Package ‘FactoMineR’

May 7, 2026

Version 2.14

Title Multivariate Exploratory Data Analysis and Data Mining

Maintainer Francois Husson <`francois.husson@institut-agro.fr`>

Depends R (>= 4.3)

Imports
car,cluster,DT,ellipse,emmeans,flashClust,graphics,grDevices,lattice,leaps,MASS,multcompView,scatterplot3d,stats,utils,

Suggests missMDA,knitr,Factoshiny,markdown

Description Exploratory data analysis methods to summarize, visualize and describe datasets. The main principal component methods are available, those with the largest potential in terms of applications: principal component analysis (PCA) when variables are quantitative, correspondence analysis (CA) and multiple correspondence analysis (MCA) when variables are categorical, Multiple Factor Analysis when variables are structured in groups, etc. and hierarchical cluster analysis. F. Husson, S. Le and J. Pages (2017).

License GPL (>= 2)

URL `http://factominer.free.fr`

BugReports `https://github.com/husson/FactoMineR/issues`

Encoding UTF-8

VignetteBuilder knitr

NeedsCompilation yes

Author Francois Husson [aut, cre] (ORCID:
<`https://orcid.org/0000-0002-7271-8877`>),
Julie Josse [aut],
Sebastien Le [aut],
Jeremy Mazet [aut]

Repository CRAN

Date/Publication 2026-04-08 10:50:02 UTC

Contents

FactoMineR-package 4
AovSum 4
autoLab 5
CA 6
CaGalt 8
catdes 10
children 11
coeffRV 12
condes 13
coord.ellipse 14
decathlon 15
descfreq 16
dimdesc 17
DMFA 18
ellipseCA 19
estim_ncp 21
FAMD 22
footsize 23
geomorphology 24
GPA 25
graph.var 26
HCPC 28
health 30
HMFA 31
hobbies 32
JO 33
LinearModel 34
MCA 35
meansComp 38
MFA 39
milk 42
mortality 42
PCA 43
plot.CA 45
plot.CaGalt 48
plot.catdes 50
plot.DMFA 51
plot.FAMD 52
plot.GPA 54
plot.HCPC 55
plot.HMFA 57
plot.MCA 58
plot.meansComp 61
plot.MFA 61
plot.PCA 65
plotellipses 68

plotGPApartial 70
plotMFApartial 71
poison 73
poison.text 73
poulet 74
predict.CA 75
predict.FAMD 75
predict.LinearModel 76
predict.MCA 77
predict.MFA 77
predict.PCA 78
prefpls 79
print.AovSum 80
print.CA 81
print.CaGalt 81
print.catdes 82
print.condes 83
print.FAMD 84
print.GPA 84
print.HCPC 85
print.HMFA 86
print.LinearModel 86
print.MCA 87
print.MFA 88
print.PCA 89
reconst 90
RegBest 91
senso 92
simule 92
summary.CA 93
summary.CaGalt 94
summary.FAMD 95
summary.MCA 96
summary.MFA 97
summary.PCA 98
svd.triplet 99
tab.disjonctif 99
tab.disjonctif.prop 100
tea 100
textual 101
wine 102
write.infile 103

Index **104**

`FactoMineR-package` *Multivariate Exploratory Data Analysis and Data Mining with R*

Description

The method proposed in this package are exploratory mutlivariate methods such as principal component analysis, correspondence analysis or clustering.

Details

FactoMineR is a package for exploratory multivariate data analysis. The package Factoshiny gives an interface to use most of the functions of FactoMineR.

Author(s)

Francois Husson, Julie Josse, Sebastien Le, Jeremy Mazet

Maintainer: `<francois.husson@institut-agro.fr>`

References

Le, S., Josse, J. & Husson, F. (2008). FactoMineR: An R Package for Multivariate Analysis. Journal of Statistical Software. 25(1). pp. 1-18. `https://www.jstatsoft.org/v25/i01/`

A website: `http://factominer.free.fr/`

Some videos: `https://www.youtube.com/playlist?list=PLnZgp6epRBbTsZEFXi_p6W48HhNyqwxIu`

`AovSum` *Analysis of variance with the contrasts sum (the sum of the coefficients is 0)*

Description

Analysis of variance with the contrasts sum (the sum of the coefficients is 0)
Test for all the coefficients
Handle missing values

Usage

```
AovSum(formula, data, na.action = na.omit, ...)
```

Arguments

`formula`	the formula for the model 'y~x1+x2+x1:x2'
`data`	a data-frame
`na.action`	(where relevant) information returned by model.frame on the special handling of NAs.
`...`	other arguments, cf the function `lm`

Value

Retourne des objets

`Ftest`	a table with the F-tests
`Ttest`	a table with the t-tests

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`aov`, `lm`

Examples

```
## Example two-way anova
data(senso)
res <- AovSum(Score~ Product + Day , data=senso)
res

## Example two-way anova with interaction
data(senso)
res2 <- AovSum(Score~ Product + Day + Product : Day, data=senso)
res2

## Example ancova
data(footsize)
res3 <- AovSum(footsize ~ size + sex + size : sex, data=footsize)
res3
```

`autoLab` *Function to better position the labels on the graphs*

Description

Function to better position the labels on the graphs.

Usage

```
autoLab(x, y = NULL, labels = seq(along = x), cex = 1,
                         method = c("SANN", "GA"),
                         allowSmallOverlap = FALSE,
                         trace = FALSE, shadotext = FALSE,
                         doPlot = TRUE, ...)
```

Arguments

`x`	the x-coordinates
`y`	the y-coordinates
`labels`	the labels
`cex`	cex
`method`	not used
`allowSmallOverlap`	boolean
`trace`	boolean
`shadotext`	boolean
`doPlot`	boolean
`...`	further arguments passed to or from other methods

Value

See the text function

`CA` *Correspondence Analysis (CA)*

Description

Performs Correspondence Analysis (CA) including supplementary row and/or column points.

Usage

```
CA(X, ncp = 5, row.sup = NULL, col.sup = NULL,
    quanti.sup=NULL, quali.sup = NULL, graph = TRUE,
axes = c(1,2), row.w = NULL, excl=NULL)
```

Arguments

`X`	a data frame or a table with *n* rows and *p* columns, i.e. a contingency table
`ncp`	number of dimensions kept in the results (by default 5)
`row.sup`	a vector indicating the indexes of the supplementary rows
`col.sup`	a vector indicating the indexes of the supplementary columns
`quanti.sup`	a vector indicating the indexes of the supplementary continuous variables
`quali.sup`	a vector indicating the indexes of the categorical supplementary variables
`graph`	boolean, if TRUE a graph is displayed
`axes`	a length 2 vector specifying the components to plot
`row.w`	an optional row weights (by default, a vector of 1 and each row has a weight equals to its margin); the weights are given only for the active rows
`excl`	numeric vector indicating the indexes of the "junk" columns (default is NULL). Useful for MCA with excl argument.

Value

Returns a list including:

`eig`	a matrix containing all the eigenvalues, the percentage of variance and the cumulative percentage of variance
`col`	a list of matrices with all the results for the column variable (coordinates, square cosine, contributions, inertia)
`row`	a list of matrices with all the results for the row variable (coordinates, square cosine, contributions, inertia)
`col.sup`	a list of matrices containing all the results for the supplementary column points (coordinates, square cosine)
`row.sup`	a list of matrices containing all the results for the supplementary row points (coordinates, square cosine)
`quanti.sup`	if quanti.sup is not NULL, a matrix containing the results for the supplementary continuous variables (coordinates, square cosine)
`quali.sup`	if quali.sup is not NULL, a list of matrices with all the results for the supplementary categorical variables (coordinates of each categories of each variables, v.test which is a criterion with a Normal distribution, square correlation ratio)
`call`	a list with some statistics

Returns the row and column points factor map.
The plot may be improved using the argument autolab, modifying the size of the labels or selecting some elements thanks to the `plot.CA` function.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`,Jeremy Mazet

References

Benzecri, J.-P. (1992) *Correspondence Analysis Handbook*, New-York : Dekker
Benzecri, J.-P. (1980) *L'analyse des donnees tome 2 : l'analyse des correspondances*, Paris : Bordas
Greenacre, M.J. (1993) *Correspondence Analysis in Practice*, London : Academic Press
Husson, F., Le, S. and Pages, J. (2009). Analyse de donnees avec R, *Presses Universitaires de Rennes*.
Husson, F., Le, S. and Pages, J. (2010). Exploratory Multivariate Analysis by Example Using R, *Chapman and Hall*.

See Also

`print.CA`, `summary.CA`, `ellipseCA`, `plot.CA`, `dimdesc`,
Video showing how to perform CA with FactoMineR

Examples

```
data(children)
res.ca <- CA (children, row.sup = 15:18, col.sup = 6:8)
summary(res.ca)
## Ellipses for all the active elements
ellipseCA(res.ca)
## Ellipses around some columns only
ellipseCA(res.ca,ellipse="col",col.col.ell=c(rep("blue",2),rep("transparent",3)),
     invisible=c("row.sup","col.sup"))

## Not run:
## Graphical interface
require(Factoshiny)
res <- Factoshiny(children)

## End(Not run)
```

`CaGalt` *Correspondence Analysis on Generalised Aggregated Lexical Table (CaGalt)*

Description

Correspondence Analysis on Generalised Aggregated Lexical Table (CaGalt) aims at expanding correspondence analysis on an aggregated lexical table to the case of several quantitative and categorical variables with the objective of establishing a typology of the variables and a typology of the frequencies from their mutual relationships. To avoid the instability issued from multicollinearity among the contextual variables and limit the influence of noisy measurements, the contextual variables are substituted by their principal components. Validation tests in the form of confidence ellipses for the frequencies and the variables are also proposed.

Usage

```
CaGalt(Y, X, type="s", conf.ellip=FALSE, nb.ellip=100, level.ventil=0,
  sx=NULL, graph=TRUE, axes=c(1,2))
```

Arguments

`Y`	a data frame with n rows (individuals) and p columns (frequencies)
`X`	a data frame with n rows (individuals) and k columns (quantitative or categorical variables)
`type`	the type of variables: "c" or "s" for quantitative variables and "n" for categorical variables. The difference is that for "s" variables are scaled to unit variance (by default, variables are scaled to unit variance)
`conf.ellip`	boolean (FALSE by default), if TRUE, draw confidence ellipses around the frequencies and the variables when "graph" is TRUE
`nb.ellip`	number of bootstrap samples to compute the confidence ellipses (by default 100)

`level.ventil`	proportion corresponding to the level under which the category is ventilated; by default, 0 and no ventilation is done. Available only when type is equal to "n"
`sx`	number of principal components kept from the principal axes analysis of the contextual variables (by default is NULL and all principal components are kept)
`graph`	boolean, if TRUE a graph is displayed
`axes`	a length 2 vector specifying the components to plot

Value

Returns a list including:

`eig`	a matrix containing all the eigenvalues, the percentage of variance and the cumulative percentage of variance
`ind`	a list of matrices containing all the results for the individuals (coordinates, square cosine)
`freq`	a list of matrices containing all the results for the frequencies (coordinates, square cosine, contributions)
`quanti.var`	a list of matrices containing all the results for the quantitative variables (coordinates, correlation between variables and axes, square cosine)
`quali.var`	a list of matrices containing all the results for the categorical variables (coordinates of each categories of each variables, square cosine)
`ellip`	a list of matrices containing the coordinates of the frequencies and variables for replicated samples from which the confidence ellipses are constructed

Returns the individuals, the frequencies and the variables factor map. If there are more than 50 frequencies, the first 50 frequencies that have the highest contribution on the 2 dimensions of your plot are drawn. The plots may be improved using the argument autolab, modifying the size of the labels or selecting some elements thanks to the plot.CaGalt function.

Author(s)

Belchin Kostov `<badriyan@clinic.ub.es>`, Monica Becue-Bertaut, Francois Husson

References

Becue-Bertaut, M., Pages, J. and Kostov, B. (2014). Untangling the influence of several contextual variables on the respondents'\ lexical choices. A statistical approach.*SORT* Becue-Bertaut, M. and Pages, J. (2014). Correspondence analysis of textual data involving contextual information: Ca-galt on principal components.*Advances in Data Analysis and Classification*

See Also

`print.CaGalt`, `summary.CaGalt`, `plot.CaGalt`

Examples

```
## Not run:
###Example with categorical variables
data(health)
res.cagalt<-CaGalt(Y=health[,1:115],X=health[,116:118],type="n")

## End(Not run)
```

`catdes` *Categories description*

Description

Description of the categories of one factor by categorical variables and/or by quantitative variables

Usage

```
catdes(donnee,num.var,proba = 0.05, row.w = NULL, na.method="NA")
```

Arguments

`donnee`	a data frame made up of at least one categorical variables and a set of quantitative variables and/or categorical variables
`num.var`	the indice of the variable to characterized
`proba`	the significance threshold considered to characterized the category (by default 0.05)
`row.w`	a vector of integers corresponding to an optional row weights (by default, a vector of 1 for uniform row weights)
`na.method`	a boolean that says how to manage missing values. If ; if na.method="NA" a new category is considered for the categorical variable; if na.method="na.omit" the missing values are deleted

Value

Returns a list including:

`test.chi`	The categorical variables which characterized the factor are listed in ascending order (from the one which characterized the most the factor to the one which significantly characterized with the proba `proba`
`category`	description of each category of the `num.var` by each category of all the categorical variables
`quanti.var`	the global description of the `num.var` variable by the quantitative variables with the square correlation coefficient and the p-value of the F-test in a one-way analysis of variance (assuming the hypothesis of homoscedsticity)
`quanti`	the description of each category of the `num.var` variable by the quantitative variables.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

References

Husson, F., Le, S. and Pages, J. (2010). Exploratory Multivariate Analysis by Example Using R, *Chapman and Hall*. Lebart, L., Morineau, A. and Piron, M. (1995) Statistique exploratoire multidimensionnelle, *Dunod*.

See Also

`plot.catdes`, `condes`

Examples

```
data(wine)
catdes(wine, num.var=2)
```

`children` *Children (data)*

Description

The data used here is a contingency table that summarizes the answers given by different categories of people to the following question : according to you, what are the reasons that can make hesitate a woman or a couple to have children?

Usage

```
data(children)
```

Format

A data frame with 18 rows and 8 columns. Rows represent the different reasons mentioned, columns represent the different categories (education, age) people belong to.

Source

Traitements Statistiques des Enquetes (D. Grange, L. Lebart, eds.) Dunod, 1993

Examples

```
data(children)
res.ca <- CA (children, row.sup = 15:18, col.sup = 6:8)
```

`coeffRV` *Calculate the RV coefficient and test its significance*

Description

Calculate the RV coefficient and test its significance.

Usage

```
coeffRV(X, Y)
```

Arguments

`X`	a matrix with *n* rows (individuals) and *p* numerous columns (variables)
`Y`	a matrix with *n* rows (individuals) and *p* numerous columns (variables)

Details

Calculates the RV coefficient between `X` and `Y`. It returns also the standardized RV, the expectation, the variance and the skewness under the permutation distribution. These moments are used to approximate the exact distribution of the RV statistic with the Pearson type III approximation and the p-value associated to this test is given.

Value

A list containing the following components:

`RV`	the RV coefficient between the two matrices
`RVs`	the standardized RV coefficients
`mean`	the mean of the RV permutation distribution
`variance`	the variance of the RV permutation distribution
`skewness`	the skewness of the RV permutation distribution
`p.value`	the p-value associated to the test of the significativity of the RV coefficient (with the Pearson type III approximation

Author(s)

Julie Josse, Francois Husson `<francois.husson@institut-agro.fr>`

References

Escouffier, Y. (1973) *Le traitement des variables vectorielles*. Biometrics **29** 751–760.
Josse, J., Husson, F., Pag\`es, J. (2007) *Testing the significance of the RV coefficient*. Computational Statististics and Data Analysis. **53** 82–91.
Kazi-Aoual, F., Hitier, S., Sabatier, R., Lebreton, J.-D., (1995) Refined approximations to permutations tests for multivariate inference. Computational Statistics and Data Analysis, **20**, 643–656

Examples

```
data(wine)
X <- wine[,3:7]
Y <- wine[,11:20]
coeffRV(X,Y)
```

`condes` *Continuous variable description*

Description

Description continuous by quantitative variables and/or by categorical variables

Usage

```
condes(donnee,num.var,weights=NULL,proba = 0.05)
```

Arguments

`donnee`	a data frame made up of at least one quantitative variable and a set of quantitative variables and/or categorical variables
`num.var`	the number of the variable to characterized
`weights`	weights for the individuals; if NULL, all individuals has a weight equals to 1; the sum of the weights can be equal to 1 and then the weights will be multiplied by the number of individuals, the sum can be greater than the number of individuals
`proba`	the significance threshold considered to characterized the category (by default 0.05)

Value

Returns a list including:

`quanti`	the description of the `num.var` variable by the quantitative variables. The variables are sorted in ascending order (from the one which characterized the most to the one which significantly characterized with the proba `proba`)
`quali`	The categorical variables which characterized the continuous variables are listed in ascending order
`category`	description of the continuous variable `num.var` by each category of all the categorical variables

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`catdes`

Examples

```
data(decathlon)
condes(decathlon, num.var=3)
```

`coord.ellipse` *Construct confidence ellipses*

Description

Construct confidence ellipses

Usage

```
coord.ellipse (coord.simul, centre = NULL, axes = c(1, 2),
    level.conf = 0.95, npoint = 100, bary = FALSE)
```

Arguments

`coord.simul`	a data frame containing the coordinates of the individuals for which the confidence ellipses are constructed. This data frame can contain more than 2 variables; the variables taken into account are chosen after. The first column must be a factor which allows to associate one row to an ellipse. The simule object of the result of the simule function correspond to a data frame.
`centre`	a data frame whose columns are the same than those of the coord.simul, and with the coordinates of the centre of each ellipse. This parameter is optional and NULL by default; in this case, the centre of the ellipses is calculated from the data
`axes`	a length 2 vector specifying the components of coord.simul that are taken into account
`level.conf`	confidence level used to construct the ellipses. By default, 0.95
`npoint`	number of points used to draw the ellipses
`bary`	boolean, if bary = TRUE, the coordinates of the ellipse around the barycentre of individuals are calculated

Value

`res`	a data frame with (npoint times the number of ellipses) rows and three columns. The first column is the factor of coord.simul, the two others columns give the coordinates of the ellipses on the two dimensions chosen.
`call`	the parameters of the function chosen

Author(s)

Jeremy Mazet

See Also

`simule`

Examples

```
data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup = 13,graph=FALSE)
aa <- cbind.data.frame(decathlon[,13],res.pca$ind$coord)
bb <- coord.ellipse(aa,bary=TRUE)
plot(res.pca,habillage=13,ellipse=bb)

## To automatically draw ellipses around the barycentres of all the categorical variables
plotellipses(res.pca)
```

`decathlon` *Performance in decathlon (data)*

Description

The data used here refer to athletes' performance during two sporting events.

