Package ‘factoextra’

April 1, 2020

Type  Package
Title  Extract and Visualize the Results of Multivariate Data Analyses
Version  1.0.7
Date  2020-04-01

Description  Provides some easy-to-use functions to extract and visualize the output of multivariate data analyses, including 'PCA' (Principal Component Analysis), 'CA' (Correspondence Analysis), 'MCA' (Multiple Correspondence Analysis), 'FAMD' (Factor Analysis of Mixed Data), 'MFA' (Multiple Factor Analysis) and 'HMFA' (Hierarchical Multiple Factor Analysis) functions from different R packages. It contains also functions for simplifying some clustering analysis steps and provides 'ggplot2' - based elegant data visualization.

License  GPL-2
LazyData  true
Encoding  UTF-8
Depends  R (>= 3.1.2), ggplot2 (>= 2.2.0)
Imports  abind, cluster, dendextend, FactoMineR, ggpubr(>= 0.1.5), grid, stats, reshape2, ggrepel, tidyr
Suggests  ade4, ca, igraph, MASS, knitr, mclust
URL  http://www.sthda.com/english/rpkgs/factoextra

BugReports  https://github.com/kassambara/factoextra/issues

Collate  'cluster_utilities.R' 'decathlon2.R' 'print.factoextra.R'
          'get_hmfa.R' 'fviz_hmfa.R' 'get_mfa.R' 'fviz_mfa.R'
          'deprecated.R' 'fviz_add.R' 'eigenvalue.R' 'utilities.R'
          'dist.R' 'fviz_dend.R' 'hcut.R' 'get_pca.R' 'fviz_cluster.R'
          'eclust.R' 'facto_summarize.R' 'fviz.R' 'fviz_ca.R'
          'fviz_contrib.R' 'fviz_cos2.R' 'fviz_ellipses.R' 'fviz_famd.R'
          'get_mca.R' 'fviz_mca.R' 'fviz_mclust.R' 'fviz_nbclust.R'
          'fviz_pca.R' 'fviz_silhouette.R' 'get_ca.R'
          'get_clust_tendency.R' 'get_famd.R' 'hkmeans.R' 'housetasks.R'
          'multishapes.R' 'poison.R' 'zzz.R'
RoxygenNote 7.1.0

NeedsCompilation no

Author Alboukadel Kassambara [aut, cre],
Fabian Mundt [aut]

Maintainer Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

Repository CRAN

Date/Publication 2020-04-01 21:20:02 UTC

R topics documented:

decathlon2 .................................................. 3
deprecated .................................................. 4
dist .......................................................... 5
eclust ........................................................ 7
eigenvalue .................................................... 9
facto_summarize ............................................ 11
fviz ........................................................... 14
fviz_add ...................................................... 18
fviz_ca ...................................................... 19
fviz_cluster ................................................ 24
fviz_contrib ............................................... 27
fviz_cos2 ................................................... 30
fviz_dend ................................................... 32
fviz_ellipses ............................................... 35
fviz_famd ................................................... 37
fviz_hmfa ................................................... 40
fviz_mca .................................................... 44
fviz_mclust ................................................ 49
fviz_mfa ..................................................... 51
fviz_nbclust ............................................... 55
fviz_pca ..................................................... 58
fviz_silhouette ............................................ 63
get_ca ....................................................... 65
get_clust_tendency ....................................... 66
get_famd ..................................................... 68
get_hmfa .................................................... 69
get_mca ...................................................... 71
get_mfa ...................................................... 73
get_pca ...................................................... 74
hcut .......................................................... 76
hkmeans ...................................................... 78
housetasks .................................................. 80
multishapes ................................................ 80
poison ....................................................... 81
print.factoextra ......................................... 82

Index 83
decathlon2

Athletes' performance in decathlon

**Description**

Athletes’ performance during two sporting meetings

**Usage**

```r
data("decathlon2")
```

**Format**

A data frame with 27 observations on the following 13 variables.

- X100m: a numeric vector
- Long.jump: a numeric vector
- Shot.put: a numeric vector
- High.jump: a numeric vector
- X400m: a numeric vector
- X110m.hurdle: a numeric vector
- Discus: a numeric vector
- Pole.vault: a numeric vector
- Javeline: a numeric vector
- X1500m: a numeric vector
- Rank: a numeric vector corresponding to the rank
- Points: a numeric vector specifying the point obtained
- Competition: a factor with levels Decastar OlympicG

**Source**

This data is a subset of decathlon data in FactoMineR package.

**Examples**

```r
data(decathlon2)
decathlon.active <- decathlon2[1:23, 1:10]
res.pca <- prcomp(decathlon.active, scale = TRUE)
fviz_pca_biplot(res.pca)
```
Description

Deprecated functions. Will be removed in the next version.

- `get_mfa_var_quanti()`. Deprecated. Use `get_mfa_var(res.mfa, "quanti.var")` instead.
- `get_mfa_var_quali()`. Deprecated. Use `get_mfa_var(res.mfa, "quali.var")` instead.
- `get_mfa_group()`. Deprecated. Use `get_mfa_var(res.mfa, "group")` instead.
- `fviz_mfa_ind_starplot()`: Star graph of individuals (draws partial points). Deprecated. Use `fviz_mfa_ind(res.mfa, partial = "All")` instead.
- `fviz_mfa_quanti_var()`: Graph of quantitative variables. Deprecated. Use `fviz_mfa(X, "quanti.var")` instead.
- `fviz_mfa_quali_var()`: Graph of qualitative variables. Deprecated. Use `fviz_mfa(X, "quali.var")` instead.
- `fviz_hmfa_ind_starplot()`: Graph of partial individuals. Deprecated. Use `fviz_hmfa_ind(X, partial = "all")` instead.
- `fviz_hmfa_quanti_var()`: Graph of quantitative variables. Deprecated. Use `fviz_hmfa_var(X, "quanti.var")` instead.
- `fviz_hmfa_quali_var()`: Graph of qualitative variables. Deprecated. Use `fviz_hmfa_var(X, "quali.var")` instead.
- `fviz_hmfa_group()`: Graph of the groups representation. Deprecated. Use `fviz_hmfa_var(X, "group")` instead.

Usage

```r
get_mfa_quanti_var(res.mfa)
get_mfa_quali_var(res.mfa)
get_mfa_group(res.mfa)
fviz_mfa_ind_starplot(X, ...)
fviz_mfa_group(X, ...)
fviz_mfa_quanti_var(X, ...)
fviz_mfa_quali_var(X, ...)
```
get.hmfa_quanti_var(res.hmfa)
get.hmfa_quali_var(res.hmfa)
get.hmfa_group(res.hmfa)
fviz.hmfa_quanti_var(X, ...)
fviz.hmfa_quali_var(X, ...)
fviz.hmfa_ind_starplot(X, ...)
fviz.hmfa_group(X, ...)

Arguments

res.mfa an object of class MFA [FactoMineR].
X an object of class MFA or HMFA [FactoMineR].
... Other arguments.
res.hmfa an object of class HMFA [FactoMineR].

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

---

dist Enhanced Distance Matrix Computation and Visualization

Description

Clustering methods classify data samples into groups of similar objects. This process requires some methods for measuring the distance or the (dis)similarity between the observations. Read more: STHDA website - clarifying distance measures.

- get.dist(): Computes a distance matrix between the rows of a data matrix. Compared to the standard dist() function, it supports correlation-based distance measures including “pearson”, “kendall” and “spearman” methods.
- fviz.dist(): Visualizes a distance matrix

Usage

get.dist(x, method = "euclidean", stand = FALSE, ...)

fviz.dist(
  dist.obj,
order = TRUE,
show_labels = TRUE,
lab_size = NULL,
gradient = list(low = "red", mid = "white", high = "blue")
)

Arguments

x                a numeric matrix or a data frame.
method           the distance measure to be used. This must be one of "euclidean", "maximum", "manhattan", "canberra", "binary", "minkowski", "pearson", "spearman" or "kendall".
stand            logical value; default is FALSE. If TRUE, then the data will be standardized using the function scale(). Measurements are standardized for each variable (column), by subtracting the variable’s mean value and dividing by the variable’s standard deviation.
...              other arguments to be passed to the function dist() when using get_dist().
dist.obj         an object of class "dist" as generated by the function dist() or get_dist().
order            logical value. if TRUE the ordered dissimilarity image (ODI) is shown.
show_labels      logical value. If TRUE, the labels are displayed.
lab_size         the size of labels.
gradien          a list containing three elements specifying the colors for low, mid and high values in the ordered dissimilarity image. The element "mid" can take the value of NULL.

Value

• get_dist(): returns an object of class "dist".
• fviz_dist(): returns a ggplot2

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

See Also

dist

Examples

data(USArrests)
res.dist <- get_dist(USArrests, stand = TRUE, method = "pearson")

fviz_dist(res.dist,
gradien = list(low = "#00AFBB", mid = "white", high = "#FC4E07"))
Description

Provides solution for enhancing the workflow of clustering analyses and ggplot2-based elegant data visualization. Read more: Visual enhancement of clustering analysis.

Usage

eclust(
  x, 
  FUNcluster = c("kmeans", "pam", "clara", "fanny", "hclust", "agnes", "diana"), 
  k = NULL, 
  k.max = 10, 
  stand = FALSE, 
  graph = TRUE, 
  hc_metric = "euclidean", 
  hc_method = "ward.D2", 
  gap_maxSE = list(method = "firstSEmax", SE.factor = 1), 
  nboot = 100, 
  verbose = interactive(), 
  seed = 123, 
  ... 
)

Arguments

x numeric vector, data matrix or data frame

FUNcluster a clustering function including "kmeans", "pam", "clara", "fanny", "hclust", "agnes" and "diana". Abbreviation is allowed.

k the number of clusters to be generated. If NULL, the gap statistic is used to estimate the appropriate number of clusters. In the case of kmeans, k can be either the number of clusters, or a set of initial (distinct) cluster centers.

k.max the maximum number of clusters to consider, must be at least two.

stand logical value; default is FALSE. If TRUE, then the data will be standardized using the function scale(). Measurements are standardized for each variable (column), by subtracting the variable’s mean value and dividing by the variable’s standard deviation.

gap_maxSE logical value. If TRUE, cluster plot is displayed.

cypher character string specifying the metric to be used for calculating dissimilarities between observations. Allowed values are those accepted by the function dist() [including "euclidean", "manhattan", "maximum", "canberra", "binary", "minkowski"] and correlation based distance measures ["pearson", "spearman" or "kendall"]. Used only when FUNcluster is a hierarchical clustering function such as one of "hclust", "agnes" or "diana".
hc_method  the agglomeration method to be used (?hclust): "ward.D", "ward.D2", "single", "complete", "average", ...

gap_maxSE a list containing the parameters (method and SE.factor) for determining the location of the maximum of the gap statistic (Read the documentation ?cluster::maxSE).

nboot integer, number of Monte Carlo ("bootstrap") samples. Used only for determining the number of clusters using gap statistic.

verbose logical value. If TRUE, the result of progress is printed.

seed integer used for seeding the random number generator.

... other arguments to be passed to FUNcluster.

Value

Returns an object of class "eclust" containing the result of the standard function used (e.g., kmeans, pam, hclust, agnes, diana, etc.).

It includes also:

• cluster: the cluster assignment of observations after cutting the tree
• nbclust: the number of clusters
• silinfo: the silhouette information of observations, including $widths (silhouette width values of each observation), $clus.avg.widths (average silhouette width of each cluster) and $avg.width (average width of all clusters)
• size: the size of clusters
• data: a matrix containing the original or the standardized data (if stand = TRUE)

The "eclust" class has method for fviz_silhouette(), fviz_dend(), fviz_cluster().

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

See Also

fviz_silhouette, fviz_dend, fviz_cluster

Examples

# Load and scale data
data("USArrests")
df <- scale(USArrests)

# Enhanced k-means clustering
# nboot >= 500 is recommended
res.km <- eclust(df, "kmeans", nboot = 2)

# Silhouette plot
fviz_silhouette(res.km)

# Optimal number of clusters using gap statistics
res.km$nbclust
# Print result
res.km

## Not run:
# Enhanced hierarchical clustering
res.hc <- eclust(df, "hclust", nboot = 2) # compute hclust
fviz_dend(res.hc) # dendrogram
fviz_silhouette(res.hc) # silhouette plot

## End(Not run)

eigenvalue

### Extract and visualize the eigenvalues/variances of dimensions

description

Eigenvalues correspond to the amount of the variation explained by each principal component (PC).

- `get_eig()`: Extract the eigenvalues/variances of the principal dimensions
- `fviz_eig()`: Plot the eigenvalues/variances against the number of dimensions
- `get_eigenvalue()`: an alias of `get_eig()`
- `fviz_screeplot()`: an alias of `fviz_eig()

These functions support the results of Principal Component Analysis (PCA), Correspondence Analysis (CA), Multiple Correspondence Analysis (MCA), Factor Analysis of Mixed Data (FAMD), Multiple Factor Analysis (MFA) and Hierarchical Multiple Factor Analysis (HMFA) functions.

Usage

gget_eig(X)
gget_eigenvalue(X)

fviz_eig(X, 
choice = c("variance", "eigenvalue"), 
geom = c("bar", "line"), 
barfill = "steelblue", 
barcolor = "steelblue", 
linecolor = "black", 
ncp = 10, 
addlabels = FALSE, 
hjust = 0, 
main = NULL, 
xlab = NULL, 
ylab = NULL,
ggtheme = theme_minimal(),
...
)

fviz_screeplot(...)

Arguments

X an object of class PCA, CA, MCA, FAMD, MFA and HMFA [FactoMineR]; prcomp and princomp [stats]; dudi, pca, coa and acm [ade4]; ca and mjca [ca package].

choice a text specifying the data to be plotted. Allowed values are "variance" or "eigenvalue".

gem a text specifying the geometry to be used for the graph. Allowed values are "bar" for barplot, "line" for lineplot or c("bar", "line") to use both types.

barfill fill color for bar plot.
barcolor outline color for bar plot.
linecolor color for line plot (when geom contains "line").
ncp a numeric value specifying the number of dimensions to be shown.
addlabels logical value. If TRUE, labels are added at the top of bars or points showing the information retained by each dimension.
hjust horizontal adjustment of the labels.

main, xlab, ylab plot main and axis titles.

ggtheme function, ggplot2 theme name. Default value is theme_pubr(). Allowed values include ggplot2 official themes: theme_gray(), theme_bw(), theme_minimal(), theme_classic(), theme_void(), ....

... optional arguments to be passed to the function ggpar.

