Package 'adjclust'

April 27, 2023

Maintainer Pierre Neuvial <pierre.neuvial@math.univ-toulouse.fr>

Date 2023-04-24

Version 0.6.7

License GPL-3

Title Adjacency-Constrained Clustering of a Block-Diagonal Similarity Matrix

Description Implements a constrained version of hierarchical agglomerative clustering, in which each observation is associated to a position, and only adjacent clusters can be merged. Typical application fields in bioinformatics include Genome-Wide Association Studies or Hi-C data analysis, where the similarity between items is a decreasing function of their genomic distance. Taking advantage of this feature, the implemented algorithm is time and memory efficient. This algorithm is described in Ambroise et al (2019) <doi:10.1186/s13015-019-0157-4>.

Depends R (>= 4.0.0)

Imports stats, graphics, grDevices, Rcpp (>= 1.0.6), Matrix, sparseMatrixStats, methods, utils, capushe, ggplot2, dendextend, rlang

Suggests knitr, testthat, rmarkdown, rioja, HiTC, snpStats, BiocGenerics

biocViews Clustering, FeatureExtraction

VignetteBuilder knitr

URL https://pneuvial.github.io/adjclust/

BugReports https://github.com/pneuvial/adjclust/issues

RoxygenNote 7.2.3

LinkingTo Rcpp, RcppArmadillo

Encoding UTF-8

Language en-US

NeedsCompilation yes

Author Christophe Ambroise [aut],
Shubham Chaturvedi [aut],
Alia Dehman [aut],
Pierre Neuvial [aut, cre],
Guillem Rigaill [aut],
Nathalie Vialaneix [aut],
Gabriel Hoffman [aut]

Repository CRAN

Date/Publication 2023-04-27 18:20:02 UTC

R topics documented:

`adjClust` *Adjacency-constrained Clustering*

Description

Adjacency-constrained hierarchical agglomerative clustering

Usage

```
adjClust(
  mat,
  type = c("similarity", "dissimilarity"),
  h = ncol(mat) - 1,
  strictCheck = TRUE
)
```

Arguments

`mat`	A similarity matrix or a dist object. Most sparse formats from `sparseMatrix` are allowed
`type`	Type of matrix : similarity or dissimilarity. Defaults to `"similarity"`
`h`	band width. It is assumed that the similarity between two items is 0 when these items are at a distance of more than band width h. Default value is `ncol(mat)-1`
`strictCheck`	Logical (default to `TRUE`) to systematically check default of positivity in input similarities. Can be disabled to avoid computationally expensive checks when the number of features is large.

Details

Adjacency-constrained hierarchical agglomerative clustering (HAC) is HAC in which each observation is associated to a position, and the clustering is constrained so as only adjacent clusters are merged. These methods are useful in various application fields, including ecology (Quaternary data) and bioinformatics (e.g., in Genome-Wide Association Studies (GWAS)).

This function is a fast implementation of the method that takes advantage of sparse similarity matrices (i.e., that have 0 entries outside of a diagonal band of width `h`). The method is fully described in (Dehman, 2015) and based on a kernel version of the algorithm. The different options for the implementation are available in the package vignette entitled "Notes on CHAC implementation in adjclust.

Value

An object of class `chac` which describes the tree produced by the clustering process. The object is a list with the same elements as an object of class `hclust` (`merge`, `height`, `order`, `labels`, `call`, `method`, `dist.method`), and two extra elements:

- `mat`: (the data on which the clustering has been performed, possibly after the pre-transformations described in the vignette entitled "Notes on CHAC implementation in adjclust".
- `correction`: the value of the correction for non positive definite similarity matrices (also described in the same vignette). If `correction == 0`, it means that the initial data were not pre-transformed.

Note

When performed on a distance matrix d with the option `type = "dissimilarity"`, `adjclust` is identical to using the option `"ward.D"` on d^2 in the function `hclust` when the ordering of the (unconstrained) clustering (in `hclust`) is compatible with the natural ordering of objects used as a constraint. It is also equivalent (under the same assumption or orderings) to the option `"ward.D2"` performed on the distance matrix d itself, except for the final heights of the merges that are equal to the square of the heights obtained with `"ward.D2"` in `hclust`. See the vignette on implementation and (Murtagh and Legendre, 2014) for further details.