Usage

```
data(decathlon)
```

Format

A data frame with 41 rows and 13 columns: the first ten columns corresponds to the performance of the athletes for the 10 events of the decathlon. The columns 11 and 12 correspond respectively to the rank and the points obtained. The last column is a categorical variable corresponding to the sporting event (2004 Olympic Game or 2004 Decastar)

Source

Department of statistics and computer science, Agrocampus Rennes

Examples

```
data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup=13)
```

`descfreq` *Description of frequencies*

Description

Description of the rows of a contingency table or of groups of rows of a contingency table

Usage

```
descfreq(donnee, by.quali = NULL, proba = 0.05)
```

Arguments

`donnee`	a data frame corresponding to a contingency table (quantitative data)
`by.quali`	a factor used to merge the data from different rows of the contingency table; by default NULL and each row is characterized
`proba`	the significance threshold considered to characterized the category (by default 0.05)

Value

Returns a list with the characterization of each rows or each group of the `by.quali`. A test corresponding to the hypergeometric distribution is performed and the probability to observe a more extreme value than the one observed is calculated. For each row (or category), each of the columns characterising the row are sorted in ascending order of p-value.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

References

Lebart, L., Morineau, A. and Piron, M. (1995) Statistique exploratoire multidimensionnelle, *Dunod*.

See Also

`catdes`, `condes`, `textual`

Examples

```
data(children)
descfreq(children[1:14,1:5])    ## desc of rows
descfreq(t(children[1:14,1:5])) ## desc of columns
```

`dimdesc` *Dimension description*

Description

This function is designed to point out the variables and the categories that are the most characteristic according to each dimension obtained by a Factor Analysis.

Usage

```
dimdesc(res, axes = 1:3, proba = 0.05)
```

Arguments

`res`	an object of class PCA, MCA, CA, MFA or HMFA
`axes`	a vector with the dimensions to describe
`proba`	the significance threshold considered to characterized the dimension (by default 0.05)

Value

Returns a list including:

`quanti`	the description of the dimensions by the quantitative variables. The variables are sorted.
`quali`	the description of the dimensions by the categorical variables

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

References

Husson, F., Le, S. and Pages, J. (2010). Exploratory Multivariate Analysis by Example Using R, *Chapman and Hall*.

See Also

`PCA`, `CA`, `MCA`, `MFA`, `HMFA`,
Video showing how to use this function

Examples

```
data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup=13, graph=FALSE)
dimdesc(res.pca)
```

`DMFA` *Dual Multiple Factor Analysis (DMFA)*

Description

Performs Dual Multiple Factor Analysis (DMFA) with supplementary individuals, supplementary quantitative variables and supplementary categorical variables.

Usage

```
DMFA(don, num.fact = ncol(don), scale.unit = TRUE, ncp = 5,
    quanti.sup = NULL, quali.sup = NULL, graph = TRUE, axes=c(1,2))
```

Arguments

`don`	a data frame with *n* rows (individuals) and *p* columns (numeric variables)
`num.fact`	the number of the categorical variable which allows to make the group of individuals
`scale.unit`	a boolean, if TRUE (value set by default) then data are scaled to unit variance
`ncp`	number of dimensions kept in the results (by default 5)
`quanti.sup`	a vector indicating the indexes of the quantitative supplementary variables
`quali.sup`	a vector indicating the indexes of the categorical supplementary variables
`graph`	boolean, if TRUE a graph is displayed
`axes`	a length 2 vector specifying the components to plot

Value

Returns a list including:

`eig`	a matrix containing all the eigenvalues, the percentage of variance and the cumulative percentage of variance
`var`	a list of matrices containing all the results for the active variables (coordinates, correlation between variables and axes, square cosine, contributions)
`ind`	a list of matrices containing all the results for the active individuals (coordinates, square cosine, contributions)
`ind.sup`	a list of matrices containing all the results for the supplementary individuals (coordinates, square cosine)
`quanti.sup`	a list of matrices containing all the results for the supplementary quantitative variables (coordinates, correlation between variables and axes)
`quali.sup`	a list of matrices containing all the results for the supplementary categorical variables (coordinates of each categories of each variables, and v.test which is a criterion with a Normal distribution)
`svd`	the result of the singular value decomposition

`var.partiel`	a list with the partial coordinate of the variables for each group
`cor.dim.gr`	
`Xc`	a list with the data centered by group
`group`	a list with the results for the groups (cordinate, normalized coordinates, cos2)
`Cov`	a list with the covariance matrices for each group

Returns the individuals factor map and the variables factor map.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`plot.DMFA`, `dimdesc`

Examples

```
## Example with the famous Fisher's iris data
res.dmfa = DMFA ( iris, num.fact = 5)
```

`ellipseCA` *Draw confidence ellipses in CA*

Description

Draw confidence ellipses in CA around rows and/or columns.

Usage

```
ellipseCA (x, ellipse=c("col","row"), method="multinomial", nbsample=100,
    axes=c(1,2), xlim=NULL, ylim=NULL, col.row="blue", col.col="red",
col.row.ell=col.row, col.col.ell=col.col,
graph.type = c("ggplot","classic"), ggoptions = NULL, ...)
```

Arguments

`x`	an object of class CA
`ellipse`	a vector of character that defines which ellipses are drawn
`method`	the method to construct ellipses (see details below)
`nbsample`	number of samples drawn to evaluate the stability of the points
`axes`	a length 2 vector specifying the components to plot
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`col.row`	a color for the rows points

`col.col`	a color for columns points
`col.row.ell`	a color for the ellipses of rows points (the color "transparent" can be used if an ellipse should not be drawn)
`col.col.ell`	a color for the ellipses of columns points (the color "transparent" can be used if an ellipse should not be drawn)
`graph.type`	a character that gives the type of graph used: "ggplot" or "classic"
`ggoptions`	a list that gives the graph options when grah.type="ggplot" is used. See the optines and the default values in the details section
`...`	further arguments passed to or from the plot.CA function, such as title, invisible, ...

Details

With `method="multinomial"`, the table X with the active elements is taken as a reference. Then new data tables are drawn in the following way: N (the sum of X) values are drawn from a multinomial distribution with theoretical frequencies equals to the values in the cells divided by N.

With `method="boot"`, the values are bootstrapped row by row: Ni (the sum of row i in the X table) values are taken in a vector with Nij equals to column j (with j varying from 1 to J).

Thus `nbsample` new datasets are drawn and projected as supplementary rows and/or supplementary columns. Then confidence ellipses are drawn for each elements thanks to the `nbsample` supplementary points.

Value

Returns the factor map with the joint plot of CA with ellipses around some elements.

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

References

Lebart, L., Morineau, A. and Piron, M. (1995) Statistique exploratoire multidimensionnelle, *Dunod*.

See Also

`plot.CA`, `CA`

Examples

```
data(children)
res.ca <- CA (children, col.sup = 6:8, row.sup = 15:18)
## Ellipses for all the active elements
ellipseCA(res.ca)
## Ellipses around some columns only
ellipseCA(res.ca,ellipse="col",col.col.ell=c(rep("red",2),rep("transparent",3)),
     invisible=c("row.sup","col.sup"))
```

`estim_ncp` *Estimate the number of components in Principal Component Analysis*

Description

Estimate the number of components in PCA .

Usage

```
estim_ncp(X, ncp.min=0, ncp.max=NULL, scale=TRUE, method="GCV")
```

Arguments

`X`	a data frame with continuous variables
`ncp.min`	minimum number of dimensions to interpret, by default 0
`ncp.max`	maximum number of dimensions to interpret, by default NULL which corresponds to the number of columns minus 2
`scale`	a boolean, if TRUE (value set by default) then data are scaled to unit variance
`method`	method used to estimate the number of components, "GCV" for the generalized cross-validation approximation or "Smooth" for the smoothing method (by default "GCV")

Value

Returns ncp the best number of dimensions to use (find the minimum or the first local minimum) and the mean error for each dimension tested

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>, Julie Josse<`Julie.Josse@agrocampus-ouest.fr`>

References

Josse, J. and Husson, F. (2012). Selecting the number of components in PCA using cross-validation approximations. Computational Statistics and Data Analysis, 56, 1869-1879.

See Also

`PCA`

Examples

```
data(decathlon)
nb.dim <- estim_ncp(decathlon[,1:10],scale=TRUE)
```

`FAMD` *Factor Analysis for Mixed Data*

Description

FAMD is a principal component method dedicated to explore data with both continuous and categorical variables. It can be seen roughly as a mixed between PCA and MCA. More precisely, the continuous variables are scaled to unit variance and the categorical variables are transformed into a disjunctive data table (crisp coding) and then scaled using the specific scaling of MCA. This ensures to balance the influence of both continous and categorical variables in the analysis. It means that both variables are on a equal foot to determine the dimensions of variability. This method allows one to study the similarities between individuals taking into account mixed variables and to study the relationships between all the variables. It also provides graphical outputs such as the representation of the individuals, the correlation circle for the continuous variables and representations of the categories of the categorical variables, and also specific graphs to visulaize the associations between both type of variables.

Usage

```
FAMD (base, ncp = 5, graph = TRUE, sup.var = NULL,
    ind.sup = NULL, axes = c(1,2), row.w = NULL, tab.disj = NULL)
```

Arguments

`base`	a data frame with *n* rows (individuals) and *p* columns
`ncp`	number of dimensions kept in the results (by default 5)
`graph`	boolean, if TRUE a graph is displayed
`ind.sup`	a vector indicating the indexes of the supplementary individuals
`sup.var`	a vector indicating the indexes of the supplementary variables
`axes`	a length 2 vector specifying the components to plot
`row.w`	an optional row weights (by default, uniform row weights); the weights are given only for the active individuals
`tab.disj`	object obtained from the imputeFAMD function of the missMDA package that allows to handle missing values

Value

Returns a list including:

`eig`	a matrix containing all the eigenvalues, the percentage of variance and the cumulative percentage of variance
`var`	a list of matrices containing all the results for the variables considered as group (coordinates, square cosine, contributions)
`ind`	a list of matrices with all the results for the individuals (coordinates, square cosine, contributions)

`quali.var`	a list of matrices with all the results for the categorical variables (coordinates, square cosine, contributions, v.test)
`quanti.var`	a list of matrices with all the results for the quantitative variables (coordinates, correlation, square cosine, contributions)
`call`	a list with some statistics

Returns the individuals factor map.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

References

Pages J. (2004). Analyse factorielle de donnees mixtes. Revue Statistique Appliquee. LII (4). pp. 93-111.

See Also

`print.FAMD`, `summary.FAMD`, `plot.FAMD`,
Video showing how to perform FAMD with FactoMineR

Examples

```
## Not run:
data(geomorphology)
res <- FAMD(geomorphology)
summary(res)

## Graphical interface
require(Factoshiny)
res <- Factoshiny(geomorphology)

### with missing values
require(missMDA)
data(ozone)
res.impute <- imputeFAMD(ozone, ncp=3)
res.afdm <- FAMD(ozone,tab.disj=res.impute$tab.disj)

## End(Not run)
```

`footsize` *footsize*

Description

Dataset for the covariance analysis (a quantitative variable explained by quantitative (continuous) and qualitative (categorical) variables)

Usage

```
data(footsize)
```

Format

Dataset with 84 rows and 3 columns: footsize, size and sex

Examples

```
data(footsize)
res3 <- AovSum (footsize ~ size + sex + size :sex, data=footsize)
res3
```

`geomorphology` *geomorphology(data)*

Description

The data used here concern a geomorphology analysis.

Usage

```
data(geomorphology)
```

Format

A data frame with 75 rows and 11 columns. Rows represent the individuals, columns represent the different questions. 10 variables are quantitative and one variable is qualitative. The dataset is analysed in: http://www.sciencedirect.com/science/article/pii/S0169555X11006362

Examples

```
## Not run:
data(geomorphology)
res <- FAMD(geomorphology)
plot(res,choix="ind",habillage=4)

## End(Not run)
```

`GPA` *Generalised Procrustes Analysis*

Description

Performs Generalised Procrustes Analysis (GPA) that takes into account missing values.

Usage

```
GPA(df, tolerance=10^-10, nbiteration=200, scale=TRUE,
    group, name.group = NULL, graph = TRUE, axes = c(1,2))
```

Arguments

`df`	a data frame with *n* rows (individuals) and *p* columns (quantitative varaibles)
`tolerance`	a threshold with respect to which the algorithm stops, i.e. when the difference between the GPA loss function at step *n* and *n+1* is less than `tolerance`
`nbiteration`	the maximum number of iterations until the algorithm stops
`scale`	a boolean, if TRUE (which is the default value) scaling is required
`group`	a vector indicating the number of variables in each group
`name.group`	a vector indicating the name of the groups (the groups are successively named group.1, group.2 and so on, by default)
`graph`	boolean, if TRUE a graph is displayed
`axes`	a length 2 vector specifying the components to plot

Details

Performs a Generalised Procrustes Analysis (GPA) that takes into account missing values: some data frames of `df` may have non described or non evaluated rows, i.e. rows with missing values only.
The algorithm used here is the one developed by Commandeur.

Value

A list containing the following components:

`RV`	a matrix of RV coefficients between partial configurations
`RVs`	a matrix of standardized RV coefficients between partial configurations
`simi`	a matrix of Procrustes similarity indexes between partial configurations
`scaling`	a vector of isotropic scaling factors
`dep`	an array of initial partial configurations
`consensus`	a matrix of consensus configuration
`Xfin`	an array of partial configurations after transformations

`correlations`	correlation matrix between initial partial configurations and consensus dimensions
`PANOVA`	a list of "Procrustes Analysis of Variance" tables, per assesor (config), per product(objet), per dimension (dimension)

Author(s)

Elisabeth Morand

References

Commandeur, J.J.F (1991) *Matching configurations*.DSWO press, Leiden University.
Dijksterhuis, G. & Punter, P. (1990) Interpreting generalized procrustes analysis "Analysis of Variance" tables, *Food Quality and Preference*, **2**, 255–265
Gower, J.C (1975) Generalized Procrustes analysis, *Psychometrika*, **40**, 33–50
Kazi-Aoual, F., Hitier, S., Sabatier, R., Lebreton, J.-D., (1995) Refined approximations to permutations tests for multivariate inference. Computational Statistics and Data Analysis, **20**, 643–656
Qannari, E.M., MacFie, H.J.H, Courcoux, P. (1999) Performance indices and isotropic scaling factors in sensory profiling, *Food Quality and Preference*, **10**, 17–21

Examples

```
## Not run:
data(wine)
res.gpa <- GPA(wine[,-(1:2)], group=c(5,3,10,9,2),
    name.group=c("olf","vis","olfag","gust","ens"))

### If you want to construct the partial points for some individuals only
plotGPApartial (res.gpa)

## End(Not run)
```

`graph.var` *Make graph of variables*

Description

Plot the graphs of the variables after a Factor Analysis.

Usage

```
graph.var(x, axes = c(1, 2),
    xlim = NULL, ylim = NULL, col.sup = "blue",
    col.var = "black", draw="all", label=draw, lim.cos2.var = 0.1,
    cex = 1, title = NULL, new.plot = TRUE, ...)
```

Arguments

`x`	an object of class PCA, MCA, MFA or HMFA
`axes`	a length 2 vector specifying the components to plot
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`col.sup`	a color for the quantitative supplementary variables
`col.var`	a color for the variables
`draw`	a list of character for the variables which are drawn (by default, all the variables are drawn). You can draw all the active variables by putting "var" and/or all the supplementary variables by putting "quanti.sup" and/or a list with the names of the variables which should be drawn
`label`	a list of character for the variables which are labelled (by default, all the drawn variables are labelled). You can label all the active variables by putting "var" and/or all the supplementary variables by putting "quanti.sup" and/or a list with the names of the variables which should be labelled
`lim.cos2.var`	value of the square cosinus under the variables are not drawn
`cex`	cf. function `par` in the **graphics** package
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`new.plot`	boolean, if TRUE, a new graphical device is created
`...`	further arguments passed to or from other methods

Value

Returns the variables factor map.

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`PCA`, `MFA`, `MCA`, `DMFA`, `HMFA`

Examples

```
data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup = 13, graph = FALSE)
graph.var (res.pca, draw = c("var","Points"),
    label = c("Long.jump", "Points"))
```

`HCPC` *Hierarchical Clustering on Principle Components (HCPC)*

Description

Performs an agglomerative hierarchical clustering on results from a factor analysis. It is possible to cut the tree by clicking at the suggested (or an other) level. Results include paragons, description of the clusters, graphics.