Value

- get_eig() (or get_eigenvalue()): returns a data.frame containing 3 columns: the eigenvalues, the percentage of variance and the cumulative percentage of variance retained by each dimension.
- fviz_eig() (or fviz_screeplot()): returns a ggplot2

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References

http://www.sthda.com/english/

See Also

fviz_pca, fviz_ca, fviz_mca, fviz_mfa, fviz_hmfa
Examples

# Principal Component Analysis
# ++++++++++++++++++++++++++
data(iris)
res.pca <- prcomp(iris[, -5], scale = TRUE)

# Extract eigenvalues/variances
get_eig(res.pca)

# Default plot
fviz_eig(res.pca, addlabels = TRUE, ylim = c(0, 85))

# Scree plot - Eigenvalues
fviz_eig(res.pca, choice = "eigenvalue", addlabels=TRUE)

# Use only bar or line plot: geom = "bar" or geom = "line"
fviz_eig(res.pca, geom="line")

## Not run:
# Correspondence Analysis
# +++++++++++++++++++++++++++++++++
library(FactoMineR)
data(housetasks)
res.ca <- CA(housetasks, graph = FALSE)
get_eig(res.ca)
fviz_eig(res.ca, linecolor = "#FC4E07",
barcolor = "#00AFBB", barfill = "#00AFBB")

# Multiple Correspondence Analysis
# +++++++++++++++++++++++++++++++++
library(FactoMineR)
data(poison)
res.mca <- MCA(poison, quanti.sup = 1:2,
quali.sup = 3:4, graph=FALSE)
get_eig(res.mca)
fviz_eig(res.mca, linecolor = "#FC4E07",
barcolor = "#2E9FDF", barfill = "#2E9FDF")

## End(Not run)

---

facto_summarize  Subset and summarize the output of factor analyses

Description

Subset and summarize the results of Principal Component Analysis (PCA), Correspondence Analysis (CA), Multiple Correspondence Analysis (MCA), Factor Analysis of Mixed Data (FAMD), Multiple Factor Analysis (MFA) and Hierarchical Multiple Factor Analysis (HMFA) functions from several packages.
Usage

```r
facto_summarize(
  X,
  element,
  node.level = 1,
  group.names,
  result = c("coord", "cos2", "contrib"),
  axes = 1:2,
  select = NULL
)
```

Arguments

- **X**: an object of class PCA, CA, MCA, FAMD, MFA and HMFA [FactoMineR]; prcomp and princomp [stats]; dudi, pca, coa and acm [ade4]; ca [ca package]; expoOutput [ExPosition].
- **element**: the element to subset from the output. Possible values are "row" or "col" for CA; "var" or "ind" for PCA and MCA; "mca.cor" for MCA; 'quanti.var', 'quali.var', 'group' or 'ind' for FAMD, MFA and HMFA.
- **node.level**: a single number indicating the HMFA node level.
- **group.names**: a vector containing the name of the groups (by default, NULL and the group are named group.1, group.2 and so on).
- **result**: the result to be extracted for the element. Possible values are the combination of c("cos2", "contrib", "coord")
- **axes**: a numeric vector specifying the axes of interest. Default values are 1:2 for axes 1 and 2.
- **select**: a selection of variables. Allowed values are NULL or a list containing the arguments name, cos2 or contrib. Default is list(name = NULL, cos2 = NULL, contrib = NULL):
  - name: is a character vector containing variable names to be selected
  - cos2: if cos2 is in [0, 1], ex: 0.6, then variables with a cos2 > 0.6 are selected. if cos2 > 1, ex: 5, then the top 5 variables with the highest cos2 are selected
  - contrib: if contrib > 1, ex: 5, then the top 5 variables with the highest cos2 are selected.

Details

If length(axes) > 1, then the columns contrib and cos2 correspond to the total contributions and total cos2 of the axes. In this case, the column coord is calculated as x^2 + y^2 + ...+; x, y, ... are the coordinates of the points on the specified axes.

Value

A data frame containing the (total) coord, cos2 and the contribution for the axes.
Author(s)
Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References
http://www.sthda.com/english/

Examples

# Principal component analysis
# +++++++++++++++++++++++++++++
data(decathlon2)
decathlon2.active <- decathlon2[1:23, 1:10]
res.pca <- prcomp(decathlon2.active, scale = TRUE)

# Summarize variables on axes 1:2
facto_summarize(res.pca, "var", axes = 1:2)[,-1]
# Select the top 5 contributing variables
facto_summarize(res.pca, "var", axes = 1:2,
    select = list(contrib = 5))[,-1]
# Select variables with cos2 >= 0.6
facto_summarize(res.pca, "var", axes = 1:2,
    select = list(cos2 = 0.6))[,-1]
# Select by names
facto_summarize(res.pca, "var", axes = 1:2,
    select = list(name = c("X100m", "Discus", "Javeline")))[,-1]

# Summarize individuals on axes 1:2
facto_summarize(res.pca, "ind", axes = 1:2)[,-1]

# Correspondence Analysis
# ++++++++++++++++++++++++++++
# Install and load FactoMineR to compute CA
# install.packages("FactoMineR")
library("FactoMineR")
data("housetasks")
res.ca <- CA(housetasks, graph = FALSE)

# Summarize row variables on axes 1:2
facto_summarize(res.ca, "row", axes = 1:2)[,-1]
# Summarize column variables on axes 1:2
facto_summarize(res.ca, "col", axes = 1:2)[,-1]

# Multiple Correspondence Analysis
# ++++++++++++++++++++++++++++++++++
library(FactoMineR)
data(poison)
res.mca <- MCA(poison, quanti.sup = 1:2,
    quali.sup = 3:4, graph=FALSE)

# Summarize variables on axes 1:2
res <- facto_summarize(res.mca, "var", axes = 1:2)
head(res)
# Summarize individuals on axes 1:2
res <- facto_summarize(res.mca, "ind", axes = 1:2)
head(res)

# Multiple factor Analysis
# +++++++++++++++++++++++++++++++++
library(FactoMineR)
data(poison)
res.mfa <- MFA(poison, group=c(2,2,5,6), type=c("s","n","n","n"),
nam.group=c("desc","desc2","symptom","eat"),
num.group.sup=1:2, graph=FALSE)
# Summarize categorical variables on axes 1:2
res <- facto_summarize(res.mfa, "quali.var", axes = 1:2)
head(res)
# Summarize individuals on axes 1:2
res <- facto_summarize(res.mfa, "ind", axes = 1:2)
head(res)

fviz

Visualizing Multivariate Analyse Outputs

Description

Generic function to create a scatter plot of multivariate analyse outputs, including PCA, CA, MCA and MFA.

Usage

fviz(
  X,
  element,
  axes = c(1, 2),
  geom = "auto",
  label = "all",
  invisible = "none",
  labelsize = 4,
  pointsize = 1.5,
  pointshape = 19,
  arrowsize = 0.5,
  habillage = "none",
  addEllipses = FALSE,
  ellipse.level = 0.95,
  ellipse.type = "norm",
  ellipse.alpha = 0.1,
  mean.point = TRUE,
  color = "black",
  fill = "white",
  alpha = 1,
  gradient.cols = NULL,
fviz

col.row.sup = "darkblue",
col.col.sup = "darkred",
select = list(name = NULL, cos2 = NULL, contrib = NULL),
title = NULL,
axes.linetype = "dashed",
repel = FALSE,
col.circle = "grey70",
circlesize = 0.5,
ggtheme = theme_minimal(),
ggpl = NULL,
font.family = "",
...)

Arguments

X

an object of class PCA, CA, MCA, FAMD, MFA and HMFA [FactoMineR]; prcomp and princomp [stats]; dudi, pca, coa and acm [ade4]; ca [ca package]; expoOutput [ExPosition].

element

the element to subset from the output. Possible values are "row" or "col" for CA; "var" or "ind" for PCA and MCA; "mca.cor" for MCA; 'quanti.var', 'quali.var', 'group' or 'ind' for FAMD, MFA and HMFA.

axes

a numeric vector specifying the axes of interest. Default values are 1:2 for axes 1 and 2.

geom

a text specifying the geometry to be used for the graph. Default value is "auto". Allowed values are the combination of c("point", "arrow", "text"). Use "point" (to show only points); "text" to show only labels; c("point", "text") or c("arrow", "text") to show both types.

label

a text specifying the elements to be labelled. Default value is "all". Allowed values are "none" or the combination of c("ind", "ind.sup", "quali", "var", "quanti.sup", "group.sup"). "ind" can be used to label only active individuals. "ind.sup" is for supplementary individuals. "quali" is for supplementary qualitative variables. "var" is for active variables. "quanti.sup" is for quantitative supplementary variables.

invisible

a text specifying the elements to be hidden on the plot. Default value is "none". Allowed values are the combination of c("ind", "ind.sup", "quali", "var", "quanti.sup", "group.sup").

labelsize

font size for the labels

pointsize

the size of points

pointshape

the shape of points

arrowsize

the size of arrows. Controls the thickness of arrows.

habillage

an optional factor variable for coloring the observations by groups. Default value is "none". If X is a PCA object from FactoMineR package, habillage can also specify the supplementary qualitative variable (by its index or name) to be used for coloring individuals by groups (see ?PCA in FactoMineR).
addEllipses logical value. If TRUE, draws ellipses around the individuals when habillage != "none".

ellipse.level the size of the concentration ellipse in normal probability.

ellipse.type Character specifying frame type. Possible values are "convex", "confidence" or types supported by \texttt{stat_ellipse()} including one of c("t", "norm", "euclid") for plotting concentration ellipses.
  - "convex": plot convex hull of a set of points.
  - "confidence": plot confidence ellipses around group mean points as \texttt{coord.ellipse() [in FactoMineR].}
  - "t": assumes a multivariate t-distribution.
  - "norm": assumes a multivariate normal distribution.
  - "euclid": draws a circle with the radius equal to level, representing the euclidean distance from the center. This ellipse probably won’t appear circular unless \texttt{coord_fixed()} is applied.

ellipse.alpha Alpha for ellipse specifying the transparency level of fill color. Use alpha = 0 for no fill color.

mean.point logical value. If TRUE (default), group mean points are added to the plot.

color color to be used for the specified geometries (point, text). Can be a continuous variable or a factor variable. Possible values include also: "cos2", "contrib", "coord", "x" or "y". In this case, the colors for individuals/variables are automatically controlled by their qualities of representation ("cos2"). contributions ("contrib"), coordinates (\(x^2+y^2\), "coord"), x values ("x") or y values ("y"). To use automatic coloring (by cos2, contrib, ...), make sure that habillage = "none".

fill same as the argument \texttt{color}, but for point fill color. Useful when pointshape = 21, for example.

alpha controls the transparency of individual and variable colors, respectively. The value can variate from 0 (total transparency) to 1 (no transparency). Default value is 1. Possible values include also: "cos2", "contrib", "coord", x" or "y". In this case, the transparency for the individual/variable colors are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates (\(x^2+y^2\), "coord"), x values(x) or y values ("y"). To use this, make sure that habillage = "none".

gradient.cols vector of colors to use for n-colour gradient. Allowed values include brewer and ggsci color palettes.

col.col.sup, col.row.sup colors for the supplementary column and row points, respectively.

select a selection of individuals/variables to be drawn. Allowed values are NULL or a list containing the arguments name, cos2 or contrib:
  - name: is a character vector containing individuals/variables to be drawn
  - cos2: if cos2 is in [0, 1], ex: 0.6, then individuals/variables with a cos2 > 0.6 are drawn. if cos2 > 1, ex: 5, then the top 5 individuals/variables with the highest cos2 are drawn.
  - contrib: if contrib > 1, ex: 5, then the top 5 individuals/variables with the highest contrib are drawn
fviz

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>the title of the graph</td>
</tr>
<tr>
<td>axes.linetype</td>
<td>linetype of x and y axes.</td>
</tr>
<tr>
<td>repel</td>
<td>a boolean, whether to use ggrepel to avoid overplotting text labels or not.</td>
</tr>
<tr>
<td>col.circle</td>
<td>a color for the correlation circle. Used only when X is a PCA output.</td>
</tr>
<tr>
<td>circlesize</td>
<td>the size of the variable correlation circle.</td>
</tr>
<tr>
<td>ggtheme</td>
<td>function, ggplot2 theme name. Default value is theme_pubr(). Allowed values include ggplot2 official themes: theme_gray(), theme_bw(), theme_minimal(), theme_classic(), theme_void(), ....</td>
</tr>
<tr>
<td>ggp</td>
<td>a ggplot. If not NULL, points are added to an existing plot.</td>
</tr>
<tr>
<td>font.family</td>
<td>character vector specifying font family.</td>
</tr>
<tr>
<td>...</td>
<td>Arguments to be passed to the functions ggpubr::ggscatter() &amp; ggpubr::ggpar().</td>
</tr>
</tbody>
</table>

**Value**

a ggplot

**Author(s)**

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

**Examples**

```r
# Principal component analysis
# +++++++++++++++++++++++++++++
data(decathlon2)
decathlon2.active <- decathlon2[1:23, 1:10]
res.pca <- prcomp(decathlon2.active, scale = TRUE)
fviz(res.pca, "ind") # Individuals plot
fviz(res.pca, "var") # Variables plot

# Correspondence Analysis
# ++++++++++++++++++++++++++
# Install and load FactoMineR to compute CA
# install.packages("FactoMineR")
library("FactoMineR")
data("housetasks")
res.ca <- CA(housetasks, graph = FALSE)
fviz(res.ca, "row") # Rows plot
fviz(res.ca, "col") # Columns plot

# Multiple Correspondence Analysis
# +++++++++++++++++++++++++++++++++
library(FactoMineR)
data(poison)
res.mca <- MCA(poison, quanti.sup = 1:2,
               quali.sup = 3:4, graph=FALSE)
fviz(res.mca, "ind") # Individuals plot
```
```
fviz(res.mca, "var") # Variables plot
```

---

### fviz_add

**Add supplementary data to a plot**

**Description**

Add supplementary data to a plot

**Usage**

```
fviz_add(
  ggp,
  df,
  axes = c(1, 2),
  geom = c("point", "arrow"),
  color = "blue",
  addlabel = TRUE,
  labelsize = 4,
  pointsize = 2,
  shape = 19,
  linetype = "dashed",
  repel = FALSE,
  font.family = "",
  ...
)
```

**Arguments**

- **ggp**: a ggplot2 plot.
- **df**: a data frame containing the x and y coordinates.
- **axes**: a numeric vector of length 2 specifying the components to be plotted.
- **geom**: a character specifying the geometry to be used for the graph. Allowed values are "point" or "arrow" or "text".
- **color**: the color to be used.
- **addlabel**: a logical value. If TRUE, labels are added.
- **labelsize**: the size of labels. Default value is 4.
- **pointsize**: the size of points.
- **shape**: point shape when geom = "point".
- **linetype**: the linetype to be used when geom = "arrow".
- **repel**: a boolean, whether to use ggrepel to avoid overplotting text labels or not.
- **font.family**: character vector specifying font family.
- **...**: Additional arguments, not used.
Value

a ggplot2 plot

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References

http://www.sthda.com

Examples

# Principal component analysis
data(decathlon2)
decathlon2.active <- decathlon2[1:23, 1:10]
res.pca <- prcomp(decathlon2.active, scale = TRUE)

# Visualize variables
p <- fviz_pca_var(res.pca)
print(p)

# Add supplementary variables
coord <- data.frame(PC1 = c(-0.7, 0.9), PC2 = c(0.25, -0.07))
rownames(coord) <- c("Rank", "Points")
print(coord)
fviz_add(p, coord, color = "blue", geom = "arrow")

fviz_ca

Visualize Correspondence Analysis

Description

Correspondence analysis (CA) is an extension of Principal Component Analysis (PCA) suited to
analyze frequencies formed by two categorical variables. fviz_ca() provides ggplot2-based elegant
visualization of CA outputs from the R functions: CA [in FactoMineR], ca [in ca], coa [in ade4],
correspondence [in MASS] and expOutput/epCA [in ExPosition]. Read more: Correspondence
Analysis

• fviz_ca_row(): Graph of row variables
• fviz_ca_col(): Graph of column variables
• fviz_ca_biplot(): Biplot of row and column variables
• fviz_ca(): An alias of fviz_ca_biplot()
Usage

fviz_ca_row(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  geom.row = geom,
  shape.row = 19,
  col.row = "blue",
  alpha.row = 1,
  col.row.sup = "darkblue",
  select.row = list(name = NULL, cos2 = NULL, contrib = NULL),
  map = "symmetric",
  repel = FALSE,
  ...
)

fviz_ca_col(
  X,
  axes = c(1, 2),
  shape.col = 17,
  geom = c("point", "text"),
  geom.col = geom,
  col.col = "red",
  col.col.sup = "darkred",
  alpha.col = 1,
  select.col = list(name = NULL, cos2 = NULL, contrib = NULL),
  map = "symmetric",
  repel = FALSE,
  ...
)

fviz_ca_biplot(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  geom.row = geom,
  geom.col = geom,
  label = "all",
  invisible = "none",
  arrows = c(FALSE, FALSE),
  repel = FALSE,
  title = "CA - Biplot",
  ...
)

fviz_ca(X, ...)
Arguments

X an object of class CA [FactoMineR], ca [ca], coa [ade4]; correspondence [MASS] and expOutput/epCA [ExPosition].

axes a numeric vector of length 2 specifying the dimensions to be plotted.

geom a character specifying the geometry to be used for the graph. Allowed values are the combination of c("point", "arrow", "text"). Use "point" (to show only points); "text" to show only labels; c("point", "text") or c("arrow", "text") to show both types.

geom.row, geom.col as geom but for row and column elements, respectively. Default is geom.row = c("point", "text), geom.col = c("point", "text").

shape.row, shape.col the point shapes to be used for row/column variables. Default values are 19 for rows and 17 for columns.

map character string specifying the map type. Allowed options include: "symmetric", "rowprincipal", "colprincipal", "symbiplot", "rowgab", "colgab", "rowgreen" and "colgreen". See details

repel a boolean, whether to use ggrepel to avoid overplotting text labels or not.

