Package: apcluster (via r-universe)

October 23, 2024

Type Package

Title Affinity Propagation Clustering

Version 1.4.13

Date 2024-04-26

Depends R (>= 3.3.0)

Imports Rcpp (>= 0.11.1), methods, Matrix, stats, graphics, grDevices

Suggests knitr

Author Ulrich Bodenhofer [aut, cre], Johannes Palme [ctb], Chrats Melkonian [ctb], Andreas Kothmeier [aut], Nikola Kostic [ctb]

Maintainer Ulrich Bodenhofer <ulrich@bodenhofer.com>

Description Implements Affinity Propagation clustering introduced by Frey and Dueck (2007) <DOI:10.1126/science.1136800>. The algorithms are largely analogous to the 'Matlab' code published by Frey and Dueck. The package further provides leveraged affinity propagation and an algorithm for exemplar-based agglomerative clustering that can also be used to join clusters obtained from affinity propagation. Various plotting functions are available for analyzing clustering results.

License GPL (>= 2)

Collate AllClasses.R AllGenerics.R access-methods.R coerce-methods.R show-methods.R labels-methods.R length-methods.R revDend.R heatmap-methods.R plot-methods.R cutree-methods.R sort-methods.R aggExCluster-methods.R apcluster-methods.R apclusterL-methods.R apclusterK-methods.R apclusterDemo.R preferenceRange-methods.R similarity.R simpleDist.R conversions.R

URL https://github.com/UBod/apcluster

VignetteBuilder knitr

LinkingTo Rcpp

LazyLoad yes

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

RemoteUrl https://github.com/ubod/apcluster

RemoteRef HEAD

RemoteSha 096bede013c45322da2632cabf6e0ab4da315aac

Contents

`apcluster-package` *APCluster Package*

Description

The apcluster package implements affinity propagation according to Frey and Dueck and a method for exemplar-based agglomerative clustering. It further offers various functions for plotting clustering results.

Details

The central function is `apcluster`. It runs affinity propagation on a given similarity matrix or it creates a similarity matrix for a given data set and similarity measure and runs affinity propagation on this matrix. The function returns an `APResult` object from which the clustering itself and information about the affinity propagation run can be obtained. Leveraged affinity propagation clustering `apclusterL` allows efficient clustering of large datasets by using only a subset of the similarities. The package further implements an exemplar-based agglomerative clustering method `aggExCluster` that can be used for computing a complete cluster hierarchy, but also for joining fine-grained clusters previously obtained by affinity propagation clustering. Further functions are implemented to visualize the results and to create distance matrices.

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, Johannes Palme, Chrats Melkonian, and Nikola Kostic

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

Frey, B. J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science* **315**, 972-976. DOI: doi:10.1126/science.1136800.

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(100, 0.2, 0.05), rnorm(100, 0.8, 0.06))
cl2 <- cbind(rnorm(100, 0.7, 0.08), rnorm(100, 0.3, 0.05))
x <- rbind(cl1, cl2)

## compute similarity matrix (negative squared Euclidean)
sim <- negDistMat(x, r=2)

## run affinity propagation
apres <- apcluster(sim, details=TRUE)

## show details of clustering results
show(apres)

## plot information about clustering run
plot(apres)

## plot clustering result
plot(apres, x)

## employ agglomerative clustering to join clusters
aggres <- aggExCluster(sim, apres)

## show information
show(aggres)
show(cutree(aggres, 2))

## plot dendrogram
plot(aggres)

## plot clustering result for k=2 clusters
plot(aggres, x, k=2)

## plot heatmap
heatmap(apres, sim)

## leveraged apcluster
apresL <- apclusterL(s=negDistMat(r=2), x=x,  frac=0.2, sweeps=3)
```

```
## show details of clustering results
show(apresL)

## plot clustering result
plot(apresL, x)
```

`aggExCluster` *Exemplar-based Agglomerative Clustering*

Description

Runs exemplar-based agglomerative clustering

Usage

```
## S4 method for signature 'matrix,missing'
aggExCluster(s, x, includeSim=FALSE)
## S4 method for signature 'matrix,ExClust'
aggExCluster(s, x, includeSim=FALSE)
## S4 method for signature 'Matrix,missing'
aggExCluster(s, x, includeSim=FALSE)
## S4 method for signature 'Matrix,ExClust'
aggExCluster(s, x, includeSim=FALSE)
## S4 method for signature 'missing,ExClust'
aggExCluster(s, x, includeSim=TRUE)
## S4 method for signature 'function,ANY'
aggExCluster(s, x, includeSim=TRUE, ...)
## S4 method for signature 'character,ANY'
aggExCluster(s, x, includeSim=TRUE, ...)
```

Arguments

`s`	an $l \times l$ similarity matrix or a similarity function either specified as the name of a package-provided similarity function as character string or a user provided function object
`x`	either a prior clustering of class `ExClust` (or `APResult`) or, if called with `s` being a function or function name, input data to be clustered (see `apcluster` for a detailed specification)
`includeSim`	if `TRUE`, the similarity matrix (either computed internally or passed via the `s` argument) is stored to the slot `sim` of the returned `AggExResult` object. The default is `FALSE` if `aggExCluster` has been called for a similarity matrix, otherwise the default is `TRUE`.
`...`	all other arguments are passed to the selected similarity function as they are.

Details

aggExCluster performs agglomerative clustering. Unlike other methods, e.g., the ones implemented in hclust, aggExCluster is computing exemplars for each cluster and its merging objective is geared towards the identification of meaningful exemplars, too.

For each pair of clusters, the merging objective is computed as follows:

1. An intermediate cluster is created as the union of the two clusters.
2. The potential exemplar is selected from the intermediate cluster as the sample that has the largest average similarity to all other samples in the intermediate cluster.
3. Then the average similarity of the exemplar with all samples in the first cluster and the average similarity with all samples in the second cluster is computed. These two values measure how well the joint exemplar describes the samples in the two clusters.
4. The merging objective is finally computed as the average of the two measures above. Hence, we can consider the merging objective as some kind of "balanced average similarity to the joint exemplar".

In each step, all pairs of clusters are considered and the pair with the largest merging objective is actually merged. The joint exemplar is then chosen as the exemplar of the merged cluster.

aggExCluster can be used in two ways, either by performing agglomerative clustering of an entire data set or by performing agglomerative clustering of data previously clustered by affinity propagation or another clustering algorithm.

1. Agglomerative clustering of an entire data set can be accomplished either by calling aggExCluster on a quadratic similarity matrix without further argument or by calling aggExCluster for a function or function name along with data to be clustered (as argument x). A full agglomeration run is performed that starts from 1 clusters (all samples in separate one-element clusters) and ends with one cluster (all samples in one single cluster).
2. Agglomerative clustering starting from a given clustering result can be accomplished by calling aggExCluster for an APResult or ExClust object passed as parameter x. The similarity matrix can either be passed as argument s or, if missing, aggExCluster looks if the similarity matrix is included in the clustering object x. A cluster hierarchy with numbers of clusters ranging from the number of clusters in x down to 1 is created.

The result is stored in an AggExResult object. The slot height is filled with the merging objective of each of the maxNoClusters-1 merges. The slot order contains a permutation of the samples/clusters for dendrogram plotting. The algorithm for computing this permutation is the same as the one used in hclust. If aggExCluster was called for an entire data set, the slot label contains the names of the objects to be clustered (if available, otherwise the indices are used). If aggExCluster was called for a prior clustering, then labels are set to 'Cluster 1', 'Cluster 2', etc.

Value

Upon successful completion, the function returns an AggExResult object.

Note

Similarity matrices can be supplied in dense or sparse format. Note, however, that sparse matrices are converted to full dense matrices before clustering which may lead to memory and/or performance bottlenecks for larger data sets.

Author(s)

Ulrich Bodenhofer, Johannes Palme, and Nikola Kostic

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

AggExResult, apcluster-methods, plot-methods, heatmap-methods, cutree-methods

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(50, 0.2, 0.05), rnorm(50, 0.8, 0.06))
cl2 <- cbind(rnorm(50, 0.7, 0.08), rnorm(50, 0.3, 0.05))
x <- rbind(cl1, cl2)

## compute agglomerative clustering from scratch
aggres1 <- aggExCluster(negDistMat(r=2), x)

## show results
show(aggres1)

## plot dendrogram
plot(aggres1)

## plot heatmap along with dendrogram
heatmap(aggres1)

## plot level with two clusters
plot(aggres1, x, k=2)

## run affinity propagation
apres <- apcluster(negDistMat(r=2), x, q=0.7)

## create hierarchy of clusters determined by affinity propagation
aggres2 <- aggExCluster(x=apres)

## show results
show(aggres2)

## plot dendrogram
plot(aggres2)
plot(aggres2, showSamples=TRUE)

## plot heatmap
heatmap(aggres2)
```

```
## plot level with two clusters
plot(aggres2, x, k=2)
```

AggExResult-class *Class "AggExResult"*

Description

S4 class for storing results of exemplar-based agglomerative clustering

Objects

Objects of this class can be created by calling aggExCluster for a given similarity matrix.

