Package ‘WeightedCluster’

February 19, 2015

Version 1.2
Date 2014-01-16
Title Clustering of Weighted Data
Depends R (>= 3.0.0), TraMineR (>= 1.8-8), cluster
Imports utils, RColorBrewer
Suggests RUnit, knitr, isotone, vegan, lattice
VignetteBuilder knitr
Description The WeightedCluster library provides functions to cluster states sequences and weighted data. These functionalities include aggregating replicated cases, an optimized weighted PAM algorithm, function computing cluster quality measures for a range of clustering solutions and miscellaneous functions to plot clustering solutions of state sequences.
License GPL (>= 2)
URL http://mephisto.unige.ch/weightedcluster
Author Matthias Studer [aut, cre]
Maintainer Matthias Studer <matthias.studer@unige.ch>
NeedsCompilation yes
Repository CRAN
Date/Publication 2014-01-19 14:22:57

R topics documented:

as.clustrange ................................................. 2
as.seqtree ....................................................... 4
seqclustname ..................................................... 5
wcAggregateCases ............................................. 6
wcClusterQuality .............................................. 7
wcCmpCluster ................................................... 8
wcKMedoids ...................................................... 10
wcKMedRange .................................................... 12
wcSilhouetteObs ............................................... 13

Index 16
Build a clustrange object to compare different clustering solutions.

**Usage**

```r
as.clustrange(object, diss=NULL, weights=NULL, r=1, samplesize=NULL, ncluster=20, ...)
```

**Arguments**

- **object**
  The object to convert such as a data.frame.
- **diss**
  A dissimilarity matrix or a dist object (see `dist`).
- **weights**
  Optional numerical vector containing weights.
- **r**
  Optional number of bootstrap that can be used to build confidence intervals.
- **samplesize**
  Size of bootstrap sample. Default to sum of weights.
- **ncluster**
  Integer. Maximum number of cluster. The range will include all clustering solution starting from two to ncluster.
- **x**
  A clustrange object to be plotted.
- **stat**
  Character. The list of statistics to plot or "noCH" to plot all statistics except "CH" and "CHsq" or "all" for all statistics. See `wcClusterQuality` for a list of possible values. It is also possible to use "RHC" to plot the quality measure 1-HC. Unlike HC, RHC should be maximized as all other quality measures.
- **legendpos**
  Character. legend position, see `legend`.
- **norm**
  Character. Normalization method of the statistics can be one of "none" (no normalization), "range" (given as (value-min)/(max-min)), "zscore" (adjusted by mean and standard deviation) or "zscoremed" (adjusted by median and median of the difference to the median).
- **withlegend**
  Logical. If FALSE, the legend is not plotted.
- **lwd**
  Numeric. Line width, see `par`.
- **col**
  A vector of line colors, see `par`. If NULL, a default set of color is used.
- **xlab**
  x axis label.
- **ylab**
  y axis label.
as.clustrange

As "clustrange"

- **conf.int**: Confidence to build the confidence interval (default: 0.9).
- **ci.method**: Method used to build the confidence interval (only if bootstrap has been used, see R above). One of "none" (do not plot confidence interval), "norm" (based on normal approximation), "perc" (based on percentile).
- **ci.alpha**: Alpha color value used to plot the interval.
- **line**: Which value should be plotted by the line? One of "t0" (value for actual sample), "mean" (average over all bootstraps), "median" (median over all bootstraps).
- **...**: Additional parameters passed to/from methods.

**Details**

as.clustrange convert objects to clustrange objects. clustrange objects contains a list of clustering solution with associated statistics and can be used to find the optimal clustering solution.

If object is a data.frame or a matrix, each column should be a clustering solution to be evaluated.

If object is an hclust or twins objects (i.e. hierarchical clustering output, see hclust, diana or agnes), the function compute all clustering solution ranging from two to ncluster and compute the associated statistics.

**Value**

An object of class clustrange with the following elements:

- **clustering**: A data.frame of all clustering solutions.
- **stats**: A matrix containing the clustering statistics of each cluster solution.

**See Also**

See also wcKMedRange, wcClusterQuality.

**Examples**

```r
data(mvad)
## Aggregating state sequence
aggMvad <- wcAggregateCases(mvad[, 17:86], weights=mvad$weight)

## Creating state sequence object
mvad.seq <- seqdef(mvad[aggMvad$aggIndex, 17:86], weights=aggMvad$aggWeights)

## Compute distance using Hamming distance
diss <- seqdist(mvad.seq, method="HAM")

## Ward clustering
wardCluster <- hclust(as.dist(diss), method="ward", members=aggMvad$aggWeights)

## Computing clustrange from Ward clustering
wardRange <- as.clustrange(wardCluster, diss=diss,
                           weights=aggMvad$aggWeights, ncluster=15)

## Plot all statistics (standardized)
```
as.seqtree

Convert a hierarchical clustering object to a seqtree object.

