

Package: yaConsensus (via r-universe)

October 8, 2024

Type Package

Title Consensus Clustering of Omic Data

Version 1.0

Date 2021-06-30

Description Procedures to perform consensus clustering starting from a dissimilarity matrix or a data matrix. It's allowed to select if the subsampling has to be by samples or features. In case of computational heavy load, the procedures can run in parallel.

Depends R (>= 4.1.0), foreach (>= 1.5.1), pheatmap (>= 1.0.12), doParallel (>= 1.0.16)

License MIT + file LICENSE

URL <https://github.com/stefanoMP/yaConsensus>

Repository <https://stefanomp.r-universe.dev>

RemoteUrl <https://github.com/stefanomp/yaconsensus>

RemoteRef HEAD

RemoteSha d9fc70e99665eef26bf15682945cf25bed0db9d6

Contents

consensus.diss	2
plot.yaConsensus	3
yaConsensus	4

Index	8
--------------	----------

consensus.diss *Computes the consensus dissimilarity matrix.*

Description

Computes the consensus dissimilarity according to the algorithm of Monti et al. (2003).

Usage

```
consensus.diss(cclusters, similarity = FALSE)
```

Arguments

`cclusters` a matrix of integers where the column are the samples, and the rows are different clusterings of the samples.

`similarity` a logical value signaling if the similarity matrix is required.

Details

In any row of the ccluster matrix, the value 0 means that the corresponding sample is not assigned to any cluster. In this case, the dissimilarity is computed consistently.

Value

An object of the 'dist' class.

Author(s)

Stefano M. Pagnotta

References

Monti et al. (2003) - Consensus Clustering: A Resampling-Based Method for Class Discovery and Visualization of Gene Expression Microarray Data - Machine Learning 52(1-2):91-118 <DOI: 10.1023/A:1023949509487>

See Also

[dist](#)

Examples

```
clusters <- rep(1:3, c(3, 9, 18))
clusterings <- matrix(NA, ncol = 30, nrow = 50)
for(k in 1:50) clusterings[k,] <- sample(clusters)
ddist <- consensus.diss(clusterings)
class(ddist)
attr(ddist, "method")
```

plot.yaConsensus	<i>Plot the consensus dissimilarity matrix and provide the consensus clustering.</i>
------------------	--

Description

This function processes the output of yaConsensus and acts as a wrapper to the pheatmap function.

Usage

```
## S3 method for class 'yaConsensus'
plot(x, G = 2, annotation = NULL, annotation.colorCode = NULL,
     matching_clustering = NULL, consensus_colors = NULL,...)
```

Arguments

x	an object coming from yaConsensus().
G	an integer value indicating the number of clusters required for the consensus clustering. Default is 2.
annotation	a data frame where the variables are annotations (as labels) of samples. The row-names have to match the names of the samples.
annotation.colorCode	a string named list of color names. The names have to be values stored in the annotation data-frame.
matching_clustering	a string value matching one of the annotation variables in the annotation data-frame. The function tries to match at best the color coding of the selected variable in the data-frame.
consensus_colors	a list of color provided to annotate the consensus clustering. If provided, the matching_clustering is overridden.
...	parameters compatible with pheatmap function.

Details

In the slot 'statistics', the function returns the same statistics of yaConsensus().

Value

A named list with the following slots:

annotation	a data frame. It is the same given in input, with 'consensus' and 'consensus.col' more variables.
ann_colors	a named list of colors associated with each variable in the annotation data-frame.
hclust	an object of hclust class. It's the result of the hclust() applied to the consensus dissimilarity with the complete linkage.
statistics	see Note

Note

In case an 'annotation' is provided, with `summary()`, additional statistics are provided.

The variable specified as given in `summary(..., given = "some clustering")` is assumed as a theoretical clustering, while the consensus clustering is the empirical one. The entropy accuracy, precision, and average (Risso and Pagnotta, 2021) are computed with the former assumptions. `summary()` itself returns all the statistics. "some clustering" has to be one of the column of the annotation `data.frame`.