Usage

```
HCPC(res, nb.clust=0, consol=TRUE, iter.max=10, min=3,
  max=NULL, metric="euclidean", method="ward", order=TRUE,
  graph.scale="inertia", nb.par=5, graph=TRUE, proba=0.05,
  cluster.CA="rows",kk=Inf,description=TRUE,...)
```

Arguments

`res`	Either the result of a factor analysis or a dataframe.
`nb.clust`	an integer. If 0, the tree is cut at the level the user clicks on. If -1, the tree is automatically cut at the suggested level (see details). If a (positive) integer, the tree is cut with nb.cluters clusters.
`consol`	a boolean. If TRUE, a k-means consolidation is performed (consolidation cannot be performed if kk is used and equals a number).
`iter.max`	An integer. The maximum number of iterations for the consolidation.
`min`	an integer. The least possible number of clusters suggested.
`max`	an integer. The higher possible number of clusters suggested; by default the minimum between 10 and the number of individuals divided by 2.
`metric`	The metric used to built the tree. See `agnes` for details. Defaults to "euclidean".
`method`	The method used to built the tree. See `agnes` for details. Defaults to "ward".
`order`	A boolean. If TRUE, clusters are ordered following their center coordinate on the first axis.
`graph.scale`	A character string. By default "inertia" and the height of the tree corresponds to the inertia gain, else "sqrt-inertia" the square root of the inertia gain.
`nb.par`	An integer. The number of edited paragons.
`graph`	If TRUE, graphics are displayed. If FALSE, no graph are displayed.
`proba`	The probability used to select axes and variables in catdes (see `catdes` for details.
`cluster.CA`	A string equals to "rows" or "columns" for the clustering of Correspondence Analysis results.

`kk`	An integer corresponding to the number of clusters used in a Kmeans preprocessing before the hierarchical clustering; the top of the hierarchical tree is then constructed from this partition. This is very useful if the number of individuals is high. Note that consolidation cannot be performed if kk is different from Inf and some graphics are not drawn. Inf is used by default and no preprocessing is done, all the graphical outputs are then given.
`description`	boolean; if TRUE the clusters are characterized by the variables and the dimensions
`...`	Other arguments from other methods.

Details

The function first built a hierarchical tree. Then the sum of the within-cluster inertia are calculated for each partition. The suggested partition is the one with the higher relative loss of inertia (i(clusters n+1)/i(cluster n)).

The absolute loss of inertia (i(cluster n)-i(cluster n+1)) is plotted with the tree.

If the ascending clustering is constructed from a data-frame with a lot of rows (individuals), it is possible to first perform a partition with kk clusters and then construct the tree from the (weighted) kk clusters.

Value

Returns a list including:

`data.clust`	The original data with a supplementary column called clust containing the partition.
`desc.var`	The description of the classes by the variables. See `catdes` for details or `descfreq` if clustering is performed on CA results.
`desc.axes`	The description of the classes by the factors (axes). See `catdes` for details.
`call`	A list or parameters and internal objects. `call$t` gives the results for the hierarchical tree; `call$bw.before.consol` and `call$bw.after.consol` give the between inertia before consolidation (i.e. for the clustering obtained from the hierarchical tree) and after the consolidation with Kmeans.
`desc.ind`	The paragons (para) and the more typical individuals of each cluster. See details.

Returns the tree and a barplot of the inertia gains, the individual factor map with the tree (3D), the factor map with individuals coloured by cluster (2D).

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`, Guillaume Le Ray, Quentin Molto

See Also

`plot.HCPC`, `catdes`,
Video showing how to perform clustering with FactoMineR

Examples

```
## Not run:
data(iris)
# Principal Component Analysis:
res.pca <- PCA(iris[,1:4], graph=FALSE)
# Clustering, auto nb of clusters:
hc <- HCPC(res.pca, nb.clust=-1)

### Construct a hierarchical tree from a partition (with 10 clusters)
### (useful when the number of individuals is very important)
hc2 <- HCPC(iris[,1:4], kk=10, nb.clust=-1)

## Graphical interface
require(Factoshiny)
res <- Factoshiny(iris[,1:4])

## End(Not run)
```

`health` *health (data)*

Description

In 1989-1990 the Valencian Institute of Public Health (IVESP) conducted a survey to better know the attitudes and opinions related to health for the non-expert population. The first question included in the questionnaire "What does health mean to you?" required free and spontaneous answers. A priori, the variables Age group (under 21, 21-35, 36-50 and over 50), Health condition (poor, fair, good and very good health) and Gender were considered as possibly conditioning the respondents' viewpoint on health.

Usage

```
data(health)
```

Format

A data frame with 392 rows and 118 columns. Rows represent the individuals (respondents), columns represent the words used at least 10 times to answer the open-ended question (columns 1 to 115) and respondents' characteristics (age, health condition and gender)

Examples

```
## Not run:
data(health)
res.cagalt<-CaGalt(Y=health[,1:115],X=health[,116:118],type="n")

## End(Not run)
```

HMFA *Hierarchical Multiple Factor Analysis*

Description

Performs a hierarchical multiple factor analysis, using an object of class `list` of `data.frame`.

Usage

```
HMFA(X,H,type = rep("s", length(H[[1]])), ncp = 5, graph = TRUE,
    axes = c(1,2), name.group = NULL)
```

Arguments

`X`	a `data.frame`
`H`	a list with one vector for each hierarchical level; in each vector the number of variables or the number of group constituting the group
`type`	the type of variables in each group in the first partition; three possibilities: "c" or "s" for quantitative variables (the difference is that for "s", the variables are scaled in the program), "n" for categorical variables; by default, all the variables are quantitative and the variables are scaled unit
`ncp`	number of dimensions kept in the results (by default 5)
`graph`	boolean, if TRUE a graph is displayed
`axes`	a length 2 vector specifying the components to plot
`name.group`	a list of vector containing the name of the groups for each level of the hierarchy (by default, NULL and the group are named L1.G1, L1.G2 and so on)

Value

Returns a list including:

`eig`	a matrix containing all the eigenvalues, the percentage of variance and the cumulative percentage of variance
`group`	a list with first a list of matrices with the coordinates of the groups for each level and second a matrix with the canonical correlation (correlation between the coordinates of the individuals and the partial points))
`ind`	a list of matrices with all the results for the active individuals (coordinates, square cosine, contributions)
`quanti.var`	a list of matrices with all the results for the quantitative variables (coordinates, correlation between variables and axes)
`quali.var`	a list of matrices with all the results for the supplementary categorical variables (coordinates of each categories of each variables, and v.test which is a criterion with a Normal distribution)
`partial`	a list of arrays with the coordinates of the partial points for each partition

Author(s)

Sebastien Le, Francois Husson <francois.husson@institut-agro.fr>

References

Le Dien, S. & Pages, J. (2003) Hierarchical Multiple factor analysis: application to the comparison of sensory profiles, *Food Quality and Preferences*, **18 (6)**, 453-464.

See Also

`print.HMFA`, `plot.HMFA`, `dimdesc`

Examples

```
data(wine)
hierar <- list(c(2,5,3,10,9,2), c(4,2))
res.hmfa <- HMFA(wine, H = hierar, type=c("n",rep("s",5)))
```

`hobbies` *hobbies (data)*

Description

The data used here concern a questionnaire on hobbies. We asked to 8403 individuals how answer questions about their hobbies (18 questions). The following four variables were used to label the individuals: sex (male, female), age (15-25, 26-35, 36-45, 46-55, 56-65, 66-75, 76-85, 86-100), marital status (single, married, widowed,divorced, remarried), profession (manual labourer, unskilled worker, technician, foreman, senior management, employee, other). And finally, a quantitative variable indicates the number of hobbies practised out of the 18 possible choices.

Usage

```
data(tea)
```

Format

A data frame with 8403 rows and 23 columns. Rows represent the individuals, columns represent the different questions. The first 18 questions are active ones, and the 4 following ones are supplementary categorical variables and the 23th is a supplementary quantitative variable (the number of activities)

Examples

```
data(hobbies)
## Not run:
res.mca <- MCA(hobbies,quali.sup=19:22,quanti.sup=23,method="Burt")
plot(res.mca,invisible=c("ind","quali.sup"),hab="quali") ### active var. only
plot(res.mca,invisible=c("var","quali.sup"),cex=.5,label="none") ### individuals only
plot(res.mca,invisible=c("ind","var"),hab="quali") ### supp. qualitative var. only

dimdesc(res.mca)
plotellipses(res.mca,keepvar=1:4)

## End(Not run)
```

`JO` *Number of medals in athletism during olympic games per country*

Description

This data frame is a contengency table with the athletism events (in row) and the coutries (in columns). Each cell gives the number of medals obtained during the 5 olympis games from 1992 to 2008 (Barcelona 1992, Atlanta 1996, Sydney 2000, Athens 2004, Beijing 2008).

Usage

```
data(JO)
```

Format

A data frame with the 24 events in athletism and in colum the 58 countries who obtained at least on medal

Examples

```
## Not run:
data(JO)
res.ca <- CA(JO)
res.ca <- CA(JO, axes = 3:4)

## End(Not run)
```

`LinearModel` *Linear Model with AIC or BIC selection, and with the contrasts sum (the sum of the coefficients is 0) if any categorical variables*

Description

Linear Model with AIC or BIC selection, and with the contrasts sum (the sum of the coefficients is 0) if any categorical variables
Test for all the coefficients
Handle missing values

Usage

```
LinearModel(formula, data, na.action = na.omit, type = c("III","II", 3, 2),
       selection=c("none","aic","bic"), ...)
```

Arguments

`formula`	the formula for the model 'y~x1+x2+x1:x2'
`data`	a data-frame
`na.action`	(where relevant) information returned by model.frame on the special handling of NAs.
`type`	type of test, "III", "II", 3 or 2. Roman numerals are equivalent to the corresponding Arabic numerals.
`selection`	a string that defines the model selection according to "BIC" for Bayesian Information Criterion or "AIC" for Akaike Information Criterion; "none", by defaut, means that there is no selection.
`...`	other arguments, cf the function `lm`

Details

The Anova function of the package car is used to calculate the F-tests.

The t-tests are obtained using the contrasts "contr.sum" which means that 'sum to zero contrasts'.

A stepwise procedure (using both backword and forward selections) is performed to select a model if selection="AIC" or selection="BIC".

Value

The outouts

`Ftest`	a table with the F-tests
`Ttest`	a table with the t-tests
`lmResult`	the summary of the function lm
`call`	the matched call

`lmResultComp`	the summary of the lm function for the complete model (given only if a selection is performed)
`callComp`	the matched call for the complete model (given only if a selection is performed)

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`aov`, `lm`, `print.LinearModel`

Examples

```
## Example two-way anova
data(senso)
res <- LinearModel(Score~ Product + Day , data=senso, selection="none")
res
## Perform means comparison
meansComp(res,~Product)

## Example two-way anova with interaction
data(senso)
res2 <- LinearModel(Score~ Product + Day + Product : Day, data=senso, selection="none")
res2
meansComp(res,~Product:Day)

## Example two-way anova with selection
data(senso)
res2 <- LinearModel(Score~ Product + Day + Product : Day, data=senso, selection="BIC")
res2

## Example ancova
data(footsize)
res3 <- LinearModel(footsize ~ size + sex + size : sex, data=footsize)
res3
```

`MCA` *Multiple Correspondence Analysis (MCA)*

Description

Performs Multiple Correspondence Analysis (MCA) with supplementary individuals, supplementary quantitative variables and supplementary categorical variables.
Performs also Specific Multiple Correspondence Analysis with supplementary categories and supplementary categorical variables.
Missing values are treated as an additional level, categories which are rare can be ventilated

Usage

```
MCA(X, ncp = 5, ind.sup = NULL, quanti.sup = NULL,
    quali.sup = NULL, excl=NULL, graph = TRUE,
level.ventil = 0, axes = c(1,2), row.w = NULL,
method="Indicator", na.method="NA", tab.disj=NULL)
```

Arguments

`X`	a data frame with *n* rows (individuals) and *p* columns (categorical variables)
`ncp`	number of dimensions kept in the results (by default 5)
`ind.sup`	a vector indicating the indexes of the supplementary individuals
`quanti.sup`	a vector indicating the indexes of the quantitative supplementary variables
`quali.sup`	a vector indicating the indexes of the categorical supplementary variables
`excl`	vector indicating the indexes of the "junk" categories (default is NULL), it can be a vector of the names of the categories or a vector of the indexes in the disjunctive data table
`graph`	boolean, if TRUE a graph is displayed
`level.ventil`	a proportion corresponding to the level under which the category is ventilated; by default, 0 and no ventilation is done
`axes`	a length 2 vector specifying the components to plot
`row.w`	an optional row weights (by default, a vector of 1 for uniform row weights); the weights are given only for the active individuals
`method`	a string corresponding to the name of the method used: "Indicator" (by default) is the CA on the Indicator matrix, "Burt" is the CA on the Burt table. For Burt and the Indicator, the graph of the individuals and the graph of the categories are given
`na.method`	a string corresponding to the name of the method used if there are missing values; available methods are "NA" or "Average" (by default, "NA")
`tab.disj`	optional data.frame corresponding to the disjunctive table used for the analysis; it corresponds to a disjunctive table obtained from imputation method (see package missMDA).

Value

Returns a list including:

`eig`	a matrix containing all the eigenvalues, the percentage of variance and the cumulative percentage of variance
`var`	a list of matrices containing all the results for the active variables (coordinates, square cosine, contributions, v.test, square correlation ratio)
`ind`	a list of matrices containing all the results for the active individuals (coordinates, square cosine, contributions)
`ind.sup`	a list of matrices containing all the results for the supplementary individuals (coordinates, square cosine)

`quanti.sup`	a matrix containing the coordinates of the supplementary quantitative variables (the correlation between a variable and an axis is equal to the variable coordinate on the axis)
`quali.sup`	a list of matrices with all the results for the supplementary categorical variables (coordinates of each categories of each variables, square cosine and v.test which is a criterion with a Normal distribution, square correlation ratio)
`call`	a list with some statistics

Returns the graphs of the individuals and categories and the graph with the variables.
The plots may be improved using the argument autolab, modifying the size of the labels or selecting some elements thanks to the `plot.MCA` function.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`, Julie Josse, Jeremy Mazet

References

Husson, F., Le, S. and Pages, J. (2010). Exploratory Multivariate Analysis by Example Using R, *Chapman and Hall*.

See Also

`plotellipses`, `summary.MCA`,`print.MCA`, `plot.MCA`, `dimdesc`,
Video showing how to perform MCA with FactoMineR

Examples

```
## Not run:
## Tea example
 data(tea)
 res.mca <- MCA(tea,quanti.sup=19,quali.sup=20:36)
 summary(res.mca)
 plot(res.mca,invisible=c("var","quali.sup","quanti.sup"),cex=0.7)
 plot(res.mca,invisible=c("ind","quali.sup","quanti.sup"),cex=0.8)
 plot(res.mca,invisible=c("quali.sup","quanti.sup"),cex=0.8)
 dimdesc(res.mca)
 plotellipses(res.mca,keepvar=1:4)
 plotellipses(res.mca,keepvar="Tea")

## Hobbies example
data(hobbies)
res.mca <- MCA(hobbies,quali.sup=19:22,quanti.sup=23)
plot(res.mca,invisible=c("ind","quali.sup"),hab="quali")
plot(res.mca,invisible=c("var","quali.sup"),cex=.5,label="none")
plot(res.mca,invisible=c("ind","var"),hab="quali")
dimdesc(res.mca)
plotellipses(res.mca,keepvar=1:4)

## Specific MCA: some categories are supplementary
data (poison)
```

```
res <- MCA (poison[,3:8],excl=c(1,3))

## Graphical interface
require(Factoshiny)
res <- Factoshiny(tea)

## Example with missing values : use the missMDA package
require(missMDA)
data(vnf)
completed <- imputeMCA(vnf,ncp=2)
res.mca <- MCA(vnf,tab.disj=completed$tab.disj)

## End(Not run)
```

`meansComp` *Perform pairwise means comparisons*

Description

Perform means comparisons and give groups of means that are not significantly different.

Usage

```
meansComp(object, spec, graph=TRUE, ...)
```

Arguments

`object`	A fitted model object that is supported, such as the result of a call to LinearModel, lm or aov.
`spec`	spec may also be a formula or a list (optionally named) of valid specs. Use of formulas is described in the Overview section below.
`graph`	Boolean; plot the graph to compare the means.
`...`	other arguments, cf the function `emmeans`.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`plot.meansComp`

Examples

```
  data(senso)
  res <- LinearModel(Score~ Product + Day , data=senso, selection="none")
  meansComp(res,~Product)

## Not run:
  ## and with the sidak correction
  meansComp(res,~Product,adjust="sidak")

## End(Not run)
```

`MFA` *Multiple Factor Analysis (MFA)*

Description

Performs Multiple Factor Analysis in the sense of Escofier-Pages with supplementary individuals and supplementary groups of variables. Groups of variables can be quantitative, categorical or contingency tables.
Specific Multiple Fac tor Analysis can be performed using the argument excl.
Missing values in numeric variables are replaced by the column mean.
Missing values in categorical variables are treated as an additional level.

