Package 'functClust'

October 13, 2022

Title Functional Clustering of Redundant Components of a System

Version 0.1.6

Date 2020-11-12

URL https://gitlab.com/ecosols/functclust/

Type Package

Author Benoît Jaillard [aut, cre] (<https://orcid.org/0000-0002-5156-1866>),
Philippe Deleporte [aut] (<https://orcid.org/0000-0002-2160-4888>),
Michel Loreau [aut],
Cyrille Violle [aut] (<https://orcid.org/0000-0002-2471-9226>)

Description Cluster together the components that make up an interactive
system on the basis of their functional redundancy
for one or more collective, systemic performances.
Plot the hierarchical tree of component clusters,
the modelled and predicted performances of component assemblages,
and other results associated with a functional clustering.
Test and prioritize the significance of the different components
that make up the interactive system,
of the different assemblages of components that make up the dataset,
and of the different performances observed on the component assemblages.
The method finds application in ecology, for instance,
where the system is an ecosystem, the components are organisms or species,
and the systemic performance is the production of biomass
or the respiration of the ecosystem.
The method is extensively described in
Jaillard B, Deleporte P, Loreau M, Violle C (2018)
``A combinatorial analysis using observational data identifies species
that govern ecosystem functioning''
<doi:10.1371/journal.pone.0201135>.

License GPL-3

Encoding UTF-8

LazyData true

NeedsCompilation no

RoxygenNote 6.1.1

Depends R (>= 3.4.0)

Imports stats, utils, graphics, grDevices, multcompView, clusterCrit

Suggests testthat, rmarkdown, knitr, R.rsp

VignetteBuilder knitr, R.rsp

Collate 'functClust-package.R' 'data.R' 'constants.R' 'stats.R' 'tools.R' 'labelling.R' 'calibrating.R' 'predicting.R' 'clustering.R' 'validating.R' 'validating_loo.R' 'validating_jack.R' 'fclust.R' 'plot_fclust.R' 'test_fclust.R' 'boot_fclust.R' 'fexport.R'

Maintainer Benoît Jaillard <benito.jaillard@wanadoo.fr>

Repository CRAN

Date/Publication 2020-12-02 10:30:02 UTC

R topics documented:

`functClust-package` *Functional Clustering of Redundant Components of a System*

Description

Cluster together the components that make up an interactive system on the basis of their functional redundancy on one or more collective, systemic performances. Plot the hierarchical tree of component clusters, the modelled and predicted performances of component assemblages, and other results associated with a functional clustering. Test and prioritize the significance of the different components that make up the interactive system, of the different assemblages of components that make up the dataset, and of the different performances observed on the component assemblages. The package contains three groups of functions:

`fclust`, `fclust_plot`, `fclust_write` and `fclust_read`, fits a clustering tree of functional components to observed assemblage performances, plots, saves and loads the results obtained using the function `fclust`, respectively;

`ftest`, `ftest_plot`, `ftest_write` and `ftest_read`, tests the significance of the different components, assemblages and performances depending on the option choisen, plots, saves and loads the results obtained using the function `ftest`, respectively;

`fboot`, `fboot_plot`, `fboot_write` and `fboot_read`, evaluates the robustness of a functional clustering by bootstrapping from 1 to (all-1) observations of assemblages or performances depending on the option choisen, plots, saves and loads the results obtained using the function `fboot`, respectively;

The function `fclust` needs to be run first. The functions `ftest` and `fboot` use the result of the function `fclust`. Both the last ones are time-consumming.

Details

None.

Author(s)

Benoît Jaillard (maintainer)

References

Jaillard B., Richon C., Deleporte P., Loreau M. and Violle C. (2018) *An a posteriori species clustering for quantifying the effects of species interactions on ecosystem functioning*. Methods in Ecology and Evolution, 9:704-715. https://doi.org/10.1111/2041-210X.12920.

Jaillard B., Deleporte P., Loreau M. and Violle C. (2018) *A combinatorial analysis using observational data identifies species that govern ecosystem functioning*. PLoS ONE 13(8): e0201135. https://doi.org/10.1371/journal.pone.0201135.

`CedarCreek.2004.2006.boot.assemblages`

Evaluate by bootstrapping the robustness of species clustering to the number of assemblages (plots) used in the cluster analysis.

Description

Species assemblages are randomly removed from the dataset, from 1 to m-1 assemblages (with m = total number of assemblages), a functional clustering of species is built, and the effect of this assemblage remove on species clustering is evaluated by comparing the resulting species clustering with the species clustering obtained with all observed species assemblages. The perturbation is measured by using different clustering indices.

Usage

```
CedarCreek.2004.2006.boot.assemblages
```

Format

A list of matrices, each matrix containing the results for a given clustering index. The indices are "Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index.

Details

None.

`CedarCreek.2004.2006.boot.performances`

Evaluate by bootstrapping the robustness of species clustering to the number of performances (annual biomass) used in the cluster analysis.

Description

Performances are randomly removed from the dataset, from 1 to n-1 performances (with n = total number of performances), a functional clustering of species is built, and the effect of this performance remove on species clustering is evaluated by comparing the resulting species clustering with the species clustering obtained with all systemic performances. The perturbation is measured by using different clustering indices.

Here, the total number of performances is 3 only, then the interest of this process is very limited.

Usage

```
CedarCreek.2004.2006.boot.performances
```

Format

A list of matrices, each matrix containing the results for a given clustering index. The indices are "Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index.

Details

None.

CedarCreek.2004.2006.dat

Data of Cedar Creek experiment for the years 2004, 2005 and 2006

Description

Data from Cedar Creek Ecosystem Science Reserve. The data used here as an example come from the *e120* experiment, more commonly known as *BioDIV* experiment. We have selected the "unsorted biomass" of the harvests of years 2004, 2005 and 2006, and the plots planted without any oak.

Usage

```
CedarCreek.2004.2006.dat
```

Format

A data.frame. Each row corresponds to an *species assemblage*, that is here a plot. The different columns of the data.frame are:

Plot: numero of the plot. Each assemblage must have a specific elemental composition, different from this of other assemblages. This first column is the *identity* of assemblages.

Achmi, Agrsm, Amocan, Andge, Asctu, Elyca, Koecr, Lesca, Liaas, Luppe, Monfi, Panvi, Petpu, Poapr, Schsc, Sornu: abbreviated names of species. The set of species makes up an ecosystem. Each species is informed by `1` if it belongs to the assemblage, and by `0` if it does not belong to the assemblage. The whole is the *matrix of occurrence* of different *components* that belong to, and thus make up, the *interactive system* under consideration.

y2004, y2005, y2006: "unsorted biomass" of plots, *i.e.* species assemblages, harvested in August 2004, July 2005 and June 2006. Each harvest is a *collective, systemic performance* of species assemblages. The performances can be treated separately, or collectively.