Slots

The following slots are defined for AggExResult objects:

l: number of samples in the data set

sel: subset of samples used for leveraged clustering (empty for normal clustering)

maxNoClusters: maximum number of clusters in the cluster hierarchy, i.e. it contains clusterings with 1 - maxNoClusters clusters.

exemplars: list of length maxNoClusters; the i-th component of the list is a vector of i exemplars (corresponding to the level with i clusters).

clusters: list of length maxNoClusters; the i-th component of clusters is a list of i clusters, each of which is a vector of sample indices.

merge: a maxNoClusters-1 by 2 matrix that contains the merging hierarchy; fully analogous to the slot merge in the class hclust.

height: a vector of length maxNoClusters-1 that contains the merging objective of each merge; largely analogous to the slot height in the class hclust except that the slot height in AggExResult objects is supposed to be non-increasing, since aggExCluster is based on similarities, whereas hclust uses dissimilarities.

order: a vector containing a permutation of indices that can be used for plotting proper dendrograms without crossing branches; fully analogous to the slot order in the class hclust.

labels: a character vector containing labels of clustered objects used for plotting dendrograms.

sim: similarity matrix; only available if aggExCluster was called with similarity function and includeSim=TRUE.

call: method call used to produce this clustering result

Methods

plot `signature(x="AggExResult")`: see `plot-methods`

plot `signature(x="AggExResult", y="matrix")`: see `plot-methods`

heatmap `signature(x="AggExResult")`: see `heatmap-methods`

heatmap `signature(x="AggExResult", y="matrix")`: see `heatmap-methods`

show `signature(object="AggExResult")`: see `show-methods`

cutree `signature(object="AggExResult", k="ANY",h="ANY")`: see `cutree-methods`

length `signature(x="AggExResult")`: gives the number of clustering levels in the clustering result.

as.hclust `signature(x="AggExResult")`: see `coerce-methods`

as.dendrogram `signature(object="AggExResult")`: see `coerce-methods`

Accessors

In the following code snippets, `x` is an `AggExResult` object.

[[`signature(x="AggExResult", i="index", j="missing")`: `x[[i]]` returns an object of class `ExClust` corresponding to the clustering level with `i` clusters; synonymous to `cutree(x, i)`.

[`signature(x="AggExResult", i="index", j="missing", drop="missing")`: `x[i]` returns a list of `ExClust` objects with all clustering levels specified in vector `i`. So, the list has as many components as the argument `i` has elements. A list is returned even if `i` is a single level.

similarity `signature(x="AggExResult")`: gives the similarity matrix.

Author(s)

Ulrich Bodenhofer, Johannes Palme, and Johannes Palme

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

`aggExCluster`, `show-methods`, `plot-methods`, `cutree-methods`

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(50, 0.2, 0.05), rnorm(50, 0.8, 0.06))
cl2 <- cbind(rnorm(50, 0.7, 0.08), rnorm(50, 0.3, 0.05))
x <- rbind(cl1, cl2)

## compute similarity matrix (negative squared Euclidean)
sim <- negDistMat(x, r=2)
```

```
## compute agglomerative clustering from scratch
aggres1 <- aggExCluster(sim)

## show results
show(aggres1)

## plot dendrogram
plot(aggres1)

## plot heatmap along with dendrogram
heatmap(aggres1, sim)

## plot level with two clusters
plot(aggres1, x, k=2)

## run affinity propagation
apres <- apcluster(sim, q=0.7)

## create hierarchy of clusters determined by affinity propagation
aggres2 <- aggExCluster(sim, apres)

## show results
show(aggres2)

## plot dendrogram
plot(aggres2)

## plot heatmap
heatmap(aggres2, sim)

## plot level with two clusters
plot(aggres2, x, k=2)
```

---

apcluster *Affinity Propagation*

Description

Runs affinity propagation clustering

Usage

```
## S4 method for signature 'matrix,missing'
apcluster(s, x, p=NA, q=NA, maxits=1000,
    convits=100, lam=0.9, includeSim=FALSE, details=FALSE,
    nonoise=FALSE, seed=NA)
## S4 method for signature 'dgTMatrix,missing'
apcluster(s, x, p=NA, q=NA, maxits=1000,
    convits=100, lam=0.9, includeSim=FALSE, details=FALSE,
    nonoise=FALSE, seed=NA)
```

```
## S4 method for signature 'sparseMatrix,missing'
apcluster(s, x, ...)
## S4 method for signature 'Matrix,missing'
apcluster(s, x, ...)
## S4 method for signature 'character,ANY'
apcluster(s, x, p=NA, q=NA, maxits=1000,
    convits=100, lam=0.9, includeSim=TRUE, details=FALSE,
    nonoise=FALSE, seed=NA, ...)
## S4 method for signature 'function,ANY'
apcluster(s, x, p=NA, q=NA, maxits=1000,
    convits=100, lam=0.9, includeSim=TRUE, details=FALSE,
    nonoise=FALSE, seed=NA, ...)
```

Arguments

s	an $l \times l$ similarity matrix or a similarity function either specified as the name of a package-provided similarity function as character string or a user provided function object. s may also be a sparse matrix according to the **Matrix** package. Internally, apcluster uses the dgTMatrix class; all other sparse matrices are cast to this class (if possible, otherwise the function quits with an error). If s is any other object of class Matrix, s is cast to a regular matrix internally (if possible, otherwise the function quits with an error).
x	input data to be clustered; if x is a matrix or data frame, rows are interpreted as samples and columns are interpreted as features; apart from matrices or data frames, x may be any other structured data type that contains multiple data items - provided that an appropriate length function is available that returns the number of items
p	input preference; can be a vector that specifies individual preferences for each data point. If scalar, the same value is used for all data points. If NA, exemplar preferences are initialized according to the distribution of non-Inf values in s. How this is done is controlled by the parameter q.
q	if p=NA, exemplar preferences are initialized according to the distribution of non-Inf values in s. If q=NA, exemplar preferences are set to the median of non-Inf values in s. If q is a value between 0 and 1, the sample quantile with threshold q is used, whereas q=0.5 again results in the median.
maxits	maximal number of iterations that should be executed
convits	the algorithm terminates if the examplars have not changed for convits iterations
lam	damping factor; should be a value in the range [0.5, 1); higher values correspond to heavy damping which may be needed if oscillations occur
includeSim	if TRUE, the similarity matrix (either computed internally or passed via the s argument) is stored to the slot sim of the returned APResult object. The default is FALSE if apcluster has been called for a similarity matrix, otherwise the default is TRUE.
details	if TRUE, more detailed information about the algorithm's progress is stored in the output object (see APResult)

nonoise	apcluster adds a small amount of noise to s to prevent degenerate cases; if TRUE, this is disabled
seed	for reproducibility, the seed of the random number generator can be set to a fixed value before adding noise (see above), if NA, the seed remains unchanged
...	for the methods with signatures character,ANY and function,ANY, all other arguments are passed to the selected similarity function as they are; for the methods with signatures Matrix,missing and sparseMatrix,missing, further arguments are passed on to the apcluster methods with signatures Matrix,missing and dgTMatrix,missing, respectively.

Details

Affinity Propagation clusters data using a set of real-valued pairwise data point similarities as input. Each cluster is represented by a cluster center data point (the so-called exemplar). The method is iterative and searches for clusters maximizing an objective function called net similarity.

When called with a similarity matrix as input (which may also be a sparse matrix according to the **Matrix** package), the function performs AP clustering. When called with the name of a package-provided similarity function or a user-provided similarity function object and input data, the function first computes the similarity matrix before performing AP clustering. The similarity matrix is returned for later use as part of the APResult object depending on whether includeSim was set to TRUE (see argument description above).

Apart from minor adaptations and optimizations, the AP clustering functionality of the function apcluster is largely analogous to Frey's and Dueck's Matlab code (see https://psi.toronto.edu/research/affinity-propagation-clustering-by-message-passing/).

The new argument q allows for better controlling the number of clusters without knowing the distribution of similarity values. A meaningful range for the parameter p can be determined using the function preferenceRange. Alternatively, a certain fixed number of clusters may be desirable. For this purpose, the function apclusterK is available.

Value

Upon successful completion, the function returns an APResult object.

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, Johannes Palme, and Chrats Melkonian

References

https://github.com/UBod/apcluster

Frey, B. J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science* **315**, 972-976. DOI: doi:10.1126/science.1136800.

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

APResult, show-methods, plot-methods, labels-methods, preferenceRange, apclusterL-methods, apclusterK

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(100, 0.2, 0.05), rnorm(100, 0.8, 0.06))
cl2 <- cbind(rnorm(50, 0.7, 0.08), rnorm(50, 0.3, 0.05))
x <- rbind(cl1, cl2)

## compute similarity matrix and run affinity propagation
## (p defaults to median of similarity)
apres <- apcluster(negDistMat(r=2), x, details=TRUE)

## show details of clustering results
show(apres)

## plot clustering result
plot(apres, x)

## plot heatmap
heatmap(apres)

## run affinity propagation with default preference of 10% quantile
## of similarities; this should lead to a smaller number of clusters
## reuse similarity matrix from previous run
apres <- apcluster(s=apres@sim, q=0.1)
show(apres)
plot(apres, x)

## now try the same with RBF kernel
sim <- expSimMat(x, r=2)
apres <- apcluster(s=sim, q=0.2)
show(apres)
plot(apres, x)

## create sparse similarity matrix
cl1 <- cbind(rnorm(20, 0.2, 0.05), rnorm(20, 0.8, 0.06))
cl2 <- cbind(rnorm(20, 0.7, 0.08), rnorm(20, 0.3, 0.05))
x <- rbind(cl1, cl2)

sim <- negDistMat(x, r=2)
ssim <- as.SparseSimilarityMatrix(sim, lower=-0.2)

## run apcluster() on the sparse similarity matrix
apres <- apcluster(ssim, q=0)
apres
```

apclusterDemo *Affinity Propagation Demo*

Description

Runs affinity propagation demo for randomly generated data set according to Frey and Dueck

Usage

```
apclusterDemo(l=100, d=2, seed=NA, ...)
```

Arguments

l	number of data points to be generated
d	dimension of data to be created
seed	for reproducibility, the seed of the random number generator can be set to a fixed value; if NA, the seed remains unchanged
...	all other arguments are passed on to apcluster

Details

apclusterDemo creates l d-dimensional data points that are uniformly distributed in $[0, 1]^d$. Affinity propagation is executed for this data set with default parameters. Alternative settings can be passed to apcluster with additional arguments. After completion of affinity propagation, the results are shown and the performance measures are plotted.