Usage

```
MFA (base, group, type = rep("s",length(group)), excl = NULL,
    ind.sup = NULL, ncp = 5, name.group = NULL,
    num.group.sup = NULL, graph = TRUE, weight.col.mfa = NULL,
    row.w = NULL, axes = c(1,2), tab.comp=NULL)
```

Arguments

`base`	a data frame with *n* rows (individuals) and *p* columns (variables)
`group`	a vector with the number of variables in each group
`type`	the type of variables in each group; four possibilities: "c" or "s" for quantitative variables (the difference is that for "s" variables are scaled to unit variance), "n" for categorical variables "m" for group of mixed variables and "f" for frequencies (from a contingency tables); by default, all variables are quantitative and scaled to unit variance
`excl`	an argument that may possible to exclude categories of active variables of categorical variable groups. NULL by default, it is a list with indexes of categories that are excluded per group
`ind.sup`	a vector indicating the indexes of the supplementary individuals
`ncp`	number of dimensions kept in the results (by default 5)
`name.group`	a vector containing the name of the groups (by default, NULL and the group are named group.1, group.2 and so on)

`num.group.sup`	the indexes of the illustrative groups (by default, NULL and no group are illustrative)
`graph`	boolean, if TRUE a graph is displayed
`weight.col.mfa`	vector of weights, useful for HMFA method (by default, NULL and an MFA is performed)
`row.w`	an optional row weights (by default, a vector of 1 for uniform row weights); the weights are given only for the active individuals
`axes`	a length 2 vector specifying the components to plot
`tab.comp`	object obtained from the imputeMFA function of the missMDA package that allows to handle missing values

Value

`summary.quali`	a summary of the results for the categorical variables
`summary.quanti`	a summary of the results for the quantitative variables
`separate.analyses`	the results for the separate analyses
`eig`	a matrix containing all the eigenvalues, the percentage of variance and the cumulative percentage of variance
`group`	a list of matrices containing all the results for the groups (Lg and RV coefficients, coordinates, square cosine, contributions, distance to the origin, the correlations between each group and each factor)
`rapport.inertie`	inertia ratio
`ind`	a list of matrices containing all the results for the active individuals (coordinates, square cosine, contributions)
`ind.sup`	a list of matrices containing all the results for the supplementary individuals (coordinates, square cosine)
`quanti.var`	a list of matrices containing all the results for the quantitative variables (coordinates, correlation between variables and axes, contribution, cos2)
`quali.var`	a list of matrices containing all the results for categorical variables (coordinates of each categories of each variables, contribution and v.test which is a criterion with a Normal distribution)
`freq`	a list of matrices containing all the results for the frequencies (coordinates, contribution, cos2)
`quanti.var.sup`	a list of matrices containing all the results for the supplementary quantitative variables (coordinates, correlation between variables and axes, cos2)
`quali.var.sup`	a list of matrices containing all the results for the supplementary categorical variables (coordinates of each categories of each variables, cos2 and v.test which is a criterion with a Normal distribution)
`freq.sup`	a list of matrices containing all the results for the supplementary frequencies (coordinates, cos2)
`partial.axes`	a list of matrices containing all the results for the partial axes (coordinates, correlation between variables and axes, correlation between partial axes)

`global.pca`	the result of the analysis when it is considered as a unique weighted PCA

Returns the individuals factor map, the variables factor map and the groups factor map.
The plots may be improved using the argument autolab, modifying the size of the labels or selecting some elements thanks to the `plot.MFA` function.

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>, J. Mazet

References

Escofier, B. and Pages, J. (1994) Multiple Factor Analysis (AFMULT package). *Computational Statistics and Data Analysis*, 18, 121-140.
Becue-Bertaut, M. and Pages, J. (2008) Multiple factor analysis and clustering of a mixture of quantitative, categorical and frequency data. *Computational Statistice and Data Analysis*, 52, 3255-3268.

See Also

`print.MFA`, `summary.MFA`, `plot.MFA`, `dimdesc`,
Video showing how to perform MFA with FactoMineR

Examples

```
## Not run:
data(wine)
res <- MFA(wine, group=c(2,5,3,10,9,2), type=c("n",rep("s",5)),
    ncp=5, name.group=c("orig","olf","vis","olfag","gust","ens"),
    num.group.sup=c(1,6))
summary(res)
barplot(res$eig[,1],main="Eigenvalues",names.arg=1:nrow(res$eig))

#### Confidence ellipses around categories per variable
plotellipses(res)
plotellipses(res,keepvar="Label") ## for 1 variable

#### Interactive graph
liste = plotMFApartial(res)
plot(res,choix="ind",habillage = "Terroir")

###Example with groups of categorical variables
data (poison)
MFA(poison, group=c(2,2,5,6), type=c("s","n","n","n"),
    name.group=c("desc","desc2","symptom","eat"),
    num.group.sup=1:2)

###Example with groups of frequency tables
data(mortality)
res<-MFA(mortality,group=c(9,9),type=c("f","f"),
    name.group=c("1979","2006"))
```

```
## Graphical interface
require(Factoshiny)
res <- Factoshiny(wine)

### with missing values
require(missMDA)
data(orange)
res.impute <- imputeMFA(orange, group=c(5,3), type=rep("s",2),ncp=2)
res.mfa <- MFA(res.impute$completeObs,group=c(5,3),type=rep("s",2))

## End(Not run)
```

`milk` *milk*

Description

Dataset to illustrate the selection of variables in regression

Usage

```
data(milk)
```

Format

Dataset with 85 rows and 6 columns : 85 milks described by the 5 variables: density, fat content, protein, casein, dry, yield

Examples

```
data(milk)
res = RegBest(y=milk[,6],x=milk[,-6])
res$best
```

`mortality` *The cause of mortality in France in 1979 and 2006*

Description

The cause of mortality in France in 1979 and 2006.

Usage

```
data(mortality)
```

Format

A data frame with 62 rows (the different causes of death) and 18 columns. Each column corresponds to an age interval (15-24, 25-34, 35-44, 45-54, 55-64, 65-74, 75-84, 85-94, 95 and more) in a year. The 9 first columns correspond to data in 1979 and the 9 last columns to data in 2006. In each cell, the counts of deaths for a cause of death in an age interval (in a year) is given.

Source

Centre d'epidemiologie sur les causes medicales

Examples

```
data(mortality)

## Not run:
res<-MFA(mortality,group=c(9,9),type=c("f","f"),
    name.group=c("1979","2006"))

plot(res,choix="freq",invisible="ind",graph.type = "classic")
lines(res$freq$coord[1:9,1],res$freq$coord[1:9,2],col="red")
lines(res$freq$coord[10:18,1],res$freq$coord[10:18,2],col="green")

## End(Not run)
```

`PCA` *Principal Component Analysis (PCA)*

Description

Performs Principal Component Analysis (PCA) with supplementary individuals, supplementary quantitative variables and supplementary categorical variables.
Missing values are replaced by the column mean.

Usage

```
PCA(X, scale.unit = TRUE, ncp = 5, ind.sup = NULL,
    quanti.sup = NULL, quali.sup = NULL, row.w = NULL,
    col.w = NULL, graph = TRUE, axes = c(1,2))
```

Arguments

`X`	a data frame with *n* rows (individuals) and *p* columns (numeric variables)
`ncp`	number of dimensions kept in the results (by default 5)
`scale.unit`	a boolean, if TRUE (value set by default) then data are scaled to unit variance
`ind.sup`	a vector indicating the indexes of the supplementary individuals
`quanti.sup`	a vector indicating the indexes of the quantitative supplementary variables

`quali.sup`	a vector indicating the indexes of the categorical supplementary variables
`row.w`	an optional row weights (by default, a vector of 1 for uniform row weights); the weights are given only for the active individuals
`col.w`	an optional column weights (by default, uniform column weights); the weights are given only for the active variables
`graph`	boolean, if TRUE a graph is displayed
`axes`	a length 2 vector specifying the components to plot

Value

Returns a list including:

`eig`	a matrix containing all the eigenvalues, the percentage of variance and the cumulative percentage of variance
`var`	a list of matrices containing all the results for the active variables (coordinates, correlation between variables and axes, square cosine, contributions)
`ind`	a list of matrices containing all the results for the active individuals (coordinates, square cosine, contributions)
`ind.sup`	a list of matrices containing all the results for the supplementary individuals (coordinates, square cosine)
`quanti.sup`	a list of matrices containing all the results for the supplementary quantitative variables (coordinates, correlation between variables and axes)
`quali.sup`	a list of matrices containing all the results for the supplementary categorical variables (coordinates of each categories of each variables, v.test which is a criterion with a Normal distribution, and eta2 which is the square correlation corefficient between a qualitative variable and a dimension)

Returns the individuals factor map and the variables factor map.
The plots may be improved using the argument autolab, modifying the size of the labels or selecting some elements thanks to the `plot.PCA` function.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`, Jeremy Mazet

References

Husson, F., Le, S. and Pages, J. (2010). Exploratory Multivariate Analysis by Example Using R, *Chapman and Hall.*

See Also

`print.PCA`, `summary.PCA`, `plot.PCA`, `dimdesc`,
Video showing how to perform PCA with FactoMineR

Examples

```
data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup=13)
## plot of the eigenvalues
## barplot(res.pca$eig[,1],main="Eigenvalues",names.arg=1:nrow(res.pca$eig))
summary(res.pca)
plot(res.pca,choix="ind",habillage=13)
## Not run:
## To describe the dimensions
dimdesc(res.pca, axes = 1:2)

## To draw ellipses around the categories of the 13th variable (which is categorical)
plotellipses(res.pca,13)

## Graphical interface
require(Factoshiny)
res <- Factoshiny(decathlon)

## Example with missing data
## use package missMDA
require(missMDA)
data(orange)
nb <- estim_ncpPCA(orange,ncp.min=0,ncp.max=5,method.cv="Kfold",nbsim=50)
imputed <- imputePCA(orange,ncp=nb$ncp)
res.pca <- PCA(imputed$completeObs)

## End(Not run)
```

`plot.CA` *Draw the Correspondence Analysis (CA) graphs*

Description

Draw the Correspondence Analysis (CA) graphs.

Usage

```
## S3 method for class 'CA'
plot(x, axes = c(1, 2),
    xlim = NULL, ylim = NULL,
invisible = c("none","row","col","row.sup","col.sup","quali.sup"),
choix = c("CA","quanti.sup"), col.row="blue", col.col="red",
col.row.sup="darkblue", col.col.sup="darkred",
    col.quali.sup="magenta", col.quanti.sup="blue",
label = c("all","none","row","row.sup","col","col.sup","quali.sup","quanti.sup"),
    title = NULL, palette = NULL, autoLab = c("auto","yes","no"),
new.plot=FALSE, selectRow = NULL, selectCol = NULL,
unselect = 0.7, shadowtext = FALSE, habillage = "none",
legend = list(bty = "y", x = "topleft"),
graph.type = c("ggplot","classic"), ggoptions = NULL, ...)
```

Arguments

`x`	an object of class CA
`axes`	a length 2 vector specifying the components to plot
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`invisible`	string indicating if some points should be unlabelled ("row", "col", "row.sup", "col.sup","quali.sup")
`choix`	the graph to plot ("CA" for the CA map, "quanti.sup" for the supplementary quantitative variables)
`col.row`	a color for the rows points
`col.col`	a color for columns points
`col.row.sup`	a color for the supplementary rows points
`col.col.sup`	a color for supplementary columns points
`col.quali.sup`	a color for the supplementary categorical variables
`col.quanti.sup`	a color for the supplementary quantitative variables
`label`	a list of character for the elements which are labelled (by default, all the elements are labelled ("row", "row.sup", "col", "col.sup","quali.sup","quanti.sup")
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=25
`autoLab`	if `autoLab="auto"`, `autoLab` is equal to "yes" if there are less than 50 elements and "no" otherwise; if "yes", the labels of the drawn elements are placed in a "good" way (can be time-consuming if many elements), and if "no" the elements are placed quickly but may overlap
`new.plot`	boolean, if TRUE, a new graphical device is created
`selectRow`	a selection of the rows that are drawn; see the details section
`selectCol`	a selection of the columns that are drawn; see the details section
`unselect`	may be either a value between 0 and 1 that gives the transparency of the unselected objects (if `unselect=1` the transparceny is total and the elements are not drawn, if `unselect=0` the elements are drawn as usual but without any label) or may be a color (for example `unselect="grey60"`)
`shadowtext`	boolean; if true put a shadow on the labels (rectangles are written under the labels which may lead to difficulties to modify the graph with another program)
`habillage`	color the individuals among a categorical variable (give the number of the categorical supplementary variable or its name)
`legend`	a list of arguments that defines the legend if needed (when individuals are drawn according to a variable); see the arguments of the function `legend`
`graph.type`	a character that gives the type of graph used: "ggplot" or "classic"
`ggoptions`	a list that gives the graph options when grah.type="ggplot" is used. See the optines and the default values in the details section
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Details

The argument `autoLab = "yes"` is time-consuming if there are many labels that overlap. In this case, you can modify the size of the characters in order to have less overlapping, using for example cex=0.7.
The `selectRow` and `selectCol` arguments can be used in order to select a part of the elements that are drawn. For example, you can use:
`selectRow = 1:5` and then the rows 1 to 5 are drawn.
`select = c("name1","name5")` and then the rows that have the names name1 and name5 are drawn.
`select = "coord 10"` and then the 10 rows (10 active and 10 supplementaries) that have the highest (squared) coordinates on the 2 chosen dimensions are drawn.
`select = "contrib 10"` and then the 10 rows (10 active) that have the highest contribution on the 2 dimensions of your plot are drawn.
`select = "cos2 5"` and then the 5 rows (5 actives and 5 supplementaries) that have the highest cos2 on the 2 dimensions of your plot are drawn.
`select = "dist 8"` and then the 8 rows (8 actives and 8 supplementaries) that have the highest distance to the center of gravity are drawn.

Value

Returns the factor map with the joint plot of CA.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`CA`

Examples

```
data(children)
res.ca <- CA (children, col.sup = 6:8, row.sup = 15:18)

## select rows and columns that have a cos2 greater than 0.8
plot(res.ca, selectCol="cos2 0.8", selectRow="cos2 0.8")

## Not run:
## You can modify the ggplot graphs as ususal with ggplot2
require(ggplot2)
gr <- plot(res.ca)
gr + theme(panel.grid.major = element_blank(),
   plot.title=element_text(size=14, color="blue"),
   axis.title = element_text(size=12, color="red"))

## End(Not run)
```

`plot.CaGalt` *Draw the Correspondence Analysis on Generalised Aggregated Lexical Table (CaGalt) graphs*

Description

Plot the graphs for a Correspondence Analysis on Generalised Aggregated Lexical Table (CaGalt).

Usage

```
## S3 method for class 'CaGalt'
plot(x, axes = c(1, 2), choix = c("ind", "freq", "quali.var", "quanti.var"),
  conf.ellip = FALSE, contr.ellipse = 3, xlim = NULL, ylim = NULL, col.ind = "black",
   col.freq = "darkred", col.quali = "blue", col.quanti = "blue", label = TRUE,
   lim.cos2.var = 0, title = NULL, palette = NULL,
   autoLab = c("auto", "yes", "no"), new.plot = FALSE, select = NULL,
   unselect = 0.7, shadowtext = FALSE, ...)
```

Arguments

`x`	an object of class CaGalt
`axes`	a length 2 vector specifying the components to plot
`choix`	the graph to plot ("ind" for the individuals, "freq" for the frequencies, "quali.var" for the categorical variables, "quanti.var" for the quantitative variables)
`conf.ellip`	boolean (FALSE by default), if TRUE, draw ellipses around the frequencies and the variables
`contr.ellipse`	the confidence ellipses were drawn for the frequencies with a contribution higher than X times of mean contribution on the 2 dimensions of your plot (by default 3)
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`col.ind`	a color for the individuals (by default "black")
`col.freq`	a color for the frequencies (by default "darkred")
`col.quali`	a color for the categories of categorical variables (by default "blue")
`col.quanti`	a color for the quantitative variables (by default "blue")
`label`	the labels are drawn (by default TRUE)
`lim.cos2.var`	value of the square cosinus under the variables are not drawn
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=2

`autoLab`	if autoLab="auto", autoLab is equal to "yes" if there are less than 50 elements and "no" otherwise; if "yes", the labels of the drawn elements are placed in a "good" way (can be time-consuming if many elements), and if "no" the elements are placed quickly but may overlap
`new.plot`	boolean, if TRUE, a new graphical device is created
`select`	a selection of the elements that are drawn; see the details section
`unselect`	may be either a value between 0 and 1 that gives the transparency of the unselected objects (if unselect=1 the transparency is total and the elements are not drawn, if unselect=0 the elements are drawn as usual but without any label) or may be a color (for example unselect="grey60")
`shadowtext`	boolean; if true put a shadow on the labels (rectangles are written under the labels which may lead to difficulties to modify the graph with another program)
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Details

The argument autoLab = "yes" is time-consuming if there are many labels that overlap. In this case, you can modify the size of the characters in order to have less overlapping, using for example cex=0.7. The select argument can be used in order to select a part of the elements (individuals if you draw the graph of individuals, or variables if you draw the graph of variabless) that are drawn. For example, you can use: select = 1:5 and then the elements 1:5 are drawn. select = c("name1","name5") and then the elements that have the names name1 and name5 are drawn. select = "coord 10" and then the 10 elements that have the highest (squared) coordinates on the 2 chosen dimensions are drawn. select = "contrib 10" and then the 10 elements that have the highest contribution on the 2 dimensions of your plot are drawn (available only when frequencies are drawn). select = "cos2 5" and then the 5 elements that have the highest cos2 on the 2 dimensions of your plot are drawn.

Value

Returns the individuals, the frequencies and the variables factor map.

Author(s)

Belchin Kostov <`badriyan@clinic.ub.es`>, Monica Becue-Bertaut, Francois Husson

See Also

`CaGalt`

Examples

```
## Not run:
data(health)
res.cagalt<-CaGalt(Y=health[,1:115],X=health[,116:118],type="n")
plot(res.cagalt,choix="quali.var",conf.ellip=TRUE,axes=c(1,4))

## Selection of some individuals,categories and frequencies
plot(res.cagalt,choix="freq",col.freq="darkgreen",cex=1.5,select="contrib 10")
```

```
plot(res.cagalt,choix="ind",select="coord 10")
plot(res.cagalt,choix="quali.var",select="cos2 0.5")

## End(Not run)
```

`plot.catdes` *Plots for description of clusters (catdes)*

Description

Plots a graph from a catdes output.

