Package ‘ADPclust’

October 12, 2022

Type Package

Title Fast Clustering Using Adaptive Density Peak Detection

Version 0.7

Description An implementation of ADPclust clustering procedures (Fast Clustering Using Adaptive Density Peak Detection). The work is built and improved upon the idea of Rodriguez and Laio (2014)<DOI:10.1126/science.1242072>. ADPclust clusters data by finding density peaks in a density-distance plot generated from local multivariate Gaussian density estimation. It includes an automatic centroids selection and parameter optimization algorithm, which finds the number of clusters and cluster centroids by comparing average silhouettes on a grid of testing clustering results; It also includes a user interactive algorithm that allows the user to manually selects cluster centroids from a two dimensional “density-distance plot”. Here is the research article associated with this package: “Wang, Xiao-Feng, and Yifan Xu (2015)<DOI:10.1177/0962280215609948> Fast clustering using adaptive density peak detection.” Statistical methods in medical research", url: http://smm.sagepub.com/content/early/2015/10/15/0962280215609948.abstract.

Depends R (>= 3.0.0),

Imports dplyr, cluster, fields, knitr

URL https://github.com/ethanyxu/ADPclust

BugReports https://github.com/ethanyxu/ADPclust/issues

VignetteBuilder knitr

License GPL (>= 2)

LazyData true

Suggests rmarkdown, testthat

RoxygenNote 5.0.1

NeedsCompilation no

Author Yifan (Ethan) Xu [aut, cre],
Xiao-Feng Wang [aut]

Maintainer Yifan (Ethan) Xu <ethan.yifanxu@gmail.com>

Repository CRAN

Date/Publication 2016-10-15 11:37:01
adpclust

Fast Clustering Using Adaptive Density Peak Detection

Description

Clustering of data by finding cluster centers from estimated density peaks. ADPclust is a non-iterative procedure that incorporates multivariate Gaussian density estimation. The number of clusters as well as bandwidths can either be selected by the user or selected automatically through an internal clustering criterion.

Usage

adpclust(x = NULL, distm = NULL, p = NULL, centroids = "auto", h = NULL, htype = "amise", nclust = 2:10, ac = 1, f.cut = c(0.1, 0.2, 0.3), fdelta = "mnorm", dmethod = "euclidean", draw = FALSE)

Arguments

x numeric data frame where rows are observations and columns are variables. One of x and distm must be provided.

distm distance matrix of class 'dist'. distm is ignored if x is given.
p number of variables (ncol(x)). This is only needed if neither x nor h is given.
centroids  character string specifying how cluster centroids are selected. Valid options are "user" and "auto".

h         nonnegative number specifying the bandwidth in density estimation. If h is NULL, the algorithm attempts to find h in a neighborhood centered at either the AMISE bandwidth or ROT bandwidth (see htype).

htype     character string specifying the method used to calculate a reference bandwidth for the density estimation. htype is ignored if h is given. Valid options of are "ROT" and "AMISE" (see details).

nclust    integer, or a vector of integers specifying the pool of the number of clusters in automatic variation. The default is 2:10.

ac        integer indicating which automatic cut method is used. This is ignored if centroids = 'user'. The valid options are:

• ac = 1: centroids are chosen to be the data points x’s with the largest delta values such that f(x) >= a’th percentile of all f(x). The number of centroids is given by the parameter nclust. The cutting percentile(s) is given by the parameter f.cut.

• ac = 2: let l denote the straight line connecting (min(f), max(delta)) and (max(f), min(delta)). The centroids are selected to be data points above l and farthest away from it. The number of centroids is given by the parameter nclust.

f.cut     number between (0, 1) or numeric vector of numbers between (0, 1). f.cut is used when centroids = "auto" and ac = 1 to automatically select cluster centroids from the decision plot (see ac). The default is c(0.1, 0.2, 0.3).

fdelta    character string that specifies the method used to estimate local density f(x) at each data point x. The default (recommended) is "mnorm" that uses a multivariate Gaussian density estimation to calculate f. Other options are listed below. Here ‘distsm’ denotes the distance matrix.