References

Murtagh F., and Legendre P. (2014). Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion? *Journal of Classification*, **31**, 274-295. DOI: doi:10.1007/s003570149161z.

Dehman A. (2015). *Spatial Clustering of Linkage Disequilibrium Blocks for Genome-Wide Association Studies*, PhD thesis, Universite Paris Saclay, France.

Ambroise C., Dehman A., Neuvial P., Rigaill G., and Vialaneix N (2019). Adjacency-constrained hierarchical clustering of a band similarity matrix with application to genomics. *Algorithms for Molecular Biology*, **14**(22). DOI: doi:10.1007/s1122201898066.

Randriamihamison N., Vialaneix N., and Neuvial P. (2020). Applicability and interpretability of Ward's hierarchical agglomerative clustering with or without contiguity constraints. *Journal of Classification*, **38**, 1-27. DOI: doi:10.1007/s0035702009377y.

See Also

snpClust to cluster SNPs based on linkage disequilibrium

hicClust to cluster Hi-C data

Examples

```
sim <- matrix(
c(1.0, 0.1, 0.2, 0.3,
  0.1, 1.0 ,0.4 ,0.5,
  0.2, 0.4, 1.0, 0.6,
  0.3, 0.5, 0.6, 1.0), nrow = 4)

## similarity, full width
fit1 <- adjClust(sim, "similarity")
plot(fit1)

## similarity, h < p-1
fit2 <- adjClust(sim, "similarity", h = 2)
plot(fit2)

## dissimilarity
dist <- as.dist(sqrt(2-(2*sim)))

## dissimilarity, full width
fit3 <- adjClust(dist, "dissimilarity")
plot(fit3)

## dissimilarity, h < p-1
fit4 <- adjClust(dist, "dissimilarity", h = 2)
plot(fit4)
```

---

chac *Class chac*

Description

S3 class for Constrained Hierarchical Agglomerative Clustering results

Usage

```
## S3 method for class 'chac'
as.hclust(x, ...)

## S3 method for class 'chac'
print(x, ...)

## S3 method for class 'chac'
```

```
head(x, ...)

## S3 method for class 'chac'
summary(object, ...)

## S3 method for class 'chac'
plot(
  x,
  y,
  ...,
  mode = c("standard", "corrected", "total-disp", "within-disp", "average-disp"),
  nodeLabel = FALSE
)

diagnose(x, graph = TRUE, verbose = TRUE)

correct(x)

cutree_chac(tree, k = NULL, h = NULL)
```

Arguments

x, object, tree	an object of class 'chac'
...	for plot, arguments passed to the function plot.dendrogram. Default values for type and leaflab are respectively set to "triangle" and "none"
y	not used
mode	type of dendrogram to plot (see Details). Default to "standard"
nodeLabel	(logical) whether the order of merging has to be displayed or not. nodeLabel=TRUE prints orders of fusion at corresponding nodes. Default to FALSE
graph	(logical) whether the diagnostic plot has to be displayed or not. Default to TRUE
verbose	(logical) whether to print a summary of the result or not. Default to TRUE
k	an integer scalar or vector with the desired number of groups
h	numeric scalar or vector with heights where the tree should be cut. Only available when the heights are increasing

Details

Methods for class 'chac'

When plot.chac is called with mode = "standard", the standard dendrogram is plotted, even though, due to contingency constrains, some branches are reversed (decreasing merges). When plot.chac is called with mode = "corrected", a correction is applied to original heights so as to have only non decreasing merges). It does not change the result of the clustering, only the look of the dendrogram for easier interpretation.

Other modes are provided that correspond to different alternatives described in Grimm (1987):

- in mode = "within-disp", heights correspond to within-cluster dispersion, *i.e.*, for a corresponding cluster, its height is
$$I(C) = \sum_{i \in C} d(i, g_C)$$
where d is the dissimilarity used to cluster objects and g_C is the center of gravity of cluster C. In this case, heights are always non decreasing;
- in mode = "total-disp", heights correspond to the total within-cluster dispersion. It is obtained from mode = "standard" by the cumulative sum of its heights. In this case, heights are always non decreasing;
- in mode = "average-disp", heights correspond to the within-cluster dispersion divided by the cluster size. In this case, there is no guaranty that the heights are non decreasing. When reversals are detected, a warning is printed to advice the user to change the mode of the representation.