Details

The Cedar Creek Ecosystem Science Reserve work was supported by grants from the US National Science Foundation Long-Term Ecological Research Program (LTER) including DEB-0620652 and DEB-1234162. Further support was provided by the Cedar Creek Ecosystem Science Reserve and the University of Minnesota.

Source

https://www.cedarcreek.umn.edu/research/data/dataset?ple120

`CedarCreek.2004.2006.res`

Functional clustering of species used in Cedar Creek experiment for the years 2004, 2005 and 2006

Description

Cedar Creek results obtained for the biomass production for the years 2004, 2005 and 2006. Each year is equally weighted.

Usage

```
CedarCreek.2004.2006.res
```

Format

A hierarchical tree of species clustering:

tree$aff a square-matrix of dimensions species number x species number

tree$cor a vector of coefficient of determination

Details

None.

`CedarCreek.2004.2006.test.assemblages`

Test of significance of species assemblages (plots) used in Cedar Creek experiment for the years 2004, 2005 and 2006

Description

Each species assemblage is successively removed from the data set, a functional clustering of species is built, and the effect of this assemblage remove on species clustering is evaluated by comparing the resulting species clustering with the species clustering obtained with all species assemblages. The perturbation is measured by using different clustering indices.

Usage

```
CedarCreek.2004.2006.test.assemblages
```

Format

A list of matrices, each matrix containing the results for a given clustering index. The indices are "Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index.

Details

None.

CedarCreek.2004.2006.test.components

Test of significance of components (species) used in Cedar Creek experiment for the years 2004, 2005 and 2006

Description

Each component is successively removed from the data set, a functional clustering of remaining species is built, and the effect of this species remove on remaining species clustering is evaluated by comparing the resulting species clustering with the species clustering obtained with all species. The perturbation is measured by using different clustering indices.

Usage

```
CedarCreek.2004.2006.test.components
```

Format

A list of matrices, each matrix containing the results for a given clustering index. The indices are "Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index.

Details

None.

CedarCreek.2004.2006.test.performances

Test of significance of different performances (yearly biomass) used in Cedar Creek experiment for the years 2004, 2005 and 2006

Description

Each performance is successively removed from the data set, a functional clustering of species is built, and the effect of this performance remove on species clustering is evaluated by comparing the resulting species clustering with the species clustering obtained with all systemic performances. The perturbation is measured by using different clustering indices.

Usage

```
CedarCreek.2004.2006.test.performances
```

Format

A list of matrices, each matrix containing the results for a given clustering index. The indices are "Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index.

Details

None.

CedarCreek.2004.res *Functional clustering of species used in Cedar Creek experiment for the year 2004*

Description

Cedar Creek results obtained for the biomass production in the year 2004 only.

Usage

```
CedarCreek.2004.res
```

Format

A hierarchical tree of species clustering:

tree$aff a square-matrix of dimensions species number x species number

tree$cor a vector of coefficient of determination

Details

None.

fboot *Evaluate the robustness of a functional clustering by bootstrapping from 1 to (all-1) observations*

Description

Evaluate by bootstrapping the robustness of a functional clustering to perturbations of data. The perturbed data can be the number of assemblages taken into account, or the number of performances taken into account.

Usage

```
fboot(fres,
      opt.var = c("assemblages", "performances"), nbIter = 1,
      opt.nbMax = fres$nbOpt, opt.R2 = FALSE, opt.plot = FALSE,
      filename = "" )
```

Arguments

fres	an object resulting from a functional clustering obtained using the function fclust.
opt.var	a string, that indicates the variable to test. The option can be "assemblages" or "performances".
nbIter	an integer, that indicates the number of random drawing to do.
opt.nbMax	a logical. If opt.plot = TRUE, the trees resulting from leaving out each performance is plotted.
opt.R2	a logical. If opt.R2 = TRUE, the primary tree is validated and the vectors of coefficient of determination (R^2) and efficiency (E) are computed.
opt.plot	a logical. If opt.plot = TRUE, the primary trees resulting from leaving out each performance are plotted. If opt.R2 = TRUE, the secondary trees resulting from leaving out each performance are plotted.
filename	a string, used as radical for naming the file "filename.components.csv", "filemane.assemblages.csv" or "filemane.performances.csv" according to the dimensions of matrices.

Details

The trees obtained by bootstrapping of performances to omit are compared to the reference tree obtained with all components using different criteria : "Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index. For more informations, see the notice of R-package clusterCrit.

Value

a list of lists, each containing a matrix by clustering index.

References

Package "clusterCrit": Clustering Indices, by Bernard Desgraupes (University of Paris Ouest - Lab Modal'X)

Examples

```
# Enable the comments
oldOption <- getOption("verbose")
if (!oldOption) options(verbose = TRUE)
layout(matrix(c(1,2,3,4), nrow = 2, ncol = 2, byrow = TRUE))



test.boot <- fboot(fres = CedarCreek.2004.2006.res,
                   opt.var  = "performances",
                   nbIter   = 4,
                   opt.plot = TRUE)


layout(1)
options(verbose = oldOption)
```

---

fboot_plot *Plot the robustness of a functional clustering evaluated by bootstrapping from 1 to (all-1) observations*

Description

Evaluate by bootstrapping the robustness of a functional clustering to perturbations of data. The perturbed data can be the number of assemblages taken into account, or the number of performances taken into account.

Usage

```
fboot_plot(fres, lboot, main = "", opt.crit = "Jaccard",
           opt.var  = c("assemblages", "performances"))
```

Arguments

fres	an object resulting from a functional clustering obtained using the function fclust.
lboot	an object resulting from a functional clustering obtained using the function fclust.
main	a string, used as main title for all plots.

opt.crit — an object resulting from a functional clustering obtained using the function fclust.

opt.var — an object resulting from a functional clustering obtained using the function fclust.

Details

The trees obtained by bootstrapping of performances to omit are compared to the reference tree obtained with all components using different criteria : "Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index. For more informations, see the notice of R-package clusterCrit.

Value

a list of lists, each containing a matrix by clustering index.

References

Package "clusterCrit": Clustering Indices, by Bernard Desgraupes (University of Paris Ouest - Lab Modal'X)

Examples

```
# Plot the significance of each component within each components cluster

layout(matrix(c(1,2,3,4), nrow = 2, ncol = 2, byrow = TRUE))
fboot_plot(fres  = CedarCreek.2004.2006.res,
           lboot = CedarCreek.2004.2006.boot.assemblages,
           main  = "BioDIV2",
           opt.var = "assemblages", opt.crit = "Jaccard")
layout(1)
```

---

fboot_read — *Read the robustness of a functional clustering evaluated by bootstrapping from 1 to (all-1) observations*

Description

Read a file of results obtained by a test of significance of functional clustering.