Description

Convert a hierarchical clustering object to a seqtree object which can then be displayed using seqtreedisplay.

Usage

as.seqtree(object, seqdata, diss, weighted=TRUE, ...)  
## S3 method for class 'twins'
as.seqtree(object, seqdata, diss, weighted=TRUE, ncluster, ...)
## S3 method for class 'hclust'
as.seqtree(object, seqdata, diss, weighted=TRUE, ncluster, ...)

Arguments

- **object**: An object to be converted to a seqtree.
- **seqdata**: State sequence object.
- **diss**: A dissimilarity matrix or a dist object (see dist)
- **weighted**: Logical. If TRUE, weights of the seqdata object are taken to build the tree.
- **ncluster**: Maximum number of cluster. The tree will be builded until this number of cluster.
- **...**: Additionnal parameters passed to/from methods.

Details

By default as.seqtree try to convert the object to a data.frame assuming that it contains a list of nested clustering solutions. Be aware that seqtree and as.seqtree only support binary splits.

If object is an hclust or twins objects (i.e. hierarchical clustering output, see hclust, diana or agnes), the function returns a seqtree object reproducing the agglomerative schedule.

Value

A seqtree object.
Examples

data(mvad)
## Aggregating state sequence
aggMvad <- wcAggregateCases(mvad[, 17:86], weights=mvad$weight)

## Creating state sequence object
mvad.seq <- seqdef(mvad[, aggMvad$aggIndex, 17:86], weights=aggMvad$aggWeights)

## Compute distance using Hamming distance
diss <- seqdist(mvad.seq, method="HAM")

## Ward clustering
wardCluster <- hclust(as.dist(diss), method="ward", members=aggMvad$weight)

st <- as.seqtree(wardCluster, seqdata=mvad.seq, diss=diss, weighted=TRUE, ncluster=10)
print(st)

## You typically want to run (You need to install GraphViz before)
## seqtreedisplay(st, type="d", border=NA)

---

seqclustname

**Automatic labeling of cluster using sequence medoids**

Description

This function automatically name the cluster using the sequence medoid of each cluster.

Usage

```
seqclustname(seqdata, group, diss, weighted = TRUE, perc = FALSE)
```

Arguments

- `seqdata` State sequence object (see `seqdef`).
- `group` A vector of clustering membership.
- `diss` a dissimilarity matrix or a dist object.
- `weighted` Logical. If TRUE, weights of the seqdata object are taken to find the medoids.
- `perc` Logical. If TRUE, the percentage of sequences in each cluster is added to the label of each group.

Value

A factor of clustering membership. The labels are defined using sequences medoids and optionally percentage of case in each cluster.
Examples

data(mvad)
## Aggregating state sequence
aggMvad <- wcAggregateCases(mvad[, 17:86], weights=mvad$weight)

## Creating state sequence object
mvad.seq <- seqdef(mvad[, aggMvad$aggIndex, 17:86], weights=aggMvad$aggWeights)
## Computing Hamming distance between sequence
diss <- seqdist(mvad.seq, method="HAM")

## KMedoids using PAMonce method (clustering only)
clust5 <- wckMedoids(diss, k=5, weights=aggMvad$aggWeights)

clust5.labels <- seqclustname(mvad.seq, clust5$clustering, diss=diss, perc=TRUE)
seqdplot(mvad.seq, group=clust5.labels)

wcAggregateCases  Aggregate identical cases.

Description

Function to aggregate identical cases.

Usage

wcAggregateCases(x, weights = NULL, ...)

## S3 method for class 'data.frame'
wcagregateCases(x, weights=NULL, ...)

## S3 method for class 'matrix'
wcagregateCases(x, weights=NULL, ...)

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

Arguments

x  The object to aggregate.
weights  Numeric. An optional case weights vector.
...  Optional additioanl arguments.