Grimm (1987) indicates that heights as provided by mode = "within-disp" are highly dependent on cluster sizes and that the most advisable representation is the one provided by mode = "total-disp". Further details are provided in the vignette "Notes on CHAC implementation in adjclust".

Value

The function plot.chac displays the dendrogram and additionally invisibly returns an object of class dendrogram with heights as specified by the user through the option mode.

diagnose invisibly exports a data frame with the numbers of decreasing merges described by the labels of the clusters being merged at this step and at the previous one, as well as the corresponding merge heights.

The function correct returns a chac objects with modified heights so as they are increasing. The new heights are calculated in an way identical to the option mode = "corrected" of the function plot.chac (see Details). In addition, the chac object has its field method modified from adjClust to adjClust-modified.

The function cutree_chac returns the clustering with k groups or with the groups obtained by cutting the tree at height h. If the heights are not increasing, the cutting of the tree is based on the corrected heights as provided by the function correct.

References

Grimm, E.C. (1987) CONISS: a fortran 77 program for stratigraphically constrained analysis by the method of incremental sum of squares. *Computer & Geosciences*, **13**(1), 13-35.

---

hicClust *Adjacency-constrained Clustering of Hi-C contact maps*

Description

Adjacency-constrained hierarchical agglomerative clustering of Hi-C contact maps

Usage

```
hicClust(x, h = NULL, log = FALSE, ...)
```

Arguments

x	either: 1. A pxp contact sparse or dense matrix (classes matrix, Matrix, dscMatrix, dgTMatrix, dgCMatrix, dgeMatrix). Its entries are the number of counts of physical interactions observed between all pairs of loci. 2. An object of class HiTC::HTCexp. The corresponding Hi-C data is stored as a Matrix::dsCMatrix object in the intdata slot. 3. A text file path with one line per pair of loci for which an interaction has been observed (in the format: locus1<tab>locus2<tab>signal) or a matrix or data frame with similar data (3 columns).
h	band width. If not provided, h is set to default value 'p-1'.
log	logical. Whether to log-transform the count data. Default to FALSE.
...	further arguments to be passed to read.table function when x is a text file name. If not provided, the text file is supposed to be separated by tabulations, with no header.

Details

Adjacency-constrained hierarchical agglomerative clustering (HAC) is HAC in which each observation is associated to a position, and the clustering is constrained so as only adjacent clusters are merged. Genomic regions (loci) are clustered according to information provided by high-throughput conformation capture data (Hi-C).

Value

An object of class chac.

References

Ambroise C., Dehman A., Neuvial P., Rigaill G., and Vialaneix N (2019). *Adjacency-constrained hierarchical clustering of a band similarity matrix with application to genomics*, Algorithms for Molecular Biology 14(22)"

Servant N. *et al* (2012). *HiTC : Exploration of High-Throughput 'C' experiments. Bioinformatics*.

See Also

adjClust

Examples

```
# input as HiTC::HTCexp object
## Not run:
if (require("HiTC", quietly = TRUE)) {
  load(system.file("extdata", "hic_imr90_40_XX.rda", package = "adjclust"))
  res1 <- hicClust(hic_imr90_40_XX)
}
```

```
## End(Not run)

# input as Matrix::dsCMatrix contact map
## Not run:
mat <- HiTC::intdata(hic_imr90_40_XX)
res2 <- hicClust(mat)

## End(Not run)

# input as text file
res3 <- hicClust(system.file("extdata", "sample.txt", package = "adjclust"))
```

---

plotSim *Plot (dis)similarity matrix*

Description

Heatmap of the (dis)similarity matrix

Usage

```
plotSim(
  mat,
  type = c("similarity", "dissimilarity"),
  clustering = NULL,
  dendro = NULL,
  k = NULL,
  log = TRUE,
  legendName = "intensity",
  main = NULL,
  priorCount = 0.5,
  stats = c("R.squared", "D.prime"),
  h = NULL,
  axis = FALSE,
  naxis = min(10, nrow(mat)),
  axistext = NULL,
  xlab = "objects",
  cluster_col = "darkred",
  mode = c("standard", "corrected", "total-disp", "within-disp", "average-disp")
)
```