Usage

```
fboot_read(filename,
           opt.var = c("assemblages", "performances"))
```

Arguments

filename	a string, used as radical for naming the file "filename.components.csv", "filemane.assemblages.csv" or "filemane.performances.csv" according to the dimensions of matrices.
opt.var	a string, that indicates the variable to test. The option can be "assemblages" or "performances".

Details

The function fboot , generate a list of lists, each one containing a matrix by clustering index ("Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index). Only their dimensions change according the used functions. Consequently, a same function is used for recording and reading the results of both the test-functions.

Value

a list of list of matrices, identical to this resulting from the function fboot.

Examples

```
# save "rtest" in the file "myRecord.*".

filename <- tempfile(pattern = "myRecord", tmpdir = tempdir())

fboot_write(fres = CedarCreek.2004.2006.res,
            lboot = CedarCreek.2004.2006.boot.performances,
            filename = filename,
            opt.var  = "performances")

lboot <- fboot_read(filename = filename, opt.var = "performances")

all.equal(lboot, CedarCreek.2004.2006.boot.performances)
```

---

fboot_write — *Record the robustness of a functional clustering evaluated by bootstrapping from 1 to (all-1) observations*

Description

Write a file of results obtained by a test of significance of functional clustering.

Usage

```
fboot_write(fres, lboot, filename,
            opt.var = c("assemblages", "performances"))
```

Arguments

fres	an object resulting from a functional clustering obtained using the function fclust.
lboot	a list of list of matrices, generated by the function fboot.
filename	a string, used as radical for naming the file "filename.components.csv", "filemane.assemblages.csv" or "filemane.performances.csv" according to the dimensions of matrices.
opt.var	a string, that indicates the variable to test. The option can be "assemblages" or "performances".

Details

The function fboot , generate a list of lists, each one containing a matrix by clustering index ("Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index). Only their dimensions change according the used functions. Consequently, a same function is used for recording and reading the results of both the test-functions.

Value

Nothing. It is a procedure.

Examples

```
# save "rtest" in the file "myRecord.*".

fboot_write(fres = CedarCreek.2004.2006.res,
            lboot = CedarCreek.2004.2006.boot.performances,
            filename = tempfile(pattern = "myRecord", tmpdir = tempdir()),
            opt.var = "performances")
```

---

fclust *Build a functional clustering for one or more performances*

Description

Fit a primary tree of component clustering to observed assemblage performances, then prune the primary tree for its predicting ability and its parcimony, finally retain a validated secondary tree and the corresponding predictions, statistics and other informations.

Usage

```
fclust(dat, nbElt,
       weight     = rep(1, dim(dat)[2] - nbElt - 1),
       opt.na     = FALSE,
       opt.repeat = FALSE,
       opt.method = "divisive",
       affectElt  = rep(1, nbElt),
       opt.mean   = "amean",
       opt.model  = "byelt",
       opt.jack   = FALSE,   jack = c(3,4) )
```

Arguments

dat	a data.frame or matrix that brings together: a vector of assemblage identity, a matrix of occurrence of components within the system, one or more vectors of observed performances. Consequently, the data.frame or matrix dimensions are: dim(dat)[1]= the number of observed assemblages, * dim(dat)[2]= 1 + number of system components + number of observed performances. On a first line (colnames): assemblage identity, a list of components identified by their names, a list of performances identified by their names. On following lines (a line by assemblage), name of the assemblage (read as character), a sequence of 0 (absence) and 1 (presence of component within each assemblage) (this is the matrix of occurrence of components within the system), a sequence of numeric values for informed each observed performances (this is the set of observed performances).
nbElt	an integer, that specifies the number of components belonging to interactive system. nbElt is used to know the dimension of matrix of occurrence.
weight	a vector of numerics, that specifies the weight of each performance. By default, each performance is equally weighted. If weight is informed, it must have the same length as the number of observed performances.
opt.na	a logical. The records for each assemblage can have NA in matrix of occurrence or in observed assemblage performances. If opt.na = FALSE (by default), an error is returned. If opt.na = TRUE, the records with NA are ignored.
opt.repeat	a logical. in any case, the function looks for different assemblages with identical elemental composition. Messages indicate these identical assemblages. If opt.repeat = FALSE (by default), their performances are averaged. If opt.repeat = TRUE, nothing is done, and the data are processed as they are.
opt.method	a string that specifies the method to use. opt.method = c("divisive", "agglomerative", "apriori"). The three methods generate hierarchical trees. Each tree is complete, running from a unique trunk to as many leaves as components. If opt.method = "divisive", the components are clustered by using a divisive method, from the trivial cluster where all components are together, towards the clustering where each component is a cluster. This method gives the best result for several reasons, exposed in detail in joined vignettes (see "The options of fclust").

If opt.method = "agglomerative", the components are clustered by using an agglomerative method, from the trivial clustering where each component is a cluster, towards the cluster where all components are brought together If all possible assemblages are not observed (that is generally he case in practice), the first clustering of few components can have no effect on convergence criterion, indicing a non-optimum result.

If opt.method = "apriori", the user knows and gives an "a priori" partitioning of the system components he is studying. The partition is arbitrary, in any number of clusters of components, but it must be specified (see following option affectElt). The tree is then built: *(i)* by using opt.method = "divisive" from the defined component clustering towards as many leaves as components; *(ii)* by using opt.method = "agglomerative" from the component clustering towards the trunk of tree.

affectElt — a vector of characters or integers, as long as the number of components nbElt, that indicates the labels of different functional clusters to which each component belongs. Each functional cluster is labelled as a character or an integer, and each component must be identified by its name in names(affectElt). The number of functional clusters defined in affectElt determines an *a priori* level of component clustering (level <- length(unique(affectElt))).

If affectElt = NULL (by default), the option opt.method must be specified. If affectElt is specified, the option opt.method switchs to apriori.

opt.mean — a character, equals to "amean" or "gmean". If opt.mean = "amean", means are computed using an arithmetic formula, if opt.mean = "gmean", mean are computed using a geometric formula.

opt.model — a character equals to "bymot" or "byelt". If opt.model = "bymot", the modelled performances are means of performances of assemblages that share a same assembly motif by including all assemblages that belong to a same assembly motif.