Author(s)

Stefano M. Pagnotta

References

Risso and Pagnotta (2021) - Per-sample standardization and asymmetric winsorization lead to accurate clustering of RNA-seq expression profiles - Bioinformatics, btob091, <DOI: 10.1093/bioinformatics/btab091>

See Also

[pheatmap](#), [yaConsensus](#)

Examples

```
# see the examples in yaConsensus help.
```

yaConsensus	<i>yaConsensus computes a number of hierarchical clusterings by sampling either samples or features.</i>
-------------	--

Description

This function mainly generates a list of "hclust" objects for downstream analysis.

Usage

```
yaConsensus(ddata, runs = 1000, epsilon = 0.65, is_by_sample = TRUE,
             distMethod = "euclidean", hcMethod = "ward.D2", prefix = NULL)
```

Arguments

ddata	either a data matrix (samples in rows, and features in columns), or a "dist" object.
runs	an integer value for the number of samplings.
epsilon	a real value indicating the sampling rate.
is_by_sample	a logical value indicating if the sampling is by samples (TRUE) or features (FALSE).

distMethod	a character indicating the kind of distance for the inner clustering. It can be any of the methods from the <code>dist</code> function.
hcMethod	a character indicating the linkage method of the inner clustering. It can be any of the methods from the <code>hclust</code> function.
prefix	string specifying a prefix to store the results in a .RData file.

Details

This function can run sequentially or in parallel. In this case, it is necessary to register a cluster of CPUs according to `doParallel` protocol.

To get the consensus clustering, the output of the function has to be processed with the `plot()` function. The consensus dissimilarity follows from the algorithm of Monti et al. (2003). The consensus clustering is from a hierarchical procedure (`hclust`) with "complete" linkage (outer `hc` method).

Value

A named list with the following slots:

distMethod	matches the input
hcMethod	matches the input
lables	a string list with the names of the samples
bySample	matches 'is_by_sample' input parameter
epsilon	matches the input
subsetDimension	actual dimension of the subsets
runs	matches the input
hclust	a list of 'hclust' objects
elapsed_time	time (in seconds) required
ncores	the number of cores used

Note

The `plot` function in the example provides an invisible result with detail and statistics of the experiment.

Author(s)

Stefano M. Pagnotta

References

Monti et al. (2003) - Consensus Clustering: A Resampling-Based Method for Class Discovery and Visualization of Gene Expression Microarray Data - *Machine Learning* 52(1-2):91-118 <DOI: 10.1023/A:1023949509487>

See Also

[dist](#), [hclust](#), [doParallel](#)

Examples

```
## Generate data and annotation
n <- 50; m <- 3000
ddata <- matrix(rnorm(n * m), ncol = m)
ddata[1:20, ] <- ddata[1:20, ] + 0.2
row.names(ddata) <- c(paste0("A", 1:20), paste0("B", 1:30))
ddist <- dist(ddata)

annotation <- data.frame(row.names = rownames(ddata), clust = substr(rownames(ddata), 1, 1))
annotation.colorCode <- c("red", "blue")
names(annotation.colorCode) <- c("A", "B")

##### run in sequential mode
##### sampling the samples ....
(aConsensus <- yaConsensus(ddist))
plot(aConsensus, G = 2)

ans <- plot(aConsensus, G = 2,
           annotation = annotation,
           annotation.colorCode = annotation.colorCode)
summary(ans)
summary(ans, given = "clust")

##### sampling the features ....
(aConsensus <- yaConsensus(ddata, runs = 20, epsilon = 0.2, is_by_sample = FALSE))
ans <- plot(aConsensus, G = 2,
           annotation = annotation,
           annotation.colorCode = annotation.colorCode,
           matching_clustering = "clust")

summary(ans, given = "clust")

##### run in parallel mode
## uncomment to run

# require(doParallel)
# cpu_cluster <- makeCluster(3)
# registerDoParallel(cpu_cluster)

# (aConsensus <- yaConsensus(ddist))
# plot(aConsensus, G = 2)

#stopCluster(cpu_cluster)
```


Index

`consensus.diss`, 2

`dist`, 2, 5, 6

`doParallel`, 6

`hclust`, 5, 6

`pheatmap`, 4

`plot.yaConsensus`, 3

`yaConsensus`, 4, 4