• unorm(f <- 1/(h * sqrt(2 * pi)) * rowSums(exp(-(distsm/h)^2/2))); Univariate Gaussian smoother

• weighted(f <- rowSums(exp(-(distsm/h)^2))); Univariate weighted smoother

• count(f <- rowSums(distsm < h) - 1); Histogram estimator (used in Rodriguez [2014])

dmethod   character string that is passed to the ‘method’ argument in function dist(), which is used to calculate the distance matrix if ’distsm’ is not given. The default is "euclidean".

draw      boolean. If draw = TRUE the clustering result is plotted after the algorithm finishes. The plot is produced by by plot.adpclust(ans), where ’ans’ is the outcome of ’adpclust()’

Details

Given n data points x’s in p dimensions, adpclust() calculates f(x) and delta(x) for each data point x, where f(x) is the local density at x, and delta(x) is the shortest distance between x and y for all y such that f(x) <= f(y). Data points with large f and large delta values are labeled class centroids. In other words, they appear as isolated points in the upper right corner of the f vs. delta plot (the
After cluster centroids are determined, other data points are clustered according to their distances to the closest centroids. A bandwidth (smoothing parameter) $h$ is used to calculate local density $f(x)$ in various ways. See parameter 'fdelta' for details. If centroids = 'user', then $h$ must be explicitly provided. If centroids = 'auto' and $h$ is not specified, then it is automatically selected from a range of testing values: First a reference bandwidth $h_0$ is calculated by one of the two methods: Scott's Rule-of-Thumb value (htype = "ROT") or Wand's Asymptotic-Mean-Integrated-Squared-Error value (htype = "AMISE"), then 10 values equally spread in the range $[1/3h_0, 3h_0]$ are tested. The value that yields the highest silhouette score is chosen as the final $h$.

Value

An 'adpclust' object that contains the list of the following items.

- clusters: Cluster assignments. A vector of the same length as the number of observations.
- centers: Indices of the clustering centers.
- silhouette: Silhouette score from the final clustering result.
- nclust: Number of clusters.
- $h$: Final bandwidth.
- $f$: Final density vector $f(x)$.
- delta: Final delta vector $\delta(x)$.
- selection.type: 'user' or 'auto'.

References

- GitHub: https://github.com/ethanyxu/ADPclust

Examples

```r
# Load a data set with 3 clusters
data(clust3)

# Automatically select cluster centroids
ans <- adpclust(clust3, centroids = "auto", draw = FALSE)
summary(ans)
plot(ans)

# Specify distm instead of data
distm <- FindDistm(clust3, normalize = TRUE)
ans.distm <- adpclust(distm = distm, p = 2, centroids = "auto", draw = FALSE)
identical(ans, ans.distm)

# Specify the grid of h and nclust
ans <- adpclust(clust3, centroids = "auto", h = c(0.1, 0.2, 0.3), nclust = 2:6)
```
# Specify that bandwidths should be searched around
# Wand's Asymptotic-Mean-Integrated-Squared-Error bandwidth
# Also test 3 to 6 clusters.
ans <- adpclust(clust3, centroids = "auto", htype = "AMISE", nclust = 3:6)

# Set a specific bandwidth value.
ans <- adpclust(clust3, centroids = "auto", h = 5)

# Change method of automatic selection of centers
ans <- adpclust(clust3, centroids = "auto", nclust = 2:6, ac = 2)

# Specify that the single "ROT" bandwidth value by
# using the 'ROT()' function
ans <- adpclust(clust3, centroids = "auto", h = ROT(clust3))

# Centroids selected by user
## Not run:
ans <- adpclust(clust3, centroids = "user", h = ROT(clust3))
## End(Not run)

# A larger data set
data(clust5)
ans <- adpclust(clust5, centroids = "auto", htype = "ROT", nclust = 3:5)
summary(ans)
plot(ans)

### AMISE

#### AMISE bandwidth

**Description**

Calculate the AMISE bandwidth from either a data frame, or from the number of observations and the dimension of the data.

**Usage**

`AMISE(x, y = NULL)`

**Arguments**

- `x` the number of variables (if `y` is given), or a data frame or a matrix (if `y` is missing).
- `y` the number of observations. If `y` is missing then `x` is interpreted as the data matrix.

**Details**

IMPORTANT NOTE: The standard deviation of each variable is omitted in this formula.
Value

AMISE bandwidth.

clust10

1000 5-dimensional data points that form ten clusters

Description

Generated from the genRandomClust() function of the "clusterGeneration" package with separation value 0.2.

Format

data frame

clust3

90 2-dimensional data points that form three clusters

Description

Randomly generated from three normal distributions.

Format

data frame

clust5

500 5-dimensional data points that form five clusters

Description

500 5-dim points in 5 clusters. Generated from the genRandomClust() function of the "clusterGeneration" package with separation value 0.1.

Format

data frame
clust5.1

500 5-dimensional data points that form five clusters

Description
Generated from the genRandomClust() function of the "clusterGeneration" package with separation value 0.01 (tightly clustered).

Format
data frame

dat_gene
243-dimensional gene expression data of 38 patients (243 genes)

Description
38 by 243 matrix. Each row represents a patient. Each column represents a gene.