If opt.model = "byelt", the modelled performances are the average of mean performances of assemblages that share a same assembly motif and that contain the same components as the assemblage to predict. This procedure corresponds to a linear model within each assembly motif based on the component occurrence in each assemblage. If no assemblage contains component belonging to assemblage to predict, performance is the mean performance of all assemblages as in opt.model = "bymot".

opt.jack — a logical, that switchs towards cross-validation method.

If opt.jack = FALSE (by default), a Leave-One-Out method is used: predicted performances are computed as the mean of performances of assemblages that share a same assembly motif, experiment by experiment, except the only assemblage to predict.

If opt.jack = TRUE, a jackknife method is used: the set of assemblages belonging to a same assembly motif is divided into jack[2] subsets of jack[1] assemblages. Predicted performances of each subset of jack[1] assemblages are computed, experiment by experiment, by using the other (jack[2] - 1) subsets

of assemblages. If the total number of assemblages belonging to the assembly motif is lower than `jack[1]*jack[2]`, predictions are computed by Leave-One-Out method.

`jack` an integer vector of length 2. The vector specifies the parameters for jackknife method. The first integer `jack[1]` specifies the size of subset, the second integer `jack[2]` specifies the number of subsets.

Details

see Vignette "The options of fclust".

Value

Return a list containing the primary tree of component clustering, predictions of assembly performances and statistics computed by using the primary and secondary trees of component clustering.

Recall of inputs:

- `nbElt`, `nbAss`, `nbXpr`: the number of components that belong to the interactive system, the number of assemblages and the number of performances observed, respectively.
- `opt.method`, `opt.mean`, `opt.model`, `opt.jack`, `jack`, `opt.na`, `opt.repeat`, `affectElt`: the options used for computing the resulting clustering trees, respectively.
- `fobs`, `mOccur`, `xpr`: the vector or matrix of observed performances of assemblages, the binary matrix of occurrence of components, and the vector of weight of different performances, respectively.

Primary and secondary, fitted and validated trees, of component clustering and associated statistics:

- `tree.I`, `tree.II`, `nbOpt`: the primary tree of component clustering, the validated secondary tree of component clustering, and the optimum number of functional clusters, respectively. A tree is a list of a square-matrix of dimensions `nbLev * nbElt` (with `nbLev = nbElt`), and of a vector of coefficient of determination (of length `nbLev`).
- `mCal`, `mPrd`, `tCal`, `tPrd`: the numeric matrix of modelled values, and of values predicted by cross-validation, using the primary tree (`mCal` and (`mPrd`) or the secondary tree (`tCal` and (`tPrd`), respectively. All matrices have the same dimension `nbLev * nbAss`. `rownames` contains the number of component clusters, that is from 1 to `nbElt` clusters. `colnames` contains the names of assemblages.
- `mMotifs`, `tNbcl`: the matrix of affectation of assemblages to different assembly motifs, coded as integers, and the matrices of the last tree levels used for predicting assemblage performances. All matrices have the same dimension `nbLev * nbAss`. `rownames` contains the number of component clusters, that is from 1 to `nbElt` clusters. `colnames` contains the names of assemblages.
- `mStats`, `tStats`: the matrices of associated statistics. `rownames` contains the number of component clusters, that is from 1 to nbElt clusters. `colnames = c("missing", "R2cal", "R2prd", "AIC", "AICc")`.

References

Jaillard, B., Richon, C., Deleporte, P., Loreau, M. and Violle, C. (2018) *An a posteriori species clustering for quantifying the effects of species interactions on ecosystem functioning*. Methods in Ecology and Evolution, 9:704-715. https://doi.org/10.1111/2041-210X.12920.

Jaillard, B., Deleporte, P., Loreau, M. and Violle, C. (2018) *A combinatorial analysis using observational data identifies species that govern ecosystem functioning*. PLoS ONE 13(8): e0201135. https://doi.org/10.1371/journal.pone.0201135.

See Also

fclust: build a functional clustering,
fclust_plot: plot the results of a functional clustering,
fclust_write: save the results of a functional clustering,
fclust_read: read the results of a functional clustering.

Examples

```
# Enable the comments
oldOption <- getOption("verbose")
if (!oldOption) options(verbose = TRUE)

nbElt <- 16   # number of components
# index = Identity, Occurrence of components, a Performance
index <- c(1, 1 + 1:nbElt, 1 + nbElt + 1)
dat.2004 <- CedarCreek.2004.2006.dat[ , index]
res <- fclust(dat.2004, nbElt)
names(res)
res$tree.II

options(verbose = oldOption)
```

`fclust_plot` *Plot various graphs of a functional clustering for one or several performances*

Description

The function plots numerous useful graphs for illustrating results and the ways by which they were obtained: hierarchical trees of component clustering, composition and mean performance of assembly motifs, mean performance of assemblages containing a given components, observed, simulated and predicted performances of assemblages labelled by assembly motif, performances of given assemblages...

Usage

```
fclust_plot(fres, nbcl = 0, main = "",
            opt.tree  = NULL, opt.perf = NULL, opt.ass = NULL,
            opt.motif = NULL, opt.comp = NULL, opt.all = NULL )
```

Arguments

fres	an object generated by the function fclust.
nbcl	an integer. The integer indicates the number of component clusters to take into account. It can be lower than or equals to the optimum number fres$nbOpt of component clusters.
main	a string, that is used as the first, reference part of the title of each graph.
opt.tree	a list, that can include opt.tree = list("cal", "prd", cols, "zoom", window, "all"). This option list manages the plot of primary and secondary trees of component clustering, simplified or not, focussed on the main component clusters or not, coloured by the user or not. The item order in list is any.

- "cal" plots the primary tree of component clustering, from trunk until leaves. At trunk level, when all components are clustered into a large, trivial cluster, the coefficient of determination R2 is low. At the leaves level, when each component is isolated in a singleton, the coefficient of determination is always equal to 1. The primary tree is therefore necessarily over-fitted near the leaves level. The optimum number fres$nbOpt of component clusters is determined by the minimum AICc.
  The blue dashed line indicates the level (optimum number fres$nbOpt of component clusters) where the tree must be optimally cut up. The red solid line indicates the value of tree efficiency E at the nbcl-level. The component clusters are named by lowercase letters, from left to right as "a", "b", "c", ...: the name and content of each component cluster is written on the following page.
- "prd" plots the validated, secondary tree of component clustering, from trunk until validated leaves. Secondary tree is the primary tree cut at the level of the optimal number nbOpt of component clusters. nbOpt is determined by the first lowest value of AIC along the primary tree.
  The red solid line indicates the value of tree efficiency E. R2 and E are stored in fres$tStats. The component clusters are named by lowercase letters, from left to right as "a", "b", "c", ...: the name and content of each component cluster is written on the following page.
- cols is a vector of colours, characters or integers, of same length as the number of components. This option specifies the colour of each component. The components labelled by the same integer have the same colour. If cols is not specified, the components that belong to a same cluster *a posteriori* determined have the same colour. This option is useful when an *a priori* clustering is known, to identify the components *a priori* clustered into the *a posteriori* clustering.
- "zoom" if "cal" or "prd" is checked, this option allows to only plot the first, significant component clusters. The cluster on the far right (the cluster

named by the last letter) is most often a large cluster, that includes many components of which the effects of assemblage performance are not significant. When the number of components is large, the tree is dense and the names of components are confusing. The option is useful to focus on the left, more signficant, part of the primary or secondary tree. If `"zoom"` is checked, `window` must be informed. If not, the function stops with an error message. Note that the large cluster, that includes many components, is always represented by at least one component.