... Additional arguments.

- in fviz_ca_row() and fviz_ca_col(): Additional arguments are passed to the functions fviz() and ggpubr::ggpar().
- in fviz_ca_biplot() and fviz_ca(): Additional arguments are passed to fviz_ca_row() and fviz_ca_col().

col.col, col.row color for column/row points. The default values are "red" and "blue", respectively. Can be a continuous variable or a factor variable. Allowed values include also: "cos2", "contrib", "coord", "x" or "y". In this case, the colors for row/column variables are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates (x^2 + y^2, "coord"), x values("x") or y values("y")

col.col.sup, col.row.sup colors for the supplementary column and row points, respectively.

alpha.col, alpha.row controls the transparency of colors. The value can vary from 0 (total transparency) to 1 (no transparency). Default value is 1. Allowed values include also: "cos2", "contrib", "coord", "x" or "y" as for the arguments col.col and col.row.

select.col, select.row a selection of columns/rows to be drawn. Allowed values are NULL or a list containing the arguments name, cos2 or contrib:

- name is a character vector containing column/row names to be drawn
- cos2 if cos2 is in [0, 1], ex: 0.6, then columns/rows with a cos2 > 0.6 are drawn. if cos2 > 1, ex: 5, then the top 5 columns/rows with the highest cos2 are drawn.
- contrib if contrib > 1, ex: 5, then the top 5 columns/rows with the highest contrib are drawn
label

A character vector specifying the elements to be labelled. Default value is "all". Allowed values are "none" or the combination of c("row", "row.sup", "col", "col.sup"). Use "col" to label only active column variables; "col.sup" to label only supplementary columns; etc.

invisible

A character value specifying the elements to be hidden on the plot. Default value is "none". Allowed values are the combination of c("row", "row.sup", "col", "col.sup").

arrows

Vector of two logicals specifying if the plot should contain points (FALSE, default) or arrows (TRUE). First value sets the rows and the second value sets the columns.

title

The title of the graph.

Details

The default plot of (M)CA is a "symmetric" plot in which both rows and columns are in principal coordinates. In this situation, it's not possible to interpret the distance between row points and column points. To overcome this problem, the simplest way is to make an asymmetric plot. This means that, the column profiles must be presented in row space or vice-versa. The allowed options for the argument map are:

- "rowprincipal" or "colprincipal": asymmetric plots with either rows in principal coordinates and columns in standard coordinates, or vice versa. These plots preserve row metric or column metric respectively.
- "symbiplot": Both rows and columns are scaled to have variances equal to the singular values (square roots of eigenvalues), which gives a symmetric biplot but does not preserve row or column metrics.
- "rowgab" or "colgab": Asymmetric maps, proposed by Gabriel & Odoroff (1990), with rows (respectively, columns) in principal coordinates and columns (respectively, rows) in standard coordinates multiplied by the mass of the corresponding point.
- "rowgreen" or "colgreen": The so-called contribution biplots showing visually the most contributing points (Greenacre 2006b). These are similar to "rowgab" and "colgab" except that the points in standard coordinates are multiplied by the square root of the corresponding masses, giving reconstructions of the standardized residuals.

Value

A ggplot

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References

http://www.sthda.com

See Also

get_ca, fviz_pca, fviz_mca
Examples

# Correspondence Analysis
# ++++++++++++++++++++++++++++++
# Install and load FactoMineR to compute CA
# install.packages("FactoMineR")

library("FactoMineR")
data(housetasks)
head(housetasks)
res.ca <- CA(housetasks, graph=FALSE)

# Biplot of rows and columns
# ++++++++++++++++++++++++++
# Symetric Biplot of rows and columns
fviz_ca_biplot(res.ca)

# Asymmetric biplot, use arrows for columns
fviz_ca_biplot(res.ca, map ="rowprincipal",
              arrow = c(FALSE, TRUE))

# Keep only the labels for row points
fviz_ca_biplot(res.ca, label ="row")

# Keep only labels for column points
fviz_ca_biplot(res.ca, label ="col")

# Select the top 7 contributing rows
# And the top 3 columns
fviz_ca_biplot(res.ca,
               select.row = list(contrib = 7),
               select.col = list(contrib = 3))

# Graph of row variables
# +++++++++++++++++++++
# Control automatically the color of row points
# using the "cos2" or the contributions "contrib"
# cos2 = the quality of the rows on the factor map
# Change gradient color
# Use repel = TRUE to avoid overplotting (slow if many points)
fviz_ca_row(res.ca, col.row = "cos2",
            gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
            repel = TRUE)

# You can also control the transparency
# of the color by the "cos2" or "contrib"
fviz_ca_row(res.ca, alpha.row="contrib")

# Select and visualize some rows with select.row argument.
# - Rows with cos2 >= 0.5: select.row = list(cos2 = 0.5)
# - Top 7 rows according to the cos2: select.row = list(cos2 = 7)
# Top 7 contributing rows: select.row = list(contrib = 7)
# Select rows by names: select.row = list(name = c("Breakfast", "Repairs", "Holidays"))

# Example: Select the top 7 contributing rows
fviz_ca_row(res.ca, select.row = list(contrib = 7))

# Graph of column points
# ++++++++++++++++++++++++++++

# Control colors using their contributions
fviz_ca_col(res.ca, col.col = "contrib",
            gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"))

# Select columns with select.col argument
# You can select by contrib, cos2 and name
# as previously described for ind
# Select the top 3 contributing columns
fviz_ca_col(res.ca, select.col = list(contrib = 3))

---

fviz_cluster  
Visualize Clustering Results

**Description**

Provides ggplot2-based elegant visualization of partitioning methods including kmeans [stats package]; pam, clara and fanny [cluster package]; dbscan [fpc package]; Mclust [mclust package]; HCPC [FactoMineR]; hkmeans [factoextra]. Observations are represented by points in the plot, using principal components if ncol(data) > 2. An ellipse is drawn around each cluster.

**Usage**

fviz_cluster(
  object,
  data = NULL,
  choose.vars = NULL,
  stand = TRUE,
  axes = c(1, 2),
  geom = c("point", "text"),
  repel = FALSE,
  show.clust.cent = TRUE,
  ellipse = TRUE,
  ellipse.type = "convex",
  ellipse.level = 0.95,
  ellipse.alpha = 0.2,
fviz_cluster

```
shape = NULL,
pointsize = 1.5,
labelsiz = 12,
main = "Cluster plot",
xlab = NULL,
ylab = NULL,
outlier.color = "black",
outlier.shape = 19,
outlier.pointsiz = pointsize,
outlier.labelsiz = labelsize,
ggtheme = theme_grey(),
```

Arguments

- **object**
  - an object of class "partition" created by the functions pam(), clara() or fanny() in cluster package; "kmeans" [in stats package]; "dbscan" [in fpc package]; "Mclust" [in mclust]; "hkmeans", "eclust" [in factoextra]. Possible value are also any list object with data and cluster components (e.g.: object = list(data = mydata, cluster = myclust)).

- **data**
  - the data that has been used for clustering. Required only when object is a class of kmeans or dbscan.

- **choose.vars**
  - a character vector containing variables to be considered for plotting.

- **stand**
  - logical value; if TRUE, data is standardized before principal component analysis

- **axes**
  - a numeric vector of length 2 specifying the dimensions to be plotted.

- **geom**
  - a text specifying the geometry to be used for the graph. Allowed values are the combination of c("point", "text"). Use "point" (to show only points); "text" to show only labels; c("point", "text") to show both types.

- **repel**
  - a boolean, whether to use ggrepel to avoid overplotting text labels or not.

- **show.clust.cent**
  - logical; if TRUE, shows cluster centers

- **ellipse**
  - logical value; if TRUE, draws outline around points of each cluster

- **ellipse.type**
  - Character specifying frame type. Possible values are 'convex', 'confidence' or types supported by stat_ellipse including one of c("t", "norm", "euclid").

- **ellipse.level**
  - the size of the concentration ellipse in normal probability. Passed for ggplot2::stat_ellipse 's level. Ignored in 'convex'. Default value is 0.95.

- **ellipse.alpha**
  - Alpha for frame specifying the transparency level of fill color. Use alpha = 0 for no fill color.

- **shape**
  - the shape of points.

- **pointsiz**
  - the size of points

- **labelsiz**
  - font size for the labels

- **main**
  - plot main title.
xlab, ylab character vector specifying x and y axis labels, respectively. Use xlab = FALSE and ylab = FALSE to hide xlab and ylab, respectively.

ggtheme function, ggplot2 theme name. Default value is theme_pubr(). Allowed values include ggplot2 official themes: theme_gray(), theme_bw(), theme_minimal(), theme_classic(), theme_void(), ....

Value

return a ggpplot.

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

See Also

fviz_silhouette, hcut, hkmeans.eclust, fviz_dend

Examples

set.seed(123)

# Data preparation
# +++++++++++++++
data("iris")
head(iris)
# Remove species column (5) and scale the data
iris.scaled <- scale(iris[, -5])

# K-means clustering
# +++++++++++++++++++++
km.res <- kmeans(iris.scaled, 3, nstart = 10)

# Visualize kmeas clustering
# use repel = TRUE to avoid overplotting
fviz_cluster(km.res, iris[, -5], ellipse.type = "norm")

# Change the color palette and theme
fviz_cluster(km.res, iris[, -5],
    palette = "Set2", ggtheme = theme_minimal())

## Not run:
# Show points only
fviz_cluster(km.res, iris[, -5], geom = "point")
# Show text only
fviz_cluster(km.res, iris[, -5], geom = "text")
```r
# PAM clustering
# ++++++++++++++++++++
require(cluster)
pam.res <- pam(iris.scaled, 3)
# Visualize pam clustering
fviz_cluster(pam.res, geom = "point", ellipse.type = "norm")

# Hierarchical clustering
# ++++++++++++++++++++++++
# Use hcut() which compute hclust and cut the tree
hc.cut <- hcut(iris.scaled, k = 3, hc_method = "complete")
# Visualize dendrogram
fviz_dend(hc.cut, show_labels = FALSE, rect = TRUE)
# Visualize cluster
fviz_cluster(hc.cut, ellipse.type = "convex")

## End(Not run)
```

---

**fviz_contrib**

*Visualize the contributions of row/column elements*

**Description**

This function can be used to visualize the contribution of rows/columns from the results of Principal Component Analysis (PCA), Correspondence Analysis (CA), Multiple Correspondence Analysis (MCA), Factor Analysis of Mixed Data (FAMD), and Multiple Factor Analysis (MFA) functions.

**Usage**

```r
caller

fviz_contrib(  
X,  
choice = c("row", "col", "var", "ind", "quanti.var", "quali.var", "group",  
"partial.axes"),  
axes = 1,  
fill = "steelblue",  
color = "steelblue",  
sort.val = c("desc", "asc", "none"),  
top = Inf,  
xtickslab.rt = 45,  
ggtheme = theme_minimal(),  
...  
)
```

fviz_pca_contrib(  
X,  
choice = c("row", "col", "var", "ind", "quanti.var", "quali.var", "group",  
"partial.axes"),  
axes = 1,  
fill = "steelblue",  
color = "steelblue",  
sort.val = c("desc", "asc", "none"),  
top = Inf,  
xtickslab.rt = 45,  
ggtheme = theme_minimal(),  
...  
)
X,
choice = c("var", "ind"),
axes = 1,
fill = "steelblue",
color = "steelblue",
sortcontrib = c("desc", "asc", "none"),
top = Inf,
...)

Arguments

X an object of class PCA, CA, MCA, FAMD, MFA and HMFA [FactoMineR]; prcomp and princomp [stats]; dudi, pca, coa and acm [ade4]; ca [ca package].
choice allowed values are "row" and "col" for CA; "var" and "ind" for PCA or MCA; "var", "ind", "quanti.var", "quali.var" and "group" for FAMD, MFA and HMFA.
axes a numeric vector specifying the dimension(s) of interest.
fill a fill color for the bar plot.
color an outline color for the bar plot.
sort.val a string specifying whether the value should be sorted. Allowed values are "none" (no sorting), "asc" (for ascending) or "desc" (for descending).
top a numeric value specifying the number of top elements to be shown.
xticks.lab rt Same as x.text.angle and y.text.angle, respectively. Will be deprecated in the near future.
ggtheme function, ggplot2 theme name. Default value is theme_pubr(). Allowed values include ggplot2 official themes: theme_gray(), theme_bw(), theme_minimal(), theme_classic(), theme_void(), ....
... other arguments to be passed to the function ggpar.
sortcontrib see the argument sort.val

Details

The function fviz_contrib() creates a barplot of row/column contributions. A reference dashed line is also shown on the barplot. This reference line corresponds to the expected value if the contribution were uniform.

For a given dimension, any row/column with a contribution above the reference line could be considered as important in contributing to the dimension.

Value

a ggplot2 plot

Functions

• fviz_pca_contrib: deprecated function. Use fviz_contrib()
Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References

http://www.sthda.com/english/

Examples

# Principal component analysis
# ++++++++++++++++++++++++++
data(decathlon2)
decathlon2.active <- decathlon2[1:23, 1:10]
res.pca <- prcomp(decathlon2.active, scale = TRUE)

# variable contributions on axis 1
fviz_contrib(res.pca, choice = "var", axes = 1, top = 10)

# Change theme and color
fviz_contrib(res.pca, choice = "var", axes = 1,
fill = "lightgray", color = "black") +
theme_minimal() +
theme(axis.text.x = element_text(angle=45))

# Variable contributions on axis 2
fviz_contrib(res.pca, choice = "var", axes = 2)

# Variable contributions on axes 1 + 2
fviz_contrib(res.pca, choice = "var", axes = 1:2)

# Contributions of individuals on axis 1
fviz_contrib(res.pca, choice = "ind", axes = 1)

## Not run:
# Correspondence Analysis
# ++++++++++++++++++++++++++
# Install and load FactoMineR to compute CA
# install.packages("FactoMineR")
library("FactoMineR")
data("housetasks")
res.ca <- CA(housetasks, graph = FALSE)

# Visualize row contributions on axes 1
fviz_contrib(res.ca, choice = "row", axes = 1)

# Visualize column contributions on axes 1
fviz_contrib(res.ca, choice = "col", axes = 1)

# Multiple Correspondence Analysis
# ++++++++++++++++++++++++++
library(FactoMineR)
data(poison)
res.mca <- MCA(poison, quanti.sup = 1:2,
fviz_cos2

Visualize the quality of representation of rows/columns

Description

This function can be used to visualize the quality of representation (cos2) of rows/columns from the results of Principal Component Analysis (PCA), Correspondence Analysis (CA), Multiple Correspondence Analysis (MCA), Factor Analysis of Mixed Data (FAMD), Multiple Factor Analysis (MFA) and Hierarchical Multiple Factor Analysis (HMFA) functions.

