- HCV(pcase$geo, pcase$feat)
smi <- getCluster(HCVobj,method="SMI")
oldpar <- par(no.readonly = TRUE)
par(mfrow=c(2,2))
labcolor <- (pcase$labels+1)%3+1
plot(pcase$feat, col = labcolor, pch=19, xlab = 'First attribute', ylab = 'Second attribute', main = 'Feature domain')
plot(pcase$geo, col = labcolor, pch=19, xlab = 'First attribute', ylab = 'Second attribute', main = 'Geometry domain')
plot(pcase$feat, col=factor(smi),pch=19, xlab = 'First attribute', ylab = 'Second attribute',main = 'Feature domain')
plot(pcase$geo, col=factor(smi),pch=19, xlab = 'First attribute', ylab = 'Second attribute',main = 'Geometry domain')
par(oldpar)

---

**plotMap**

*Drawing a Thematic Map with a Quantitative Feature*

**Description**

Plot the polygons in a SpatialPolygonsDataFrame object, and turn the values of a quantitative feature into colors over individual polygons.

**Usage**

```r
plotMap(
  map,
  feat,
  color = topo.colors(10),
  main = "",
  bar_title = "rank",
  zlim = NULL
)
```

**Arguments**

- `map` SpatialPolygonsDataFrame object consisting of data and polygons.
- `feat` numeric vector having the same elements as the number of polygons in the input map.
- `color` vector of distinct colors for converting values of `feat`.
- `main` character specifying the main title.
- `bar_title` character specifying the text over the color bar.
- `zlim` length-2 numeric vector specifying the range of values to be converted.

**Value**

A colored map.
See Also

SpatialPolygonsDataFrame

Examples

```r
require(sp)
grd <- GridTopology(c(1,1), c(1,1), c(5,5))
polys <- as(grd, "SpatialPolygons")
centroids <- coordinates(polys)
gdomain <- SpatialPolygonsDataFrame(polys, data=data.frame(x=centroids[,1],
y=centroids[,2], row.names=row.names(polys)))
feat <- gdomain$x*5+gdomain$y^2
plotMap(gdomain,feat)
```

synthetic_data Generating Point-level Data Having Several Groups

Description

Generation of synthetic point-level data based on a method proposed by Lin et al. (2005).

Usage

```r
synthetic_data(k, f, r, n, feature, geometry, homogeneity = TRUE)
```

Arguments

- `k` integer specifying the number of groups.
- `f` positive number controlling the concentration of generated samples toward large groups.
- `r` positive number controlling the variance of individual attributes on the feature domain.
- `n` integer specifying the total number of sampled points.
- `feature` integer specifying the number of attributes for the feature domain.
- `geometry` integer specifying the number of attributes for the geometry domain.
- `homogeneity` logical indicating whether to force the centers of the feature domain to be the same as those of the geometry domain. Default is TRUE.

Value

A list with two matrices and a vector of labels. One matrix is for the feature domain and the other is for the geometry domain, both of which have `n` sampled points. The vector of labels indicates which cluster each sample belongs to.
tessellation_adjacency_matrix

Author(s)
ShengLi Tzeng and Hao-Yun Hsu.

References

Examples

def set.seed(0)
def pcase <- synthetic_data(3,30,0.02,100,2,2)
def oldpar <- par(no.readonly = TRUE)
def par(mfrow=c(1,2))
def labcolor <- (pcase$labels+1)%%3+1
plot(pcase$feat, col = labcolor, pch=19, xlab = 'First attribute',
     ylab = 'Second attribute', main = 'Feature domain')
def plot(pcase$geo, col = labcolor, pch=19, xlab = 'First attribute',
     ylab = 'Second attribute', main = 'Geometry domain')
def par(oldpar)

---

tessellation_adjacency_matrix

**Adjacency Matrix from Tessellation**

Description
This function deals with spatial data having a point-level geometry domain. It converts the spatial proximity into an adjacency matrix based on Voronoi tessellation or Delaunay triangulation.

Usage
tessellation_adjacency_matrix(geometry_domain)

Arguments

- **geometry_domain**
  - A matrix of geographical coordinates for \( n \) points in \( d \)-dimensional space.

Value
An \( n \times n \) matrix with 0-1 values indicating the adjacency between the \( n \) input points.

Author(s)
ShengLi Tzeng and Hao-Yun Hsu.
References


Examples

```r
if( require(fields) & require(alphahull) ) {
  pts <- ChicagoO3$x
  rownames(pts) <- LETTERS[1:20]
  Vcells <- delvor(pts)
  plot(Vcells,wlines='vor',pch='.')
  text(pts,rownames(pts))
  Amat <- tessellation_adjacency_matrix(pts)
}
```
Index

getCluster, 2
hclust, 4
HCV, 3
M3C, 3
plotMap, 5
SpatialPolygonsDataFrame, 6
synthetic_data, 6
tessellation_adjacency_matrix, 7