- `window` an integer, that specifies the number of components to plot. `window` must be informed when `"zoom"` is checked. If `window` is higher than the number of components, it is ignored. If `window` is lower than the number of significant components, it is ajusted in such a way that the large cluster, that includes many components, is at least represented by one component.
- `"all"` plots all possible graphs. This option is equivalent to `opt.tree = list("cal", "prd", "zoom", window = 20)`. If the number of components is lower than 20, the option is equivalent to `opt.tree = list("cal", "prd")`.

`opt.perf` a list, that can include `opt.perf = list("stats_I", "stats_II","cal", "prd", "missing", "pub", "calprd","seq", "ass", "aov", pvalue, "all")`. This option list manages the plot of observed, modelled and predicted performances of assemblages, and associated statistics. It also allows to plot performances of some given, identified assemblages. The item order in list is any.

- `"stats_I", "stats_II"`: plot the statistics associated to fit of primary tree that best accounts for observed performances (`"stats_I"`), and of secondary tree that best predicts observed performances of assemblages (`"stats_II"`). Four graphs are plotted: 1. coefficient of determination `R2` and efficiency `E` of models of component clustering (on y-axis) *versus* the number of component clusters (on x-axis); 2. the ratio of assemblage perfomances that cannot be predicted by cross-validation ("predicting ratio"); 3. and 4. the Akaike Information Criterion, corrected `AICc` or not `AIC` for small datasets. The green solid line indicates the first minimum of `AIC` that corresponds to the optimum number `nbOpt` of component clusters to consider.
- `"cal", "prd"`: plot modelled performances *versus* observed performances (`"cal"`, or modelled and predicted by cross-validation performances *versus* observed performances (`"prd"`, for a number of component clusters increasing from `1` until the number of component clusters where efficiency `E` is maximum. Different symbols correspond to different assembly motifs. The prediction error induced by cross-validation is indicated by a short vertical line.

  The blue dashed lines are mean performances. The red solid line is 1:1 bissector line. The number of component clusters is indicated on graph left top. Predicting ratio and coefficient of determination `R2` of the clustering are indicated on graph right bottom. If `"prd"` is checked, efficiency `E` and `E/R2` ratio are added. If `"aov"` is checked, groups significantly different (at a p-value < `pvalue`) are indicated by differents letters on the right of graph.

- "missing": the option "prd" plot modelled and predicted by cross-validation performances *versus* observed performances, using different symbols for different assembly motifs. The option "missing" plot the same data, but in using different symbols according to the clustering model used for predicting the performances of assemblages. This option allows to identify assemblages of which the performance cannot be predicted using the clustering model of the current level. The assemblages are plotted and named using the symbol corresponding to the level of the used clustering model.

  The blue dashed lines are mean performances. The red solid line is 1:1 bissector line. The number of component clusters is indicated on graph left top. Predicting ratio and coefficient of determination of the clustering are indicated on graph right bottom. If "aov" is checked, groups significantly different (at a p-value < pvalue) are indicated by differents letters on the right of graph.
- "pub": the option "prd" plot modelled and predicted by cross-validation performances *versus* observed performances, using different symbols for different assembly motifs. The option "pub" plot the same data, but in using only one symbol. This option is useful for publication.

  The blue dashed lines are mean performances. The red solid line is 1:1 bissector line. The number of component clusters is indicated on graph left top. Predicting ratio and coefficient of determination of the clustering are indicated on graph right bottom. If "aov" is checked, groups significantly different (at a p-value < pvalue) are indicated by differents letters on the right of graph.
- "calprd": plot performances predicted by cross-validation *versus* performances predicted by clustering model ("modelled performances"). This option is useful to identify which assembly motifs become difficult to predict by cross-validation.

  The blue dashed lines are mean performances. The red solid line is 1:1 bissector line. The number of component clusters is indicated on graph left top. Predicting ratio and coefficient of determination of the clustering are indicated on graph right bottom. If "aov" is checked, groups significantly different (at a p-value < pvalue) are indicated by differents letters on the right of graph. The letters are located at mean(Fprd[motif == label]).
- "seq": plot performances of assembly motifs, from 1 to nbMax number of component clusters. Remember that number m of assembly motifs increases with the number nbcl of component clusters (m = 2^nbcl - 1). When the optimal number of component clusters is large, this option is useful to determine a number of component clusters lower than the optimal number of component clusters. Assembly motifs are named as the combinations of component clusters (see "opt.tree").
- "ass" plot the name of each assemblage close to its performance. This option can be used with the options "cal", "prd", "pub" and "calprd". It must be used only if the number of assemblages is small. If the number of assemblages is large, the following option "opt.ass" is more convenient.

- "aov": does a variance analysis of assemblage performances by assembly motifs, and plot the result on the right of graphs. Different letters correspond to groups significantly different at a p-value < pvalue. If "aov" is checked, pvalue must be informed. If not, pvalue = 0.001.
- pvalue: a probability used as threshold in the variance analysis. Then pvalue must be higher than 0 and lower than 1. pvalue must be informed when "aov" is checked. Groups significantly different (at a p-value < pvalue) are then indicated by differents letters on the right of boxplots.
- "all": plot all possible graphs. This option is equivalent to opt.pref = list("cal", "prd", "pub", "calprd", "aov", pvalue = 0.001).

opt.ass — a list, that include opt.ass = list(sample, who). This option plot modelled and predicted by cross-validation performances *versus* observed performances, for a small sample of assemblages randomly drawn (sample), or for given, identified assemblages chosen by the user (who). The item order in list is any.