Usage

fviz_cos2(
  X,
  choice = c("row", "col", "var", "ind", "quanti.var", "quali.var", "group"),
  axes = 1,
  fill = "steelblue",
  color = "steelblue",
  sort.val = c("desc", "asc", "none"),
  top = Inf,
  xtickslab.rt = 45,
  ggtheme = theme_minimal(),
  ...)

fviz_cos2

Arguments

- **X**: an object of class PCA, CA, MCA, FAMD, MFA and HMFA [FactoMineR]; prcomp and princomp [stats]; dudi, pca, coa and acm [ade4]; ca [ca package].
- **choice**: allowed values are "row" and "col" for CA; "var" and "ind" for PCA or MCA; "var", "ind", "quanti.var", "quali.var" and "group" for FAMD, MFA and HMFA.
- **axes**: a numeric vector specifying the dimension(s) of interest.
- **fill**: a fill color for the bar plot.
- **color**: an outline color for the bar plot.
- **sort.val**: a string specifying whether the value should be sorted. Allowed values are "none" (no sorting), "asc" (for ascending) or "desc" (for descending).
- **top**: a numeric value specifying the number of top elements to be shown.
- **xtickslab.rt**: Same as x.text.angle and y.text.angle, respectively. Will be deprecated in the near future.
- **ggtheme**: function, ggplot2 theme name. Default value is theme_pubr(). Allowed values include ggplot2 official themes: theme_gray(), theme_bw(), theme_minimal(), theme_classic(), theme_void(), ....
- ... not used

Value

a ggplot

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References

http://www.sthda.com/english/

Examples

# Principal component analysis
# ++++++++++++++++++++++++++
data(decathlon2)
decathlon2.active <- decathlon2[1:23, 1:10]
res.pca <- prcomp(decathlon2.active, scale = TRUE)

# variable cos2 on axis 1
fviz_cos2(res.pca, choice="var", axes = 1, top = 10)

# Change color
fviz_cos2(res.pca, choice="var", axes = 1, fill = "lightgray", color = "black")

# Variable cos2 on axes 1 + 2
fviz_cos2(res.pca, choice="var", axes = 1:2)

# cos2 of individuals on axis 1
fviz_cos2(res.pca, choice="ind", axes = 1)

## Not run:
# Correspondence Analysis
# ++++++++++++++++++++++++++
library("FactoMineR")
data("housetasks")
res.ca <- CA(housetasks, graph = FALSE)

# Visualize row cos2 on axes 1
fviz_cos2(res.ca, choice ="row", axes = 1)
# Visualize column cos2 on axes 1
fviz_cos2(res.ca, choice ="col", axes = 1)

# Multiple Correspondence Analysis
# +++++++++++++++++++++++++++++++++
library(FactoMineR)
data(poison)
res.mca <- MCA(poison, quanti.sup = 1:2,
              quali.sup = 3:4, graph=FALSE)

# Visualize individual cos2 on axes 1
fviz_cos2(res.mca, choice ="ind", axes = 1, top = 20)
# Visualize variable categorie cos2 on axes 1
fviz_cos2(res.mca, choice ="var", axes = 1)

# Multiple Factor Analysis
# ++++++++++++++++++++++++ 
library(FactoMineR)
data(poison)
res.mfa <- 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, graph=FALSE)

# Visualize individual cos2 on axes 1
# Select the top 20
fviz_cos2(res.mfa, choice ="ind", axes = 1, top = 20)
# Visualize categorical variable categorie cos2 on axes 1
fviz_cos2(res.mfa, choice ="quali.var", axes = 1)

## End(Not run)
**Description**

Draws easily beautiful dendrograms using either R base plot or ggplot2. Provides also an option for drawing a circular dendrogram and phylogenetic trees.

**Usage**

```r
defviz_dend(
  x,
  k = NULL,
  h = NULL,
  k_colors = NULL,
  palette = NULL,
  show_labels = TRUE,
  color_labels_by_k = TRUE,
  label_cols = NULL,
  labels_track_height = NULL,
  repel = FALSE,
  lwd = 0.7,
  type = c("rectangle", "circular", "phylogenetic"),
  phylo_layout = "layout.auto",
  rect = FALSE,
  rect_border = "gray",
  rect_lty = 2,
  rect_fill = FALSE,
  lower_rect,
  horiz = FALSE,
  cex = 0.8,
  main = "Cluster Dendrogram",
  xlab = "",
  ylab = "Height",
  sub = NULL,
  ggtheme = theme_classic(),
  ...
)
```

**Arguments**

- `x` an object of class dendrogram, hclust, agnes, diana, hcut, hkmeans or HCPC (FactoMineR).
- `k` the number of groups for cutting the tree.
- `h` a numeric value. Cut the dendrogram by cutting at height h. (k overrides h)
- `k_colors`, `palette` a vector containing colors to be used for the groups. It should contain k number of colors. Allowed values include also "grey" for grey color palettes; brewer palettes e.g. "RdBu", "Blues", ...; and scientific journal palettes from ggsci R package, e.g.: "npg", "aaas", "lancer", "jco", "ucsegb", "uchicago", "simpsons" and "rickandmorty".
show_labels a logical value. If TRUE, leaf labels are shown. Default value is TRUE.

color_labels_by_k logical value. If TRUE, labels are colored automatically by group when k != NULL.

label_cols a vector containing the colors for labels.

labels_track_height a positive numeric value for adjusting the room for the labels. Used only when type = "rectangle".

repel logical value. Use repel = TRUE to avoid label overplotting when type = "phylogenetic".

lwd a numeric value specifying branches and rectangle line width.

type type of plot. Allowed values are one of "rectangle", "triangle", "circular", "phylogenetic".

phylo_layout the layout to be used for phylogenetic trees. Default value is "layout.auto". Allowed values include: layout_auto, layout_with_drl, layout_as_tree, layout_gem, layout_mds and layout_with_lgl.

rect logical value specifying whether to add a rectangle around groups. Used only when k != NULL.

rect_border, rect_lty border color and line type for rectangles.

rect_fill a logical value. If TRUE, fill the rectangle.

lower_rect a value of how low should the lower part of the rectangle around clusters. Ignored when rect = FALSE.

horiz a logical value. If TRUE, an horizontal dendrogram is drawn.

cex size of labels

main, xlab, ylab main and axis titles

sub Plot subtitle. If NULL, the method used hierarchical clustering is shown. To remove the subtitle use sub = "".

ggtheme function, ggplot2 theme name. Default value is theme_classic(). Allowed values include ggplot2 official themes: theme_gray(), theme_bw(), theme_minimal(), theme_classic(), theme_void(), ....

... other arguments to be passed to the function plot.dendrogram()

Value

an object of class fviz_dend which is a ggplot with the attributes "dendrogram" accessible using attr(x, "dendrogram"), where x is the result of fviz_dend().

Examples

# Load and scale the data
data(USArrests)
df <- scale(USArrests)
fviz_ellipses

Draw confidence ellipses around the categories

Description

Draw confidence ellipses around the categories

Usage

fviz_ellipses(
  X,
  habillage,
  axes = c(1, 2),
  addEllipses = TRUE,
  ellipse.type = "confidence",
  palette = NULL,
  pointsize = 1,
  ...)
fviz_ellipses

```r
gem = c("point", "text"),
ggtheme = theme_bw(),
...
```

**Arguments**

- **X**
  - an object of class MCA, PCA or MFA.

- **habillage**
  - a numeric vector of indexes of variables or a character vector of names of variables. Can be also a data frame containing grouping variables.

- **axes**
  - a numeric vector specifying the axes of interest. Default values are 1:2 for axes 1 and 2.

- **addEllipses**
  - logical value. If TRUE, draws ellipses around the individuals when habillage != "none".

- **ellipse.type**
  - Character specifying frame type. Possible values are "convex", "confidence" or types supported by `stat_ellipse()` including one of c("t", "norm", "euclid") for plotting concentration ellipses.
    - "convex": plot convex hull of a set o points.
    - "confidence": plot confidence ellipses arrround group mean points as `coord.ellipse()` in FactoMineR.
    - "t": assumes a multivariate t-distribution.
    - "norm": assumes a multivariate normal distribution.
    - "euclid": draws a circle with the radius equal to level, representing the euclidean distance from the center. This ellipse probably won’t appear circular unless `coord_fixed()` is applied.

- **palette**
  - the color palette to be used for coloring or filling by groups. Allowed values include "grey" for grey color palettes; brewer palettes e.g. "RdBu", "Blues", ...; or custom color palette e.g. c("blue", "red"); and scientific journal palettes from ggsci R package, e.g.: "npg", "aaas", "lancet", "jco", "uscsrgb", "uchicago", "simpsons" and "rickandmorty". Can be also a numeric vector of length(groups); in this case a basic color palette is created using the function `palette`.

- **pointsize**
  - the size of points

- **geom**
  - a text specifying the geometry to be used for the graph. Allowed values are the combination of c("point", "text"). Use "point" (to show only points); "text" to show only labels; c("point", "text") to show both types.

- **ggtheme**
  - function, ggplot2 theme name. Default value is theme_pubr(). Allowed values include ggplot2 official themes: theme_gray(), theme_bw(), theme_minimal(), theme_classic(), theme_void(), ....

... Arguments to be passed to the functions ggpubr::ggscatter() & ggpubr::ggsave().

**Value**

- a ggplot
fviz_famd

Author(s)
Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

Examples

# Multiple Correspondence Analysis
# +++++++++++++++++++++++++++++++++
library(FactoMineR)
data(poison)
res.mca <- MCA(poison, quanti.sup = 1:2,
               quali.sup = 3:4, graph=FALSE)

fviz_ellipses(res.mca, 1:4, geom = "point",
              palette = "jco")

fviz_famd

Visualize Factor Analysis of Mixed Data

Description
Factor analysis of mixed data (FAMD) is, a particular case of MFA, used to analyze a data set containing both quantitative and qualitative variables. fviz_famd() provides ggplot2-based elegant visualization of FAMD outputs from the R function: FAMD [FactoMineR].

• fviz_famd_ind(): Graph of individuals
• fviz_famd_var(): Graph of variables
• fviz_famd(): An alias of fviz_famd_ind(res.famd)

Usage
fviz_famd_ind(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  repel = FALSE,
  habillage = "none",
  palette = NULL,
  addEllipses = FALSE,
  col.ind = "blue",
  col.ind.sup = "darkblue",
  alpha.ind = 1,
shape.ind = 19,
col.quali.var = "black",
select.ind = list(name = NULL, cos2 = NULL, contrib = NULL),
gradien.cols = NULL,
...
)
fviz_famd_var(
  X,
  choice = c("var", "quanti.var", "quali.var"),
  axes = c(1, 2),
  geom = c("point", "text"),
  repel = FALSE,
  col.var = "red",
  alpha.var = 1,
  shape.var = 17,
  col.var.sup = "darkgreen",
  select.var = list(name = NULL, cos2 = NULL, contrib = NULL),
  ...
)
fviz_famd(X, ...)

Arguments

X an object of class FAMD [FactoMineR].

axes a numeric vector of length 2 specifying the dimensions to be plotted.

goem a text specifying the geometry to be used for the graph. Allowed values are the combination of c("point", "arrow", "text"). Use "point" (to show only points); "text" to show only labels; c("point", "text") or c("arrow", "text") to show arrows and texts. Using c("arrow", "text") is sensible only for the graph of variables.

repel a boolean, whether to use ggrepel to avoid overplotting text labels or not.

habillage an optional factor variable for coloring the observations by groups. Default value is "none". If X is an MFA object from FactoMineR package, habillage can also specify the index of the factor variable in the data.

palette the color palette to be used for coloring or filling by groups. Allowed values include "grey" for grey color palettes; brewer palettes e.g. "RdBo", "Blues", ..., or custom color palette e.g. c("blue", "red"); and scientific journal palettes from ggsci R package, e.g.: "npg", "aaas", "lancet", "jco", "ucscgb", "uchicago", "simpsons" and "rickandmorthy". Can be also a numeric vector of length(groups); in this case a basic color palette is created using the function palette.

addEllipses logical value. If TRUE, draws ellipses around the individuals when habillage != "none".

col.ind, col.var color for individuals and variables, respectively. Can be a continuous variable or a factor variable. Possible values include also: "cos2", "contrib", "coord",
"x" or "y". In this case, the colors for individuals/variables are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates ($x^2 + y^2$, "coord"), $x$ values ("$x$") or $y$ values ("$y$"). To use automatic coloring (by cos2, contrib, ....), make sure that habillage ="none".

col.ind.sup color for supplementary individuals
alpha.ind, alpha.var controls the transparency of individuals and variables, respectively. The value can variate from 0 (total transparency) to 1 (no transparency). Default value is 1. Possible values include also : "cos2", "contrib", "coord", "$x$" or "$y$". In this case, the transparency for individual/variable colors are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates ($x^2 + y^2$, "coord"), $x$ values ("$x$") or $y$ values ("$y$"). To use this, make sure that habillage ="none".

shape.ind, shape.var point shapes of individuals, variables, groups and axes

col.quali.var color for qualitative variables in fviz_mfa_ind(). Default is "black".

select.ind, select.var a selection of individuals and variables to be drawn. Allowed values are NULL or a list containing the arguments name, cos2 or contrib:

- name is a character vector containing individuals/variables to be drawn
- cos2 if cos2 is in [0, 1], ex: 0.6, then individuals/variables with a cos2 > 0.6 are drawn. if cos2 > 1, ex: 5, then the top 5 individuals/variables with the highest cos2 are drawn.
- contrib if contrib > 1, ex: 5, then the top 5 individuals/variables with the highest cos2 are drawn

gradient.cols vector of colors to use for n-colour gradient. Allowed values include brewer and ggsci color palettes.

... Arguments to be passed to the function fviz()

choice The graph to plot infviz_mfa_var(). Allowed values include one of c("var", quanti.var", "quali.var")

col.var.sup color for supplementary variables.

Value

a ggplot

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

Examples

# Compute FAMD
library("FactoMineR")
data(wine)
res.famd <- FAMD(wine[,c(1,2, 16, 22, 29, 28,30,31)], graph = FALSE)
fviz_hmfa

Visualize Hierarchical Multiple Factor Analysis

Description

Hierarchical Multiple Factor Analysis (HMFA) is an extension of MFA, used in a situation where the data are organized into a hierarchical structure. fviz_hmfa() provides ggplot2-based elegant visualization of HMFA outputs from the R function: HMFA [FactoMineR].

- fviz_hmfa_ind(): Graph of individuals
- fviz_hmfa_var(): Graph of variables
- fviz_hmfa_quali_biplot(): Biplot of individuals and qualitative variables
- fviz_hmfa(): An alias of fviz_hmfa_ind()

Usage

fviz_hmfa_ind(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  repel = FALSE,
  habillage = "none",
  addEllipses = FALSE,
  shape.ind = 19,
  col.ind = "blue",
  col.ind.sup = "darkblue",
  alpha.ind = 1,
  select.ind = list(name = NULL, cos2 = NULL, contrib = NULL),
  partial = NULL,
  col.partial = "group",
)
fviz_hmfa

group.names = NULL,
node.level = 1,
...}

fviz_hmfa_var(
  X,
  choice = c("quanti.var", "quali.var", "group"),
  axes = c(1, 2),
  geom = c("point", "text"),
  repel = FALSE,
  col.var = "red",
  alpha.var = 1,
  shape.var = 17,
  col.var.sup = "darkgreen",
  select.var = list(name = NULL, cos2 = NULL, contrib = NULL),
  ...
)

fviz_hmfa_quali_biplot(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  repel = FALSE,
  habillage = "none",
  title = "Biplot of individuals and qualitative variables - HMFA",
  ...
)

fviz_hmfa(X, ...)