This function corresponds to the demo function in the original Matlab code of Frey and Dueck. We warn the user, however, that uniformly distributed data are not necessarily ideal for demonstrating clustering, as there can never be real clusters in uniformly distributed data - all clusters found must be random artefacts.

Value

Upon successful completion, the function returns an invisible list with three components. The first is the data set that has been created, the second is the similarity matrix, and the third is an APResult object with the clustering results (see examples below).

Author(s)

Ulrich Bodenhofer, Johannes Palme, and Johannes Palme

References

https://github.com/UBod/apcluster

Frey, B. J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science* **315**, 972-976. DOI: doi:10.1126/science.1136800.

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

APResult, plot-methods, apcluster, apclusterL

Examples

```
## create random data set and run affinity propagation
apd <- apclusterDemo()

## plot clustering result along with data set
plot(apd[[3]], apd[[1]])
```

---

apclusterK *Affinity Propagation for Pre-defined Number of Clusters*

Description

Runs affinity propagation clustering for a given similarity matrix adjusting input preferences iteratively in order to achieve a desired number of clusters

Usage

```
## S4 method for signature 'matrix,missing'
apclusterK(s, x, K, prc=10, bimaxit=20, exact=FALSE,
     maxits=1000, convits=100, lam=0.9, includeSim=FALSE, details=FALSE,
     nonoise=FALSE, seed=NA, verbose=TRUE)
## S4 method for signature 'Matrix,missing'
apclusterK(s, x, K, ...)
## S4 method for signature 'dgTMatrix,missing'
apclusterK(s, x, K, prc=10, bimaxit=20,
     exact=FALSE, maxits=1000, convits=100, lam=0.9, includeSim=FALSE,
     details=FALSE, nonoise=FALSE, seed=NA, verbose=TRUE)
## S4 method for signature 'sparseMatrix,missing'
apclusterK(s, x, K, ...)
## S4 method for signature 'function,ANY'
apclusterK(s, x, K, prc=10, bimaxit=20, exact=FALSE,
     maxits=1000, convits=100, lam=0.9, includeSim=TRUE, details=FALSE,
     nonoise=FALSE, seed=NA, verbose=TRUE, ...)
## S4 method for signature 'character,ANY'
apclusterK(s, x, K, prc=10, bimaxit=20, exact=FALSE,
     maxits=1000, convits=100, lam=0.9, includeSim=TRUE, details=FALSE,
     nonoise=FALSE, seed=NA, verbose=TRUE, ...)
```

Arguments

s	an $l \times l$ similarity matrix in sparse or dense format or a similarity function either specified as the name of a package-provided similarity function as character string or a user provided function object.

x	input data to be clustered; if x is a matrix or data frame, rows are interpreted as samples and columns are interpreted as features; apart from matrices or data frames, x may be any other structured data type that contains multiple data items - provided that an appropriate length function is available that returns the number of items
K	desired number of clusters
prc	the algorithm stops if the number of clusters does not deviate more than prc percent from desired value K; set to 0 if you want to have exactly K clusters
bimaxit	maximum number of bisection steps to perform; note that no warning is issued if the number of clusters is still not in the desired range
exact	flag indicating whether or not to compute the initial preference range exactly (see preferenceRange)
maxits	maximal number of iterations that apcluster should execute
convits	apcluster terminates if the examplars have not changed for convits iterations
lam	damping factor for apcluster; should be a value in the range [0.5, 1); higher values correspond to heavy damping which may be needed if oscillations occur
includeSim	if TRUE, the similarity matrix (either computed internally or passed via the s argument) is stored to the slot sim of the returned APResult object. The default is FALSE if apclusterK has been called for a similarity matrix, otherwise the default is TRUE.
details	if TRUE, more detailed information about the algorithm's progress is stored in the output object (see APResult)
nonoise	apcluster adds a small amount of noise to s to prevent degenerate cases; if TRUE, this is disabled
seed	for reproducibility, the seed of the random number generator can be set to a fixed value, if NA, the seed remains unchanged
verbose	flag indicating whether status information should be displayed during bisection
...	for the methods with signatures character,ANY and function,ANY, all other arguments are passed to the selected similarity function as they are; for the methods with signatures Matrix,missing and sparseMatrix,missing, further arguments are passed on to the apclusterK methods with signatures Matrix,missing and dgTMatrix,missing, respectively.

Details

apclusterK first runs preferenceRange to determine the range of meaningful choices of the input preference p. Then it decreases p exponentially for a few iterations to obtain a good initial guess for p. If the number of clusters is still too far from the desired goal, bisection is applied.

When called with a similarity matrix as input, the function performs the procedure described above. When called with the name of a package-provided similarity function or a user-provided similarity function object and input data, the function first computes the similarity matrix before running apclusterK on this similarity matrix. The similarity matrix is returned for later use as part of the APResult object depending on whether includeSim was set to TRUE (see argument description above).

Apart from minor adaptations and optimizations, the implementation is largely analogous to Frey's and Dueck's Matlab code (see https://psi.toronto.edu/research/affinity-propagation-clustering-by-message

Value

Upon successful completion, the function returns a APResult object.

Author(s)

Ulrich Bodenhofer and Andreas Kothmeier

References

https://github.com/UBod/apcluster

Frey, B. J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science* **315**, 972-976. DOI: doi:10.1126/science.1136800.

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

apcluster, preferenceRange, APResult

Examples

```
## create three Gaussian clouds
cl1 <- cbind(rnorm(70, 0.2, 0.05), rnorm(70, 0.8, 0.06))
cl2 <- cbind(rnorm(50, 0.7, 0.08), rnorm(50, 0.3, 0.05))
cl3 <- cbind(rnorm(60, 0.8, 0.04), rnorm(60, 0.8, 0.05))
x <- rbind(cl1, cl2, cl3)

## run affinity propagation such that 3 clusters are obtained
apres <- apclusterK(negDistMat(r=2), x, K=3)

## show details of clustering results
show(apres)

## plot clustering result
plot(apres, x)

## create sparse similarity matrix
cl1 <- cbind(rnorm(20, 0.2, 0.05), rnorm(20, 0.8, 0.06))
cl2 <- cbind(rnorm(20, 0.7, 0.08), rnorm(20, 0.3, 0.05))
x <- rbind(cl1, cl2)

sim <- negDistMat(x, r=2)
ssim <- as.SparseSimilarityMatrix(sim, lower=-0.2)

## run apcluster() on the sparse similarity matrix
apres <- apclusterK(ssim, K=2)
apres
```

apclusterL *Leveraged Affinity Propagation*

Description

Runs leveraged affinity propagation clustering

Usage

```
## S4 method for signature 'matrix,missing'
apclusterL(s, x,
          sel, p=NA, q=NA, maxits=1000, convits=100, lam=0.9,
          includeSim=FALSE, nonoise=FALSE, seed=NA)
## S4 method for signature 'character,ANY'
apclusterL(s, x,
          frac, sweeps, p=NA, q=NA, maxits=1000, convits=100, lam=0.9,
          includeSim=TRUE, nonoise=FALSE, seed=NA, ...)
## S4 method for signature 'function,ANY'
apclusterL(s, x,
          frac, sweeps, p=NA, q=NA, maxits=1000, convits=100, lam=0.9,
          includeSim=TRUE, nonoise=FALSE, seed=NA, ...)
```

Arguments

s	an $l \times length(sel)$ similarity matrix or a similarity function either specified as the name of a package provided similarity function as character string or a user provided function object for similarity calculation. If s is supplied as a similarity matrix, the columns must correspond to the same sub-selection of samples as specified in the sel argument and must be in the same increasing order. For a package- or user-defined similarity function, additional parameters can be specified as appropriate for the chosen method and are passed on to the similarity function via the ... argument (see below). See the package vignette for a non-trivial example or supplying a user-defined similarity measure.
x	input data to be clustered; if x is a matrix or data frame, rows are interpreted as samples and columns are interpreted as features; apart from matrices or data frames, x may be any other structured data type that contains multiple data items - provided that an appropriate length function is available that returns the number of items
frac	fraction of samples that should be used for leveraged clustering. The similarity matrix will be generated for all samples against a random fraction of the samples as specified by this parameter.
sweeps	number of sweeps of leveraged clustering performed with changing randomly selected subset of samples.
sel	selected sample indices; a vector containing the sample indices of the sample subset used for leveraged AP clustering in increasing order.