- sample: an integer. This integer specifies the number of assemblages to randomly drawn in the assemblage set, the plot as the option opt.perf = list("prd"). All chosen assemblages are plotted on a same graph.
- who: a list of assemblage names. The list contains the names of assemblages to plot. Each assemblage is plotted on a specific graph. This option is useful when ssemblage performances are observed over several experiments.

opt.motif — a list, that can include opt.motif = list("obs", "cal", "prd", cols, "hor", "ver", "seq",pvalue, "all"). This option list manages the plot of mean performances of assembly motifs as boxplots, observed, modelled or predicted by cross-validation, horizontally or vertically, sorted by increasing or decreasing mean values, from 1 to nbOpt clusters of components. The item order in list is any.

- "obs", "cal", "prd": plot the observed, modelled or predicted by cross-validation mean performances of assembly motifs as boxplots. Assembly motifs are named as the combinations of component clusters (see "opt.tree"). The coloured squares are the mean performances of assembly motifs. Size (number of observed assemblages) of assembly motifs is indicated on the left of boxplots. The red dashed line is the mean performance of assembly motifs. If "aov" is checked, groups significantly different (at a p-value < pvalue) are indicated by differents letters on the right of boxplots.
- "hor": plot boxplots as horizontal boxes: x-axis corresponds to assemblage performances, and y-axis corresponds to assembly motifs. It "hor" is not checked, boxplots are plotted as vertical boxes: x-axis corresponds to assembly motifs, and y-axis corresponds to assemblage performances. Option "ver" can also be used: "ver" = !"hor".
- "seq": plot mean performances of assembly motifs, from 2 to nbOpt number of component clusters. Remember that number m of assembly motifs increases with the number nbcl of component clusters (m = 2^nbcl - 1). When the optimal number of component clusters is large, this option is

useful to determine a number of component clusters lower than the optimal number of component clusters. Assembly motifs are named as the combinations of component clusters (see "opt.tree").

- pvalue = value: a probability used as threshold in the variance analysis. Then pvalue must be higher than 0 and lower than 1. pvalue must be informed when "aov" is checked. Groups significantly different (at a p-value < pvalue) are then indicated by differents letters on the right of boxplots.
- "all": plot all possible graphs. This option is equivalent to opt.motif = list("obs", "cal", "prd", "seq", "aov", pvalue = 0.001). ¶

opt.comp a list, that can include opt.comp = list("tree", "perf", "hor", "ver", cols,pvalue, "zoom", window, "all"). This option list manages the plot as boxplot of observed mean performances of assemblages that contain a given component, horizontally or vertically, components sorted by increasing or decreasing mean values, or components sorted like the clustering tree. The item order in list is any.

- "tree", "perf": plot the observed mean performances of assemblages that contain a given component as boxplots. Each set of assemblages that contains a given component is named by the contained component. The coloured squares are the mean performances of assemblage sets. Size (number of observed assemblages) of assemblage sets is indicated on the left of boxplots. The red dashed line is the mean performance of assemblage sets. If "aov" is checked, groups significantly different (at a p-value < pvalue) are indicated by differents letters on the right of boxplots.

  If "tree": is checked, mean performances of assemblages that contain a given component are sorted like the clustering tree. If "perf" is checked, mean performances of assemblages that contain a given component are sorted by increasing mean performances.
- "hor": plot boxplots as horizontal boxes: x-axis corresponds to assemblage performances, and y-axis corresponds to assemblage sets. It "hor" is not checked, boxplots are plotted as vertical boxes: x-axis corresponds to assemblage sets, and y-axis corresponds to assemblage performances. Option "ver" can also be used: "ver" = !"hor".
- cols: is a vector of integers, of same length as the number of components. This option specifies the colour of each component. The components labelled by the same integer have the same colour. If cols is not specified, the components that belong to a same cluster *a posteriori* determined have the same colour. This option is useful when an *a priori* clustering is known, to identify the components *a priori* clustered into the *a posteriori* clustering.
- pvalue = value: a probability used as threshold in the variance analysis. Then pvalue must be higher than 0 and lower than 1. pvalue must be informed when "aov" is checked. Groups significantly different (at a p-value < pvalue) are then indicated by differents letters on the right of boxplots.
- "all": plot all possible graphs. This option is equivalent to opt.motif = list("tree", "aov", pvalue = 0.001, "zoom", window = 20).

`opt.all` This option is equivalent to `opt.tree = "all"`, `opt.comp = "all"`, `opt.motif = "all"`,`opt.perf = "all"`. This option is convenient to overview the different options of the function `fclust_plot`.

Details

If all the options are `NULL`, that is `opt.tree = NULL`, `opt.perf = NULL`, `opt.ass = NULL`,`opt.motif = NULL`, `opt.comp = NULL`, `opt.all = NULL`, the function plot the main results, that are: the secondary tree (`opt.tree = "prd"`), assembly motifs as horizontal boxplots (`opt.motif = list("obs", "hor")`)), and modelled and predicted by cross-validation mean performances *versus* observed performances (`opt.perf = "prd"`).

Value

Nothing. It is a procedure.

References

Jaillard, B., Richon, C., Deleporte, P., Loreau, M. and Violle, C. (2018) *An a posteriori species clustering for quantifying the effects of species interactions on ecosystem functioning*. Methods in Ecology and Evolution, 9:704-715. https://doi.org/10.1111/2041-210X.12920.

Jaillard, B., Deleporte, P., Loreau, M. and Violle, C. (2018) *A combinatorial analysis using observational data identifies species that govern ecosystem functioning*. PLoS ONE 13(8): e0201135. https://doi.org/10.1371/journal.pone.0201135.

See Also

`fclust`: make a functional clustering,
`fclust_plot`: plot the results of a functional clustering,
`fclust_write`: save the results of a functional clustering,
`fclust_read`: read the results of a functional clustering.

`plot_ftrees` plot primary and secondary trees resulting from a functional clusterin,
`plot_fperf` plot observed, modelled and predicted performances resulting from a functional clustering,
`plot_fass` plot performances of some given assemblages,
`plot_fmotif` plot as boxplot mean performances of assemblages sorted by assembly motifs,
`plot_fcomp` plot as boxplot mean performances of assemblages containing a given component,
`fclust_plot` plot all possible outputs of a functional clustering.