Arguments

X                  an object of class HMFA [FactoMineR].
axes               a numeric vector of length 2 specifying the dimensions to be plotted.
geom               a text specifying the geometry to be used for the graph. Allowed values are the combination of c("point","arrow","text"). Use "point" (to show only points); "text" to show only labels; c("point","text") or c("arrow","text") to show arrows and texts. Using c("arrow","text") is sensible only for the graph of variables.
repel              a boolean, whether to use ggrepel to avoid overplotting text labels or not.
habillage          an optional factor variable for coloring the observations by groups. Default value is "none". If X is an HMFA object from FactoMineR package, habillage can also specify the index of the factor variable in the data.
addEllipses        logical value. If TRUE, draws ellipses around the individuals when habillage != "none".
shape.ind, shape.var
point shapes of individuals and variables, respectively.
col.ind, col.var
color for individuals, partial individuals and variables, respectively. Can be a continuous variable or a factor variable. Possible values include also: "cos2", "contrib", "coord", "x" or "y". In this case, the colors for individuals/variables are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates ($x^2 + y^2$, "coord"), x values ("x") or y values ("y"). To use automatic coloring (by cos2, contrib, ...), make sure that "habillage ="none".

col.ind.sup

controls the transparency of individual, partial individual and variable, respectively. The value can variate from 0 (total transparency) to 1 (no transparency). Default value is 1. Possible values include also: "cos2", "contrib", "coord", "x" or "y". In this case, the transparency for individual/variable colors are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates ($x^2 + y^2$, "coord"), x values ("x") or y values ("y"). To use this, make sure that "habillage ="none".

select.ind, select.var

a selection of individuals and variables to be drawn. Allowed values are NULL or a list containing the arguments name, cos2 or contrib:
- name is a character vector containing individuals/variables to be drawn
- cos2 if cos2 is in [0, 1], ex: 0.6, then individuals/variables with a cos2 > 0.6 are drawn. If cos2 > 1, ex: 5, then the top 5 individuals/variables with the highest cos2 are drawn.
- contrib if contrib > 1, ex: 5, then the top 5 individuals/variables with the highest cos2 are drawn

partial

list of the individuals for which the partial points should be drawn. (by default, partial = NULL and no partial points are drawn). Use partial = "All" to visualize partial points for all individuals.

col.partial
color for partial individuals. By default, points are colored according to the groups.

group.names

a vector containing the name of the groups (by default, NULL and the group are named group.1, group.2 and so on).

node.level

a single number indicating the HMFA node level to plot.

... Arguments to be passed to the function fviz() and ggpubr::ggpar()

choice

the graph to plot. Allowed values include one of c("quanti.var", "quali.var", "group") for plotting quantitative variables, qualitative variables and group of variables, respectively.

col.var.sup

color for supplementary variables.

title

the title of the graph

Value

a ggplot
fviz_hmfa

Author(s)

Fabian Mundt <f.mundt@inventionate.de>
Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References

http://www.sthda.com/english/

Examples

# Hierarchical Multiple Factor Analysis
# ++++++++++++++++++++++++  
# Install and load FactoMineR to compute MFA 
# install.packages("FactoMineR") 
library("FactoMineR") 
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) 

# Graph of individuals 
# ++++++++++++++++++++++++  
# Color of individuals: col.ind = "#2E9FDF" 
# Use repel = TRUE to avoid overplotting (slow if many points) 
fviz_hmfa_ind(res.hmfa, repel = TRUE, col.ind = "#2E9FDF") 

# Color individuals by groups, add concentration ellipses 
# Remove labels: label = "none". 
# Change color palette to "jco". See ?ggpubr::ggpar 
grp <- as.factor(wine[,1]) 
p <- fviz_hmfa_ind(res.hmfa, label="none", habillage=grp, addEllipses=TRUE, palette = "jco") 
print(p)

# Graph of variables 
# ++++++++++++++++++++++++  
# Quantitative variables 
fviz_hmfa_var(res.hmfa, "quanti.var") 
# Graph of categorical variable categories 
fviz_hmfa_var(res.hmfa, "quali.var") 
# Groups of variables (correlation square) 
fviz_hmfa_var(res.hmfa, "group")

# Biplot of categorical variable categories and individuals 
# ++++++++++++++++++++++++  
fviz_hmfa_quali_biplot(res.hmfa)

# Graph of partial individuals (starplot) 
# ++++++++++++++++++++++++  
fviz_hmfa_ind(res.hmfa, partial = "all", palette = "Dark2")
fviz_mca

Visualize Multiple Correspondence Analysis

Description

Multiple Correspondence Analysis (MCA) is an extension of simple CA to analyse a data table containing more than two categorical variables. fviz_mca() provides ggplot2-based elegant visualization of MCA outputs from the R functions: MCA [in FactoMineR], acm [in ade4], and expOutput/epMCA [in ExPosition]. Read more: Multiple Correspondence Analysis Essentials.

- fviz_mca_ind(): Graph of individuals
- fviz_mca_var(): Graph of variables
- fviz_mca_biplot(): Biplot of individuals and variables
- fviz_mca(): An alias of fviz_mca_biplot()

Usage

fviz_mca_ind(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  geom.ind = geom,
  repel = FALSE,
  habillage = "none",
  palette = NULL,
  addEllipses = FALSE,
  col.ind = "blue",
  col.ind.sup = "darkblue",
  alpha.ind = 1,
  shape.ind = 19,
  map = "symmetric",
  select.ind = list(name = NULL, cos2 = NULL, contrib = NULL),
  ...
)

fviz_mca_var(
  X,
  choice = c("var.cat", "mca.cor", "var", "quanti.sup"),
  axes = c(1, 2),
  geom = c("point", "text"),
  geom.var = geom,
  repel = FALSE,
  col.var = "red",
  alpha.var = 1,
  ...)
fviz_mca

shape.var = 17,
col.quanti.sup = "blue",
col.quali.sup = "darkgreen",
map = "symmetric",
select.var = list(name = NULL, cos2 = NULL, contrib = NULL),
...

fviz_mca_biplot(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  geom.ind = geom,
  geom.var = geom,
  repel = FALSE,
  label = "all",
  invisible = "none",
  habillage = "none",
  addEllipses = FALSE,
  palette = NULL,
  arrows = c(FALSE, FALSE),
  map = "symmetric",
  title = "MCA - Biplot",
  ...
)

fviz_mca(X, ...)

Arguments

X
  an object of class MCA [FactoMineR], acm [ade4] and expOutput/epMCA [Ex-
  Position].

axes
  a numeric vector of length 2 specifying the dimensions to be plotted.

geom
  a text specifying the geometry to be used for the graph. Allowed values are
  the combination of c("point", "arrow", "text"). Use "point" (to show only
  points); "text" to show only labels; c("point", "text") or c("arrow", "text")
  to show arrows and texts. Using c("arrow", "text") is sensible only for the
  graph of variables.

geom.ind, geom.var
  as geom but for individuals and variables, respectively. Default is geom.ind =
  c("point", "text"), geom.var = c("point", "text").

repel
  a boolean, whether to use ggrepel to avoid overplotting text labels or not.

habillage
  an optional factor variable for coloring the observations by groups. Default value
  is "none". If X is an MCA object from FactoMineR package, habillage can also
  specify the index of the factor variable in the data.

palette
  the color palette to be used for coloring or filling by groups. Allowed values
  include "grey" for grey color palettes; brewer palettes e.g. "RdBu", "Blues",
fviz_mca

...; or custom color palette e.g. c("blue", "red"); and scientific journal palettes from ggsci R package, e.g.: "npg", "aaas", "lancet", "jco", "ucscgb", "uchicago", "simpsons" and "rickandmorty". Can be also a numeric vector of length(groups); in this case a basic color palette is created using the function palette.

addEllipses logical value. If TRUE, draws ellipses around the individuals when habillage != "none".

col.ind, col.var
color for individuals and variables, respectively. Can be a continuous variable or a factor variable. Possible values include also: "cos2", "contrib", "coord", "x" or "y". In this case, the colors for individuals/variables are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates (x² + y², "coord"), x values("x") or y values("y"). To use automatic coloring (by cos2, contrib, ...), make sure that habillage = "none".

col.ind.sup color for supplementary individuals
alph.ind, alpha.var
controls the transparency of individual and variable colors, respectively. The value can variate from 0 (total transparency) to 1 (no transparency). Default value is 1. Possible values include also: "cos2", "contrib", "coord", "x" or "y". In this case, the transparency for individual/variable colors are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates (x² + y², "coord"), x values("x") or y values("y"). To use this, make sure that habillage = "none".

shape.ind, shape.var
point shapes of individuals and variables.

map character string specifying the map type. Allowed options include: "symmetric", "rowprincipal", "colprincipal", "symbiplot", "rowgab", "colgab", "rowgreen" and "colgreen". See details

select.ind, select.var
a selection of individuals/variables to be drawn. Allowed values are NULL or a list containing the arguments name, cos2 or contrib:

- name is a character vector containing individuals/variables to be drawn
- cos2 if cos2 is in [0, 1], ex: 0.6, then individuals/variables with a cos2 > 0.6 are drawn. if cos2 > 1, ex: 5, then the top 5 individuals/variables with the highest cos2 are drawn.
- contrib if contrib > 1, ex: 5, then the top 5 individuals/variables with the highest contrib are drawn

... Additional arguments.

- in fviz_mca_ind(), fviz_mca_var() and fviz_mca_cor(): Additional arguments are passed to the functions fviz() and ggpubr::ggpar().
- in fviz_mca_biplot() and fviz_mca(): Additional arguments are passed to fviz_mca_ind() and fviz_mca_var().

choice the graph to plot. Allowed values include: i) "var" and "mca.cor" for plotting the correlation between variables and principal dimensions; ii) "var.cat" for variable categories and iii) "quanti.sup" for the supplementary quantitative variables.

col.quanti.sup, col.quali.sup
a color for the quantitative/qualitative supplementary variables.
fviz_mca

label a text specifying the elements to be labelled. Default value is "all". Allowed values are "none" or the combination of c("ind", "ind.sup","var", "quali.sup", "quanti.sup"). "ind" can be used to label only active individuals. "ind.sup" is for supplementary individuals. "var" is for active variable categories. "quali.sup" is for supplementary qualitative variable categories. "quanti.sup" is for quantitative supplementary variables.

invisible a text specifying the elements to be hidden on the plot. Default value is "none". Allowed values are the combination of c("ind", "ind.sup", "var", "quali.sup", "quanti.sup").

arrows Vector of two logicals specifying if the plot should contain points (FALSE, default) or arrows (TRUE). First value sets the rows and the second value sets the columns.

title the title of the graph

Details

The default plot of MCA is a "symmetric" plot in which both rows and columns are in principal coordinates. In this situation, it's not possible to interpret the distance between row points and column points. To overcome this problem, the simplest way is to make an asymmetric plot. The argument "map" can be used to change the plot type. For more explanation, read the details section of fviz_ca documentation.

Value

a ggplot

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

See Also

get_mca, fviz_pca, fviz_ca, fviz_mfa, fviz_hmfa

Examples

# Multiple Correspondence Analysis
# ++++++++++++++++++++++++++++++
# Install and load FactoMineR to compute MCA
# install.packages("FactoMineR")
library("FactoMineR")
data(poison)
poison.active <- poison[1:55, 5:15]
head(poison.active)
res.mca <- MCA(poison.active, graph=FALSE)

# Graph of individuals
# ++++++++++++++++++++++

# Default Plot
fviz_mca

# Color of individuals: col.ind = "steelblue"
fviz_mca_ind(res.mca, col.ind = "steelblue")

# 1. Control automatically the color of individuals
# using the "cos2" or the contributions "contrib"
# cos2 = the quality of the individuals on the factor map
# 2. To keep only point or text use geom = "point" or geom = "text".
fviz_mca_ind(res.mca, col.ind = "cos2", repel = TRUE)

## Not run:
# You can also control the transparency
# of the color by the cos2
fviz_mca_ind(res.mca, alpha.ind="cos2")

## End(Not run)

# Color individuals by groups, add concentration ellipses
# Remove labels: label = "none".
grp <- as.factor(poison.active[, "Vomiting"])
p <- fviz_mca_ind(res.mca, label="none", habillage=grp,
addEllipses=TRUE, ellipse.level=0.95)
print(p)

# Change group colors using RColorBrewer color palettes
# Read more: http://www.sthda.com/english/wiki/ggplot2-colors
p + scale_color_brewer(palette="Dark2") +
  scale_fill_brewer(palette="Dark2")

# Change group colors manually
# Read more: http://www.sthda.com/english/wiki/ggplot2-colors
p + scale_color_manual(values=c("#999999", "#E69F00")) +
  scale_fill_manual(values=c("#999999", "#E69F00"))

# Select and visualize some individuals (ind) with select.ind argument.
# - ind with cos2 >= 0.4: select.ind = list(cos2 = 0.4)
# - Top 20 ind according to the cos2: select.ind = list(cos2 = 20)
# - Top 20 contributing individuals: select.ind = list(contrib = 20)
# - Select ind by names: select.ind = list(name = c("44", "38", "53", "39") )

# Example: Select the top 40 according to the cos2
fviz_mca_ind(res.mca, select.ind = list(cos2 = 20))

# Graph of variable categories
# +++++++++++++++++++++++++++++++
# Default plot: use repel = TRUE to avoid overplotting
fviz_mca_var(res.mca, col.var = "#FC4E07")

# Control variable colors using their contributions
fviz_mclust

# use repel = TRUE to avoid overplotting
fviz_mca_var(res.mca, col.var = "contrib",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"))

# Bplot
# ++++++++++++++++++++++++++
grp <- as.factor(poison.active[, "Vomiting"])
fviz_mca_biplot(res.mca, repel = TRUE, col.var = "#E7B800",
                habillage = grp, addEllipses = TRUE, ellipse.level = 0.95)

## Not run:
# Keep only the labels for variable categories:
fviz_mca_biplot(res.mca, label = "var")

# Keep only labels for individuals
fviz_mca_biplot(res.mca, label = "ind")

# Hide variable categories
fviz_mca_biplot(res.mca, invisible = "var")

# Hide individuals
fviz_mca_biplot(res.mca, invisible = "ind")

# Control automatically the color of individuals using the cos2
fviz_mca_biplot(res.mca, label = "var", col.ind="cos2")

# Change the color by groups, add ellipses
fviz_mca_biplot(res.mca, label="var", col.var = "blue",
                habillage=grp, addEllipses=TRUE, ellipse.level=0.95)

# Select the top 30 contributing individuals
# And the top 10 variables
fviz_mca_biplot(res.mca,
                select.ind = list(contrib = 30),
                select.var = list(contrib = 10))

## End(Not run)

fviz_mclust

Plot Model-Based Clustering Results using ggplot2

Description

Plots the classification, the uncertainty and the BIC values returned by the Mclust() function.