p	input preference; can be a vector that specifies individual preferences for each data point. If scalar, the same value is used for all data points. If NA, exemplar preferences are initialized according to the distribution of non-Inf values in s. How this is done is controlled by the parameter q. See also apcluster.
q	if p=NA, exemplar preferences are initialized according to the distribution of non-Inf values in s. If q=NA, exemplar preferences are set to the median of non-Inf values in s. If q is a value between 0 and 1, the sample quantile with threshold q is used, whereas q=0.5 again results in the median. See also apcluster.
maxits	maximal number of iterations that should be executed
convits	the algorithm terminates if the examplars have not changed for convits iterations
lam	damping factor; should be a value in the range [0.5, 1); higher values correspond to heavy damping which may be needed if oscillations occur
includeSim	if TRUE, the similarity matrix (either computed internally or passed via the s argument) is stored to the slot sim of the returned APResult object. The default is FALSE if apclusterL has been called for a similarity matrix, otherwise the default is TRUE.
nonoise	apcluster adds a small amount of noise to s to prevent degenerate cases; if TRUE, this is disabled
seed	for reproducibility, the seed of the random number generator can be set to a fixed value before adding noise (see above), if NA, the seed remains unchanged
...	all other arguments are passed to the selected similarity function as they are; note that possible name conflicts between arguments of apcluster and arguments of the similarity function may occur; therefore, we recommend to write user-defined similarity functions without additional parameters or to use closures to fix parameters (such as, in the example below);

Details

Affinity Propagation clusters data using a set of real-valued pairwise similarities as input. Each cluster is represented by a representative cluster center (the so-called exemplar). The method is iterative and searches for clusters maximizing an objective function called net similarity.

Leveraged Affinity Propagation reduces dynamic and static load for large datasets. Only a subset of the samples are considered in the clustering process assuming that they provide already enough information about the cluster structure.

When called with input data and the name of a package provided or a user provided similarity function the function selects a random sample subset according to the frac parameter, calculates a rectangular similarity matrix of all samples against this subset and repeats affinity propagation sweep times. A new sample subset is used for each repetition. The clustering result of the sweep with the highest net similarity is returned. Any parameters specific to the chosen method of similarity calculation can be passed to apcluster in addition to the parameters described above. The similarity matrix for the best trial is also returned in the result object when requested by the user (argument includeSim).

When called with a rectangular similarity matrix (which represents a column subset of the full similarity matrix) the function performs AP clustering on this similarity matrix. The information

about the selected samples is passed to clustering with the parameter `sel`. This function is only needed when the user needs full control of distance calculation or sample subset selection.

Apart from minor adaptations and optimizations, the implementation of the function `apclusterL` is largely analogous to Frey's and Dueck's Matlab code (see https://psi.toronto.edu/research/affinity-propagation-clustering-by-message-passing/).

Value

Upon successful completion, both functions returns an `APResult` object.

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, and Johannes Palme

References

https://github.com/UBod/apcluster

Frey, B. J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science* **315**, 972-976. DOI: doi:10.1126/science.1136800.

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

`APResult`, `show-methods`, `plot-methods`, `labels-methods`, `preferenceRange`, `apcluster-methods`, `apclusterK`

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(150, 0.2, 0.05), rnorm(150, 0.8, 0.06))
cl2 <- cbind(rnorm(100, 0.7, 0.08), rnorm(100, 0.3, 0.05))
x <- rbind(cl1, cl2)

## leveraged apcluster
apres <- apclusterL(negDistMat(r=2), x, frac=0.2, sweeps=3, p=-0.2)

## show details of leveraged clustering results
show(apres)

## plot leveraged clustering result
plot(apres, x)

## plot heatmap of clustering result
heatmap(apres)

## show net similarities of single sweeps
apres@netsimLev

## show samples on which best sweep was based
apres@sel
```

APResult-class *Class "APResult"*

Description

S4 class for storing results of affinity propagation clustering. It extends the class ExClust.

Objects

Objects of this class can be created by calling apcluster or apclusterL for a given similarity matrix or calling one of these procedures with a data set and a similarity measure.

Slots

The following slots are defined for APResult objects. Most names are taken from Frey's and Dueck's original Matlab package:

sweeps: number of times leveraged clustering ran with different subsets of samples

it: number of iterations the algorithm ran

p: input preference (either set by user or computed by apcluster or apclusterL)

netsim: final total net similarity, defined as the sum of expref and dpsim (see below)

dpsim: final sum of similarities of data points to exemplars

expref: final sum of preferences of the identified exemplars

netsimLev: total net similarity of the individual sweeps for leveraged clustering; only available for leveraged clustering

netsimAll: vector containing the total net similarity for each iteration; only available if apcluster was called with details=TRUE

exprefAll: vector containing the sum of preferences of the identified exemplars for each iteration; only available if apcluster was called with details=TRUE

dpsimAll: vector containing the sum of similarities of data points to exemplars for each iteration; only available if apcluster was called with details=TRUE

idxAll: matrix with sample-to-exemplar indices for each iteration; only available if apcluster was called with details=TRUE

Extends

Class "ExClust", directly.

Methods

plot signature(x="APResult"): see plot-methods

plot signature(x="ExClust", y="matrix"): see plot-methods

heatmap signature(x="ExClust"): see heatmap-methods

heatmap signature(x="ExClust", y="matrix"): see heatmap-methods

show `signature(object="APResult")`: see show-methods

labels `signature(object="APResult")`: see labels-methods

cutree `signature(object="APResult")`: see cutree-methods

length `signature(x="APResult")`: gives the number of clusters.

sort `signature(x="ExClust")`: see sort-methods

as.hclust `signature(x="ExClust")`: see coerce-methods

as.dendrogram `signature(object="ExClust")`: see coerce-methods

Accessors

In the following code snippets, `x` is an `APResult` object.

[[`signature(x="APResult", i="index", j="missing")`: `x[[i]]` returns the i-th cluster as a list of indices of samples belonging to the i-th cluster.

[`signature(x="APResult", i="index", j="missing", drop="missing")`: `x[i]` returns a list of integer vectors with the indices of samples belonging to this cluster. The list has as many components as the argument `i` has elements. A list is returned even if `i` is a single integer.

similarity `signature(x="APResult")`: gives the similarity matrix.

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, Johannes Palme

References

https://github.com/UBod/apcluster

APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

Frey, B. J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science* **315**, 972-976. DOI: doi:10.1126/science.1136800.

See Also

apcluster, apclusterL, show-methods, plot-methods, labels-methods, cutree-methods

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(100, 0.2, 0.05), rnorm(100, 0.8, 0.06))
cl2 <- cbind(rnorm(50, 0.7, 0.08), rnorm(50, 0.3, 0.05))
x <- rbind(cl1, cl2)

## compute similarity matrix (negative squared Euclidean)
sim <- negDistMat(x, r=2)

## run affinity propagation
apres <- apcluster(sim, details=TRUE)
```

```
## show details of clustering results
show(apres)

## plot information about clustering run
plot(apres)

## plot clustering result
plot(apres, x)

## plot heatmap
heatmap(apres, sim)
```

---

coerce-methods *Coercion of cluster hierarchies*

Description

Functions for coercing clustering object to hclust and dendrogram objects

Usage

```
## S4 method for signature 'AggExResult'
as.hclust(x, base=0.05)
## S4 method for signature 'ExClust'
as.hclust(x, base=0.05, ...)
## S4 method for signature 'AggExResult'
as.dendrogram(object, base=0.05, useNames=TRUE)
## S4 method for signature 'ExClust'
as.dendrogram(object, base=0.05, useNames=TRUE, ...)
```

Arguments

x	a clustering result object of class APResult, ExClust, or AggExResult
object	a clustering result object of class APResult, ExClust, or AggExResult
base	fraction of height used for the very first join; defaults to 0.05, i.e. the first join appears at 5% of the total height of the dendrogram (see details below).
useNames	if TRUE (default), the labels of the dendrogram are the sample/cluster names (if available); otherwise, the labels are indices.
...	all other arguments are passed on to aggExCluster (see details below).

Details

If called for an AggExResult object, as.hclust creates an hclust object. The heights are transformed to the interval from base (height of lowest join) to 1 (height of highest join). If called for an ExClust or APResult object, aggExCluster is called internally to create a cluster hierarchy first. This is only possible if the pairwise similarities are included in the sim slot of x (see aggExCluster on how to ensure this).

If x is an AggExResult object obtained by clustering an entire data set, as.hclust produces a complete hierarchy. If, however, x is an ExClust (or APResult) object or an AggExResult obtained by running aggExCluster on an ExClust or APResult object, then as.hclust produces a hierarchy of clusters, not of samples.

If called for an AggExResult object, as.dendrogram creates an dendrogram object. Analogously to as.hclust, the heights are transformed to the interval ranging from base (height of lowest join) to 1 (height of highest join). So, any information about heights of merges is lost. If the original join heights are relevant, call plot on the original AggExResult object directly without coercing it to a dendrogram object first. If called for an ExClust or APResult object, aggExCluster is called first to create a cluster hierarchy. Again this is only possible if the pairwise similarities are included in the sim slot of object.