Usage

```
## S3 method for class 'catdes'
plot(x, show="all",output=c("figure","dt") , level=0.01, sort=NULL,
   col.upper="indianred2", col.lower="royalblue1", numchar = 10,
   barplot = FALSE,cex.names=1, ...)
```

Arguments

`x`	A catdes object, see `catdes` for details.
`show`	a string. If "quali", only the categorical variables are used. If "quanti", only the the quantitative variables are used. If "all", both quali and quanti are used. If "quanti.var" is used the characterization of the quantitative variables is given; if "test.chi2" is used the characterization of the qualitative variables is given.
`output`	string: "dt" for a datatable or "figure" for a figure
`level`	a positive float. Indicates a critical value the p-value.
`sort`	NULL (default) or an integer between 1 and the number of clusters or a character (the name of a group). If it is an integer or the name of a group, the features are sorted according to their significances in the construction of the given cluster.
`col.upper`	The color used for under-represented features.
`col.lower`	The color used for over-represented features.
`numchar`	number of characters for the labels
`barplot`	a boolean; if true a barplot per category is drawn, else a table
`cex.names`	the magnification to be used for the names
`...`	further arguments passed to or from other methods

Value

if barplot is true, a barplot is drawn per category with variables that significantly describe the category.
If barplot is false; it returns a grid. The rows stand for the clusters and the columns for the significant variables. A cell colored in col.lower (resp. col.upper) i.e. by default in blue (resp. red) for a quantitative variable means that the average value of the variable in the given cluster is significantly lower (resp. higher) than in the overall data. A cell colored in col.lower (resp. col.upper) for a categorical variable means that the given value of the variable is significantly under-represented (resp. over-represented) in the given cluster than in the overall data. The degree of transparency of the color also indicates the significance of the difference between the behavior of the variable in the given cluster and in the overall data. Indeed, the more transparent the cell is, the less significant the difference is.

Author(s)

Guillaume Le Ray, Camille Chanial, Elise Dumas, Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`catdes`

Examples

```
## Not run:
data(wine)
res.c=catdes(wine, num.var=2)
plot(res.c)

## End(Not run)
```

`plot.DMFA` *Draw the Dual Multiple Factor Analysis (DMFA) graphs*

Description

Plot the graphs for a Principal Component Analysis (DMFA) with supplementary individuals, supplementary quantitative variables and supplementary categorical variables.

Usage

```
## S3 method for class 'DMFA'
plot(x, axes = c(1, 2), choix = "ind", label="all",
    lim.cos2.var = 0., xlim=NULL, ylim=NULL, title = NULL,
    palette = NULL, new.plot = FALSE,
autoLab = c("auto","yes","no"), ...)
```

Arguments

`x`	an object of class DMFA
`axes`	a length 2 vector specifying the components to plot
`choix`	the graph to plot ("ind" for the individuals, "var" for the variables)
`label`	a list of character for the elements which are labelled (by default, all the elements are labelled ("ind", ind.sup", "quali", "var", "quanti.sup"))
`lim.cos2.var`	value of the square cosinus under the variables are not drawn
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=25
`new.plot`	boolean, if TRUE, a new graphical device is created
`autoLab`	if `autoLab="auto"`, `autoLab` is equal to "yes" if there are less than 50 elements and "no" otherwise; if "yes", the labels of the drawn elements are placed in a "good" way (can be time-consuming if many elements), and if "no" the elements are placed quickly but may overlap
`...`	further arguments passed to or from other methods

Value

Returns the individuals factor map and the variables factor map, the partial variables representation and the groups factor map.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`DMFA`

`plot.FAMD` *Draw the Multiple Factor Analysis for Mixt Data graphs*

Description

It provides the graphical outputs associated with the principal component method for mixed data: FAMD.

Usage

```
## S3 method for class 'FAMD'
plot(x, choix = c(”ind”,”var”,”quanti”,”quali”), axes = c(1, 2),
    lab.var = TRUE, lab.ind = TRUE, habillage = ”none”, col.lab = FALSE,
    col.hab = NULL, invisible = NULL, lim.cos2.var = 0., xlim = NULL,
    ylim = NULL, title = NULL, palette=NULL, autoLab = c(”auto”,”yes”,”no”),
new.plot = FALSE, select = NULL, unselect = 0.7, shadowtext = FALSE,
legend = list(bty = ”y”, x = ”topleft”),
graph.type = c(”ggplot”,”classic”), ggoptions = NULL, ...)
```

Arguments

`x`	an object of class FAMD
`choix`	a string corresponding to the graph that you want to do ("ind" for the individual or categorical variables graph, "var" for all the variables (quantitative and categorical), "quanti" for the correlation circle)
`axes`	a length 2 vector specifying the components to plot
`lab.var`	boolean indicating if the labelled of the variables should be drawn on the map
`lab.ind`	boolean indicating if the labelled of the individuals should be drawn on the map
`habillage`	string corresponding to the color which are used. If "ind", one color is used for each individual else if it is the name or the position of a categorical variable, it colors according to the different categories of this variable
`col.lab`	boolean indicating if the labelled should be colored
`col.hab`	vector indicating the colors to use to labelled the rows or columns elements chosen in habillage
`invisible`	list of string; for choix ="ind", the individuals can be omit (invisible = "ind"), or supplementary individuals (invisible="ind.sup") or the centerg of gravity of the categorical variables (invisible= "quali"); if invisible = c("ind","ind.sup"), just the centers of gravity are drawn
`lim.cos2.var`	value of the square cosinus under the variables are not drawn
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=25
`autoLab`	if `autoLab="auto"`, `autoLab` is equal to "yes" if there are less than 50 elements and "no" otherwise; if "yes", the labels of the drawn elements are placed in a "good" way (can be time-consuming if many elements), and if "no" the elements are placed quickly but may overlap
`new.plot`	boolean, if TRUE, a new graphical device is created
`select`	a selection of the elements that are drawn; see the details section

`unselect`	may be either a value between 0 and 1 that gives the transparency of the unselected objects (if `unselect=1` the transparceny is total and the elements are not drawn, if `unselect=0` the elements are drawn as usual but without any label) or may be a color (for example `unselect="grey60"`)
`shadowtext`	boolean; if true put a shadow on the labels (rectangles are written under the labels which may lead to difficulties to modify the graph with another program)
`legend`	a list of arguments that defines the legend if needed (when individuals are drawn according to a variable); see the arguments of the function `legend`
`graph.type`	a character that gives the type of graph used: "ggplot" or "classic"
`ggoptions`	a list that gives the graph options when grah.type="ggplot" is used. See the optines and the default values in the details section
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Value

Returns the individuals factor map and the variables factor map.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`FAMD`

Examples

```
## Not run:
data(geomorphology)
res <- FAMD(geomorphology)
plot(res,choix="ind",habillage=4)

## End(Not run)
```

`plot.GPA` *Draw the General Procrustes Analysis (GPA) map*

Description

Draw the General Procrustes Analysis (GPA) map.

Usage

```
## S3 method for class 'GPA'
plot(x, axes = c(1, 2),
    lab.ind.moy = TRUE, habillage = "ind",
    partial = "all", chrono = FALSE, xlim = NULL, ylim = NULL,
    cex = 1, title = NULL, palette = NULL, ...)
```

Arguments

`x`	an object of class GPA
`axes`	a length 2 vector specifying the components to plot
`lab.ind.moy`	boolean, if TRUE, the label of the mean points are drawn
`habillage`	string corresponding to the color which are used. If "ind", one color is used for each individual; if "group" the individuals are colored according to the group
`partial`	list of the individuals or of the center of gravity for which the partial points should be drawn (by default, partial = "none" and no partial points are drawn)
`chrono`	boolean, if TRUE, the partial points of a same point are linked (useful when groups correspond to different moment)
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`cex`	cf. function `par` in the **graphics** package
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=2
`...`	further arguments passed to or from other methods

Value

Returns the General Procrustes Analysis map.

Author(s)

Elisabeth Morand, Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`GPA`

`plot.HCPC` *Plots for Hierarchical Classification on Principle Components (HCPC) results*

Description

Plots graphs from a HCPC result: tree, barplot of inertia gains and first factor map with or without the tree, in 2 or 3 dimensions.

Usage

```
## S3 method for class 'HCPC'
plot(x, axes=c(1,2), choice="3D.map", rect=TRUE,
  draw.tree=TRUE, ind.names=TRUE, t.level="all", title=NULL,
  new.plot=FALSE, max.plot=15, tree.barplot=TRUE,
  centers.plot=FALSE, ...)
```

Arguments

`x`	A HCPC object, see HCPC for details.
`axes`	a two integers vector.Defines the axes of the factor map to plot.
`choice`	A string. "tree" plots the tree. "bar" plots bars of inertia gains. "map" plots a factor map, individuals colored by cluster. "3D.map" plots the same factor map, individuals colored by cluster, the tree above.
`rect`	a boolean. If TRUE, rectangles are drawn around clusters if choice ="tree".
`tree.barplot`	a boolean. If TRUE, the barplot of intra inertia losses is added on the tree graph.
`draw.tree`	A boolean. If TRUE, the tree is projected on the factor map if choice ="map".
`ind.names`	A boolean. If TRUE, the individuals names are added on the factor map when choice="3D.map" or choice="map"
`t.level`	Either a positive integer or a string. A positive integer indicates the starting level to plot the tree on the map when draw.tree=TRUE. If "all", the whole tree is ploted. If "centers", it draws the tree starting t the centers of the clusters.
`title`	a string. Title of the graph. NULL by default and a title is automatically defined
`centers.plot`	a boolean. If TRUE, the centers of clusters are drawn on the 3D factor maps.
`new.plot`	a boolean. If TRUE, the plot is done in a new window.
`max.plot`	The max for the bar plot
`...`	Other arguments from other methods.

Value

Returns the chosen plot.

Author(s)

Guillaume Le Ray, Quentin Molto, Francois Husson <francois.husson@institut-agro.fr>

See Also

HCPC

Examples

```
data(iris)
# Clustering, auto nb of clusters:
res.hcpc=HCPC(iris[1:4], nb.clust=3)
# 3D graph from a different point of view:
plot(res.hcpc, choice="3D.map", angle=60)
```

`plot.HMFA`	*Draw the Hierarchical Multiple Factor Analysis (HMFA) graphs*

Description

Draw the Hierarchical Multiple Factor Analysis (HMFA) graphs

Usage

```
## S3 method for class 'HMFA'
plot(x, axes = c(1,2),num=6, choix = "ind",
    lab.grpe = TRUE, lab.var = TRUE, lab.ind.moy = TRUE,
    invisible = NULL, lim.cos2.var = 0.,
    xlim = NULL, ylim = NULL, cex = 1, title = NULL, new.plot = FALSE, ...)
```

Arguments

`x`	an object of class HMFA
`axes`	a length 2 vector specifying the components to plot
`num`	number of grpahs in a same windows
`choix`	a string corresponding to the graph that you want to do ("ind" for the individual or categorical variables graph, "var" for the quantitative variables graph, "axes" for the graph of the partial axes, "group" for the groups representation)
`lab.grpe`	boolean, if TRUE, the label of the groups are drawn
`lab.var`	boolean, if TRUE, the label of the variables are drawn
`lab.ind.moy`	boolean, if TRUE, the label of the mean points are drawn
`invisible`	list of string; for choix ="ind", the individuals can be omit (invisible = "ind"), or the centers of gravity of the categorical variables (invisible= "quali")
`lim.cos2.var`	value of the square cosinus under with the points are not drawn
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`cex`	cf. function `par` in the **graphics** package
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`new.plot`	boolean, if TRUE, a new graphical device is created
`...`	further arguments passed to or from other methods

Value

Returns the individuals factor map and the variables factor map.

Author(s)

Jeremy Mazet, Francois Husson `<francois.husson@institut-agro.fr>`

See Also

HMFA

Examples

```
data(wine)
hierar <- list(c(2,5,3,10,9,2), c(4,2))
res.hmfa <- HMFA(wine, H = hierar, type=c("n",rep("s",5)), graph = FALSE)
plot(res.hmfa, invisible="quali")
plot(res.hmfa, invisible="ind")
```

---

plot.MCA *Draw the Multiple Correspondence Analysis (MCA) graphs*

Description

Draw the Multiple Correspondence Analysis (MCA) graphs.

Usage

```
## S3 method for class 'MCA'
plot(x, axes = c(1, 2), choix=c("ind","var","quanti.sup"),
    xlim = NULL, ylim = NULL,
    invisible = c("none","ind","var","ind.sup","quali.sup","quanti.sup"),
    col.ind = "black", col.var = "red", col.quali.sup = "darkgreen",
    col.ind.sup = "blue", col.quanti.sup = "blue",
    label = c("all","none","ind","var","ind.sup","quali.sup","quanti.sup"),
    title = NULL, habillage = "none",
    palette = NULL, autoLab = c("auto","yes","no"), new.plot = FALSE,
    select = NULL, selectMod = NULL, unselect = 0.7,
shadowtext = FALSE, legend = list(bty = "y", x = "topleft"),
graph.type = c("ggplot","classic"), ggoptions = NULL, ...)
```

Arguments

`x`	an object of class MCA
`axes`	a length 2 vector specifying the components to plot
`choix`	the graph to plot ("ind" for the individuals and the categories, "var" for the variables, "quanti.sup" for the supplementary quantitative variables)
`xlim`	range for the plotted ’x’ values, defaulting to the range of the finite values of ’x’
`ylim`	range for the plotted ’y’ values, defaulting to the range of the finite values of ’y’
`invisible`	string indicating if some points should not be drawn ("ind", "var", "ind.sup", "quali.sup", "quanti.sup")
`col.ind`	a color for the individuals, if color ="none" the label is not written

`col.var`	a color for the categories of categorical variables, if color ="none" the label is not written
`col.quali.sup`	a color for the categorical supplementary variables, if color ="none" the label is not written
`col.ind.sup`	a color for the supplementary individuals only if there is not habillage, if color ="none" the label is not written
`col.quanti.sup`	a color for the supplementary quantitative variables, if color ="none" the label is not written
`label`	print the labels of the points; "all" print all the labels; may be a vector with "ind" (for the individuals),"ind.sup" (for the supplementary individuals),"var" (for the active categories), "quali.sup" "var" (for the supplementary categories)
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`habillage`	string corresponding to the color which are used. If "none", one color is used for the individual, another one for the categorical variables; if "quali", one color is used for each categorical variables; else if it is the position of a categorical variable, it colors according to the different categories of this variable
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=25
`autoLab`	if `autoLab="auto"`, `autoLab` is equal to "yes" if there are less than 50 elements and "no" otherwise; if "yes", the labels of the drawn elements are placed in a "good" way (can be time-consuming if many elements), and if "no" the elements are placed quickly but may overlap
`new.plot`	boolean, if TRUE, a new graphical device is created
`select`	a selection of the elements that are drawn; see the details section
`selectMod`	a selection of the categories that are drawn; see the details section
`unselect`	may be either a value between 0 and 1 that gives the transparency of the unselected objects (if `unselect=1` the transparceny is total and the elements are not drawn, if `unselect=0` the elements are drawn as usual but without any label) or may be a color (for example `unselect="grey60"`)
`shadowtext`	boolean; if true put a shadow on the labels (rectangles are written under the labels which may lead to difficulties to modify the graph with another program)
`legend`	a list of arguments that defines the legend if needed (when individuals are drawn according to a variable); see the arguments of the function `legend`
`graph.type`	a character that gives the type of graph used: "ggplot" or "classic"
`ggoptions`	a list that gives the graph options when grah.type="ggplot" is used. See the optines and the default values in the details section
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Details

The argument `autoLab = "yes"` is time-consuming if there are many labels that overlap. In this case, you can modify the size of the characters in order to have less overlapping, using for example

cex=0.7.
The `select` argument can be used in order to select a part of the elements (individuals if you draw the graph of individuals, or variables if you draw the graph of variabless) that are drawn. For example, you can use:
`select = 1:5` and then the elements 1:5 are drawn.
`select = c("name1","name5")` and then the elements that have the names name1 and name5 are drawn.
`select = "coord 10"` and then the 10 elements that have the highest (squared) coordinates on the 2 chosen dimensions are drawn.
`select = "contrib 10"` and then the 10 elements that have the highest contribution on the 2 dimensions of your plot are drawn.
`select = "cos2 5"` and then the 5 elements that have the highest cos2 on the 2 dimensions of your plot are drawn.
`select = "dist 8"` and then the 8 elements that have the highest distance to the center of gravity are drawn.
The `selectMod` argument can be used in order to select the categories that are drawn.

Value

Returns the individuals factor map and the variables factor map.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`MCA`

Examples

```
data (poison)
res.mca = MCA (poison, quali.sup = 3:4, quanti.sup = 1:2, graph=FALSE)
plot(res.mca,invisible=c("var","quali.sup"))
plot(res.mca,invisible="ind")
plot(res.mca,choix="var")
plot(res.mca,invisible=c("ind"), selectMod="cos2 10")
## Not run:
plot(res.mca, selectMod="cos2 5", select="cos2 5")

## You can modify the ggplot graphs as ususal with ggplot2
require(ggplot2)
gr <- plot(res.mca)
gr + theme(panel.grid.major = element_blank(),
   plot.title=element_text(size=14, color="blue"),
   axis.title = element_text(size=12, color="red"))

## End(Not run)
```

`plot.meansComp` *Draw the means comparisons*

Description

Plot the graphs for the means comparisons.

Usage

```
## S3 method for class 'meansComp'
plot(x, ...)
```

Arguments

`x`	an object of class meansComp.
`...`	further arguments passed to or from other methods, such as ggplot, ...

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`meansComp`

Examples

```
  data(senso)
  res <- LinearModel(Score~ Product + Day , data=senso, selection="none")
  meansComp(res,~Product)

## Not run:
  ## and with the sidak correction
  meansComp(res,~Product,adjust="sidak")

## End(Not run)
```

`plot.MFA` *Draw the Multiple Factor Analysis (MFA) graphs*

Description

Draw the Multiple Factor Analysis (MFA) graphs.