Examples

```
res <- CedarCreek.2004.res

# plot the hierarchical tree of functionally redundant components
fclust_plot(res, main = "BioDiv2 2004", opt.tree = "prd")
```

```
# plot AIC and AICc versus the number of clusters of components
layout(matrix(c(1,2,3,4), nrow = 2, ncol = 2, byrow = TRUE))
fclust_plot(res, main = "BioDiv2 2004", opt.perf = "stats_II")
layout(1)

# plot the performances modelled and predicted versus observed performances
fclust_plot(res, main = "BioDiv2 2004", opt.perf = "prd")

# plot the performances sorted by assembly motifs
layout(matrix(c(1,2), nrow = 1, ncol = 2, byrow = TRUE))
fclust_plot(res, main = "BioDiv2 2004",
            opt.motif = c("obs", "prd", "hor"))
layout(1)
```

`fclust_read` *Read a functional clustering for one or several performances*

Description

Read the files resulting from a functional clustering and saved in text format in 6 different files by using function `fclust_write()`.

Usage

```
fclust_read(filename = "")
```

Arguments

`filename`	a string, used as radical for the 6 file names.

Details

The results are saved in 5 different files.

- `"filename.options.csv"`: contains `nbElt`, `nbAss`, `nbOpt`, `"opt.method"`, `"opt.mean"`, `"opt.model"`.
- `"filename.inputs.csv"`: contains `fobs` and `names(fobs)`, `xpr` and `names(xpr)`.
- `"filename.trees.csv"`: contains the hierarchical tree `tree$aff` and `tree$cor`.
- `"filename.matrices.csv"`: contains the matrices `mCal`, `mPrd`, `mMotifs`, `tCal`, `tPrd`, and `tNbcl`.
- `"filename.stats.csv"`: contains both statistical matrices `mStats` and `tStats`.

If only a file does not exist or is corrupted, the function is stopped.

Value

The result of the functional clustering recorded in the files `"filename.*.csv"`.

See Also

fclust: make a functional clustering,
fclust_plot: plot the results of a functional clustering,
fclust_write: save the results of a functional clustering,
fclust_read: read the results of a functional clustering.

Examples

```
# save "res" in the files "myRecord.*" then read them again.

res <- CedarCreek.2004.res
filename <- tempfile(pattern = "myRecord", tmpdir = tempdir())

fclust_write(res, filename)
res <- fclust_read(filename)

all.equal(res, CedarCreek.2004.res)
```

---

fclust_write *Record a functional clustering for one or several performances*

Description

Write the results of a functional clustering in text format in 6 different files.

Usage

```
fclust_write(fres = NULL, filename = "")
```

Arguments

fres	a list containing predictions of assembly performances and statistics computed by using a species clustering tree. The list is generated by the function validate_ftree, also called by the function fclust.
filename	a string, used as radical for the 6 file names.

Details

The results are splitted in 5 different files.

- "filename.options.csv": contains nbElt, nbAss, nbXpr, "opt.method", "opt.mean", "opt.model", "opt.jack","jack", "opt.na", "opt.repeat" and "affectElt".
- "filename.inputs.csv": contains fobs, xpr and mOccur.

- "filename.trees.csv": contains the optimum number of functional clusters nbOpt, and the hierarchical trees tree.I and tree.II.
- "filename.matrices.csv": contains the matrices mCal, mPrd, mMotifs, tCal, tPrd, and tNbcl.
- "filename.stats.csv": contains both statistical matrices mStats and tStats.

Value

Nothing. It is a procedure.

See Also

fclust: make a functional clustering,
fclust_plot: plot the results of a functional clustering,
fclust_write: save the results of a functional clustering,
fclust_read: read the results of a functional clustering.

Examples

```
# save "res" in the files "myRecord.*".

res <-  CedarCreek.2004.res
filename <- tempfile(pattern = "myRecord", tmpdir = tempdir())

fclust_write(res, filename)
```

---

ftest *Test the significance of different variables of a functional clustering*

Description

The function allows to test the relative significance of each component, of each assemblage and of each performance on the result of the functional clustering. The method is based on removing one after the other each component, assemblage or performance, then evaluating the effect of these deletions on the functional clustering. Each new functional clustering is compared with the functional clustering obtained with the whole dataset. The process is time-consuming.

Usage

```
ftest(fres,
      opt.var = c("components", "assemblages", "performances"),
      opt.nbMax = fres$nbOpt, opt.R2 = FALSE, opt.plot = FALSE )
```

Arguments

fres	an object resulting from a functional clustering obtained with the whole dataset using the function fclust.
opt.var	a string, that indicates the variable to test. The option can be "components", "assemblages" or "performances".
opt.nbMax	a logical. If opt.plot = TRUE, at each test, the tree resulting from removing each component, assemblage or performance is plotted.
opt.R2	a logical. If opt.R2 = TRUE, the primary tree is validated and the vectors of coefficient of determination (R^2) and efficiency (E) are computed.
opt.plot	a logical. If opt.plot = TRUE, at each test, the tree resulting from removing each component, assemblage or performance is plotted.

Details

None.

Value

a list of matrices, each matrix containing the results for a given clustering index.

Examples

```
# Enable the comments
oldOption <- getOption("verbose")
if (!oldOption) options(verbose = TRUE)
layout(matrix(c(1,2,3,4), nrow = 2, ncol = 2, byrow = TRUE))



# Test the significance of annual biomass production
test.perf <- ftest(fres = CedarCreek.2004.2006.res,
                   opt.var = c("performance"), opt.plot = TRUE)

# Test the significance of each component within each component cluster
test.comp <- ftest(fres = CedarCreek.2004.res,
                   opt.var = c("components"), opt.plot = TRUE)


layout(1)
options(verbose = oldOption)
```

`ftest_plot` *Plot the significance of different variables of a functional clustering*

Description

Different plots are built according to the tested variable.

Usage

```
ftest_plot(fres, rtest,
           main     = "Title",
           opt.var  = c("components", "assemblages", "performances"),
           opt.crit = "Jaccard",
           opt.comp = NULL, opt.ass = NULL, opt.perf = NULL)
```

Arguments

`fres`	an object resulting from a functional clustering obtained with the whole dataset using the function `fclust`.
`rtest`	a list of matrices, each containing the results for a clustering index. `rtest` is an object generated by the function `ftest`.
`main`	a string, that is used as the first, reference part of the title of each graph.
`opt.var`	a string, that indicates the variable to test. The option can be `"components"`, `"assemblages"` or `"performances"`.
`opt.crit`	a list of strings, indicating the clustering indices to plot. The indices can be: "Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" or "Sokal_Sneath2". For more informations, see the notice of R-package `clusterCrit`.
`opt.comp`	a list, that can include `opt.comp = list("all.together", "fgroups.together","comps.together", "comps.byfg", "sorted.tree", "sorted.leg", "all")`. This option list manages the plot of results obtained using the function `ftest` with `opt.var = "components"`. The item order in list is any.