Format
matrix

defCol
Default colors

Description
Returns 10 default colors

Usage
defCol()

Value
vector of colors
FindCentersAutoD

Automatically finds centers with diagonal \( f(x) \) vs \( \delta(x) \) thresholds

Description

Automatically finds centers with diagonal \( f(x) \) vs \( \delta(x) \) thresholds. This is used in adpclust() with \( ac = 2 \). It finds points that are above and farthest from the diagonal line in the \( f \) vs. \( \delta \) plots, and label them to be centers.

Usage

FindCentersAutoD(f, delta, nclust)

Arguments

- \( f \) vector of local distance.
- \( \delta \) vector of minimal distances to higher ground.
- \( nclust \) number of clusters. Can be a single integer or a vector of integers. Duplicates are silently removed.

Value

a list of vectors. Each vector gives the locations of centers.

Author(s)

Ethan Xu

FindCentersAutoV

Automatically find centers with vertical threshold

Description

Automatically find centers with vertical threshold vertical \( f(x) \) thresholds.

Usage

FindCentersAutoV(f, delta, f.cut = c(0.1, 0.2, 0.3), nclust, rm.dup = TRUE)
**Arguments**

- **f**
  vector of local distance $f(x)$. See the detail section of the help(adpclust).

- **delta**
  vector of minimal distances to higher ground $\delta(x)$. See the detail section of the help(adpclust).

- **f.cut**
  number between (0, 1) or numeric vector of numbers between (0, 1). Data points whose $f$ values are larger than $f\text{.cut}$ with large $\delta$ values are selected as centers. The default is $c(0.1, 0.2, 0.3)$.

- **nclust**
  number of clusters. It can be either a single integer or a vector of integers.

- **rm.dup**
  boolean. If TRUE (default) duplicated centers vectors are removed from returned list.

**Details**

Given $f$'s and $\delta$'s, cluster centers are chosen to be the data points whose $\delta$ values are high and $f$ values are larger than a fixed threshold. To be more specific, let $F$ denote the set of all $f(x)$. centers are selected as points with the largest $m$ $\delta$ values in the set $x \mid f(x) > a^{th}$ percentile of $F$. The number of centers $m$ is given by the parameter `nclust`. The cutting percentile $a$ is given by the parameter `f.cut`. When at least one of these two parameters are vectors, centers are selected based all combinations of them, and returned in a list.

**Value**

a list of vectors. Each vector contains the indices of selected centers.

**Author(s)**

Ethan Xu

---

**FindClustersAuto**

Automatically find cluster assignment given $f$ and $\delta$.

**Description**

This is the subroutine that automatically finds cluster assignments from given $f$ and $\delta$ by testing various parameter settings and find the one that maximizes the silhouette.

**Usage**

FindClustersAuto(distm, f, delta, ac = 1, nclust = 2:10, f.cut = c(0.1, 0.2, 0.3))
**FindClustersGivenCenters**

**Arguments**

- `distm`: the distance matrix
- `f`: vector of local distance f(x). See the help of adpclust() for details.
- `delta`: vector of minimal distances to higher ground delta(x). See the help of adpclust() for details.
- `ac`: type of auto selection. The valid options are 1 and 2. See the help of adpclust() for details.
- `nclust`: number of clusters to test. Either a single integer or a vector of integers.
- `f.cut`: number between (0, 1) or numeric vector of numbers between (0, 1). Data points whose f values are larger than f.cut with large delta values are selected as centers. The default is c(0.1, 0.2, 0.3). See the help of FindCentersAutoV() for more details.

**Value**

- `clusters`: Cluster assignments. A vector of the same length as the number of observations.
- `centers`: Indices of the clustering centers.
- `silhouette`: Silhouette score from the final clustering result.
- `nclust`: Number of clusters.

**Author(s)**

Ethan Xu

---

**Description**

Find cluster assignments from given centers and distance matrix. Each point is assigned to the center that has the shortest Euclidean distance.

**Usage**

FindClustersGivenCenters(distm, centers)

**Arguments**

- `distm`: distance matrix
- `centers`: vector of integers that gives the indices of centers. Duplications will be silently dropped.
Value
Cluster assignments. A vector of the same length as the number of observations.

Description
Plot the f vs. delta plot, then wait for the user to select centers of clusters by left clicking the points. In general points with both large f and large delta are good candidates of cluster centroids. Selected centers are highlighted. Press ESC to end the selection.

Usage
FindClustersManual(distm, f, delta)

Arguments
- distm: distance matrix.
- f: vector of local densities f(x). Same length of the number of observations.
- delta: vector of distances to the closest high ground delta(x). Same length of the number of observations.

Value
A list of the following items:
- clusters: Cluster assignments. A vector of the same length as the number of observations.
- centers: Indices of the clustering centers.
- silhouette: Silhouette score from the final clustering result.
- nclust: Number of clusters.