Usage

```
## S3 method for class 'MFA'
plot(x, axes = c(1, 2), choix = c("ind","var","group","axes","freq"),
    ellipse=NULL, ellipse.par=NULL,
    lab.grpe=TRUE, lab.var=TRUE, lab.ind=TRUE,
    lab.par=FALSE, lab.col=TRUE, ncp=2, habillage="group", col.hab=NULL,
    invisible = c("none","ind","ind.sup","quanti","quanti.sup",
"quali","quali.sup","row","row.sup","col","col.sup"),
partial = NULL, lim.cos2.var = 0.,
    chrono = FALSE, xlim = NULL, ylim = NULL,
    title = NULL, palette = NULL,
autoLab = c("auto","yes","no"), new.plot = FALSE,
select = NULL, unselect = 0.7, shadowtext = FALSE,
legend = list(bty = "y", x = "topleft"),
graph.type = c("ggplot","classic"), ggoptions = NULL, ...)
```

Arguments

`x`	an object of class MFA
`choix`	a string corresponding to the graph that you want to do ("ind" for the individual or categorical variables graph, "var" for the quantitative variables graph, "freq" for the frequence or contingency tables, "axes" for the graph of the partial axes, "group" for the groups representation)
`axes`	a length 2 vector specifying the components to plot
`ellipse`	boolean (NULL by default), if not null, draw ellipses around the individuals, and use the results of `coord.ellipse`
`ellipse.par`	boolean (NULL by default), if not null, draw ellipses around the partial individuals, and use the results of `coord.ellipse`
`lab.grpe`	boolean, if TRUE, the labels of the groups are drawn
`lab.var`	boolean, if TRUE, the labels of the variables are drawn
`lab.ind`	boolean, if TRUE, the labels of the mean points are drawn
`lab.par`	boolean, if TRUE, the labels of the partial points are drawn
`lab.col`	boolean, if TRUE, the labels of the columns for the contingency tables are drawn
`ncp`	number of principal components drawn for the separate analyses for the graph of the partial axes
`habillage`	string corresponding to the color which are used. If "ind", one color is used for each individual; if "group" the individuals are colored according to the group; else if it is the name or the position of a categorical variable, it colors according to the different categories of this variable
`col.hab`	the colors to use. By default, colors are chosen
`invisible`	list of string; for choix ="ind", the individuals can be omit (invisible = "ind"), or supplementary individuals (invisible="ind.sup") or the center of gravity of the categorical variables (invisible= "quali" or "quali.sup" for the supplementary categories); if invisible = c("ind","ind.sup"), just the centers of gravity are

	drawn; if choix="var", invisible="quanti" suppress the active variable and invisible = "quanti.sup" suppress the supplementary variables
`partial`	list of the individuals or of the center of gravity for which the partial points should be drawn (by default, partial = NULL and no partial points are drawn)
`lim.cos2.var`	value of the square cosinus under with the points are not drawn
`chrono`	boolean, if TRUE, the partial points of a same point are linked (useful when groups correspond to different moment)
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=2
`autoLab`	if `autoLab="auto"`, `autoLab` is equal to "yes" if there are less than 50 elements and "no" otherwise; if "yes", the labels of the drawn elements are placed in a "good" way (can be time-consuming if many elements), and if "no" the elements are placed quickly but may overlap
`new.plot`	boolean, if TRUE, a new graphical device is created
`select`	a selection of the elements that are drawn; see the details section
`unselect`	may be either a value between 0 and 1 that gives the transparency of the unselected objects (if `unselect=1` the transparceny is total and the elements are not drawn, if `unselect=0` the elements are drawn as usual but without any label) or may be a color (for example `unselect="grey60"`)
`shadowtext`	boolean; if true put a shadow on the labels (rectangles are written under the labels which may lead to difficulties to modify the graph with another program)
`legend`	a list of arguments that defines the legend if needed (when individuals are drawn according to a variable); see the arguments of the function `legend`
`graph.type`	a character that gives the type of graph used: "ggplot" or "classic"
`ggoptions`	a list that gives the graph options when grah.type="ggplot" is used. See the optines and the default values in the details section
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Details

The argument `autoLab = "yes"` is time-consuming if there are many labels that overlap. In this case, you can modify the size of the characters in order to have less overlapping, using for example cex=0.7.

The `select` argument can be used in order to select a part of the elements (individuals if you draw the graph of individuals, or variables if you draw the graph of variabless) that are drawn. For example, you can use:

`select = 1:5` and then the elements 1:5 are drawn.

`select = c("name1","name5")` and then the elements that have the names name1 and name5 are drawn.

`select = "coord 10"` and then the 10 elements (individuals, variables, frequencies) that have the highest (squared) coordinates on the 2 chosen dimensions are drawn.
`select = "contrib 10"` and then the 10 elements (individuals, variables, frequencies) that have the highest contribution on the 2 dimensions of your plot are drawn.
`select = "cos2 5"` and then the 5 elements (individuals, variables, frequencies) that have the highest cos2 on the 2 dimensions of your plot are drawn.
`selectMod = "contrib 5"`, the 5 categories that contribute the most to the two dimensions are drawn.
`selectMod = "cos2 0.5"`, the categories that have a cos2 greater than 0.5 on the two dimensions are drawn.
`selectMod = "v.test 2"`, the categories that have a v.test higher than the value on one of the two dimensions are drawn.

Value

Returns the individuals factor map and the variables factor map.

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>, Jeremy Mazet

See Also

`MFA`

Examples

```
## Not run:
data(wine)
res <- MFA(wine,group=c(2,5,3,10,9,2),type=c("n",rep("s",5)),ncp=5,
    name.group=c("orig","olf","vis","olfag","gust","ens"),
    num.group.sup=c(1,6),graph=FALSE)
plot(res, choix = "ind")
plot(res, choix = "ind", partial="all")
plot(res, choix = "ind", habillage="Label")
plot(res, choix = "var", habillage="group")
plot(res, choix = "axes")

data(wine)
res <- MFA(wine, group=c(2,5,3,10,9,2), type=c("n",rep("s",5)),
    ncp=5, name.group=c("orig","olf","vis","olfag","gust","ens"),
    num.group.sup=c(1,6))
summary(res)
barplot(res$eig[,1],main="Eigenvalues",names.arg=1:nrow(res$eig))

#### Confidence ellipses around categories per variable
plotellipses(res)
plotellipses(res,keepvar="Label") ## for 1 variable

#### Interactive graph
```

```
liste = plotMFApartial(res)
plot(res,choix="ind",habillage = "Terroir")

###Example with groups of categorical variables
data (poison)
MFA(poison, group=c(2,2,5,6), type=c("s","n","n","n"),
    name.group=c("desc","desc2","symptom","eat"),
    num.group.sup=1:2)

###Example with groups of frequency tables
data(mortality)
res<-MFA(mortality,group=c(9,9),type=c("f","f"),
    name.group=c("1979","2006"))

## End(Not run)
```

`plot.PCA` *Draw the Principal Component Analysis (PCA) graphs*

Description

Plot the graphs for a Principal Component Analysis (PCA) with supplementary individuals, supplementary quantitative variables and supplementary categorical variables.

Usage

```
## S3 method for class 'PCA'
plot(x, axes = c(1, 2), choix = c("ind","var","varcor"),
    ellipse = NULL, xlim = NULL, ylim = NULL, habillage="none",
    col.hab = NULL, col.ind="black", col.ind.sup="blue",
    col.quali="magenta", col.quanti.sup="blue", col.var="black",
    label = c("all","none","ind","ind.sup","quali","var","quanti.sup"),
invisible = c("none","ind","ind.sup","quali","var","quanti.sup"),
    lim.cos2.var = 0., title = NULL, palette=NULL,
    autoLab = c("auto","yes","no"), new.plot = FALSE, select = NULL,
unselect = 0.7, shadowtext = FALSE, legend = list(bty = "y", x = "topleft"),
graph.type = c("ggplot","classic"), ggoptions = NULL, ...)
```

Arguments

`x`	an object of class PCA
`axes`	a length 2 vector specifying the components to plot
`choix`	the graph to plot ("ind" for the individuals, "var" for the variables, "varcor" for a graph with the correlation circle when `scale.unit=FALSE`)
`ellipse`	boolean (NULL by default), if not null, draw ellipses around the individuals, and use the results of `coord.ellipse`

`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`habillage`	give no color for the individuals ("none"), a color for each individual ("ind"), or color the individuals among a categorical variable (give the number of the categorical variable)
`col.hab`	a vector with the color to use for the individuals
`col.ind`	a color for the individuals only if there is not habillage
`col.ind.sup`	a color for the supplementary individuals only if there is not habillage
`col.quali`	a color for the categories of categorical variables only if there is not habillage
`col.quanti.sup`	a color for the quantitative supplementary variables
`col.var`	a color for the variables
`label`	a list of character for the elements which are labelled (by default, all the elements are labelled ("ind", ind.sup", "quali", "var", "quanti.sup"))
`invisible`	string indicating if some points should not be drawn ("ind", "ind.sup" or "quali" for the individual graph and "var" or "quanti.sup" for the correlation circle graph)
`lim.cos2.var`	value of the square cosinus under the variables are not drawn
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=25
`autoLab`	if `autoLab="auto"`, `autoLab` is equal to "yes" if there are less than 50 elements and "no" otherwise; if "yes", the labels of the drawn elements are placed in a "good" way (can be time-consuming if many elements), and if "no" the elements are placed quickly but may overlap
`new.plot`	boolean, if TRUE, a new graphical device is created; only used when `graph.type="classic"`
`select`	a selection of the elements that are drawn; see the details section
`unselect`	may be either a value between 0 and 1 that gives the transparency of the unselected objects (if `unselect=1` the transparceny is total and the elements are not drawn, if `unselect=0` the elements are drawn as usual but without any label) or may be a color (for example `unselect="grey60"`)
`shadowtext`	boolean; if true put a shadow on the labels (rectangles are written under the labels which may lead to difficulties to modify the graph with another program); only used when `graph.type="classic"`
`legend`	a list of arguments that defines the legend if needed (when individuals are drawn according to a variable); see the arguments of the function `legend`
`graph.type`	a character that gives the type of graph used: "ggplot" or "classic"
`ggoptions`	a list that gives the graph options when grah.type="ggplot" is used. See the optines and the default values in the details section
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Details

The argument `autoLab = "yes"` is time-consuming if there are many labels that overlap. In this case, you can modify the size of the characters in order to have less overlapping, using for example cex=0.7.
The `select` argument can be used in order to select a part of the elements (individuals if you draw the graph of individuals, or variables if you draw the graph of variables) that are drawn. For example, you can use:
`select = 1:5` and then the elements 1:5 are drawn.
`select = c("name1","name5")` and then the elements that have the names name1 and name5 are drawn.
`select = "coord 10"` and then the 10 elements that have the highest (squared) coordinates on the 2 chosen dimensions are drawn.
`select = "contrib 10"` and then the 10 elements that have the highest contribution on the 2 dimensions of your plot are drawn.
`select = "cos2 5"` and then the 5 elements that have the highest cos2 on the 2 dimensions of your plot are drawn.
`select = "dist 8"` and then the 8 elements that have the highest distance to the center of gravity are drawn.

`ggoptions` is a list that gives some ggplot2 options when the graph.type="ggplot" is used. Use for instance ggoptions(list(size=3,title.size=10,bg.color="orange")) if you want to modify the size of the points and labels, the title size and the background color.
Below you can see the options and the default values:
size = 4, #label size (point size = size/3)
point.shape = 19, #points shape
line.lty = 2, #origin linetypes (0="blank", 1="solid", 2="dashed", 3="dotted",...)
line.lwd = 0, #origin lines width
line.color = "black", #origin lines color
segment.lty = 1, #arrow linetypes (0="blank", 1="solid", 2="dashed", 3="dotted",...)
segment.lwd = 0, #arrow width
circle.lty = 1, #circle linetypes (0="blank", 1="solid", 2="dashed", 3="dotted",...)
circle.lwd = 0, #circle width
circle.color = "black", #circle color
low.col.quanti = "blue", #for quantitative variables, low color to be used
high.col.quanti = "red3", #for quantitative variables, high color to be used

Value

Returns the individuals factor map and the variables factor map.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`PCA`

Examples

```
data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup = 13)
plot(res.pca, habillage = 13, cex=0.8)
## Not run:
plot(res.pca, habillage = "cos2")
plot(res.pca, habillage = "100m")
plot(res.pca, habillage = "Competition")

## End(Not run)
## To automatically draw ellipses around the barycentres of the categorical variables
plotellipses(res.pca)

## Not run:
## Selection of some individuals
plot(res.pca,select="contrib 7") # plot the 7 individuals with the highest contribution
plot(res.pca,select="cos2 0.8")  # plot the individuals with cos2 greater than 0.8
plot(res.pca,select="cos2 5")    # plot the 5 individuals with the highest cos2
plot(res.pca,choix="var",select="cos2 0.6")  # plot the variables with cos2 greater than 0.6

plot(res.pca,habillage="100m",
  ggoptions=list(low.col.quanti="grey90",high.col.quanti="grey10"),legend=list(x="bottom"))

## You can modify the ggplot graphs as ususal with ggplot2
require(ggplot2)
gr <- plot(res.pca)
gr + theme(panel.grid.major = element_blank(),
   plot.title=element_text(size=14, color="blue"),
   axis.title = element_text(size=12, color="red"))

## To draw classical R graphs
plot(res.pca, graph.type = "classic")

## End(Not run)
```

`plotellipses` *Draw confidence ellipses around the categories*

Description

Draw confidence ellipses around the categories.

Usage

```
plotellipses(model, keepvar = "all", axes = c(1, 2), means=TRUE, level = 0.95,
    magnify = 2, cex = 1, pch = 20, pch.means=15, type = c("g","p"),
    keepnames = TRUE, namescat = NULL, xlim=NULL, ylim=NULL, lwd=1,
    label="all", autoLab=c("auto","yes","no"),
graph.type = c("ggplot","classic"), ...)
```

Arguments

`model`	an object of class MCA or PCA or MFA
`keepvar`	a boolean or numeric vector of indexes of variables or a character vector of names of variables. If keepvar is "all", "quali" or "quali.sup" variables which are plotted are all the categorical variables, only those which are used to compute the dimensions (active variables) or only the supplementary categorical variables. If keepvar is a numeric vector of indexes or a character vector of names of variables, only relevant variables are plotted.
`axes`	a length 2 vector specifying the components to plot
`means`	boolean which indicates if the confidence ellipses are for (the coordinates of) the means of the categories (the empirical variance is divided by the number of observations) or for (the coordinates of) the observations of the categories
`level`	the confidence level for the ellipses
`magnify`	numeric which control how the level names are magnified. A value of 2 means that the level names have character expansion equal to two times cex
`cex`	cf. function `par` in the **graphics** package
`pch`	plotting character for coordinates, cf. function `par` in the **graphics** package
`pch.means`	plotting character for means, cf. function `par` in the **graphics** package
`type`	cf. function `xyplot`
`keepnames`	a boolean or numeric vector of indexes of variables or a character vector of names of variables. If keepnames is TRUE, names of levels are taken from the (modified) dataset extracted from modele, if FALSE trimming names is done. When trimming, names of levels are taken from the (modified) dataset extracted from modele, then, the corresponding number of characters of names of original variables plus 1 is removed. If keepnames is a vector of indexes or names, trimming is done on all variables excepted whose in keepnames
`namescat`	a vector giving for each observation the value of categorical variable, each variable are stacked under each other. If NULL, names are taken from the (modified) dataset extracted from modele
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`lwd`	The line width, a positive number, defaulting to 1
`label`	a list of character for the elements which are labelled (by default, "all", you can use "none", "ind", ind.sup"))
`autoLab`	if `autoLab="auto"`, `autoLab` is equal to "y" if there are less than 50 elements and "no" otherwise; if "y", the labels of the drawn elements are placed in a "good" way (can be time-consuming if many elements), and if "no" the elements are placed quickly but may overlap
`graph.type`	a character that gives the type of graph used: "ggplot" or "classic"
`...`	further arguments passed to or from other methods

Value

Return a graph with the ellipses. If only one variable is chosen, the graph is different.

Author(s)

Pierre-Andre Cornillon, Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`MCA`, `PCA`

Examples

```
## Not run:
data(poison)
res.mca = MCA(poison, quali.sup = 3:4, quanti.sup = 1:2)
plotellipses(res.mca)
plotellipses(res.mca,keepvar=3:6)

## End(Not run)

data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup=13)
plotellipses(res.pca,keepvar=13)
```

`plotGPApartial` *Draw an interactive General Procrustes Analysis (GPA) map*

Description

Draw an interactive General Procrustes Analysis (GPA) map. The graph is interactive and clicking on a point will draw the partial points, if you click on a point for which the partial points are yet drawn, the partial points are deleted. To stop the interactive plot, click on the title (or in the top of the graph)

Usage

```
plotGPApartial(x, axes = c(1, 2),
    lab.ind.moy = TRUE, habillage = "ind",
    chrono = FALSE, draw.partial = NULL,
    xlim = NULL, ylim = NULL, cex = 1, title = NULL, palette = NULL, ...)
```

Arguments

`x`	an object of class GPA
`axes`	a length 2 vector specifying the components to plot
`lab.ind.moy`	boolean, if TRUE, the label of the mean points are drawn
`habillage`	string corresponding to the color which are used. If "ind", one color is used for each individual; if "group" the individuals are colored according to the group
`chrono`	boolean, if TRUE, the partial points of a same point are linked (useful when groups correspond to different moment)

`draw.partial`	data frame of a boolean variable for all the individuals and all the centers of gravity and with for which the partial points should be drawn (by default, NULL and no partial points are drawn)
`xlim`	range for the plotted 'x' values, defaulting to the range of the finite values of 'x'
`ylim`	range for the plotted 'y' values, defaulting to the range of the finite values of 'y'
`cex`	cf. function `par` in the **graphics** package
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=25
`...`	further arguments passed to or from other methods

Value

Returns the General Procrustes Analysis map.