Value

A wcAggregateCases object with the following components:

aggIndex  Index of the unique cases in the original object data.
aggWeights  Aggregated case weights
disaggIndex  Index of the original object data in the unique cases.
disaggWeights  Original weights used.
Examples

```r
data(mvad)
## Taking only the father unemployment and
## success at the end of compulsory schooling.
myData <- mvad[, c("funemp", "gcse5eq")]
## Computing aggregated cases informations
ac <- wcAggregateCases(myData, weights=mvad$weight)
print(ac)
## Retrieving unique cases in the original data set
uniqueData <- myData[ac$aggIndex, ]
## Table from original data
table.orig <- xtabs(mvad$weight~funemp+gcse5eq, data=myData)
## Table from aggregated data
table.agg <- xtabs(ac$aggWeights~funemp+gcse5eq, data=uniqueData)

## Both table are equal, no information is lost
## (only the call command is different)
all(table.orig == table.agg)
```

---

### wcClusterQuality  
*Cluster quality statistics*

**Description**

Compute several quality statistics of a given clustering solution.

**Usage**

```r
wcClusterQuality(diss, clustering, weights = NULL)
```

**Arguments**

- `diss`  
  A dissimilarity matrix or a `dist` object (see `dist`)
- `clustering`  
  Factor. A vector of clustering membership.
- `weights`  
  optional numerical vector containing weights.

**Details**

Compute several quality statistics of a given clustering solution. See value for details.

**Value**

A list with two elements `stats` and `ASW`:

- `stats`  
  with the following statistics:
**PBC** Point Biserial Correlation. Correlation between the given distance matrix and a distance which equal to zero for individuals in the same cluster and one otherwise.

**HG** Hubert’s Gamma. Same as previous but using Kendall’s Gamma coefficient.

**HGSD** Hubert’s Gamma (Somers’D). Same as previous but using Somers’ D coefficient.

**ASW** Average Silhouette width (observation).

**ASWw** Average Silhouette width (weighted).

**CH** Calinski-Harabasz index (Pseudo F statistics computed from distances).

**R2** Share of the discrepancy explained by the clustering solution.

**CHsq** Calinski-Harabasz index (Pseudo F statistics computed from squared distances).

**R2sq** Share of the discrepancy explained by the clustering solution (computed using squared distances).

**HC** Hubert’s C coefficient.

ASW: The Average Silhouette Width of each cluster, one column for each ASW measure.

### Examples

```
data(mvad)
## Aggregating state sequence
aggMvad <- wcAggregateCases(mvad[, 17:86], weights=mvad$weight)

## Creating state sequence object
mvad.seq <- seqdef(mvad[aggMvad$aggIndex, 17:86], weights=aggMvad$aggWeights)

## Computing Hamming distance between sequence
diss <- seqdist(mvad.seq, method="HAM")

## KMedoids using PAMonce method (clustering only)
clust5 <- wKMedoids(diss, k=5, weights=aggMvad$aggWeights, cluster.only=TRUE)

## Compute the silhouette of each observation
qual <- wcClusterQuality(diss, clust5, weights=aggMvad$aggWeights)

print(qual)
```

---

**wcCmpCluster**

*Automatic comparison of clustering methods.*

### Description

Automatically compute different clustering solutions and associated quality measures to help identifying the best one.
Usage

wcCmpCluster(diss, weights = NULL, maxcluster, method = "all", pam.combine = TRUE)
## S3 method for class 'clustrangetfamily'
print(x, max.rank=1, ...)
## S3 method for class 'clustrangetfamily'
summary(object, max.rank=1, ...)
## S3 method for class 'clustrangetfamily'
plot(x, group="stat", method="all", pam.combine=FALSE,
  stat="noCH", norm="none", withlegend=TRUE, lwd=1, col=NULL, legend.prop=NA,
  rows=NA, cols=NA, main=NULL, xlab="", ylab="", ...) 

Arguments

diss            A dissimilarity matrix or a dist object (see dist).
weights         Optional numerical vector containing weights.
maxcluster      Integer. Maximum number of cluster. The range will include all clustering
                solution starting from two to ncluster.
method          A vector of hierarchical clustering methods to compute or "all" for all
                methods. Possible values include "ward", "single", "complete", "average", "mc-
                quitty", "median", "centroid" (using hclust), "pam" (using wcKMedRange), "di-
                ana" (only for unweighted datasets using diana), "beta.flexible" (only for un-
                weighted datasets using agnes)
pam.combine     Logical. Should we try all combinations of hierarchical and PAM clustering?
x              A clustrangetfamily object to plot or print
object          A clustrangetfamily object to summarize
max.rank        Integer. The different number of solution to print/summarize
stat            One of "stat" or "method". If "stat", plots are grouped by statistics, other-
                 wise by clustering methods.
            Character. The list of statistics to plot or "noCH" to plot all statistics except
            "CH" and "CHsq" or "all" for all statistics. See wcClusterQuality for a list
            of possible values. It is also possible to use "RHC" to plot the quality measure
            1-HC. Unlike HC, RHC should be maximized as all other quality measures.
norm            Character. Normalization method of the statistics can be one of "none" (no
                normalization), "range" (given as (value -min)/(max-min), "zscore" (adjusted by
                mean and standard deviation) or "zscoremed" (adjusted by median and median
                of the difference to the median).
withlegend      Logical. If FALSE, the legend is not plotted.
lwd             Numeric. Line width, see par.
col             A vector of line colors, see par. If NULL, a default set of color is used.
legend.prop     When withlegend=TRUE, sets the proportion of the graphic area used for plot-
                ting the legend. Default value is set according to the place (bottom or right of
                the graphic area) where the legend is plotted. Values from 0 to 1.
rows,cols        optional arguments to arrange plots.
wcKMedoids