If object is an AggExResult object obtained by clustering an entire data set, as.dendrogram produces a complete dendrogram. If object is an ExClust (or APResult) object or an AggExResult obtained by previously running aggExCluster on an ExClust or APResult object, then as.dendrogram produces a complete dendrogram of all samples, too, but with the difference that entire clusters of the previous ExClust or APResult object are not further split up hierarchically. Consequently, if x is not a complete cluster hierarchy, but a hierarchy of clusters, as.dendrogram(as.hclust(x)) produces a dendrogram of clusters, whereas as.dendrogram(x) in any case produces a dendrogram of samples (with the special property mentioned above).

Value

see details above

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, and Johannes Palme

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

APResult, AggExResult, ExClust, heatmap-methods, apcluster, apclusterL, aggExCluster, cutree-methods

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(20, 0.2, 0.05), rnorm(20, 0.8, 0.06))
cl2 <- cbind(rnorm(20, 0.7, 0.08), rnorm(20, 0.3, 0.05))
x <- rbind(cl1, cl2)

## run affinity propagation
apres <- apcluster(negDistMat(r=2), x, q=0.7, details=TRUE)
```

```
## perform agglomerative clustering of affinity propagation clusters
aggres1 <- aggExCluster(x=apres)

## compute and plot dendrogram
dend1 <- as.dendrogram(aggres1)
dend1
plot(dend1)

## compute and show dendrogram computed from hclust object
dend2 <- as.dendrogram(as.hclust(aggres1))
dend2
plot(dend2)

## perform agglomerative clustering of whole data set
aggres2 <- aggExCluster(negDistMat(r=2), x)

## compute and plot dendrogram
dend3 <- as.dendrogram(aggres2)
dend3
plot(dend3)
```

`conversions` *Conversions Between Dense and Sparse Similarity Matrices*

Description

Converts a dense similarity matrix into a sparse one or vice versa

Usage

```
## S4 method for signature 'matrix'
as.SparseSimilarityMatrix(s, lower=-Inf)
## S4 method for signature 'Matrix'
as.SparseSimilarityMatrix(s, lower=-Inf)
## S4 method for signature 'sparseMatrix'
as.SparseSimilarityMatrix(s, lower=-Inf)
## S4 method for signature 'matrix'
as.DenseSimilarityMatrix(s, fill=-Inf)
## S4 method for signature 'Matrix'
as.DenseSimilarityMatrix(s, fill=-Inf)
## S4 method for signature 'sparseMatrix'
as.DenseSimilarityMatrix(s, fill=-Inf)
```

Arguments

`s`	a similarity matrix in sparse or dense format (see details below)
`lower`	cut-off threshold to apply when converting similarity matrices into sparse format. All similarities lower than or equal to `lower` will be omitted from the result. The default is `-Inf`), i.e. only `-Inf` values are removed.

`fill` value to fill in for entries that are missing from sparse similarity matrix 's' (defaults to `-Inf`).

Details

The function `as.SparseSimilarityMatrix` takes a matrix argument, removes all diagonal elements and all values that are lower than or equal to the cut-off threshold `lower` and returns a sparse matrix of class `dgTMatrix`.

If the function `as.DenseSimilarityMatrix` is called for a sparse matrix (class `sparseMatrix` or any class derived from this class), a dense matrix is returned, where all values that were missing in the sparse matrix are replaced with `fill`.

`as.DenseSimilarityMatrix` can also be called for dense `matrix` and `Matrix` objects. In this case, `as.DenseSimilarityMatrix` assumes that the matrices have three columns that encode for a sparse matrix in the same way as the Matlab implementation of Frey's and Dueck's sparse affinity propagation accepts it: the first column contains 1-based row indices, the second column contains 1-based column indices, and the third column contains the similarity values. The same format is also accepted by `as.SparseSimilarityMatrix` to convert a sparse similarity matrix of this format into a `dgTMatrix` object. Note that, for matrices of this format, `as.DenseSimilarityMatrix` replaces the deprectated function `sparseToFull` that was used in older versions of the package.

Note that `as.SparseSimilarityMatrix` and `as.DenseSimilarityMatrix` are no S4 coercion methods. There are no classes named `SparseSimilarityMatrix` or `DenseSimilarityMatrix`.

Value

returns a square similarity matrix in sparse format (class `dgTMatrix` or in dense format (standard class `matrix`).

Author(s)

Ulrich Bodenhofer

References

https://github.com/UBod/apcluster

Frey, B. J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science* **315**, 972-976. DOI: doi:10.1126/science.1136800.

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

Examples

```
## create similarity matrix in sparse format according to Frey and Dueck
sp <- matrix(c(1, 2, 0.5, 3, 1, 0.2, 5, 4, -0.2, 3, 4, 1.2), 4, 3, byrow=TRUE)
sp

## perform conversions
as.DenseSimilarityMatrix(sp, fill=0)
as.SparseSimilarityMatrix(sp)
```

```
## create dense similarity matrix
cl1 <- cbind(rnorm(20, 0.2, 0.05), rnorm(20, 0.8, 0.06))
cl2 <- cbind(rnorm(20, 0.7, 0.08), rnorm(20, 0.3, 0.05))
x <- rbind(cl1, cl2)

sim <- negDistMat(x, r=2)
ssim <- as.SparseSimilarityMatrix(sim, lower=-0.2)

## run apcluster() on the sparse similarity matrix
apres <- apcluster(ssim, q=0)
apres
```

`cutree-methods` *Cut Out Clustering Level from Cluster Hierarchy*

Description

Cut out a clustering level from a cluster hierarchy

Usage

```
## S4 method for signature 'AggExResult'
cutree(tree, k, h)
## S4 method for signature 'APResult'
cutree(tree, k, h)
```

Arguments

`tree`	an object of class `AggExResult` containing a cluster hierarchy; may also be an object of class `APResult`
`k`	the level (i.e. the number of clusters) to be selected
`h`	alternatively, the level can be selected by specifying a cut-off for the merging objective

Details

The function `cutree` extracts a clustering level from a cluster hierarchy stored in an `AggExResult` object. Which level is selected can be determined by one of the two arguments `k` and `h` (see above). If both `k` and `h` are specified, `k` overrides `h`. This is done largely analogous to the standard function `cutree`. The differences are (1) that only one level can be extracted at a time and (2) that an `ExClust` is returned instead of an index list.

The function `cutree` may further be used to convert an `APResult` object into an `ExClust` object. In this case, the arguments `k` and `h` are ignored.

Value

returns an object of class `ExClust`

Author(s)

Ulrich Bodenhofer and Andreas Kothmeier

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

AggExResult, ExClust

Examples

```
## create two simple clusters
x <- c(1, 2, 3, 7, 8, 9)
names(x) <- c("a", "b", "c", "d", "e", "f")

## compute similarity matrix (negative squared distance)
sim <- negDistMat(x, r=2)

## run affinity propagation
aggres <- aggExCluster(sim)

## show details of clustering results
show(aggres)

## retrieve clustering with 2 clusters
cutree(aggres, 2)

## retrieve clustering with cut-off h=-1
cutree(aggres, h=-1)
```

ExClust-class *Class "ExClust"*

Description

S4 class for storing exemplar-based clusterings

Objects

Objects of this class can be created by calling cutree to cut out a clustering level from a cluster hierarchy of class AggExResult. Moreover, cutree can also be used to convert an object of class APResult to class ExClust.

Slots

The following slots are defined for ExClust objects:

`l`: number of samples in the data set

`sel`: subset of samples used for leveraged clustering

`exemplars`: vector containing indices of exemplars

`clusters`: list containing the clusters; the i-th component is a vector of indices of data points belonging to the i-th exemplar (including the exemplar itself)

`idx`: vector of length `l` realizing a sample-to-exemplar mapping; the i-th entry contains the index of the exemplar the i-th sample belongs to

`sim`: similarity matrix; only available if the preceding clustering method was called with `includeSim=TRUE`.

`call`: method call of the preceding clustering method

Methods

plot `signature(x="ExClust")`: see `plot-methods`

plot `signature(x="ExClust", y="matrix")`: see `plot-methods`

heatmap `signature(x="ExClust")`: see `heatmap-methods`

heatmap `signature(x="ExClust", y="matrix")`: see `heatmap-methods`

show `signature(object="ExClust")`: see `show-methods`

labels `signature(object="ExClust")`: see `labels-methods`

cutree `signature(object="ExClust", k="ANY", h="ANY")`: see `cutree-methods`

length `signature(x="ExClust")`: gives the number of clusters.

sort `signature(x="ExClust")`: see `sort-methods`

as.hclust `signature(x="ExClust")`: see `coerce-methods`

as.dendrogram `signature(object="ExClust")`: see `coerce-methods`

Accessors

In the following code snippets, `x` is an `ExClust` object.

[[`signature(x="ExClust", i="index", j="missing")`: `x[[i]]` returns the i-th cluster as a list of indices of samples belonging to the i-th cluster.

[`signature(x="ExClust", i="index", j="missing", drop="missing")`: `x[i]` returns a list of integer vectors with the indices of samples belonging to this cluster. The list has as many components as the argument `i` has elements. A list is returned even if `i` is a single integer.

similarity `signature(x="ExClust")`: gives the similarity matrix.

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, and Johannes Palme

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

aggExCluster, show-methods, plot-methods, labels-methods, cutree-methods, AggExResult, APResult

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(20, 0.2, 0.05), rnorm(20, 0.8, 0.06))
cl2 <- cbind(rnorm(25, 0.7, 0.08), rnorm(25, 0.3, 0.05))
x <- rbind(cl1, cl2)

## compute similarity matrix (negative squared Euclidean)
sim <- negDistMat(x, r=2)

## run affinity propagation
aggres <- aggExCluster(sim)

## extract level with two clusters
excl <- cutree(aggres, k=2)

## show details of clustering results
show(excl)

## plot information about clustering run
plot(excl, x)
```