Author(s)

Elisabeth Morand, Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`GPA`

`plotMFApartial` *Plot an interactive Multiple Factor Analysis (MFA) graph*

Description

Draw an interactive Multiple Factor Analysis (MFA) graphs.

Usage

```
plotMFApartial(x, axes = c(1, 2),
    lab.ind = TRUE, lab.par = FALSE, habillage = "group",
    chrono = FALSE, col.hab = NULL, invisible = NULL,
    draw.partial = NULL, xlim = NULL, ylim = NULL,
    cex = 1, title = NULL, palette = NULL, ...)
```

Arguments

`x`	an object of class MFA
`axes`	a length 2 vector specifying the components to plot
`lab.ind`	boolean, if TRUE, the label of the mean points are drawn
`lab.par`	boolean, if TRUE, the label of the partial points are drawn
`habillage`	string corresponding to the color which are used. If "group", one color is used for each group of variables; if "quali" the individuals are colored according to one categorical variable; if "group" the individuals are colored according to the group
`chrono`	boolean, if TRUE, the partial points of a same point are linked (useful when groups correspond to different moment)
`col.hab`	the colors to use. By default, colors are chosen
`invisible`	list of string; for choix ="ind", the individuals can be omit (invisible = "ind"), or supplementary individuals (invisible="ind.sup") or the centerg of gravity of the categorical variables (invisible= "quali"); if invisible = c("ind","ind.sup"), just the centers of gravity are drawn
`draw.partial`	data frame of a boolean variable for all the individuals and all the centers of gravity and with for which the partial points should be drawn (by default, NULL and no partial points are drawn)
`xlim`	range for the plotted ’x’ values, defaulting to the range of the finite values of ’x’
`ylim`	range for the plotted ’y’ values, defaulting to the range of the finite values of ’y’
`cex`	cf. function `par` in the **graphics** package
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`palette`	the color palette used to draw the points. By default colors are chosen. If you want to define the colors : palette=palette(c("black","red","blue")); or you can use: palette=palette(rainbow(30)), or in black and white for example: palette=palette(gray(seq(0,.9,len=25
`...`	further arguments passed to or from other methods

Value

Draw a graph with the individuals and the centers of gravity. The graph is interactive and clicking on a point will draw the partial points, if you click on a point for which the partial points are yet drawn, the partial points are deleted. To stop the interactive plot, click on the title (or in the top of the graph).
Return the names of the points for which the partial points are drawn.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`MFA`, `plot.MFA`

Examples

```
## Not run:
data(wine)
res.wine = MFA(wine,group=c(2,5,3,10,9,2),type=c("n",rep("s",5)),ncp=5,
    name.group=c("orig","olf","vis","olfag","gust","ens"),
    num.group.sup=c(1,6),graph=FALSE)
liste = plotMFApartial(res.wine)
plot(res.wine, partial = liste)

## End(Not run)
```

`poison` *Poison*

Description

The data used here refer to a survey carried out on a sample of children of primary school who suffered from food poisoning. They were asked about their symptoms and about what they ate.

Usage

```
data(poison)
```

Format

A data frame with 55 rows and 15 columns.

Examples

```
## Not run:
data(poison)
res.mca <- MCA(poison, quanti.sup = 1:2, quali.sup=c(3,4))

## End(Not run)
```

`poison.text` *Poison*

Description

The data used here refer to a survey carried out on a sample of children of primary school who suffered from food poisoning. They were asked about their symptoms and about what they ate.

Usage

```
data(poison)
```

Format

A data frame with 55 rows and 3 columns (the sex, if they are sick or not, and a textual variable with their symptom and what they eat).

Examples

```
data(poison.text)
res.text <- textual(poison.text, num.text = 3, contingence.by = c(1,2))
## Contingence table for the sex variable, the sich variable and the couple
## of variable sick-sex
res.text2 <- textual(poison.text, num.text = 3, contingence.by = list(1,2,c(1,2)))
```

`poulet` *Genomic data for chicken*

Description

Genomic data for chicken

Usage

```
data(poulet)
```

Format

A data frame with 43 chickens and 7407 variables. A factor with levels J16 J16R16 J16R5 J48 J48R24 N
And many continuous variables corresponding to the gene expression

Examples

```
## Not run:
data(poulet)
res.pca = PCA(poulet,quali.sup=1, graph=FALSE)
plot(res.pca)
plot(res.pca,habillage=1,label="quali",
    palette=palette(c("black","red","blue","darkgreen","purple","orange")))
dimdesc(res.pca)
## Dessine des ellipses autour des centres de gravite
plotellipses(res.pca)
## End(Not run)
```

---

predict.CA *Predict projection for new rows with Correspondence Analysis*

Description

Predict the projection of new rows with Correspondence Analysis.

Usage

```
## S3 method for class 'CA'
predict(object, newdata, ...)
```

Arguments

`object`	an object of class CA
`newdata`	A data frame or a matrix in which to look for variables with which to predict. newdata must contain columns with the same names as the original data.
`...`	Other options.

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`CA`

---

predict.FAMD *Predict projection for new rows with Factor Analysis of Mixed Data*

Description

Predict the projection of new rows with Factor Analysis of Mixed Data.

Usage

```
## S3 method for class 'FAMD'
predict(object, newdata, ...)
```

Arguments

`object`	an object of class FAMD
`newdata`	A data frame or a matrix in which to look for variables with which to predict. newdata must contain columns with the same names as the original data.
`...`	Other options.

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`FAMD`

`predict.LinearModel` *Predict method for Linear Model Fits*

Description

Predicted values based on LinearModel object.

Usage

```
## S3 method for class 'LinearModel'
predict(object, newdata, interval = c("none", "confidence", "prediction"),
        level = 0.95, type = c("response", "terms"), ...)
```

Arguments

`object`	Object of class inheriting from "LinearModel"
`newdata`	An optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
`interval`	Type of interval calculation. Can be abbreviated.
`level`	Tolerance/confidence level.
`type`	Type of prediction (response or model term). Can be abbreviated.
`...`	further arguments passed to or from other methods such as `lm`.

Details

See the help of `predict.lm` function.

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`LinearModel`

`predict.MCA` *Predict projection for new rows with Multiple Correspondence Analysis*

Description

Predict the projection of new rows with Multiple Correspondence Analysis.

Usage

```
## S3 method for class 'MCA'
predict(object, newdata, ...)
```

Arguments

`object`	an object of class MCA
`newdata`	A data frame or a matrix in which to look for variables with which to predict. newdata must contain columns with the same names as the original data.
`...`	Other options.

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`MCA`

`predict.MFA` *Predict projection for new rows with Multiple Factor Analysis*

Description

Predict the projection of new rows with Multiple Factor Analysis.

Usage

```
## S3 method for class 'MFA'
predict(object, newdata, ...)
```

Arguments

`object`	an object of class MFA
`newdata`	A data frame or a matrix in which to look for variables with which to predict. newdata must contain columns with the same names as the original data.
`...`	Other options.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`MFA`

`predict.PCA` *Predict projection for new rows with Principal Component Analysis*

Description

Predict the projection of new rows with Principal Component Analysis.

Usage

```
## S3 method for class 'PCA'
predict(object, newdata, ...)
```

Arguments

`object`	an object of class PCA
`newdata`	A data frame or a matrix in which to look for variables with which to predict. newdata must contain columns with the same names as the original data.
`...`	Other options.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`PCA`

---

prefpls — *Scatter plot and additional variables with quality of representation contour lines*

Description

This function is useful to interpret the usual graphs (x,y) with additional quantitative variables.

Usage

```
prefpls(donnee, var1 = 1, var2 = 2, firstvar = 3,
    lastvar = ncol(donnee), levels = c(0.2,0.4,0.6,0.7,0.8,0.9,1),
    asp = 1, nbchar = max(nchar(colnames(donnee))), title = NULL,
    choix="var")
```

Arguments

`donnee`	a data frame made up of quantitative variables
`var1`	the position of the variable corresponding to the x-axis
`var2`	the position of the variable corresponding to the y-axis
`firstvar`	the position of the first endogenous variable
`lastvar`	the position of the last endogenous variable (by default the last column of donnee)
`levels`	a list of the levels displayed in the graph of variables
`asp`	aspect ratio for the graph of the individuals
`nbchar`	the number of characters used for the labels of the variables
`title`	string corresponding to the title of the graph you draw (by default NULL and a title is chosen)
`choix`	the graph to plot ("ind" for the individuals, "var" for the variables)

Details

This function is very useful when there is a strong correlation between two variables x and y

Value

A scatter plot of the invividuals
A graph with additional variables and the quality of representation contour lines.

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

References

Husson, F. & Pages, J. (2005). Scatter plot and additional variables. *Journal of applied statistics*

Examples

```
data(decathlon)
prefpls(decathlon[,c(11,12,1:10)])
```

`print.AovSum` *Print the AovSum results*

Description

Print the results of the ANOVA obtained by the function AovSum.

Usage

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

Arguments

`x`	an object of class AovSum
`...`	further arguments passed to or from other methods

Author(s)

Vincent Guyader, Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`AovSum`

Examples

```
## Not run:
data(senso)
res <- AovSum(Score~ Product + Day , data=senso)
res

## End(Not run)
```

`print.CA` *Print the Correspondance Analysis (CA) results*

Description

Print the Correspondance Analysis (CA) results.

Usage

```
## S3 method for class 'CA'
print(x, file = NULL, sep = ";", ...)
```

Arguments

`x`	an object of class CA
`file`	A connection, or a character string naming the file to print to. If NULL (the default), the results are not printed in a file
`sep`	character string to insert between the objects to print (if the argument file is not NULL
`...`	further arguments passed to or from other methods

Author(s)

Jeremy Mazet, Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`CA`, `write.infile`

`print.CaGalt` *Print the Correspondence Analysis on Generalised Aggregated Lexical Table (CaGalt) results*

Description

Print the Correspondence Analysis on Generalised Aggregated Lexical Table (CaGalt) results

Usage

```
## S3 method for class 'CaGalt'
## S3 method for class 'CaGalt'
print(x, file = NULL, sep = ";", ...)
```

Arguments

`x`	an object of class CaGalt
`file`	A connection, or a character string naming the file to print to. If NULL (the default), the results are not printed in a file
`sep`	character string to insert between the objects to print (if the argument file is not NULL)
`...`	further arguments passed to or from other methods

Author(s)

Belchin Kostov <`badriyan@clinic.ub.es`>, Monica Becue-Bertaut, Francois Husson

See Also

`CaGalt`, `write.infile`

Examples

```
## Not run:
data(health)
res.cagalt<-CaGalt(Y=health[,1:115],X=health[,116:118],type="n")
print(res.cagalt)

## End(Not run)
```

`print.catdes` *Print the catdes results*

Description

Print the results of the function catdes.

Usage

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

Arguments

`x`	an object of class catdes
`...`	further arguments passed to or from other methods

Author(s)

Vincent Guyader, Francois Husson <`francois.husson@institut-agro.fr`>

See Also

catdes

Examples

```
## Not run:
data(wine)
res <- catdes(wine, num.var=2)
print(res)

## End(Not run)
```

`print.condes` *Print the condes results*

Description

Print the results of the function condes.

Usage

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

Arguments

`x`	an object of class condes
`...`	further arguments passed to or from other methods

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

condes

Examples

```
## Not run:
data(wine)
res <- condes(wine, num.var=3)
print(res)

## End(Not run)
```

`print.FAMD` *Print the Multiple Factor Analysis of mixt Data (FAMD) results*

Description

Print the Multiple Factor Analysis of mixt Data (FAMD) results.

Usage

```
## S3 method for class 'FAMD'
print(x, file = NULL, sep = ";", ...)
```

Arguments

`x`	an object of class FAMD
`file`	A connection, or a character string naming the file to print to. If NULL (the default), the results are not printed in a file
`sep`	character string to insert between the objects to print (if the argument file is not NULL
`...`	further arguments passed to or from other methods

Author(s)

Jeremy Mazet, Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`FAMD`

`print.GPA` *Print the Generalized Procrustes Analysis (GPA) results*

Description

Print the Generalized Procrustes Analysis (GPA) results.

Usage

```
## S3 method for class 'GPA'
print(x, file = NULL, sep = ";", ...)
```

Arguments

`x`	an object of class GPA
`file`	A connection, or a character string naming the file to print to. If NULL (the default), the results are not printed in a file
`sep`	character string to insert between the objects to print (if the argument file is not NULL
`...`	further arguments passed to or from other methods

Author(s)

Elisabeth Morand, Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`GPA`, `write.infile`

`print.HCPC` *Print the Hierarchical Clustering on Principal Components (HCPC) results*

Description

Print the Hierarchical Clustering on Principal Components (HCPC) results.

Usage

```
## S3 method for class 'HCPC'
print(x, file = NULL, sep = ";", ...)
```

Arguments

`x`	an object of class HCPC
`file`	A connection, or a character string naming the file to print to. If NULL (the default), the results are not printed in a file
`sep`	character string to insert between the objects to print (if the argument file is not NULL
`...`	further arguments passed to or from other methods

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`HCPC`, `write.infile`

`print.HMFA` *Print the Hierarchical Multiple Factor Analysis results*

Description

Print the Hierarchical Multiple Factor Analysis results.

Usage

```
## S3 method for class 'HMFA'
print(x, file = NULL, sep = ";", ...)
```

Arguments

`x`	an object of class HMFA
`file`	A connection, or a character string naming the file to print to. If NULL (the default), the results are not printed in a file
`sep`	character string to insert between the objects to print (if the argument file is not NULL
`...`	further arguments passed to or from other methods

Author(s)

Sebastien Le, Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`HMFA`, `write.infile`

`print.LinearModel` *Print the LinearModel results*

Description

Print the results of the ANOVA obtained by the function LinearModel.

Usage

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

Arguments

`x`	an object of class LinearModel
`...`	further arguments passed to or from other methods

Details

Gives the results of the LinearModel function. If a model selection is performed, the global F-test for the complete model is first given, then all the results are given for the selected model (global F-test, the F-tests for main effects and interaction, the t-tests)

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`LinearModel`

Examples

```
## Not run:
data(senso)
res <- LinearModel(Score~ Product + Day , data=senso)
res

res2 <- LinearModel(Score~ Product + Day , data=senso, selection="BIC")
res2

## End(Not run)
```

`print.MCA` *Print the Multiple Correspondance Analysis (MCA) results*

Description

Print the Multiple Correspondance Analysis (spMCA) results.

Usage

```
## S3 method for class 'MCA'
print(x, file = NULL, sep = ";", ...)
```

Arguments

`x`	an object of class MCA
`file`	A connection, or a character string naming the file to print to. If NULL (the default), the results are not printed in a file
`sep`	character string to insert between the objects to print (if the argument file is not NULL
`...`	further arguments passed to or from other methods

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`MCA`, `write.infile`

`print.MFA` *Print the Multiple Factor Analysis results*

Description

Print the Multiple Factor Analysis results.

Usage

```
## S3 method for class 'MFA'
print(x, file = NULL, sep = ";", ...)
```

Arguments

`x`	an object of class MFA
`file`	A connection, or a character string naming the file to print to. If NULL (the default), the results are not printed in a file
`sep`	character string to insert between the objects to print (if the argument file is not NULL
`...`	further arguments passed to or from other methods

Author(s)

Jeremy Mazet, Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`MFA`, `write.infile`

`print.PCA` *Print the Principal Component Analysis (PCA) results*

Description

Print the Principal Component Analysis (PCA) results.

Usage

```
## S3 method for class 'PCA'
print(x, file = NULL, sep = ";", ...)
```

Arguments

`x`	an object of class PCA
`file`	A connection, or a character string naming the file to print to. If NULL (the default), the results are not printed in a file
`sep`	character string to insert between the objects to print (if the argument file is not NULL
`...`	further arguments passed to or from other methods

Author(s)

Jeremy Mazet, Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`PCA`, `write.infile`

Examples

```
## Not run:
data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup = 13)
print(res.pca, file="c:/essai.csv", sep = ";")

## End(Not run)
```

`reconst` *Reconstruction of the data from the PCA, CA or MFA results*

Description

Reconstruct the data from the PCA, CA or MFA results.

Usage

```
reconst(res, ncp=NULL)
```

Arguments

`res`	an object of class PCA, CA or MFA
`ncp`	number of dimensions used to reconstitute the data (by default NULL and the number of dimensions calculated for the PCA, CA or MFA is used)

Value

Returns a data frame with the number of individuals and the number of variables used for the PCA, CA or MFA

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`, Julie Josse`<Julie.Josse@agrocampus-ouest.fr>`

See Also

`PCA`,`CA`, `MFA`

Examples

```
data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup=13, graph=FALSE)
rec <- reconst(res.pca,ncp=2)
```

RegBest *Select variables in multiple linear regression*

Description

Find an optimal submodel

Usage

```
RegBest(y,x, int = TRUE, wt=NULL, na.action = na.omit,
    method=c("r2","Cp", "adjr2"), nbest=1)
```

Arguments

`y`	A response vector
`x`	A matrix of predictors
`int`	Add an intercept to the model
`wt`	Optional weight vector
`na.action`	Handling missing values
`method`	Calculate R-squared, adjusted R-squared or Cp to select the model. By default a the F-test on the r-square is used
`nbest`	number of best models for each set of explained variables (by default 1)

Value

Returns the objects

`all`	gives all the `nbest` best models for a given number of variables
`best`	the best model

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`lm`

Examples

```
data(milk)
res = RegBest(y=milk[,6],x=milk[,-6])
res$best
```

---

senso *senso*

Description

Dataset to illustrate one-way and Two-way analysis of variance

Usage

```
data(senso)
```

Format

Dataset with 45 rows and 3 columns: Score, Product and Day

Examples

```
## Example of 2-way analysis of variance
data(senso)
res <- AovSum (Score~ Product + Day, data=senso)
res

## Example of 2-way analysis of variance with interaction
data(senso)
res2 <- AovSum (Score~ Product + Day + Product : Day, data=senso)
res2
```

---

simule *Simulate by bootstrap*

Description

Simulate by bootstrap

Usage

```
simule(data, nb.simul)
```

Arguments

`data`	A data frame from which the rows are the original data from which the simualte data are calculated (by the average of a bootstrap sample. The columns corresponds to the variables for which the simulation should be done. The first column must be a factor allowing to group the rows. A bootstrap simulation is done for each level of this factor.
`nb.simul`	The number of simulations.