Examples
```r
data(clust3)
distm <- FindDistm(clust3, normalize = TRUE)
## Not run:
fd <- FindFD(distm, 2, "mnorm")
ans <- FindClustersManual(distm, fd$f, fd$delta)
names(ans)
ans$centers
## End(Not run)
```
FindDistm

Find the distance matrix from data.

Description

A wrapper of the dist() method, with the option to rescale the data with standard deviation of each dimension before calculating the distance matrix. NOTE: If fdelta='mnorm' is passed to adpclust(), then the distm is calculated from rescaled data internally, i.e. distm <- FindDistm(x, normalize = TRUE).

Usage

FindDistm(x, normalize = FALSE, method = "euclidean")

Arguments

- x: data
- normalize: boolean. Normalize data before calculating distance?
- method: passed to 'dist()'

Value

distance matrix of class dist.

Author(s)

Ethan Xu

FindFD

Find f and delta from distance matrix.

Description

Calculate f(x) and delta(x) from distm and h.

Usage

FindFD(distm, h, fdelta)
**Arguments**

- `distm`: distance matrix of class 'dist'.
- `h`: bandwidth.
- `fdelta`: character string that specifies the method used to estimate local density \( f(x) \) at each data point \( x \). The default is "mnorm" that uses a multivariate Gaussian density estimation to calculate \( f \). Other options are listed below. Here 'distm' denotes the distance matrix.
  - `unorm(f <- 1/(h * sqrt(2 * pi)) * rowSums(exp(-(distm/h)^2/2)));` Univariate Gaussian smoother
  - `weighted(f <- rowSums(exp(-(distm/h)^2)));` Univariate weighted smoother
  - `count(f <- rowSums(distm < h) - 1);` Histogram estimator (used in Rodriguez [2014])

**Value**

list of two items: \( f \) and \( \delta \).

---

**FindH**

*Find bandwidth \( h \).*

**Description**

Find bandwidth \( h \) from the number of observations \( n \) and the dimension \( p \).

**Usage**

`FindH(p, n, htype)`

**Arguments**

- `p`: dimension of data. The number of variables.
- `n`: the number of observations.
- `htype`: methods to calculate \( h \). The valid options are (case insensitive) "amise" or "rot".

**Value**

bandwidth \( h \).
### plot.adpclust

**Visualize the result of adpclust()**

**Description**

Plot the f vs. delta plot with selected centroids.

**Usage**

```r
## S3 method for class "adpclust"
plot(x, cols = "default", to.plot = c("cluster.sil", "fd"), ...)
```

**Arguments**

- `x`: an object of class "adpclust". Result of adpclust().
- `cols`: vector of colors used to distinguish different clusters. Recycled if necessary.
- `to.plot`: string vector that indicate which plot(s) to show. The two options are 'cluster.sil' (nclust vs. silhouette) and 'fd' (f vs. delta).
- `...`: Not used.

**Examples**

```r
## Load a data set with 3 clusters
data(clust3)
## Automatically select cluster centroids
ans <- adpclust(clust3, centroids = "auto")
plot(ans)
plot(ans, to.plot = "fd")
plot(ans, to.plot = "cluster.sil")
plot(ans, to.plot = c("cluster.sil", "fd")) #Default
```

---

### ROT

**Calculate ROT bandwidth**

**Description**

Calculate the ROT bandwidth either from a data frame, or from p and n.

**Usage**

```r
ROT(x, y = NULL)
```
Arguments

- **x**: the number of variables (if y is missing), or a data frame or a matrix (if y is not missing).
- **y**: the number of observations. If y is missing, x should be the data matrix.

Details

IMPORTANT NOTE: The standard deviation of each variable is omitted in this formula.

Value

- ROT bandwidth.

---

**summary.adpclust**

**Summary of adpclust**

**Description**

Summarizes the result from the adpclust() function.

**Usage**

```r
## S3 method for class 'adpclust'
summary(object, ...)
```

**Arguments**

- **object**: object of class "adpclust" that is returned from adpclust().
- **...**: other arguments. NOT used.
Index

* datasets
  clust10, 6
  clust3, 6
  clust5, 6
  clust5.1, 7
  dat_gene, 7

adpclust, 2
AMISE, 5

clust10, 6
clust3, 6
clust5, 6
clust5.1, 7

dat_gene, 7
defCol, 7

FindCentersAutoD, 8
FindCentersAutoV, 8
FindClustersAuto, 9
FindClustersGivenCenters, 10
FindClustersManual, 11
FindDistm, 12
FindFD, 12
FindH, 13

plot.adpclust, 14

ROT, 14

summary.adpclust, 15