---

heatmap *Plot Heatmap*

Description

Functions for Plotting of Heatmap

Usage

```
## S4 method for signature 'ExClust,missing'
heatmap(x, y, ...)
## S4 method for signature 'ExClust,matrix'
heatmap(x, y, ...)
## S4 method for signature 'ExClust,Matrix'
heatmap(x, y, ...)
## S4 method for signature 'ExClust,sparseMatrix'
```

```
heatmap(x, y, ...)
## S4 method for signature 'AggExResult,missing'
heatmap(x, y, ...)
## S4 method for signature 'AggExResult,matrix'
heatmap(x, y, Rowv=TRUE, Colv=TRUE,
    sideColors=NULL, col=heat.colors(12),
    base=0.05,  add.expr, margins=c(5, 5, 2),
    cexRow=max(min(35 / nrow(y), 1), 0.1),
    cexCol=max(min(35 / ncol(y), 1), 0.1), main=NULL, dendScale=1,
    barScale=1, legend=c("none", "col"), ...)
## S4 method for signature 'matrix,missing'
heatmap(x, y, ...)
## S4 method for signature 'missing,matrix'
heatmap(x, y, ...)
```

Arguments

x	a clustering result object of class APResult, ExClust, or AggExResult; for compatibility, x may also be a similarity matrix (see details below).
y	a similarity matrix
sideColors	character vector of colors to be used for plotting color bars that visualize clusters of the finest clustering level in x. This is done in a fashion similar to using RowSideColors or ColSideColors in the standard heatmap function. However, color bars are plotted either on both sides or not at all. The sideColors argument determines the coloring of both horizontal and vertical bars. If sideColors is shorter than the number of clusters in the finest clustering level, sideColors is recycled. In any case, a minimum number of two colors (two elements of sideColors) is required. If NA, no color bars are plotted. If NULL (default), color bars are only plotted if the finest cluster level does not only consist of single samples. In this case, the rainbow function is used to compute the vector of colors which is shuffled such that dissimilar colors are placed next to each other in the color bar.
col	color ramp used for the heatmap image; see image
Rowv	determines whether or not a row dendrogram should be plotted. If FALSE or NA, no row dendrogram is plotted. In any other case, a row dendrogram is plotted unless the number of clusters in the finest clustering level is less than 2. Note that, in the latter case, the actual values in Rowv are ignored, so this argument cannot be used to supply a previously computed dendrogram or re-ordering of elements as in the standard heatmap function.
Colv	determines whether or not a column dendrogram should be plotted. Fully analogous to Rowv, except that column dendrograms are never plotted if the similarity matrix y is non-quadratic.
base	fraction of height used for the very first join in dendrograms; see coerce-methods.
add.expr, margins, cexRow, cexCol, main	largely analogous to the standard heatmap function; to omit row/column labeling, set cexRow/cexCol to 0. The default for margins is a vector of length 3, where the third element is the right-hand side margin for the color legend (see

	legend argument). It remains unused (and can also be omitted) if no color legend is plotted.
dendScale	factor scaling the width of vertical and height of horizontal dendrograms; values have to be larger than 0 and no larger than 2. The default is 1 which corresponds to the same size as the dendrograms plot by the standard heatmap function
barScale	factor scaling the width of color bars; values have to be larger than 0 and no larger than 4. The default is 1 which corresponds to half the width of the color bars plot by the standard heatmap function
legend	if "col", then a color legend similar to filled.contour is added on the right-hand side of the heatmap plot; if "none" (default), no such legend is added.
...	see details below

Details

The heatmap functions provide plotting of heatmaps from several different types of input object. The implementation is similar to the standard graphics function heatmap. Plotting heatmaps via the plot command as available in previous versions of this package is still available for backward compatibility.

If heatmap is called for objects of classes APResult or ExClust, a heatmap of the similarity matrix in slot sim of the parameter x is created with clusters grouped together and highlighted in different colors. The order of clusters is determined by running aggExCluster on the clustering result x. This variant of heatmap returns an invisible AggExResult object.

If heatmap is called for an AggExResult object that contains all levels of clustering, the heatmap is displayed with the corresponding clustering dendrogram. If the AggExResult object is the result of running aggExCluster on a prior clustering result, the same heatmap plot is produced as if heatmap had been called on this prior clustering result, however, returning the cluster hierarchy's dendrogram. In the latter case, color bars are plotted to visualize the prior clustering result (see description of argument sideColors above).

All variants described above only work if the input object x contains a slot sim with the similarity matrix (which is only the case if the preceding clustering method has been called with includeSim=TRUE). In case the slot sim of x does not contain the similarity matrix, the similarity matrix must be supplied as second argument y.

All variants described above internally use heatmap with signature AggExResult,matrix, so all arguments list above can be used for all variants, as they are passed through using the ... argument. All other arguments, analogously to the standard heatmap function, are passed on to the standard function image. This is particularly useful for using alternative color schemes via the col argument.

The two variants with one of the two arguments being a matrix and one being missing are just wrappers around the standard heatmap function with the aim to provide compatibility with this standard case.

Value

see details above

Note

Similarity matrices can be supplied in dense or sparse format. Note, however, that sparse matrices are converted to full dense matrices before plotting heatmaps which may lead to memory and/or performance bottlenecks for larger data sets.

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, and Johannes Palme

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

APResult, AggExResult, ExClust, apcluster, apclusterL, aggExCluster, cutree-methods, plot-methods

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(50, 0.2, 0.05), rnorm(50, 0.8, 0.06))
cl2 <- cbind(rnorm(50, 0.7, 0.08), rnorm(50, 0.3, 0.05))
x <- rbind(cl1, cl2)

## run affinity propagation using negative squared Euclidean
apres <- apcluster(negDistMat(r=2), x, p=-0.1)

## plot heatmap clustering run
heatmap(apres)

## rerun affinity propagation
## reuse similarity matrix from previous run
apres2 <- apcluster(s=apres@sim, q=0.6)

## plot heatmap of second run
heatmap(apres2, apres@sim)

## with alternate heatmap coloring, alternating color bars, and no dendrograms
heatmap(apres2, apres@sim, Rowv=NA, Colv=NA,
        sideColors=c("darkgreen", "yellowgreen"), col=terrain.colors(12))

## perform agglomerative clustering of affinity propagation clusters
aggres1 <- aggExCluster(apres@sim, apres2)

## plot heatmap
heatmap(cutree(aggres1, 2), apres@sim)

## perform agglomerative clustering of whole data set
```

```
aggres2 <- aggExCluster(negDistMat(r=2), x)

## show heatmap along with dendrogram
heatmap(aggres2)
```

`labels-methods` *Generate label vector from clustering result*

Description

Generate a label vector from an clustering result

Usage

```
## S4 method for signature 'ExClust'
labels(object, type="names")
```

Arguments

`object`	object of class `APResult` or `ExClust`
`type`	specifies which kind of label vector should be created, see details below

Details

The function `labels` creates a label vector from a clustering result. Which kind of labels are produced is controlled by the argument `type`:

"names" (default) returns the name of the exemplar to which each data sample belongs to; if no names are available, the function stops with an error;

"enum" returns the index of the cluster to which each data sample belongs to, where clusters are enumerated consecutively from 1 to the number of clusters (analogous to other clustering methods like `kmeans`);

"exemplars" returns the index of the exemplar to which each data sample belongs to, where indices of exemplars are within the original data, which is nothing else but the slot `object@idx` with attributes removed.

Value

returns a label vector as long as the number of samples in the original data set

Author(s)

Ulrich Bodenhofer and Andreas Kothmeier

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

APResult, ExClust, cutree

Examples

```
## create two simple clusters
x <- c(1, 2, 3, 7, 8, 9)
names(x) <- c("a", "b", "c", "d", "e", "f")

## compute similarity matrix (negative squared distance)
sim <- negDistMat(x, r=2)

## run affinity propagation
apres <- apcluster(sim)

## show details of clustering results
show(apres)

## label vector (names of exemplars)
labels(apres)

## label vector (consecutive index of exemplars)
labels(apres, type="enum")

## label vector (index of exemplars within original data set)
labels(apres, type="exemplars")

## now with agglomerative clustering
aggres <- aggExCluster(sim)

## label (names of exemplars)
labels(cutree(aggres, 2))
```

---

plot *Plot Clustering Results*

Description

Functions for Visualizing Clustering Results

Usage

```
## S4 method for signature 'APResult,missing'
plot(x, y, type=c("netsim", "dpsim", "expref"),
    xlab="# Iterations", ylab="Similarity", ...)
## S4 method for signature 'ExClust,matrix'
plot(x, y, connect=TRUE, xlab="", ylab="",
labels=NA, limitNo=15, ...)
## S4 method for signature 'ExClust,data.frame'
```