Details

The simulation is independently done for each level of the factor. The number of rows can be different for each levels.

Value

`mean`	Data.frame with all the levels of the factor variable, and for each variable, the mean of the original data.
`simul`	Data.frame with all the levels of the factor variable, and for each variable, the nb.simul bootstrap simulations.
`simul.mean`	Data.frame with all the levels of the factor variable, and for each variable, the mean of the simulated data.

Author(s)

Jeremy Mazet

`summary.CA` *Printing summeries of ca objects*

Description

Printing summaries of correspondence analysis objects

Usage

```
## S3 method for class 'CA'
summary(object, nb.dec = 3, nbelements=10,
   ncp = 3, align.names=TRUE, file="", ...)
```

Arguments

`object`	an object of class CA
`nb.dec`	number of decimal printed
`nbelements`	number of elements written (rows, columns, ...) ; use `nbelements = Inf` if you want to have all the elements
`ncp`	number of dimensions printed
`align.names`	boolean, if TRUE the names of the objects are written using the same number of characters
`file`	a connection, or a character string naming the file to print to
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

CA

`summary.CaGalt` *Printing summaries of CaGalt objects*

Description

Printing summaries of Correspondence Analysis on Generalised Aggregated Lexical Table (CaGalt) objects

Usage

```
## S3 method for class 'CaGalt'
summary(object, nb.dec = 3, nbelements=10, nbind = nbelements,
  ncp = 3, align.names=TRUE, file="", ...)
```

Arguments

`object`	an object of class CaGalt
`nb.dec`	number of printed decimals
`nbelements`	number of written elements (variables, categories, frequencies); use nbelements = Inf if you want to have all the elements
`nbind`	number of written elements; use nbind = Inf to have the results for all the individuals and nbind = 0 if you do not want the results for individuals
`ncp`	number of printed dimensions
`align.names`	boolean, if TRUE the names of the objects are written using the same number of characters
`file`	a connection, or a character string naming the file to print to
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Author(s)

Belchin Kostov <`badriyan@clinic.ub.es`>, Monica Becue-Bertaut, Francois Husson

See Also

CaGalt

Examples

```
## Not run:
data(health)
res.cagalt<-CaGalt(Y=health[,1:115],X=health[,116:118],type="n")
summary(res.cagalt)

## End(Not run)
```

`summary.FAMD` *Printing summeries of FAMD objects*

Description

Printing summaries of factor analysis on mixed data objects

Usage

```
## S3 method for class 'FAMD'
summary(object, nb.dec = 3, nbelements=10,
    nbind=nbelements, ncp = 3, align.names=TRUE , file="", ...)
```

Arguments

`object`	an object of class FAMD
`nb.dec`	number of decimal printed
`nbelements`	number of elements written (variables, categories, ...); use `nbelements = Inf` if you want to have all the elements
`nbind`	number of individuals written (individuals and supplementary individuals, ...); use `nbind = Inf` to have the results for all the individuals and `nbind = 0` if you do not want the results for individuals
`ncp`	number of dimensions printed
`align.names`	boolean, if TRUE the names of the objects are written using the same number of characters
`file`	a connection, or a character string naming the file to print to
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`FAMD`

`summary.MCA` *Printing summeries of MCA objects*

Description

Printing summaries of multiple correspondence analysis objects

Usage

```
## S3 method for class 'MCA'
summary(object, nb.dec = 3, nbelements=10,
    nbind=nbelements, ncp = 3, align.names=TRUE, file="", ...)
```

Arguments

`object`	an object of class MCA
`nb.dec`	number of decimal printed
`nbelements`	number of elements written (variables, categories, ...); use `nbelements = Inf` if you want to have all the elements
`nbind`	number of individuals written (individuals and supplementary individuals, ...); use `nbind = Inf` to have the results for all the individuals and `nbind = 0` if you do not want the results for individuals
`ncp`	number of dimensions printed
`align.names`	boolean, if TRUE the names of the objects are written using the same number of characters
`file`	a connection, or a character string naming the file to print to
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`MCA`

`summary.MFA` *Printing summaries of MFA objects*

Description

Printing summaries of multiple factor analysis objects

Usage

```
## S3 method for class 'MFA'
summary(object, nb.dec = 3, nbelements=10,
    nbind = nbelements, ncp = 3, align.names=TRUE, file="", ...)
```

Arguments

`object`	an object of class MFA
`nb.dec`	number of decimal printed
`nbelements`	number of elements written (groups, variables, categories, ...); use `nbelements = Inf` if you want to have all the elements
`nbind`	number of individuals written (individuals and supplementary individuals, ...); use `nbind = Inf` to have the results for all the individuals and `nbind = 0` if you do not want the results for individuals
`ncp`	number of dimensions printed
`align.names`	boolean, if TRUE the names of the objects are written using the same number of characters
`file`	a connection, or a character string naming the file to print to
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Author(s)

Francois Husson <`francois.husson@institut-agro.fr`>

See Also

`MFA`

`summary.PCA` *Printing summeries of PCA objects*

Description

Printing summaries of principal component analysis objects

Usage

```
## S3 method for class 'PCA'
summary(object, nb.dec = 3, nbelements=10,
   nbind = nbelements, ncp = 3, align.names=TRUE, file="", ...)
```

Arguments

`object`	an object of class PCA
`nb.dec`	number of decimal printed
`nbelements`	number of elements written (variables, categories, ...); use `nbelements = Inf` if you want to have all the elements
`nbind`	number of individuals written (individuals and supplementary individuals, ...); use `nbind = Inf` to have the results for all the individuals and `nbind = 0` if you do not want the results for individuals
`ncp`	number of dimensions printed
`align.names`	boolean, if TRUE the names of the objects are written using the same number of characters
`file`	a connection, or a character string naming the file to print to
`...`	further arguments passed to or from other methods, such as cex, cex.main, ...

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`PCA`

`svd.triplet` *Singular Value Decomposition of a Matrix*

Description

Compute the singular-value decomposition of a rectangular matrix with weights for rows and columns.

Usage

```
svd.triplet(X, row.w=NULL, col.w=NULL, ncp=Inf)
```

Arguments

`X`	a data matrix
`row.w`	vector with the weights of each row (NULL by default and the weights are uniform)
`col.w`	vector with the weights of each column (NULL by default and the weights are uniform)
`ncp`	the number of components kept for the outputs

Value

`vs`	a vector containing the singular values of 'x';
`u`	a matrix whose columns contain the left singular vectors of 'x';
`v`	a matrix whose columns contain the right singular vectors of 'x'.

See Also

`svd`

`tab.disjonctif` *Make a disjonctif table*

Description

Make a disjonctif table.

Usage

```
tab.disjonctif(tab)
```

Arguments

`tab`	a data frame with factors

Value

The disjonctif table

`tab.disjonctif.prop` *Make a disjunctive table when missing values are present*

Description

Create a disjunctive table. The missing values are replaced by the proportion of the category.

Usage

```
tab.disjonctif.prop(tab,seed=NULL,row.w=NULL)
```

Arguments

`tab`	a data frame with factors
`row.w`	an optional row weights (by default, a vector of 1 for uniform row weights)
`seed`	a single value, interpreted as an integer for the set.seed function (if seed = NULL, missing values are initially imputed by the mean of each variable)

Value

The disjonctif table.prop

`tea` *tea (data)*

Description

The data used here concern a questionnaire on tea. We asked to 300 individuals how they drink tea (18 questions), what are their product's perception (12 questions) and some personal details (4 questions).

Usage

```
data(tea)
```

Format

A data frame with 300 rows and 36 columns. Rows represent the individuals, columns represent the different questions. The first 18 questions are active ones, the 19th is a supplementary quantitative variable (the age) and the last variables are supplementary categorical variables.

Examples

```
## Not run:
data(tea)
res.mca=MCA(tea,quanti.sup=19,quali.sup=20:36)
plot(res.mca,invisible=c("var","quali.sup","quanti.sup"),cex=0.7)
plot(res.mca,invisible=c("ind","quali.sup","quanti.sup"),cex=0.8)
plot(res.mca,invisible=c("quali.sup","quanti.sup"),cex=0.8)
dimdesc(res.mca)
plotellipses(res.mca,keepvar=1:4)

## make a hierarchical clustering: click on the tree to define the number of clusters
## HCPC(res.mca)

## End(Not run)
```

`textual` *Text mining*

Description

Calculates the number of occurence of each words and a contingence table

Usage

```
textual(tab, num.text, contingence.by=1:ncol(tab),
    maj.in.min = TRUE, sep.word=NULL)
```

Arguments

`tab`	a data frame with one textual variable
`num.text`	indice of the textual variable
`contingence.by`	a list with the indices of the variables for which a contingence table is calculated by default a contingence table is calculated for all the variables (except the textual one). A contingence table can also be calculated for couple of variables. If `contingence.by` is equal to num.text, then the contingence table is calculated for each row of the data table
`maj.in.min`	boolean, if TRUE majuscule are transformed in minuscule
`sep.word`	a string with all the characters which correspond to separator of words

Value

Returns a list including:

`cont.table`	the contingence table with in rows the categories of the categorical variables (or the couple of categories), and in column the words, and in each cell the number of occurence
`nb.words`	a data.frame with all the words and for each word, the number of lists in which it is present, and the number of occurence

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

See Also

`CA`, `descfreq`

Examples

```
data(poison.text)
res.text <- textual(poison.text, num.text = 3, contingence.by = 1)
descfreq(res.text$cont.table)
## Contingence table for the couple of variable sick-sex
res.text2 <- textual(poison.text, num.text = 3, contingence.by = list(c(1,2)))
descfreq(res.text2$cont.table)
## Contingence table for sex, sick and the couple of variable sick-sex
res.text2 <- textual(poison.text, num.text = 3, contingence.by = list(1,2,c(1,2)))
```

`wine` *Wine*

Description

The data used here refer to 21 wines of Val de Loire.

Usage

```
data(wine)
```

Format

A data frame with 21 rows (the number of wines) and 31 columns: the first column corresponds to the label of origin, the second column corresponds to the soil, and the others correspond to sensory descriptors.

Source

Centre de recherche INRA d'Angers

Examples

```
data(wine)

## Example of PCA
res.pca = PCA(wine,ncp=5, quali.sup = 1:2)

## Not run:
## Example of MCA
res.mca = MCA(wine,ncp=5, quanti.sup = 3:ncol(wine))
```

```
## Example of MFA
res.mfa = MFA(wine,group=c(2,5,3,10,9,2),type=c("n",rep("s",5)),ncp=5,
    name.group=c("orig","olf","vis","olfag","gust","ens"),
    num.group.sup=c(1,6),graph=FALSE)
plotellipses(res.mfa)
plotellipses(res.mfa,keepvar="Label") ## for 1 variable

## End(Not run)
```

`write.infile` *Print in a file*

Description

Print in a file.

Usage

```
write.infile(X, file, sep=";", append = FALSE, nb.dec=4)
```

Arguments

`X`	an object of class list, data.frame, matrix, ...
`file`	A connection, or a character string naming the file to print to
`sep`	character string to insert between the objects to print (if the argument file is not NULL)
`append`	logical. If TRUE output will be appended to file; otherwise, it will overwrite the contents of file.
`nb.dec`	number of decimal printed, by default 4

Author(s)

Francois Husson `<francois.husson@institut-agro.fr>`

Examples

```
## Not run:
data(decathlon)
res.pca <- PCA(decathlon, quanti.sup = 11:12, quali.sup = 13)
write.infile(res.pca, file="c:/essai.csv", sep = ";")

## End(Not run)
```

Index

∗ **Factor analysis**
    FactoMineR-package, 4
∗ **algebra**
    svd.triplet, 99
    tab.disjonctif, 99
    tab.disjonctif.prop, 100
∗ **datasets**
    children, 11
    decathlon, 15
    footsize, 23
    geomorphology, 24
    health, 30
    hobbies, 32
    JO, 33
    milk, 42
    mortality, 42
    poison, 73
    poison.text, 73
    poulet, 74
    senso, 92
    tea, 100
    wine, 102
∗ **dplot**
    autoLab, 5
    coord.ellipse, 14
    graph.var, 26
    plot.CA, 45
    plot.catdes, 50
    plot.DMFA, 51
    plot.FAMD, 52
    plot.GPA, 54
    plot.HCPC, 55
    plot.HMFA, 57
    plot.MCA, 58
    plot.meansComp, 61
    plot.MFA, 61
    plot.PCA, 65
    plotGPApartial, 70
    plotMFApartial, 71
    prefpls, 79
    simule, 92
∗ **models**
    AovSum, 4
    LinearModel, 34
    meansComp, 38
    RegBest, 91
∗ **multivariate**
    CA, 6
    CaGalt, 8
    catdes, 10
    coeffRV, 12
    condes, 13
    descfreq, 16
    dimdesc, 17
    DMFA, 18
    ellipseCA, 19
    estim_ncp, 21
    FAMD, 22
    GPA, 25
    HCPC, 28
    HMFA, 31
    MCA, 35
    MFA, 39
    PCA, 43
    plot.CaGalt, 48
    plotellipses, 68
    print.CaGalt, 81
    reconst, 90
    summary.CaGalt, 94
    textual, 101
∗ **print**
    print.AovSum, 80
    print.CA, 81
    print.catdes, 82
    print.condes, 83
    print.FAMD, 84
    print.GPA, 84
    print.HCPC, 85

`print.HMFA`, 86
`print.LinearModel`, 86
`print.MCA`, 87
`print.MFA`, 88
`print.PCA`, 89
`write.infile`, 103

`agnes`, *28*
`aov`, *5*, *35*
`AovSum`, 4, *80*
`autoLab`, 5

`CA`, 6, *17*, *20*, *47*, *75*, *81*, *90*, *94*, *102*
`CaGalt`, 8, *49*, *82*, *94*
`catdes`, 10, *13*, *16*, *28*, *29*, *50*, *51*, *83*
`children`, 11
`coeffRV`, 12
`condes`, *11*, 13, *16*, *83*
`coord.ellipse`, 14

`decathlon`, 15
`descfreq`, 16, *29*, *102*
`dimdesc`, *7*, 17, *19*, *32*, *37*, *41*, *44*
`DMFA`, 18, *27*, *52*

`ellipseCA`, *7*, 19
`estim_ncp`, 21

`FactoMineR` *(`FactoMineR-package`)*, 4
`FactoMineR-package`, 4
`FAMD`, 22, *54*, *76*, *84*, *95*
`footsize`, 23

`geomorphology`, 24
`GPA`, 25, *55*, *71*, *85*
`graph.var`, 26

`HCPC`, 28, *56*, *85*
`health`, 30
`HMFA`, *17*, *27*, 31, *58*, *86*
`hobbies`, 32

`JO`, 33

`LinearModel`, 34, *76*, *87*
`lm`, *5*, *35*, *91*

`MCA`, *17*, *27*, 35, *60*, *70*, *77*, *88*, *96*
`meansComp`, 38, *61*
`MFA`, *17*, *27*, 39, *64*, *72*, *78*, *88*, *90*, *97*
`milk`, 42
`mortality`, 42

`par`, *27*, *55*, *57*, *69*, *71*, *72*
`PCA`, *17*, *21*, *27*, 43, *67*, *70*, *78*, *89*, *90*, *98*
`plot.CA`, *7*, *20*, 45
`plot.CaGalt`, *9*, 48
`plot.catdes`, *11*, 50
`plot.DMFA`, *19*, 51
`plot.FAMD`, *23*, 52
`plot.GPA`, 54
`plot.HCPC`, *29*, 55
`plot.HMFA`, *32*, 57
`plot.MCA`, *37*, 58
`plot.meansComp`, *38*, 61
`plot.MFA`, *41*, 61, *72*
`plot.PCA`, *44*, 65
`plotellipses`, *37*, 68
`plotGPApartial`, 70
`plotMFApartial`, 71
`poison`, 73
`poison.text`, 73
`poulet`, 74
`predict.CA`, 75
`predict.FAMD`, 75
`predict.LinearModel`, 76
`predict.lm`, *76*
`predict.MCA`, 77
`predict.MFA`, 77
`predict.PCA`, 78
`prefpls`, 79
`print.AovSum`, 80
`print.CA`, *7*, 81
`print.CaGalt`, *9*, 81
`print.catdes`, 82
`print.condes`, 83
`print.FAMD`, *23*, 84
`print.GPA`, 84
`print.HCPC`, 85
`print.HMFA`, *32*, 86
`print.LinearModel`, *35*, 86
`print.MCA`, *37*, 87
`print.MFA`, *41*, 88
`print.PCA`, *44*, 89

`reconst`, 90
`RegBest`, 91

`senso`, 92

`simule`, *15*, 92
`summary.CA`, *7*, 93
`summary.CaGalt`, *9*, 94
`summary.FAMD`, *23*, 95
`summary.MCA`, *37*, 96
`summary.MFA`, *41*, 97
`summary.PCA`, *44*, 98
`svd`, *99*
`svd.triplet`, 99

`tab.disjonctif`, 99
`tab.disjonctif.prop`, 100
`tea`, 100
`textual`, *16*, 101

`wine`, 102
`write.infile`, *81*, *82*, *85*, *86*, *88*, *89*, 103

`xyplot`, *69*