Usage

fviz_mclust(
object,
what = c("classification", "uncertainty", "BIC"),
etype = "norm",
etype.level = 0.4,
ggtheme = theme_classic(),
...)

fviz_mclust_bic(
  object,
  model.names = NULL,
  shape = 19,
  color = "model",
  palette = NULL,
  legend = NULL,
  main = "Model selection",
  xlab = "Number of components",
  ylab = "BIC",
  ...
)

Arguments

object an object of class Mclust
what choose from one of the following three options: 'classification' (default), 'uncertainty' and 'BIC'.
etype Character specifying frame type. Possible values are 'convex', 'confidence' or types supported by stat_ellipse including one of c("t", "norm", "euclid").
etype.level the size of the concentration ellipse in normal probability. Passed for ggplot2::stat_ellipse 's level. Ignored in 'convex'. Default value is 0.95.
ggtheme function, ggplot2 theme name. Default value is theme_pubr(). Allowed values include ggplot2 official themes: theme_gray(), theme_bw(), theme_minimal(), theme_classic(), theme_void(), ....
... other arguments to be passed to the functions fviz_cluster and ggpar.
model.names one or more model names corresponding to models fit in object. The default is to plot the BIC for all of the models fit.
shape point shape. To change point shape by model names use shape = "model".
color point and line color.
palette the color palette to be used for coloring or filling by groups. Allowed values include 'grey' for grey color palettes; brewer palettes e.g. "RdBu", "Blues", ...; or custom color palette e.g. c("blue", "red"); and scientific journal palettes from ggsj R package, e.g.: "npg", "aaas", "lancet", "jco", "ucscgb", "uchicago", "simpsons" and "rickandmorty". Can be also a numeric vector of length(groups); in this case a basic color palette is created using the function palette.
legend character specifying legend position. Allowed values are one of c("top", "bottom", "left", "right", "none"). To remove the legend use legend = "none". Legend position can be also specified using a numeric vector c(x, y); see details section.

main plot main title.
xlab character vector specifying x axis labels. Use xlab = FALSE to hide xlab.

Functions

• fviz_mclust: Plots classification and uncertainty.
• fviz_mclust_bic: Plots the BIC values.

Examples

if(require("mclust")){
  # Compute model-based-clustering
  require("mclust")
  data("diabetes")
  mc <- Mclust(diabetes[, -1])

  # Visualize BIC values
  fviz_mclust_bic(mc)

  # Visualize classification
  fviz_mclust(mc, "classification", geom = "point")
}
Usage

fviz_mfa_ind(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  repel = FALSE,
  habillage = "none",
  palette = NULL,
  addEllipses = FALSE,
  col.ind = "blue",
  col.ind.sup = "darkblue",
  alpha.ind = 1,
  shape.ind = 19,
  col.quali.var.sup = "black",
  select.ind = list(name = NULL, cos2 = NULL, contrib = NULL),
  partial = NULL,
  col.partial = "group",
  ...
)

fviz_mfa_quali_biplot(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  repel = repel,
  title = "Biplot of individuals and qualitative variables - MFA",
  ...
)

fviz_mfa_var(
  X,
  choice = c("quanti.var", "group", "quali.var"),
  axes = c(1, 2),
  geom = c("point", "text"),
  repel = FALSE,
  habillage = "none",
  col.var = "red",
  alpha.var = 1,
  shape.var = 17,
  col.var.sup = "darkgreen",
  palette = NULL,
  select.var = list(name = NULL, cos2 = NULL, contrib = NULL),
  ...
)

fviz_mfa_axes(
  X,
  axes = c(1, 2),
  ...)
fviz_mfa

geom = c("arrow", "text"),
col.axes = NULL,
alpha.axes = 1,
col.circle = "grey70",
select.axes = list(name = NULL, contrib = NULL),
repel = FALSE,
...

fviz_mfa(X, partial = "all", ...)

Arguments

X an object of class MFA [FactoMineR].

axes a numeric vector of length 2 specifying the dimensions to be plotted.

geom a text specifying the geometry to be used for the graph. Allowed values are the combination of c("point","arrow","text"). Use "point" (to show only points); "text" to show only labels; c("point","text") or c("arrow","text") to show arrows and texts. Using c("arrow","text") is sensible only for the graph of variables.

repel a boolean, whether to use ggrepel to avoid overplotting text labels or not.

habillage an optional factor variable for coloring the observations by groups. Default value is "none". If X is an MFA object from FactoMineR package, habillage can also specify the index of the factor variable in the data.

palette the color palette to be used for coloring or filling by groups. Allowed values include 'grey' for grey color palettes; brewer palettes e.g. "RdBu", "Blues", ...; or custom color palette e.g. c("blue", "red"); and scientific journal palettes from ggsci R package, e.g.: "npg", "aaas", "lancet", "jco", "ucscgb", "uchicago", "simpsons" and "rickandmorty". Can be also a numeric vector of length(groups); in this case a basic color palette is created using the function palette.

addEllipses logical value. If TRUE, draws ellipses around the individuals when habillage != "none".

col.ind, col.var, col.axes color for individuals, variables and col.axes respectively. Can be a continuous variable or a factor variable. Possible values include : "cos2", "contrib", "coord", "x" or "y". In this case, the colors for individuals/variables are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates (x^2 + y^2 , "coord"), x values("x") or y values("y"). To use automatic coloring (by cos2, contrib, ...), make sure that habillage = "none".

col.ind.sup color for supplementary individuals

alpha.ind, alpha.var, alpha.axes controls the transparency of individual, variable, group and axes colors, respectively. The value can variate from 0 (total transparency) to 1 (no transparency). Default value is 1. Possible values include : "cos2", "contrib", "coord", "x" or "y". In this case, the transparency for individual/variable colors are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates
fviz_mfa

(x^2 + y^2 , "coord"), x values("x") or y values("y"). To use this, make sure that
habillage = "none".

shape.ind, shape.var
point shapes of individuals, variables, groups and axes

col.quali.var.sup
color for supplementary qualitative variables. Default is "black".

select.ind, select.var, select.axes
a selection of individuals/partial individuals/ variables/groups/axes to be drawn.
Allowed values are NULL or a list containing the arguments name, cos2 or
contrib:
• name is a character vector containing individuals/variables to be drawn
• cos2 if cos2 is in [0, 1], ex: 0.6, then individuals/variables with a cos2 > 0.6
  are drawn. if cos2 > 1, ex: 5, then the top 5 individuals/variables with the
  highest cos2 are drawn.
• contrib if contrib > 1, ex: 5, then the top 5 individuals/variables with the
  highest cos2 are drawn

partial
list of the individuals for which the partial points should be drawn. (by default,
partial = NULL and no partial points are drawn). Use partial = "All" to visualize
partial points for all individuals.

col.partial
color for partial individuals. By default, points are colored according to the
groups.

... Arguments to be passed to the function fviz()

title
the title of the graph

choice
the graph to plot. Allowed values include one of c("quanti.var", "quali.var",
"group") for plotting quantitative variables, qualitative variables and group of
variables, respectively.

col.var.sup
color for supplementary variables.

col.circle
da color for the correlation circle. Used only when X is a PCA output.

Value
a ggplot2 plot

Author(s)

Fabian Mundt <f.mundt@inventionate.de>
Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References
http://www.sthda.com/english/
Examples

```r
# Compute Multiple Factor Analysis
library("FactoMineR")
data(wine)
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)

# Eigenvalues/variances of dimensions
fviz_screeplot(res.mfa)
# Group of variables
fviz_mfa_var(res.mfa, "group")
# Quantitative variables
fviz_mfa_var(res.mfa, "quanti.var", palette = "jco",
   col.var.sup = "violet", repel = TRUE)
# Graph of individuals colored by cos2
fviz_mfa_ind(res.mfa, col.ind = "cos2",
   gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
   repel = TRUE)
# Partial individuals
fviz_mfa_ind(res.mfa, partial = "all")
# Partial axes
fviz_mfa_axes(res.mfa)

# Graph of categorical variable categories
# ++++++++++++++++++++++++++++++++++++++++
data(poison)
res.mfa <- MFA(poison, group=c(2,5,6), type=c("s","n","n","n"),
   name.group=c("desc","desc2","symptom","eat"),
   num.group.sup=1:2, graph=FALSE)

# Plot of qualitative variables
fviz_mfa_var(res.mfa, "quali.var")

# Biplot of categorical variable categories and individuals
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# Use repel = TRUE to avoid overplotting
grp <- as.factor(poison[, "Vomiting"])
fviz_mfa_quaili_biplot(res.mfa, repel = FALSE, col.var = "#E7B800",
   habillage = grp, addEllipses = TRUE, ellipse.level = 0.95)
```

**fviz_nbclust**

Determining and Visualizing the Optimal Number of Clusters
Description

Partitioning methods, such as k-means clustering require the users to specify the number of clusters to be generated.

- `fviz_nbclust()`: Determines and visualize the optimal number of clusters using different methods: **within cluster sums of squares**, **average silhouette** and **gap statistics**.

- `fviz_gap_stat()`: Visualize the gap statistic generated by the function `clusGap()` [in cluster package]. The optimal number of clusters is specified using the "firstmax" method (?:cluster::clusGap).

Read more: Determining the optimal number of clusters

Usage

```r
fviz_nbclust(
  x,
  FUNcluster = NULL,
  method = c("silhouette", "wss", "gap_stat"),
  diss = NULL,
  k.max = 10,
  nboot = 100,
  verbose = interactive(),
  barfill = "steelblue",
  barcolor = "steelblue",
  linecolor = "steelblue",
  print.summary = TRUE,
  ...
)
```

```r
fviz_gap_stat(
  gap.stat,
  linecolor = "steelblue",
  maxSE = list(method = "firstSEmax", SE.factor = 1)
)
```

Arguments

- **x**: numeric matrix or data frame. In the function `fviz_nbclust()`, x can be the results of the function `NbClust()`.

- **FUNcluster**: a partitioning function which accepts as first argument a (data) matrix like x, second argument, say k, k >= 2, the number of clusters desired, and returns a list with a component named cluster which contains the grouping of observations. Allowed values include: `kmeans`, `cluster::pam`, `cluster::clara`, `cluster::fanny`, `hcut`, etc. This argument is not required when x is an output of the function `NbClust:::NbClust()`.

- **method**: the method to be used for estimating the optimal number of clusters. Possible values are "silhouette" (for average silhouette width), "wss" (for total within sum of square) and "gap_stat" (for gap statistics).
fviz_nbclust

diss
k.max
nboot
verbose
barfill, barcolor
linecolor
print.summary
...
gap_stat
maxSE

Value

• fviz_nbclust, fviz_gap_stat: return a ggplot2

Author(s)
Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

See Also

fviz_cluster, eclust

Examples

set.seed(123)

# Data preparation
```r
# ++++++++++++++
data(iris)
head(iris)
# Remove species column (5) and scale the data
iris.scaled <- scale(iris[, -5])

# Optimal number of clusters in the data
# ++++++++++++++++++++++++++++++++++++++
# Examples are provided only for kmeans, but
# you can also use cluster::pam (for pam) or
# hcut (for hierarchical clustering)

### Elbow method (look at the knee)
# Elbow method for kmeans
fviz_nbclust(iris.scaled, kmeans, method = "wss") +
  geom_vline(xintercept = 3, linetype = 2)

# Average silhouette for kmeans
fviz_nbclust(iris.scaled, kmeans, method = "silhouette")

### Gap statistic
library(cluster)
set.seed(123)
# Compute gap statistic for kmeans
# we used B = 10 for demo. Recommended value is ~500
gap_stat <- clusGap(iris.scaled, FUN = kmeans, nstart = 25,
  K.max = 10, B = 10)
print(gap_stat, method = "firstmax")
fviz_gap_stat(gap_stat)

gap_stat <- clusGap(iris.scaled, FUN = hcut, K.max = 10, B = 10)
fviz_gap_stat(gap_stat)

--------
fviz_pca Visualize Principal Component Analysis
--------

Description

Principal component analysis (PCA) reduces the dimensionality of multivariate data, to two or three
that can be visualized graphically with minimal loss of information. fviz_pca() provides ggplot2-
based elegant visualization of PCA outputs from: i) prcomp and princomp [in built-in R stats],
ii) PCA [in FactoMineR], iii) dudi.pca [in ade4] and EpPCA [ExPosition]. Read more: Principal Component Analysis

• fviz_pca_ind(): Graph of individuals
• fviz_pca_var(): Graph of variables
```
fviz_pca

- fviz_pca_biplot(): Biplot of individuals and variables
- fviz_pca(): An alias of fviz_pca_biplot()

Note that, fviz_pca_xxx() functions are wrapper around the core function fviz(), which is also a wrapper around the function ggscatter() [in ggpubr]. Therefore, further arguments, to be passed to the function fviz() and ggscatter(), can be specified in fviz_pca_ind() and fviz_pca_var().

Usage

fviz_pca(X, ...)

fviz_pca_ind(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  geom.ind = geom,
  repel = FALSE,
  habillage = "none",
  palette = NULL,
  addEllipses = FALSE,
  col.ind = "black",
  fill.ind = "white",
  col.ind.sup = "blue",
  alpha.ind = 1,
  select.ind = list(name = NULL, cos2 = NULL, contrib = NULL),
  ...
)

fviz_pca_var(
  X,
  axes = c(1, 2),
  geom = c("arrow", "text"),
  geom.var = geom,
  repel = FALSE,
  col.var = "black",
  fill.var = "white",
  alpha.var = 1,
  col.quanti.sup = "blue",
  col.circle = "grey70",
  select.var = list(name = NULL, cos2 = NULL, contrib = NULL),
  ...
)

fviz_pca_biplot(
  X,
  axes = c(1, 2),
  geom = c("point", "text"),
  geom.ind = geom,
fviz_pca

geom.var = c("arrow", "text"),
col.ind = "black",
fill.ind = "white",
col.var = "steelblue",
fill.var = "white",
gradiant.cols = NULL,
label = "all",
invisible = "none",
repel = FALSE,
habillage = "none",
palette = NULL,
addEllipses = FALSE,
title = "PCA - Biplot",
...
)

Arguments

X an object of class PCA [FactoMineR]; prcomp and princomp [stats]; dudi and pca [ade4]; expOutput/epPCA [ExPosition].

... Additional arguments.