K-Medoids or PAM clustering of weighted data.

Description

K-Medoids or PAM clustering of weighted data.

Value

An object of class clustrangefamily with the following elements:

- **Method name**: the results of as.clustrange objects under each method name (see argument method for a list of possible values)
- **allstats**: A matrix containing the clustering statistics for each cluster solution and method.
- **param**: The parameters set when the function was called.

See Also

See Also as.clustrange

Examples

data(mvad)

#Creating state sequence object
mvad.seq <- seqdef(mvad[, 17:86])

# Compute distance using Hamming distance
diss <- seqdist(mvad.seq, method="HAM")

# Ward clustering
allClust <- wcCmpCluster(diss, maxcluster=15, method=c("average", "pam", "beta.flexible"), pam.combine=FALSE)

summary(allClust, max.rank=3)

### Plot PBC, RHC and ASW
plot(allClust, stat=c("PBC", "RHC", "ASW"), norm="zscore", lwd=2)

### Plot PBC, RHC and ASW grouped by cluster method
plot(allClust, group="method", stat=c("PBC", "RHC", "ASW"), norm="zscore", lwd=2)
Usage

wcKMedoids(diss, k, weights=NULL, npass = 1, initialclust=NULL, method="PAMonce", cluster.only = FALSE, debuglevel=0)

Arguments

diss A dissimilarity matrix or a dist object (see dist).
k Integer. The number of cluster.
weights Numeric. Optional numerical vector containing case weights.
npass Integer. Number of random start solution to test.
initialclust An integer vector, a factor, an "hclust" or a "twins" object. Can be either the index of the initial medoids (length should equal to k) or a vector specifying an initial clustering solution (length should then be equal to the number of observation.). If initialclust is an "hclust" or a "twins" object, then the initial clustering solution is taken from the hierarchical clustering in k groups.
method Character. One of "KMedoids", "PAM" or "PAMonce" (default). See details.
cluster.only Logical. If FALSE, the quality of the retained solution is computed.
debuglevel Integer. If greater than zero, print some debugging messages.

Details

K-Medoids algorithms aim at finding the best partition of the data in a k predefined number of groups. Based on a dissimilarity matrix, those algorithms seeks to minimize the (weighted) sum of distance to the medoid of each group. The medoid is defined as the observation that minimize the sum of distance to the other observations of this group. The function wcKMedoids support three different algorithms specified using the method argument:

"KMedoids" Start with a random solution and then iteratively adapt the medoids using an algorithm similar to kmeans. Part of the code is inspired (but completely rewritten) by the C clustering library (see de Hoon et al. 2010). If you use this solution, you should set npass>1 to try several solution.

"PAM" See pam in the cluster library. This code is based on the one available in the cluster library (Maechler et al. 2011). The advantage over the previous method is that it try to minimize a global criteria instead of a local one.

"PAMonce" Same as previous but with two optimizations. First, the optimization presented by Reynolds et al. 2006. Second, only evaluate possible swap if the dissimilarity is greater than zero. This algorithm is used by default.

wcKMedoids works differently according to the diss argument. It may be faster using a matrix but require more memory (since all distances are stored twice). All combination between method and diss argument are possible, except for the "PAM" algorithm were only distance matrix may be used (use the "PAMonce" algorithm instead).

Value

An integer vector with the index of the medoids associated with each observation.
References

See Also
pam in the cluster library, wcClusterQuality, wcKMedRange.