```
plot(x, y, connect=TRUE, xlab="",
ylab="", labels=NA, limitNo=15, ...)
## S4 method for signature 'AggExResult,missing'
plot(x, y, main="Cluster dendrogram",
    xlab="", ylab="", ticks=4, digits=2, base=0.05, showSamples=FALSE,
    horiz=FALSE, ...)
## S4 method for signature 'AggExResult,matrix'
plot(x, y, k=NA, h=NA, ...)
## S4 method for signature 'AggExResult,data.frame'
plot(x, y, k=NA, h=NA, ...)
```

Arguments

x	a clustering result object of class APResult, ExClust, or AggExResult
y	a matrix or data frame (see details below)
type	a string or array of strings indicating which performance measures should be plotted; valid values are "netsim", "dpsim", and "expref" which can be used in any combination or order; all other strings are ignored (for the meaning see APResult)
xlab, ylab	labels for axes of 2D plots; ignored if y has more than two columns
labels	names used for variables in scatter plot matrix (displayed if y has more than two columns). If NA (default), column names are used. If no column names are available, labels such as x[, 2] are displayed.
limitNo	if the number of columns/features in y is too large, problems may occur when attempting to plot a scatter plot matrix. To avoid problems, the plot method throws an error if the number of columns exceeds limitNo. For special applications, users can increase the value (15 by default). If limitNo is set to NA or any other non-numeric value, the limit is ignored entirely. Please note that attempting to plot scatter plot matrices with too many features may corrupt the graphics device. So users are making changes at their own risk. If plotting of many features is necessary, make sure that the graphics device is large enough to accommodate the plot (e.g. by using a sufficiently large graphics file device).
connect	used only if clustering is plotted on original data, ignored otherwise. If connect is TRUE, lines are drawn connecting exemplars with their cluster members.
main	title of plot
ticks	number of ticks used for the axis on the left side of the plot (applies to dendrogram plots only, see below)
digits	number of digits used for the axis tickmarks on the left side of the plot (applies to dendrogram plots only, see below)
base	fraction of height used for the very first join; defaults to 0.05, i.e. the first join appears at 5% of the total height of the dendrogram.
showSamples	if TRUE, a complete cluster hierarchy is shown, otherwise, in case that x is a hierarchy of clusters, the dendrogram of clusters is shown. For backward compatibility, the default is FALSE.
horiz	if TRUE, the dendrogram is plotted horizontally (analogous to plot.dendrogram). The default is FALSE.

k	level to be selected when plotting a single clustering level of cluster hierarchy (i.e. the number of clusters; see cutree-methods)
h	cut-off to be used when plotting a single clustering level of cluster hierarchy (see cutree-methods)
...	all other arguments are passed to the plotting command that are used internally, plot or heatmap.

Details

If plot is called for an APResult object without specifying the second argument y, a plot is created that displays graphs of performance measures over execution time of the affinity propagation run. This only works if apcluster was called with details=TRUE.

If plot is called for an APResult object along with a matrix or data frame as argument y, then the dimensions of the matrix determine the behavior of plot:

1. If the matrix y has two columns, y is interpreted as the original data set. Then a plot of the clustering result superimposed on the original data set is created. Each cluster is displayed in a different color. The exemplar of each cluster is highlighted by a black square. If connect is TRUE, lines connecting the cluster members to their exemplars are drawn. This variant of plot does not return any value.
2. If y has more than two columns, clustering results are superimposed in a sort of scatter plot matrix. The variant that y is interpreted as similarity matrix if it is quadratic has been removed in version 1.3.2. Use heatmap instead.
3. If y has only one column, an error is displayed.

If plot is called for an ExClust object along with a matrix or data frame as argument y, then plot behaves exactly the same as described in the previous paragraph.

If plot is called for an AggExResult object without specifying the second argument y, then a dendrogram plot is drawn. This variant returns an invisible dendrogram object. The showSamples argument determines whether a complete dendrogram or a dendrogram of clusters is plotted (see above). If the option horiz=TRUE is used, the dendrogram is rotated. Note that, in this case, the margin to the right of the plot may not be wide enough to accommodate long cluster/sample labels. In such a case, the figure margins have to be widened before plot is called.

If plot is called for an AggExResult object along with a matrix or data frame y, y is again interpreted as original data set. If one of the two arguments k or h is present, a clustering is cut out from the cluster hierarchy using cutree and this clustering is displayed with the original data set as described above. This variant of plot returns an invisible ExClust object containing the extracted clustering.

Value

see details above

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, and Johannes Palme

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

APResult, AggExResult, ExClust, heatmap-methods, apcluster, apclusterL, aggExCluster, cutree-methods

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(50, 0.2, 0.05), rnorm(50, 0.8, 0.06))
cl2 <- cbind(rnorm(50, 0.7, 0.08), rnorm(50, 0.3, 0.05))
x <- rbind(cl1, cl2)

## run affinity propagation
apres <- apcluster(negDistMat(r=2), x, q=0.7, details=TRUE)

## plot information about clustering run
plot(apres)

## plot clustering result
plot(apres, x)

## perform agglomerative clustering of affinity propagation clusters
aggres1 <- aggExCluster(x=apres)

## show dendrograms
plot(aggres1)
plot(aggres1, showSamples=TRUE)

## show clustering result for 4 clusters
plot(aggres1, x, k=4)

## perform agglomerative clustering of whole data set
aggres2 <- aggExCluster(negDistMat(r=2), x)

## show dendrogram
plot(aggres2)

## show heatmap along with dendrogram
heatmap(aggres2)

## show clustering result for 2 clusters
plot(aggres2, x, k=2)

## cluster iris data set
data(iris)
apIris <- apcluster(negDistMat(r=2), iris, q=0)
```

```
plot(apIris, iris)
```

---

preferenceRange *Determine Meaningful Ranges for Input Preferences*

Description

Determines meaningful ranges for affinity propagation input preference

Usage

```
## S4 method for signature 'matrix'
preferenceRange(s, exact=FALSE)
## S4 method for signature 'Matrix'
preferenceRange(s, exact=FALSE)
## S4 method for signature 'dgTMatrix'
preferenceRange(s, exact=FALSE)
## S4 method for signature 'sparseMatrix'
preferenceRange(s, exact=FALSE)
```

Arguments

s	an $l \times l$ similarity matrix in sparse or dense format
exact	flag indicating whether exact ranges should be computed, which is relatively slow; if bounds are sufficient, supply FALSE (default)

Details

Affinity Propagation clustering relies on an appropriate choice of input preferences. This function helps in finding a good choice by determining meaningful lower and upper bounds.

If the similarity matrix s is sparse or if it contains -Inf similarities, only the similarities are taken into account that are specified in s and larger than -Inf. In such cases, the lower bound returned by preferenceRange need not correspond to one or two clusters. Moreover, it may also happen in degenerate cases that the lower bound exceeds the upper bound. In such a case, no warning or error is issued, so it is the user's responsibility to ensure a proper interpretation of the results. The method apclusterK makes use of this function internally and checks the plausibility of the result returned by preferenceRange.

Value

returns a vector with two entries, the first of which is the minimal input preference (which would lead to 1 or 2 clusters) and the second of which is the maximal input prefence (which would lead to as many clusters as data samples).

Author(s)

Ulrich Bodenhofer and Andreas Kothmeier

References

https://github.com/UBod/apcluster

Frey, B. J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science* **315**, 972-976. DOI: doi:10.1126/science.1136800.

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

apcluster

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(100, 0.2, 0.05), rnorm(100, 0.8, 0.06))
cl2 <- cbind(rnorm(50, 0.7, 0.08), rnorm(50, 0.3, 0.05))
x <- rbind(cl1, cl2)

## create similarity matrix
sim <- negDistMat(x, r=2)

## determine bounds
preferenceRange(sim)

## determine exact range
preferenceRange(sim, exact=TRUE)
```

---

show-methods *Display Clustering Result Objects*

Description

Display methods for S4 classes APResult, ExClust, and AggExResult

Usage

```
## S4 method for signature 'APResult'
show(object)
## S4 method for signature 'ExClust'
show(object)
## S4 method for signature 'AggExResult'
show(object)
```

Arguments

object	an object of class APResult, ExClust, or AggExResult

Details

show displays the most important information stored in `object`.

For `APResult` objects, the number of data samples, the number of clusters, the number of iterations, the input preference, the final objective function values, the vector of exemplars, the list of clusters and for leveraged clustering the selected sample subset are printed.

For `ExClust` objects, the number of data samples, the number of clusters, the vector of exemplars, and list of clusters are printed.

For `AggExResult` objects, only the number of data samples and the maximum number of clusters are printed. For retrieving a particular clustering level, use the function `cutree`.

For accessing more detailed information, it is necessary to access the slots of `object` directly. Use `str` to get a compact overview of all slots of an object.

Value

show returns an invisible `NULL`

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, and Johannes Palme

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

`APResult`, `ExClust`, `AggExResult`, `cutree-methods`

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(100, 0.2, 0.05), rnorm(100, 0.8, 0.06))
cl2 <- cbind(rnorm(50, 0.7, 0.08), rnorm(50, 0.3, 0.05))
x <- rbind(cl1, cl2)

## compute similarity matrix (negative squared Euclidean)
sim <- negDistMat(x, r=2)

## run affinity propagation
apres <- apcluster(sim)

## show details of clustering results
show(apres)

## apply agglomerative clustering to apres
aggres <- aggExCluster(sim, apres)
```

```
## display overview of result
show(aggres)

## show clustering level with two clusters
show(cutree(aggres, 2))
```

`similarities` *Methods for Computing Similarity Matrices*

Description

Compute similarity matrices from data set

Usage

```
negDistMat(x, sel=NA, r=1, method="euclidean", p=2)
expSimMat(x, sel=NA, r=2, w=1, method="euclidean", p=2)
linSimMat(x, sel=NA, w=1, method="euclidean", p=2)
corSimMat(x, sel=NA, r=1, signed=TRUE, method="pearson")
linKernel(x, sel=NA, normalize=FALSE)
```

Arguments

`x`	input data to be clustered; if `x` is a vector, it is interpreted as a list of scalar values that are to be clustered; if `x` is a matrix or data frame, rows are interpreted as samples and columns are interpreted as features; in the case that `x` is a data frame, only numerical columns/features are taken into account, whereas categorical features are neglected. If `x` is missing, all functions return a function that can be used as similarity measure, in particular, as `s` argument for `apclusterL`.
`sel`	selected samples subset; vector of row indices for `x` in increasing order (see details below)
`r`	exponent (see details below)
`w`	radius (see details below)
`signed`	take sign of correlation into account (see details below)
`normalize`	see details below
`method`	type of distance measure to be used; for `negDistMat`, `expSimMat` and `linSimMat`, this argument is analogous to the `method` argument of `dist`. For `corSimMat`, this argument is analogous to the `method` argument of `cor`.
`p`	exponent for Minkowski distance; only used for `method="minkowski"`, otherwise ignored. See `dist`.