• in fviz_pca_ind() and fviz_pca_var(): Additional arguments are passed to the functions fviz() and ggpubr::ggpar().
• in fviz_pca_biplot() and fviz_pca(): Additional arguments are passed to fviz_pca_ind() and fviz_pca_var().

axes a numeric vector of length 2 specifying the dimensions to be plotted.

geom a text specifying the geometry to be used for the graph. Allowed values are the combination of c("point", "arrow", "text"). Use "point" (to show only points); "text" to show only labels; c("point", "text") or c("arrow", "text") to show arrows and texts. Using c("arrow", "text") is sensible only for the graph of variables.

geom.ind, geom.var as geom but for individuals and variables, respectively. Default is geom.ind = c("point", "text"), geom.var = c("arrow", "text").

repel a boolean, whether to use ggrepel to avoid overplotting text labels or not.

habillage an optional factor variable for coloring the observations by groups. Default value is "none". If X is a PCA object from FactoMineR package, habillage can also specify the supplementary qualitative variable (by its index or name) to be used for coloring individuals by groups (see ?PCA in FactoMineR).

palette the color palette to be used for coloring or filling by groups. Allowed values include "grey" for grey color palettes; brewer palettes e.g. "RdBu", "Blues", ...; or custom color palette e.g. c("blue", "red"); and scientific journal palettes from ggsci R package, e.g.: "npg", "aaas", "lancet", "jco", "ucscgb", "uchicago", "simpsons" and "rickandmorty". Can be also a numeric vector of length(groups); in this case a basic color palette is created using the function palette.

addEllipses logical value. If TRUE, draws ellipses around the individuals when habillage != "none". 
col.ind, col.var

color for individuals and variables, respectively. Can be a continuous variable or a factor variable. Possible values include also: "cos2", "contrib", "coord", "x" or "y". In this case, the colors for individuals/variables are automatically controlled by their qualities of representation ("cos2"), contributions ("contrib"), coordinates (x^2+y^2, "coord"), x values ("x") or y values ("y"). To use automatic coloring (by cos2, contrib, ...), make sure that habillage ="none".

fill.ind, fill.var

same as col.ind and col.var but for the fill color.

col.ind.sup
color for supplementary individuals

alpha.ind, alpha.var

controls the transparency of individual and variable colors, respectively. The value can vary from 0 (total transparency) to 1 (no transparency). Default value is 1. Possible values include also: "cos2", "contrib", "coord", "x" or "y". In this case, the transparency for the individual/variable colors are automatically controlled by their qualities ("cos2"), contributions ("contrib"), coordinates (x^2+y^2, "coord"), x values("x") or y values("y"). To use this, make sure that habillage ="none".

select.ind, select.var

a selection of individuals/variables to be drawn. Allowed values are NULL or a list containing the arguments name, cos2 or contrib:

• name: is a character vector containing individuals/variables to be drawn
• cos2: if cos2 is in [0, 1], ex: 0.6, then individuals/variables with a cos2 > 0.6 are drawn. If cos2 > 1, ex: 5, then the top 5 individuals/variables with the highest cos2 are drawn.
• contrib: if contrib > 1, ex: 5, then the top 5 individuals/variables with the highest contrib are drawn

col.quanti.sup

da color for the quantitative supplementary variables.

col.circle

a color for the correlation circle. Used only when X is a PCA output.

gradient.cols

vector of colors to use for n-colour gradient. Allowed values include brewer and ggsci color palettes.

label

a text specifying the elements to be labelled. Default value is "all". Allowed values are "none" or the combination of c("ind", "ind.sup", "quali", "var", "quanti.sup"). "ind" can be used to label only active individuals. "ind.sup" is for supplementary individuals. "quali" is for supplementary qualitative variables. "var" is for active variables. "quanti.sup" is for quantitative supplementary variables.

invisible

a text specifying the elements to be hidden on the plot. Default value is "none". Allowed values are the combination of c("ind", "ind.sup", "quali", "var", "quanti.sup").

title

the title of the graph

Value

a ggplot

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>
See Also

fviz_ca, fviz_mca

Examples

# Principal component analysis
# ++++++++++++++++++++++++++++++
data(iris)
res.pca <- prcomp(iris[, -5], scale = TRUE)

# Graph of individuals
# +++++++++++++++++++++

# Default plot
# Use repel = TRUE to avoid overplotting (slow if many points)
fviz_pca_ind(res.pca, col.ind = "#00AFBB",
             repel = TRUE)

# 1. Control automatically the color of individuals
# using the "cos2" or the contributions "contrib"
# cos2 = the quality of the individuals on the factor map
# 2. To keep only point or text use geom = "point" or geom = "text".
fviz_pca_ind(res.pca, col.ind = "cos2", geom = "point",
             gradient.cols = c(white", ", "#E9F00", ", "#56B4E9")

# Color individuals by groups, add concentration ellipses
# Change group colors using RColorBrewer color palettes
# Read more: http://www.sthda.com/english/wiki/ggplot2-colors
# Remove labels: label = "none"
fviz_pca_ind(res.pca, label = "none", habillage = iris$Species,
             addEllipses = TRUE, ellipse.level = 0.95, palette = "Dark2")

# Change group colors manually
# Read more: http://www.sthda.com/english/wiki/ggplot2-colors
fviz_pca_ind(res.pca, label = "none", habillage = iris$Species,
             addEllipses = TRUE, ellipse.level = 0.95,
             palette = c("#999999", ", "#E9F00", ", "#56B4E9")

# Select and visualize some individuals (ind) with select.ind argument.
# - ind with cos2 >= 0.96: select.ind = list(cos2 = 0.96)
# - Top 20 ind according to the cos2: select.ind = list(cos2 = 20)
# - Top 20 contributing individuals: select.ind = list(contrib = 20)
# - Select ind by names: select.ind = list(name = c("23", "42", "119") )

# Example: Select the top 40 according to the cos2
fviz_pca_ind(res.pca, select.ind = list(cos2 = 40))
fviz_silhouette

Visualize Silhouette Information from Clustering

Description

Silhouette (Si) analysis is a cluster validation approach that measures how well an observation is clustered and it estimates the average distance between clusters. fviz_silhouette() provides ggplot2-based elegant visualization of silhouette information from i) the result of silhouette(), pam(), clara() and fanny() [in cluster package]; ii) eclust() and hcut() [in factoextra].

Read more: Clustering Validation Statistics.

Usage

fviz_silhouette(sil.obj, label = FALSE, print.summary = TRUE, ...)

Arguments

sil.obj an object of class silhouette: pam, clara, fanny [in cluster package]; eclust and hcut [in factoextra].
label logical value. If true, x axis tick labels are shown
print.summary logical value. If true a summary of cluster silhouettes are printed in fviz_silhouette().
... other arguments to be passed to the function ggpubr::ggpar().
Details

- Observations with a large silhouette Si (almost 1) are very well clustered.
- A small Si (around 0) means that the observation lies between two clusters.
- Observations with a negative Si are probably placed in the wrong cluster.

Value

return a ggplot

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

See Also

fviz_cluster, hcut, hkmeans, eclust, fviz_dend

Examples

set.seed(123)

# Data preparation
# +++++++++++++++
data("iris")
head(iris)
# Remove species column (5) and scale the data
iris.scaled <- scale(iris[, -5])

# K-means clustering
# +++++++++++++++++++++
km.res <- kmeans(iris.scaled, 3, nstart = 2)

# Visualize kmeans clustering
fviz_cluster(km.res, iris[, -5], ellipse.type = "norm")+
theme_minimal()

# Visualize silhouette information
require("cluster")
sil <- silhouette(km.res$cluster, dist(iris.scaled))
fviz_silhouette(sil)

# Identify observation with negative silhouette
neg_sil_index <- which(sil[, "sil_width"] < 0)
sil[neg_sil_index, , drop = FALSE]

## Not run:
# PAM clustering
# ++++++++++++++++++++
require(cluster)
pam.res <- pam(iris.scaled, 3)
# Visualize pam clustering
fviz_cluster(pam.res, ellipse.type = "norm")+
theme_minimal()
# Visualize silhouette information
fviz_silhouette(pam.res)

# Hierarchical clustering
# ++++++++++++++++++++++++
# Use hcut() which compute hclust and cut the tree
hc.cut <- hcut(iris.scaled, k = 3, hc_method = "complete")
# Visualize dendrogram
fviz_dend(hc.cut, show_labels = FALSE, rect = TRUE)
# Visualize silhouette information
fviz_silhouette(hc.cut)

## End(Not run)

---

### get_ca

**Extract the results for rows/columns - CA**

#### Description

Extract all the results (coordinates, squared cosine, contributions and inertia) for the active row/column variables from Correspondence Analysis (CA) outputs.

- `get_ca()`: Extract the results for rows and columns
- `get_ca_row()`: Extract the results for rows only
- `get_ca_col()`: Extract the results for columns only

#### Usage

```r
get_ca(res.ca, element = c("row", "col"))
get_ca_col(res.ca)
get_ca_row(res.ca)
```

#### Arguments

- `res.ca`: an object of class CA [FactoMineR], ca [ca], coa [ade4]; correspondence [MASS].
- `element`: the element to subset from the output. Possible values are "row" or "col".

#### Value

A list of matrices containing the results for the active rows/columns including:

- `coord`: coordinates for the rows/columns
- `cos2`: cos2 for the rows/columns
- `contrib`: contributions of the rows/columns
- `inertia`: inertia of the rows/columns
get_clust_tendency

Assessing Clustering Tendency

Description

Before applying cluster methods, the first step is to assess whether the data is clusterable, a process defined as the **assessing of clustering tendency**. `get_clust_tendency()` assesses clustering tendency using Hopkins' statistic and a visual approach. An ordered dissimilarity image (ODI) is shown. Objects belonging to the same cluster are displayed in consecutive order using hierarchical clustering. For more details and interpretation, see STHDA website: Assessing clustering tendency.
get_clust_tendency

Usage

get_clust_tendency(
    data,
    n,
    graph = TRUE,
    gradient = list(low = "red", mid = "white", high = "blue"),
    seed = 123
)

Arguments

data a numeric data frame or matrix. Columns are variables and rows are samples. Computation are done on rows (samples) by default. If you want to calculate Hopkins statistic on variables, transpose the data before.
n the number of points selected from sample space which is also the number of points selected from the given sample.

graph logical value; if TRUE the ordered dissimilarity image (ODI) is shown.

gradient a list containing three elements specifying the colors for low, mid and high values in the ordered dissimilarity image. The element "mid" can take the value of NULL.

seed an integer specifying the seed for random number generator. Specify seed for reproducible results.

Details

**Hopkins statistic**: If the value of Hopkins statistic is close to 1 (far above 0.5), then we can conclude that the dataset is significantly clusterable.

**VAT (Visual Assessment of cluster Tendency)**: The VAT detects the clustering tendency in a visual form by counting the number of square shaped dark (or colored) blocks along the diagonal in a VAT image.

Value

A list containing the elements:

- hopkins_stat for Hopkins statistic value
- plot for ordered dissimilarity image. This is generated using the function `fviz_dist(dist.obj)`.

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

See Also

`fviz_dist`
Examples

```r
data(iris)

# Clustering tendency
gradient_col = list(low = "steelblue", high = "white")
get_clust_tendency(iris[, -5], n = 50, gradient = gradient_col)

# Random uniformly distributed dataset
# (without any inherent clusters)
set.seed(123)
random_df <- apply(iris[, -5], 2,
  function(x){runif(length(x), min(x), max(x))}
)
get_clust_tendency(random_df, n = 50, gradient = gradient_col)
```

get_famd  

*Extract the results for individuals and variables - FAMD*

Description

Extract all the results (coordinates, squared cosine and contributions) for the active individuals and variables from Factor Analysis of Mixed Date (FAMD) outputs.

- get_famd(): Extract the results for variables and individuals
- get_famd_ind(): Extract the results for individuals only
- get_famd_var(): Extract the results for quantitative and qualitative variables only

Usage

```r
get_famd(res.famd, element = c("ind", "var", "quanti.var", "quali.var"))
get_famd_ind(res.famd)
get_famd_var(res.famd, element = c("var", "quanti.var", "quali.var"))
```

Arguments

- `res.famd`: an object of class FAMD [FactoMineR].
- `element`: the element to subset from the output. Possible values are "ind", "quanti.var" or "quali.var".
Value

- a list of matrices containing the results for the active individuals and variables, including:
  - coord coordinates of individuals/variables.
  - cos2 cos2 values representing the quality of representation on the factor map.
  - contrib contributions of individuals/variables to the principal components.

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

Examples

```r
# Compute FAMD
library("FactoMineR")
data(wine)
res.famd <- FAMD(wine[,c(1,2,16,22,29,28,30,31)], graph = FALSE)

# Extract the results for qualitative variable categories
quali.var <- get_famd_var(res.famd, "quali.var")
print(quali.var)
head(quali.var$coord) # coordinates of qualitative variables

# Extract the results for quantitative variables
quanti.var <- get_famd_var(res.famd, "quanti.var")
print(quanti.var)
head(quanti.var$coord) # coordinates

# Extract the results for individuals
ind <- get_famd_ind(res.famd)
print(ind)
head(ind$coord) # coordinates of individuals
```

get_hmfa

Extract the results for individuals/variables/group/partial axes - HMFA

Description

Extract all the results (coordinates, squared cosine and contributions) for the active individuals/quantitative variables/qualitative variable categories/groups/partial axes from Hierarchical Multiple Factor Analysis (HMFA) outputs.

- get_hmfa(): Extract the results for variables and individuals
get_hmfa

- get_hmfa_ind(): Extract the results for individuals only
- get_mfa_var(): Extract the results for variables (quantitatives, qualitatives and groups)
- get_hmfa_partial(): Extract the results for partial.node.

Usage

get_hmfa(
  res.hmfa,
  element = c("ind", "quanti.var", "quali.var", "group", "partial.node")
)

get_hmfa_ind(res.hmfa)

get_hmfa_var(res.hmfa, element = c("quanti.var", "quali.var", "group"))

get_hmfa_partial(res.hmfa)

Arguments

res.hmfa an object of class HMFA [FactoMineR].

element the element to subset from the output. Possible values are "ind", "quanti.var",
"quali.var", "group" or "partial.node".

Value

a list of matrices containing the results for the active individuals, variables, groups and partial nodes, including :

coord coordinates

cos2 cos2

ccontrib contributions

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

Fabian Mundt <f.mundt@inventionate.de>

Examples

# Multiple Factor Analysis
# ++++++++++++++++++++++++
# Install and load FactoMineR to compute MFA
# install.packages("FactoMineR")
# library("FactoMineR")
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)

# Extract the results for qualitative variable categories
get_mca

```r
var <- get_hmfa_var(res.hmfa, "quali.var")
print(var)
head(var$coord) # coordinates of qualitative variables
head(var$cos2) # cos2 of qualitative variables
head(var$contrib) # contributions of qualitative variables

# Extract the results for individuals
ind <- get_hmfa_ind(res.hmfa)
print(ind)
head(ind$coord) # coordinates of individuals
head(ind$cos2) # cos2 of individuals
head(ind$contrib) # contributions of individuals

# You can also use the function get_hmfa()
get_hmfa(res.hmfa, "ind") # Results for individuals
get_hmfa(res.hmfa, "quali.var") # Results for qualitative variable categories
```

---

get_mca

Extract the results for individuals/variables - MCA

Description

Extract all the results (coordinates, squared cosine and contributions) for the active individuals/variable categories from Multiple Correspondence Analysis (MCA) outputs.

- `get_mca()`: Extract the results for variables and individuals
- `get_mca_ind()`: Extract the results for individuals only
- `get_mca_var()`: Extract the results for variables only

Usage

```r
get_mca(res.mca, element = c("var", "ind", "mca.cor", "quanti.sup"))

get_mca_var(res.mca, element = c("var", "mca.cor", "quanti.sup"))

get_mca_ind(res.mca)
```

Arguments

- `res.mca` an object of class MCA [FactoMineR], acm [ade4], expoOutput/epMCA [ExPosition].
- `element` the element to subset from the output. Possible values are "var" for variables, "ind" for individuals, "mca.cor" for correlation between variables and principal dimensions, "quanti.sup" for quantitative supplementary variables.
Value

a list of matrices containing the results for the active individuals/variable categories including :

coord coordinates for the individuals/variable categories

\textit{cos2} cos2 for the individuals/variable categories

\textit{contrib} contributions of the individuals/variable categories

\textit{inertia} inertia of the individuals/variable categories

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References

http://www.sthda.com/english/

Examples

# Multiple Correspondence Analysis
# ++++++++++++++++++++++++++++++++++++
# Install and load FactoMineR to compute MCA
# install.packages("FactoMineR")
library("FactoMineR")
data(poison)
poison.active <- poison[1:55, 5:15]
head(poison.active[,1:6])
res.mca <- MCA(poison.active, graph=FALSE)

# Extract the results for variable categories
var <- get_mca_var(res.mca)
print(var)
head(var$coord) # coordinates of variables
head(var$cos2) # cos2 of variables
head(var$contrib) # contributions of variables

# Extract the results for individuals
ind <- get_mca_ind(res.mca)
print(ind)
head(ind$coord) # coordinates of individuals
head(ind$cos2) # cos2 of individuals
head(ind$contrib) # contributions of individuals

# You can also use the function get_mca()
get_mca(res.mca, "ind") # Results for individuals
get_mca(res.mca, "var") # Results for variable categories
get_mfa

Extract the results for individuals/variables/group/partial axes - MFA

Description

Extract all the results (coordinates, squared cosine and contributions) for the active individuals/quantitative variables/qualitative variable categories/groups/partial axes from Multiple Factor Analysis (MFA) outputs.