Arguments

mat	matrix to plot. It can be of class 'matrix', 'dgCMatrix', 'dsCMatrix', 'dist', 'HTCexp', 'snpMatrix'.
type	input matrix type. Can be either "similarity" or "dissimilarity" (kernels are supposed to be of type "similarity").

clustering	vector of clusters to display on the matrix (if not NULL). If clustering is provided, it must be a numeric vector of length identical to the matrix size with clusters identified as consecutive integers 1, 2, 3, ...
dendro	dendrogram provided as an hclust object, or as another type of object that can be converted to hclust with as.hclust.
k	number of clusters to display. Used only when dendro is not NULL and clustering is NULL. The clustering is then deduced from the dendrogram by a standard cut.
log	logical. Should the breaks be based on log-scaled values of the matrix entries. Default to TRUE.
legendName	character. Title of the legend. Default to "intensity".
main	character. Title of the plot. Default to NULL (no title).
priorCount	numeric. Average count to be added to each entry of the matrix to avoid taking log of zero. Used only if log == TRUE and default to 0.5.
stats	input SNP correlation type. Used when mat is of type SnpMatrix.
h	positive integer. Threshold distance for SNP correlation computation. Used when mat is of type SnpMatrix.
axis	logical. Should x-axis be displayed on the plot? Default to FALSE.
naxis	integer. If axis == TRUE, number of ticks to display on the x-axis.
axistext	character vector. If axis == TRUE, labels to display of the x-axis (its length has to be equal to naxis).
xlab	character. If axis == TRUE, x-axis title.
cluster_col	colour for the cluster line if clustering is not NULL.
mode	type of dendrogram to plot (see plot.chac). Default to "standard".

Details

This function produces a heatmap for the used (dis)similarity matrix that can be used as a diagnostic plot to check the consistency between the obtained clustering and the original (dis)similarity

See Also

select, adjClust

Examples

```
## Not run:
clustering <- rep(1:3, each = 50)
dist_data <- as.matrix(dist(iris[, 1:4]))
dendro_iris <- adjClust(dist_data, type = "dissimilarity")
plotSim(dist_data, type = "dissimilarity", dendro = dendro_iris, axis = TRUE)
plotSim(dist_data, type = "dissimilarity", dendro = dendro_iris,
        clustering = clustering)
plotSim(dist_data, type = "dissimilarity", dendro = dendro_iris, axis = TRUE,
        k = 3)
plotSim(dist_data, type = "dissimilarity", legendName = "IF", axis = TRUE,
        clustering = clustering)
```

```
p <- plotSim(dist(iris[, 1:4]), type = "dissimilarity", log = FALSE,
             clustering = clustering, cluster_col = "blue")
# custom palette
p + scale_fill_gradient(low = "yellow", high = "red")
# dsCMatrix
m <- Matrix(c(0, 0, 2, 0, 3, 0, 2, 0, 0), ncol = 3)
res <- adjClust(m)
plotSim(m, axis = TRUE)
plotSim(m, dendro = res)
# dgCMatrix
m <- as(m, "generalMatrix")
plotSim(m)
m <- as.dist(m)
if (require("HiTC", quietly = TRUE)) {
  load(system.file("extdata", "hic_imr90_40_XX.rda", package = "adjclust"))
  res <- hicClust(hic_imr90_40_XX, log = TRUE)
  plotSim(hic_imr90_40_XX, axis = TRUE)
}
if (requireNamespace("snpStats", quietly = TRUE)) {
  data(testdata, package = "snpStats")
  plotSim(Autosomes[1:200, 1:5], h = 3, stats = "R.squared", axis = TRUE,
          axistext = c("A", "B", "C", "D", "E"))
}

## End(Not run)
```

`select` *Clustering selection*

Description

Clustering selection from a chac object with the slope heuristic or the broken stick heuristic

Usage

```
select(
  x,
  type = c("capushe", "bstick"),
  k.max = NULL,
  graph = FALSE,
  pct = 0.15
)
```

Arguments

`x`	an object of class 'chac'
`type`	model selection approach between slope heuristic (`"capushe"`) and broken stick approach (`"bstick"`)

k.max	maximum number of clusters that can be selected. Default to NULL, in which case it is set to $\min(\max(100, \frac{n}{\log(n)}), \frac{n}{2})$ where n is the number of objects to be clustered for capushe and to n for the broken stick model
graph	logical. Whether the diagnostic plot for the capushe selection is displayed or not. Default to FALSE
pct	minimum percentage of points for the plateau selection in capushe selection. See DDSE for further details

Value

The function returns the clustering selected by the slope heuristic, as implemented in the R package capushe.