Examples

```r
data(mvad)
## Aggregating state sequence
aggMvad <- wcAggregateCases(mvad[, 17:86], weights=mvad$weight)

## Creating state sequence object
mvad.seq <- seqdef(mvad[aggMvad$aggIndex, 17:86], weights=aggMvad$aggWeights)
## Computing Hamming distance between sequence
diss <- seqdist(mvad.seq, method="HAM")

## K-Medoids
clust5 <- wcKMedoids(diss, k=5, weights=aggMvad$aggWeights)

## clust5$clustering contains index number of each medoids
## Those medoids are unique(clust5$clustering)

## Print the medoids sequences
print(mvad.seq[unique(clust5$clustering), ], informat="SPS")

## Some info about the clustering
print(clust5)

## Plot sequences according to clustering solution.
seqdplot(mvad.seq, group=clust5$clustering)
```

---

**wcKMedRange**

*Compute wcKMedoids clustering for different number of clusters.*

**Description**

Compute **wcKMedoids** clustering for different number of clusters.

**Usage**

```r
wcKMedRange(diss, kvals=NULL, weights=(NULL, R=1, samplesize=NULL, ...)
```
Compute the silhouette of each object using weighted data.

**Arguments**
- **diss**: A dissimilarity matrix or a dist object (see `dist`).
- **kvals**: A numeric vector containing the number of cluster to compute.
- **weights**: Numeric. Optional numerical vector containing case weights.
- **R**: Optional number of bootstrap that can be used to build confidence intervals.
- **samplesize**: Size of bootstrap sample. Default to sum of weights.
- **...**: Additional parameters passed to `wckMedoids`.

**Details**
Compute a clustrange object using the `wckMedoids` method. clustrange objects contain a list of clustering solution with associated statistics and can be used to find the optimal clustering solution. See `as.clustrange` for more details.

**See Also**
See `as.clustrange`.

**Examples**
```r
data(mvad)
## Aggregating state sequence
aggMvad <- wcAggregateCases(mvad[, 17:86], weights=mvad$weight)

## Creating state sequence object
mvad.seq <- seqdef(mvad[aggMvad$aggIndex, 17:86], weights=aggMvad$aggWeights)

## Compute distance using Hamming distance
diss <- seqdist(mvad.seq, method="HAM")

## Pam clustering
pamRange <- wckMedRange(diss, 2:15)

## Plot all statistics (standardized)
plot(pamRange, stat="all", norm="zscoremed", lwd=3)

## Plotting sequences in 3 groups
seqdplot(mvad.seq, group=pamRange$clustering$cluster3)
```
Usage

wcSilhouetteObs(diss, clustering, weights = NULL, measure="ASW")

Arguments

diss A dissimilarity matrix or a dist object (see dist)
clustering Factor. A vector of clustering membership.
weights optional numerical vector containing weights.
measure "ASW" or "ASWw", the measure of the silhouette. See the WeightedCluster vignettes.

Details

See the silhouette function in the cluster package for a detailed explanation of the silhouette.

Value

A numeric vector containing the silhouette of each observation.

References


See Also

See also silhouette.

Examples

data(mvad)
## Aggregating state sequence
aggMvad <- wcAggregateCases(mvad[, 17:86], weights=mvad$weight)

## Creating state sequence object
mvad.seq <- seqdef(mvad[aggMvad$aggIndex, 17:86], weights=aggMvad$aggWeights)
## Computing Hamming distance between sequence
diss <- seqdist(mvad.seq, method="HAM")

## KMedoids using PAMonce method (clustering only)
clust5 <- wcKMedoids(diss, k=5, weights=aggMvad$aggWeights, cluster.only=TRUE)

## Compute the silhouette of each observation
sil <- wcSilhouetteObs(diss, clust5, weights=aggMvad$aggWeights, measure="ASWw")

## If you want to compute the average silhouette width,
## you should take weights into account
weighted.mean(sil, w=aggMvad$aggWeights)
## Plotting sequences ordered by silhouette width, best classified are drawn on the top.
seqIplot(mvad.seq, group=clust5, sortv=sil)
Index

agnes, 3, 4, 9
as.clustrange, 2, 10, 13
as.sectree, 4
cluster, 11, 14
diana, 3, 4, 9
dist, 2, 4, 7, 9, 11, 13, 14
hclust, 3, 4, 9
legend, 2
lines, 10
pam, 11, 12
par, 2, 9
plot.clustrange (as.clustrange), 2
plot.clustrangefamily (wcCmpCluster), 8
print.clustrangefamily (wcCmpCluster), 8
print.wcAggregateCases
  (wcAggregateCases), 6
seqclustname, 5
seqdef, 5
sectree, 4
seqtreedisplay, 4
silhouette, 14
summary.clustrangefamily
  (wcCmpCluster), 8
wcAggregateCases, 6
wcClusterQuality, 2, 3, 7, 9, 12
wcCmpCluster, 8
wcKMedoids, 10, 12, 13
wcKMedRange, 3, 9, 12, 12
wcSilhouetteObs, 13