- get_mfa(): Extract the results for variables and individuals
- get_mfa_ind(): Extract the results for individuals only
- get_mfa_var(): Extract the results for variables (quantitatives, qualitatives and groups)
- get_mfa_partial_axes(): Extract the results for partial axes only

Usage

get_mfa(
  res.mfa,
  element = c("ind", "quanti.var", "quali.var", "group", "partial.axes")
)

get_mfa_ind(res.mfa)

get_mfa_var(res.mfa, element = c("quanti.var", "quali.var", "group"))

get_mfa_partial_axes(res.mfa)

Arguments

- res.mfa: an object of class MFA [FactoMineR].
- element: the element to subset from the output. Possible values are "ind", "quanti.var", "quali.var", "group" or "partial.axes".

Value

A list of matrices containing the results for the active individuals/quantitative variable categories/qualitative variable categories/groups/partial axes including:

- coord: coordinates for the individuals/quantitative variable categories/qualitative variable categories/groups/partial axes
- cos2: cos2 for the individuals/quantitative variable categories/qualitative variable categories/groups/partial axes
- contrib: contributions of the individuals/quantitative variable categories/qualitative variable categories/groups/partial axes
- inertia: inertia of the individuals/quantitative variable categories/qualitative variable categories/groups/partial axes
get_pca

Extract the results for individuals/variables - PCA

Description

Extract all the results (coordinates, squared cosine, contributions) for the active individuals/variables from Principal Component Analysis (PCA) outputs.

- get_pca(): Extract the results for variables and individuals
- get_pca_ind(): Extract the results for individuals only
- get_pca_var(): Extract the results for variables only
get_pca

Usage

get_pca(res.pca, element = c("var", "ind"))

get_pca_ind(res.pca, ...)

get_pca_var(res.pca)

Arguments

res.pca an object of class PCA [FactoMineR]; prcomp and princomp [stats]; pca, dudi [adea4]; epPCA [ExPosition].
element the element to subset from the output. Allowed values are "var" (for active variables) or "ind" (for active individuals).
... not used

Value

a list of matrices containing all the results for the active individuals/variables including:

coord coordinates for the individuals/variables
cos2 cos2 for the individuals/variables
contrib contributions of the individuals/variables

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

References

http://www.sthda.com/english/

Examples

# Principal Component Analysis
# +++++++++++++++++++++++++++++
data(iris)
res.pca <- prcomp(iris[, -5], scale = TRUE)
# Extract the results for individuals
ind <- get_pca_ind(res.pca)
print(ind)
head(ind$coord) # coordinates of individuals
head(ind$cos2) # cos2 of individuals
head(ind$contrib) # contributions of individuals

# Extract the results for variables
var <- get_pca_var(res.pca)
print(var)
head(var$coord) # coordinates of variables
head(var$cos2) # cos2 of variables
head(var$contrib) # contributions of variables

# You can also use the function get_pca()
get_pca(res.pca, "ind") # Results for individuals
get_pca(res.pca, "var") # Results for variable categories

**hcut**

*Computes Hierarchical Clustering and Cut the Tree*

**Description**

Computes hierarchical clustering (hclust, agnes, diana) and cut the tree into k clusters. It also accepts correlation based distance measure methods such as "pearson", "spearman" and "kendall".

**Usage**

```r
hc = function(
  x, 
  k = 2, 
  isdiss = inherits(x, "dist"), 
  hc_func = c("hclust", "agnes", "diana"), 
  hc_method = "ward.D2", 
  hc_metric = "euclidean", 
  stand = FALSE, 
  graph = FALSE, 
  ... 
)
```

**Arguments**

- **x**
  - a numeric matrix, numeric data frame or a dissimilarity matrix.
- **k**
  - the number of clusters to be generated.
- **isdiss**
  - logical value specifying whether x is a dissimilarity matrix.
- **hc_func**
  - the hierarchical clustering function to be used. Default value is "hclust". Possible values is one of "hclust", "agnes", "diana". Abbreviation is allowed.
- **hc_method**
  - the agglomeration method to be used (?hclust) for hclust() and agnes(): "ward.D", "ward.D2", "single", "complete", "average", ...
- **hc_metric**
  - character string specifying the metric to be used for calculating dissimilarities between observations. Allowed values are those accepted by the function dist() [including "euclidean", "manhattan", "maximum", "canberra", "binary", "minkowski"] and correlation based distance measures ["pearson", "spearman" or "kendall"].
- **stand**
  - logical value; default is FALSE. If TRUE, then the data will be standardized using the function scale(). Measurements are standardized for each variable (column), by subtracting the variable’s mean value and dividing by the variable’s standard deviation.
graph logical value. If TRUE, the dendrogram is displayed.
...
not used.

Value

an object of class "hcut" containing the result of the standard function used (read the documentation of hclust, agnes, diana).

It includes also:

- cluster: the cluster assignment of observations after cutting the tree
- nbclust: the number of clusters
- silinfo: the silhouette information of observations (if k > 1)
- size: the size of clusters
- data: a matrix containing the original or the standardized data (if stand = TRUE)

See Also

fviz_dend, hkmeans, eclust

Examples

data(USArrests)

# Compute hierarchical clustering and cut into 4 clusters
res <- hcut(USArrests, k = 4, stand = TRUE)

# Cluster assignments of observations
res$cluster
# Size of clusters
res$size

# Visualize the dendrogram
fviz_dend(res, rect = TRUE)

# Visualize the silhouette
fviz_silhouette(res)

# Visualize clusters as scatter plots
fviz_cluster(res)
**hkmeans**

*Hierarchical k-means clustering*

**Description**

The final k-means clustering solution is very sensitive to the initial random selection of cluster centers. This function provides a solution using an hybrid approach by combining the hierarchical clustering and the k-means methods. The procedure is explained in "Details" section. Read more: Hybrid hierarchical k-means clustering for optimizing clustering outputs.

- `hkmeans()`: compute hierarchical k-means clustering
- `print.hkmeans()`: prints the result of `hkmeans`
- `hkmeans_tree()`: plots the initial dendrogram

**Usage**

```r
hkmeans(
  x,
  k,
  hc.metric = "euclidean",
  hc.method = "ward.D2",
  iter.max = 10,
  km.algorithm = "Hartigan-Wong"
)
```

## S3 method for class 'hkmeans'

```r
print(x, ...)
```

```r
hkmeans_tree(hkmeans, rect.col = NULL, ...)
```

**Arguments**

- `x` a numeric matrix, data frame or vector
- `k` the number of clusters to be generated
- `hc.metric` the distance measure to be used. Possible values are "euclidean", "maximum", "manhattan", "canberra", "binary" or "minkowski" (see `?dist`).
- `hc.method` the agglomeration method to be used. Possible values include "ward.D", "ward.D2", "single", "complete", "average", "mcquitty", "median" or "centroid" (see `?hclust`).
- `iter.max` the maximum number of iterations allowed for k-means.
- `km.algorithm` the algorithm to be used for kmeans (see `?kmeans`).
- `...` others arguments to be passed to the function `plot.hclust()` (see `?plot.hclust`)
- `hkmeans` an object of class `hkmeans` (returned by the function `hkmeans()`)
- `rect.col` Vector with border colors for the rectangles around clusters in dendrogram
Details

The procedure is as follow:
1. Compute hierarchical clustering
2. Cut the tree in k-clusters
3. Compute the center (i.e the mean) of each cluster
4. Do k-means by using the set of cluster centers (defined in step 3) as the initial cluster centers. Optimize the clustering.

This means that the final optimized partitioning obtained at step 4 might be different from the initial partitioning obtained at step 2. Consider mainly the result displayed by `fviz_cluster()`.

Value

`hkmeans` returns an object of class "hkmeans" containing the following components:

- The elements returned by the standard function `kmeans()` (see `?kmeans`)
- data: the data used for the analysis
- hclust: an object of class "hclust" generated by the function `hclust()`

Examples

```r
# Load data
data(USArrests)
# Scale the data
df <- scale(USArrests)

# Compute hierarchical k-means clustering
res.hk <- hkmeans(df, 4)

# Elements returned by `hkmeans()`
names(res.hk)

# Print the results
res.hk

# Visualize the tree
hkmeans_tree(res.hk, cex = 0.6)
# or use this
fviz_dend(res.hk, cex = 0.6)

# Visualize the `hkmeans` final clusters
fviz_cluster(res.hk, frame.type = "norm", frame.level = 0.68)
```
housetasks  

*House tasks contingency table*

**Description**

A data frame containing the frequency of execution of 13 house tasks in the couple. This table is also available in ade4 package.

**Usage**

```r
data("housetasks")
```

**Format**

A data frame with 13 observations (house tasks) on the following 4 columns.

- **Wife** a numeric vector
- **Alternating** a numeric vector
- **Husband** a numeric vector
- **Jointly** a numeric vector

**Source**

This data is from FactoMineR package.

**Examples**

```r
library(FactoMineR)
data(housetasks)
res.ca <- CA(housetasks, graph=FALSE)
fviz_ca_biplot(res.ca, repel = TRUE)+
  theme_minimal()
```

---

multishapes  

*A dataset containing clusters of multiple shapes*

**Description**

Data containing clusters of any shapes. Useful for comparing density-based clustering (DBSCAN) and standard partitioning methods such as k-means clustering.

**Usage**

```r
data("multishapes")
```
**poison**

**Format**

A data frame with 1100 observations on the following 3 variables.

- `x` a numeric vector containing the x coordinates of observations
- `y` a numeric vector containing the y coordinates of observations
- `shape` a numeric vector corresponding to the cluster number of each observations.

**Details**

The dataset contains 5 clusters and some outliers/noises.

**Examples**

```r
data(multishapes)
plot(multishapes[,1], multishapes[, 2],
     col = multishapes[, 3], pch = 19, cex = 0.8)
```

---

**poison**  

*Poison*

---

**Description**

This data is a result from a survey carried out on children of primary school who suffered from food poisoning. They were asked about their symptoms and about what they ate.

**Usage**

```r
data("poison")
```

**Format**

A data frame with 55 rows and 15 columns.

**Source**

This data is from FactoMineR package.
library(FactoMineR)
data(poison)
res.mca <- MCA(poison, quanti.sup = 1:2, quali.sup = c(3,4),
  graph = FALSE)
fviz_mca_biplot(res.mca, repel = TRUE)+
theme_minimal()

print.factoextra

Print method for an object of class factoextra

Description
Print method for an object of class factoextra

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

Arguments

x an object of class factoextra

... further arguments to be passed to print method

Author(s)

Alboukadel Kassambara <alboukadel.kassambara@gmail.com>

Examples

data(iris)
res.pca <- prcomp(iris[, -5], scale = TRUE)
ind <- get_pca_ind(res.pca, data = iris[, -5])
print(ind)
Index

clara, 63
clusGap, 56
coord.ellipse, 16, 36
coord_fixed, 16, 36
decathlon2, 3
deprecated, 4
dist, 5, 5, 6
eclust, 7, 26, 57, 63, 64, 77
eigenvalue, 9

facto_summarize, 11
fanny, 63
fviz, 14, 59
fviz_add, 18
fviz_ca, 10, 19, 47, 62
fviz_ca_biplot (fviz_ca), 19
fviz_ca_col (fviz_ca), 19
fviz_ca_row (fviz_ca), 19
fviz_cluster, 8, 24, 50, 57, 64
fviz_contrib, 27
fviz_cos2, 30
fviz_dend, 8, 26, 32, 64, 77
fviz_dist, 67
fviz_dist (dist), 5
fviz_eig (eigenvalue), 9
fviz_ellipses, 35
fviz_famd, 37
fviz_famd_ind (fviz_famd), 37
fviz_famd_var (fviz_famd), 37
fviz_gap_stat (fviz_nbclust), 55
fviz_hmfa, 10, 40, 47
fviz_hmfa_group (deprecated), 4
fviz_hmfa_ind (fviz_hmfa), 40
fviz_hmfa_ind_starplot (deprecated), 4
fviz_hmfa_quali_biplot (fviz_hmfa), 40
fviz_hmfa_quali_var (deprecated), 4
fviz_hmfa_quanti_var (deprecated), 4
fviz_hmfa_var (fviz_hmfa), 40
fviz_mca, 10, 22, 44, 62
fviz_mca_biplot (fviz_mca), 44
fviz_mca_ind (fviz_mca), 44
fviz_mca_var (fviz_mca), 44
fviz_mclust, 49
fviz_mclust_bic (fviz_mclust), 49
fviz_mfa, 10, 47, 51
fviz_mfa_axes (fviz_mfa), 51
fviz_mfa_group (deprecated), 4
fviz_mfa_ind (fviz_mfa), 51
fviz_mfa_ind_starplot (deprecated), 4
fviz_mfa quali_biplot (fviz_mfa), 51
fviz_mfa quali_var (deprecated), 4
fviz_mfa quanti_var (deprecated), 4
fviz_mfa_var (fviz_mfa), 51
fviz_nbclust, 55
fviz_pca, 10, 22, 47, 58
fviz_pca_biplot (fviz_pca), 58
fviz_pca_contrib (fviz_contrib), 27
fviz_pca_ind, 59
fviz_pca_ind (fviz_pca), 58
fviz_pca_var, 59
fviz_pca_var (fviz_pca), 58
fviz_screeplot (eigenvalue), 9
fviz_silhouette, 8, 26, 63

gca, 22, 65
gca (get_ca), 65
gca_col (get_ca), 65
gca_row (get_ca), 65
gclust_tendency, 66
gdist (dist), 5
gget_eig (eigenvalue), 9
gget_eigenvalue (eigenvalue), 9
gget_famd, 68
gget_famd_ind (get_famd), 68
gget_famd_var (get_famd), 68
gget_hmfa, 69
gget_hmfa_group (deprecated), 4
gget_hmfa_ind (get_hmfa), 69
gget_hmfa_partial (get_hmfa), 69
get_hmfa_quali_var (deprecated), 4
get_hmfa_quanti_var (deprecated), 4
get_hmfa_var (get_hmfa), 69
get_mca, 47, 71
get_mca_ind (get_mca), 71
get_mca_var (get_mca), 71
get_mfa, 73
get_mfa_group (deprecated), 4
get_mfa_ind (get_mfa), 73
get_mfa_partial_axes (get_mfa), 73
get_mfa_quali_var (deprecated), 4
get_mfa_quanti_var (deprecated), 4
get_mfa_var (get_mfa), 73
get_pca, 74
get_pca_ind (get_pca), 74
get_pca_var (get_pca), 74
ggpar, 10, 26, 28, 50
ggscatter, 26, 59

hcun, 26, 63, 64, 76
hkmeans, 26, 64, 77, 78
hkmeans_tree (hkmeans), 78
housetasks, 80

layout.auto, 34
layout.gem, 34
layout.mds, 34
layout_as_tree, 34
layout_with_drl, 34
layout_with_lgl, 34

multishapes, 80

palette, 36, 38, 46, 50, 53, 60
pam, 63
poison, 81
print.factoextra, 82
print.hkmeans (hkmeans), 78

silhouette, 63
stat_ellipse, 16, 25, 36, 50