References

Baudry J.P., Maugis C., and Michel B. (2012). Slope heuristics: overview and implementation. *Statistics and Computing*, **22**(2), 355-470. MacArthur, R.H. (1957) On the relative abundance of bird species. *Proceedings of the National Academy of Sciences*, **43**, 293-295.

Examples

```
## Not run: if (require("HiTC", quietly = TRUE)) {
  load(system.file("extdata", "hic_imr90_40_XX.rda", package = "adjclust"))
  res <- hicClust(hic_imr90_40_XX, log = TRUE)
  selected.capushe <- select(res)
  table(selected.capushe)
  selected.bs <- select(res, type = "bstick")
  table(selected.bs)
}
## End(Not run)

res <- adjClust(dist(iris[ ,1:4]))
select.clust <- select(res, "bs")
table(select.clust)
```

---

snpClust — *Adjacency-constrained Clustering of Single Nucleotide Polymorphisms*

Description

Adjacency-constrained hierarchical agglomerative clustering of Single Nucleotide Polymorphisms based on Linkage Disequilibrium

Usage

```
snpClust(x, h = ncol(x) - 1, stats = c("R.squared", "D.prime"))
```

Arguments

x	either a genotype matrix of class SnpMatrix/matrix or a linkage disequilibrium matrix of class dgCMatrix. In the latter case the LD values are expected to be in [0,1]
h	band width. If not provided, h is set to default value ‘p-1‘ where ‘p‘ is the number of columns of x
stats	a character vector specifying the linkage disequilibrium measures to be calculated (using the ld function) when x is a genotype matrix. Only "R.squared" and "D.prime" are allowed, see Details.

Details

Adjacency-constrained hierarchical agglomerative clustering (HAC) is HAC in which each observation is associated to a position, and the clustering is constrained so as only adjacent clusters are merged. SNPs are clustered based on their similarity as measured by the linkage disequilibrium.

In the special case where genotypes are given as input and the corresponding LD matrix has missing entries, the clustering cannot be performed. This can typically happen when there is insufficient variability in the sample genotypes. In this special case, the indices of the SNP pairs which yield missing values are returned.

If x is of class SnpMatrix or matrix, it is assumed to be a $n \times p$ matrix of p genotypes for n individuals. This input is converted to a LD similarity matrix using the snpStats::ld. If x is of class dgCMatrix, it is assumed to be a (squared) LD matrix.

Clustering on a LD similarity other than "R.squared" or "D.prime" can be performed by providing the LD values directly as argument x. These values are expected to be in [0,1], otherwise they are truncated to [0,1].

Value

An object of class chac (when no LD value is missing)

References

Dehman A. (2015) *Spatial Clustering of Linkage Disequilibrium Blocks for Genome-Wide Association Studies*, PhD thesis, Universite Paris Saclay.

Dehman, A. Ambroise, C. and Neuvial, P. (2015). Performance of a blockwise approach in variable selection using linkage disequilibrium information. *BMC Bioinformatics* 16:148.

Ambroise C., Dehman A., Neuvial P., Rigaill G., and Vialaneix N (2019). *Adjacency-constrained hierarchical clustering of a band similarity matrix with application to genomics*, Algorithms for Molecular Biology 14(22)"

See Also

adjClust ld

Examples

```
## a very small example
if (requireNamespace("snpStats", quietly = TRUE)) {
  data(testdata, package = "snpStats")

  # input as snpStats::SnpMatrix
  fit1 <- snpClust(Autosomes[1:200, 1:5], h = 3, stats = "R.squared")

  # input as base::matrix
  fit2 <- snpClust(as.matrix(Autosomes[1:200, 1:5]), h = 3, stats = "R.squared")

  # input as Matrix::dgCMatrix
 ldres <- snpStats::ld(Autosomes[1:200, 1:5], depth = 3, stats = "R.squared", symmetric = TRUE)
  fit3 <- snpClust(ldres, 3)
}
```

Index