Details

`negDistMat` creates a square matrix of mutual pairwise similarities of data vectors as negative distances. The argument `r` (default is 1) is used to transform the resulting distances by computing the r-th power (use `r=2` to obtain negative squared distances as in Frey's and Dueck's demos), i.e., given a distance d, the resulting similarity is computed as $s = -d^r$. With the parameter `sel` a subset of samples can be specified for distance calculation. In this case not the full distance matrix is computed but a rectangular similarity matrix of all samples (rows) against the subset (cols) as needed for leveraged clustering. Internally, the computation of distances is done using an internal method derived from `dist`. All options of this function except `diag` and `upper` can be used, especially `method` which allows for selecting different distance measures. Note that, since version 1.4.4. of the package, there is an additional method `"discrepancy"` that implements Weyl's discrepancy measure.

`expSimMat` computes similarities in a way similar to `negDistMat`, but the transformation of distances to similarities is done in the following way:

$$s = \exp\left(-\left(\frac{d}{w}\right)^r\right)$$

The parameter `sel` allows the creation of a rectangular similarity matrix. As above, r is an exponent. The parameter w controls the speed of descent. `r=2` in conjunction with Euclidean distances corresponds to the well-known Gaussian/RBF kernel, whereas `r=1` corresponds to the Laplace kernel. Note that these similarity measures can also be understood as fuzzy equality relations.

`linSimMat` provides another way of transforming distances into similarities by applying the following transformation to a distance d:

$$s = \max\left(0, 1 - \frac{d}{w}\right)$$

Thw parameter `sel` is used again for creation of a rectangular similarity matrix. Here `w` corresponds to a maximal radius of interest. Note that this is a fuzzy equality relation with respect to the Lukasiewicz t-norm.

Unlike the above three functions, `linKernel` computes pairwise similarities as scalar products of data vectors, i.e. it corresponds, as the name suggests, to the "linear kernel". Use parameter `sel` to compute only a submatrix of the full kernel matrix as described above. If `normalize=TRUE`, the values are scaled to the unit sphere in the following way (for two samples `x` and `y`:

$$s = \frac{\vec{x}^T\vec{y}}{\|\vec{x}\|\|\vec{y}\|}$$

The function `corSimMat` computes pairwise similarities as correlations. It uses `link[stats:cor]{cor}` internally. The `method` argument is passed on to `link[stats:cor]{cor}`. The argument `r` serves as an exponent with which the correlations can be transformed. If `signed=TRUE` (default), negative correlations are taken into account, i.e. two samples are maximally dissimilar if they are negatively correlated. If `signed=FALSE`, similarities are computed as absolute values of correlations, i.e. two samples are maximally similar if they are positively or negatively correlated and the two samples are maximally dissimilar if they are uncorrelated.

Note that the naming of the argument `p` has been chosen for consistency with `dist` and previous versions of the package. When using leveraged AP in conjunction with the Minkowski distance, this

leads to conflicts with the input preference parameter p of apclusterL. In order to avoid that, use the above functions without x argument to create a custom similarity measure with fixed parameter p (see example below).

Value

All functions listed above return square or rectangular matrices of similarities.

Author(s)

Ulrich Bodenhofer, Andreas Kothmeier, and Johannes Palme

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

Frey, B. J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science* **315**, 972-976. DOI: doi:10.1126/science.1136800.

Micchelli, C. A. (1986) Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constr. Approx.* **2**, 11-20.

De Baets, B. and Mesiar, R. (1997) Pseudo-metrics and T-equivalences. *J. Fuzzy Math.* **5**, 471-481.

Bauer, P., Bodenhofer, U., and Klement, E. P. (1996) A fuzzy algorithm for pixel classification based on the discrepancy norm. In *Proc. 5th IEEE Int. Conf. on Fuzzy Systems*, volume III, pages 2007–2012, New Orleans, LA. DOI: doi:10.1109/FUZZY.1996.552744.

See Also

dist, apcluster, apclusterL

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(100, 0.2, 0.05), rnorm(100, 0.8, 0.06))
cl2 <- cbind(rnorm(100, 0.7, 0.08), rnorm(100, 0.3, 0.05))
x <- rbind(cl1, cl2)

## create negative distance matrix (default Euclidean)
sim1 <- negDistMat(x)

## compute similarities as squared negative distances
## (in accordance with Frey's and Dueck's demos)
sim2 <- negDistMat(x, r=2)

## compute RBF kernel
sim3 <- expSimMat(x, r=2)

## compute similarities as squared negative distances
## all samples versus a randomly chosen subset
```

```
## of 50 samples (for leveraged AP clustering)
sel <- sort(sample(1:nrow(x), nrow(x)*0.25))
sim4 <- negDistMat(x, sel, r=2)


## example of leveraged AP using Minkowski distance with non-default
## parameter p
cl1 <- cbind(rnorm(150, 0.2, 0.05), rnorm(150, 0.8, 0.06))
cl2 <- cbind(rnorm(100, 0.7, 0.08), rnorm(100, 0.3, 0.05))
x <- rbind(cl1, cl2)

apres <- apclusterL(s=negDistMat(method="minkowski", p=2.5, r=2),
                    x, frac=0.2, sweeps=3, p=-0.2)
show(apres)
```

`sort-methods` *Sort clusters*

Description

Rearrange clusters according to sort criterion

Usage

```
## S4 method for signature 'ExClust'
sort(x, decreasing=FALSE,
                  sortBy=c("aggExCluster", "size",
                           "nameExemplar", "noExemplar"), ...)
```

Arguments

`x`	object of class `APResult` or `ExClust`
`decreasing`	logical indicating if sorting should be done in decreasing order, see details below
`sortBy`	sort criterion, see details below
`...`	further arguments are ignored; only defined for S3 method consistency

Details

The function `sort` takes an `APResult` or `ExClust` clustering object `x` and creates a new clustering object of the same class, but with clusters arranged according to the sort criterion passed as argument `sortBy`:

"aggExCluster" (default) order clusters as they would appear in the dendrogram produced by `aggExCluster`. This is also the same ordering in which the clusters are arranged by `heatmap`. Note that this only works if the similarity matrix is included in the input object `x`, otherwise an error message is produced.

"size" sorts clusters according to their size (from small to large).

"nameExemplar" sorts clusters according to the names of the examplars (if available, otherwise an error is produced).

"noExemplar" sorts clusters according to the indices of the examplars.

If `decreasing` is `TRUE`, the order is reversed and, for example, `sortBy="size"` sorts clusters with such that the larger clusters come first.

Note that the cluster numbers of `x` are not preserved by `sort`, i.e. the cluster no. 1 of the object returned by `sort` is the one that has been ranked first by `sort`, which may not necessarily coincide with cluster no. 1 of the original clustering object `x`.

Note that this is an S3 method (whereas all other methods in this package are S4 methods). This inconsistency has been introduced in order to avoid interoperability problems with the **BiocGenerics** package which may overwrite the definition of the `sort` generic if it is loaded after the **apcluster** package.

Value

returns a copy of `x`, but with slots `exemplars` and `clusters` (see `APResult` or `ExClust`) reordered.

Author(s)

Ulrich Bodenhofer

References

https://github.com/UBod/apcluster

Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011) APCluster: an R package for affinity propagation clustering. *Bioinformatics* **27**, 2463-2464. DOI: doi:10.1093/bioinformatics/btr406.

See Also

`APResult`, `ExClust`

Examples

```
## create two Gaussian clouds
cl1 <- cbind(rnorm(50,0.2,0.05),rnorm(50,0.8,0.06))
cl2 <- cbind(rnorm(50,0.7,0.08),rnorm(50,0.3,0.05))
x <- rbind(cl1,cl2)

## run affinity propagation
apres <- apcluster(negDistMat(r=2), x, q=0.7)

show(apres)
## show dendrogram
plot(aggExCluster(x=apres))

## default sort order: like in heatmap or dendrogram
show(sort(apres))

## show dendrogram (note the different cluster numbers!)
```

```
plot(aggExCluster(x=sort(apres)))

## sort by size
show(sort(apres, decreasing=TRUE, sortBy="size"))
```

Index