- `"all.together"`, `"fgroups.together"`, `"comps.together"` plot *(i)* the general mean index; *(ii)* the mean indices for each functional group on a same plot; and *(iii)* the mean index for each components on a same plot, when removing one after one each component from the dataset. This allows to evaluate the raw robustness of functional clustering to perturbation of dataset, and the weight of each cluster on functional clustering.
- `"fgroups.byfg"`, `"comps.byfg"` plot *(i)* mean component clusters, functional group by functional group; *(ii)* a graph by component, functional group by functional group; This allows to evaluate the weight of each component on functional clustering.
- `"sorted.tree"`, `"sorted.leg"` plot *(i)* the hierarchical tree of components, with components decreasingly sorted according to their weight on

functional clustering within each functional group; *(ii)* the names of component decreasingly sorted according to their weight on functional clustering within each functional group.

- `"all"` plot all possible graphs. This option is equivalent to `opt.comp = list("all.together", "fgroups.together", "comps.together", "fgroups.byfg", "comps.byfg", "sorted.tree", "sorted.leg")`.

`opt.ass` a list, that can include `opt.ass = list("all.together", "motifs.together","assemblages.togeth "assemblages.bymot", "sorted.leg", "all")`. This option list manages the plot of results obtained using the function `ftest` with `opt.var = "assemblages"`. The item order in list is any.

- `"all.together", "motifs.together", "assemblages.together"` plot *(i)* the general mean index; *(ii)* the mean indices for each assembly motif on a same plot; and *(iii)* the mean index for each assemblages on a same plot, when removing one after one each assemblage from the dataset. This allows to evaluate the raw robustness of functional clustering to perturbation of dataset, and the weight of each assemblage on functional clustering.
- `"motifs.bymot", "assemblages.bymot"` plot *(i)* mean assembly motifs, assembly motif by assembly motif; *(ii)* a graph by removed assemblage, assembly motif by assembly motif; This allows to evaluate the weight of each assemblage on functional clustering.
- `"sorted.leg"` plot the names of assemblages decreasingly sorted according to their weight on functional clustering.
- `"all"` plot all possible graphs. This option is equivalent to `opt.ass = list("all.together", "motifs.together", "assemblages.together", "motifs.bymot", "assemblages.bymot", "sorted.leg")`.

`opt.perf` a list, that can include a list, that can include `opt.comp = list("all.together", "performances.together", "sorted.leg")`. This option list manages the plot of results obtained using the function `ftest` with `opt.var = "performances"`. The item order in list is any.

- `"all.together", "performances.together"` plot *(i)* the general mean index; *(ii)* the mean indices for each removed performance on a same plot, when removing one after one each performance from the dataset. This allows to evaluate the raw robustness of functional clustering to perturbation of dataset, and the weight of each performance on functional clustering.
- `"sorted.leg"` plot the names of performances decreasingly sorted according to their weight on functional clustering.
- `"all"` plot all possible graphs. This option is equivalent to `opt.comp = list("all.together", "performances.together", "sorted.leg")`.

Details

The trees obtained by leaving out each element are compared to the reference tree obtained with all element of the variables using different criteria of clustering: "Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index. For more informations, see the notice of R-package `clusterCrit`.

Value

Nothing. It is a procedure.

References

Package "clusterCrit": Clustering Indices, by Bernard Desgraupes (University of Paris Ouest - Lab Modal'X)

Examples

```
# Plot the hierachical tree of components
layout(matrix(c(1,2,3,4), nrow = 2, ncol = 2, byrow = TRUE))
fclust_plot(fres = CedarCreek.2004.2006.res, main  = "BioDIV2",
            opt.tree = "prd")

# Plot the significance of each component within each components cluster
ftest_plot(fres  = CedarCreek.2004.2006.res,
           rtest = CedarCreek.2004.2006.test.components,
           main  = "BioDIV2",
           opt.var = c("components"), opt.crit = "Jaccard")
layout(1)
```

---

ftest_read *Read the significance of different variables of a functional clustering*

Description

Read a file of results obtained by a test of significance of functional clustering.

Usage

```
ftest_read(filename,
           opt.var = c("components", "assemblages", "performances") )
```

Arguments

filename	a string, used as radical for naming the file "filename.components.csv", "filemane.assemblages.csv" or "filemane.performances.csv" according to the dimensions of matrices.
opt.var	a string, specifying the last part of the file-name. opt can only be equal to "components", "assemblages" or "performances".

Details

The functions ftest_components, ftest_assemblages, ftest_performances, fboot_assemblages and fboot_performances. generate a list containing a matrix by clustering index ("Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index). Only their dimensions change according the used functions. Consequently, a same function is used for recording and reading the results of both the test-functions.

Value

a list of matrices, each containing the results for a clustering index.

Examples

```
# save "res" in the files "myRecord.*" then read it again.

filename <- tempfile(pattern = "myRecord", tmpdir = tempdir())

ftest_write(fres  = CedarCreek.2004.2006.res,
            rtest = CedarCreek.2004.2006.test.components,
            filename = filename,
            opt.var  = "components")

rtest <- ftest_read(filename = filename, opt.var = "components")

all.equal(rtest, CedarCreek.2004.2006.test.components)
```

---

ftest_write *Record the significance of different variables of a functional clustering*

Description

Record in a file the results of a test of significance of functional clustering.

Usage

```
ftest_write(fres, rtest, filename,
            opt.var = c("components", "assemblages", "performances") )
```

Arguments

fres	an object resulting from a functional clustering obtained with the whole dataset using the function fclust.
rtest	a list of matrices, each matrix containing the results for a given clustering index. The object rtest is generated by the function ftest.

filename	a string, used as radical for naming the file "filename.components.csv", "filemane.assemblages.csv or "filemane.performances.csv" according to the dimensions of matrices.
opt.var	a string, specifying the last part of the file-name. opt can only be equal to "components", "assemblages" or "performances".

Details

The functions ftest, ftest_components, ftest_assemblages and ftest_performances generate a list containing a matrix by clustering index ("Czekanowski_Dice", "Folkes_Mallows", "Jaccard", "Kulczynski", "Precision", "Rand", "Recall", "Rogers_Tanimoto", "Russel_Rao", "Sokal_Sneath1" and "Sokal_Sneath2" index). Only their dimensions change according the used functions. Consequently, a same function is used for recording and reading the results of both the test-functions.

Value

Nothing. It is a procedure.

Examples

```
# save "rtest" in the file "myRecord.*".

ftest_write(fres  = CedarCreek.2004.2006.res,
            rtest = CedarCreek.2004.2006.test.components,
            filename = tempfile(pattern = "myRecord", tmpdir = tempdir()),
            opt.var  = "components")
```

Index