Package: soc.ca (via r-universe)

October 13, 2024

Title Specific Correspondence Analysis for the Social Sciences

Description Specific and class specific multiple correspondence analysis on survey-like data. Soc.ca is optimized to the needs of the social scientist and presents easily interpretable results in near publication ready quality.

URL https://github.com/Rsoc/soc.ca

Version 0.8.1

Maintainer Anton Grau Larsen <agraul@ruc.dk>

Author Anton Grau Larsen and Jacob Lunding with contributions from Christoph Ellersgaard and Stefan Andrade

Depends R (>= 3.5.0), ggplot2

Imports gridExtra, ellipse, stats, utils, FactoMineR, stringr, Matrix, htmlTable, flextable, ggpp, ggrepel, shiny, purrr, RColorBrewer, tibble, dplyr, forcats, rlang, magrittr, reshape2, GDAtools, tidyr, igraph

License GPL-3

RoxygenNote 7.2.3

Suggests knitr, rmarkdown

Encoding UTF-8

Repository https://rsoc.r-universe.dev

RemoteUrl https://github.com/rsoc/soc.ca

RemoteRef HEAD

RemoteSha 753cee03681abe0a63dd42b20fc1154b68e195c6

Contents

`add.cases` *Add a layer of cases (individuals) to an mca map*

Description

Add a layer of cases (individuals) to an mca map

Usage

```
add.cases(
  object,
  dim = c(1, 2),
  ind = extract_ind(object, dim),
  mapping = aes(),
  ...
)

add.ind(
  object,
  dim = c(1, 2),
  ind = extract_ind(object, dim),
  mapping = aes(),
  ...
)
```

Arguments

`object`	a soc.mca result object
`dim`	a numeric vector with the plotted dimensions
`ind`	a data.frame with coordinates of cases as produced by extract_ind. This controls the plotted points.
`mapping`	a call to aes from the ggplot2 package. Here you can map aesthetics to variables such as color, fill, alpha, size and shape.
`...`	further arguments are passed on to geom_point()

Value

a ggplot2 object that can be added to an existing plot like those produced by map.ca.base

Examples

```
example(soc.mca)
map.ca.base() + add.cases(result)
```

`add.categories` *Add a layer of categories (modalities) to an mca map*

Description

Add a layer of categories (modalities) to an mca map

Usage

```
The presets adds, replaces or filters from the categories in cats.
  "ctr" returns all categories contributing above average to the plane
  defined in dim. "sup" returns the supplementary categories from object.
  "all" returns both supplementary and active categories.
```

Arguments

`object`	a soc.mca result object
`preset`	a character string selecting among presets. If "active" - no change is made to "cats".
`dim`	a numeric vector with the dimensions for the plane
`cats`	a data.frame with coordinates of categories as produced by extract_mod or extract_sup. This controls the plotted points.
`mapping`	a call to aes from the ggplot2 package. Here you can map aesthetics to variables such as color, alpha, size and family.
`repel`	if TRUE label position is adjusted to lower overlap
`check_overlap`	if TRUE overlapping categories are removed
`points`	if TRUE points are plotted
`...`	further arguments are passed onto geom_text or geom_text_repel

Value

a ggplot2 object that can be added to an existing plot like those produced by map.ca.base

Examples

```
example(soc.mca)
map.ca.base() + add.categories(result, check_overlap = TRUE)
map.ca.base() + add.categories(result, preset = "all", mapping = aes(color = type, label = Modality), repel = TRUE)
```

`add.count` *Add a new layer of points on top of an existing plot with output from the min_cut function*

Description

Add a new layer of points on top of an existing plot with output from the min_cut function

Usage

```
add.count(x, p, label = TRUE, ...)
```

Arguments

`x`	a matrix created by the min_cut function
`p`	is a ggplot object, preferably from one of the mapping functions in soc.ca
`label`	if TRUE the labels of points will be shown
`...`	further arguments are passed on to geom_path, geom_point and geom_text

`add.density` *Add a layer with density curves to an mca map.*

Description

Add a layer with density curves to an mca map.

Usage

```
add.density(
  object,
  dim = c(1, 2),
  ind = extract_ind(object, dim),
  mapping = aes(),
  ...
)
```

Arguments

`object`	a soc.mca result object
`dim`	a numeric vector with the dimensions for the plane
`ind`	a data.frame with coordinates of cases as produced by extract_ind. This controls the points that are used for the density curves.
`mapping`	a call to aes from the ggplot2 package. Here you can map aesthetics to variables such as color, fill, alpha, size and linetype.
`...`	further arguments are passed onto geom_density_2d

`add.ellipse` *Add a layer with concentration ellipses to an mca map.*

Description

Add a layer with concentration ellipses to an mca map.

Usage

```
add.ellipse(
  object,
  var = NULL,
  draw = unique(var),
  dim = c(1, 2),
  el = ellipses(object, var = var, dim = dim),
  mapping = aes(color = Category),
  draw.axis = TRUE,
  ...
)
```

Arguments

`object`	a soc.mca result object
`var`	a factor
`draw`	a character vector with the levels to draw ellipses for
`dim`	a numeric vector with the dimensions for the plane
`el`	a data.frame produced by the ellipses function.
`mapping`	a call to aes from the ggplot2 package. Here you can map aesthetics to variables such as color, fill, alpha, size and linetype.
`draw.axis`	if TRUE the axis within the concentration ellipse is drawn.
`...`	further arguments is passed onto geom_path and geom_line

Value

a ggplot2 object that can be added to an existing plot like those produced by map.ca.base

Examples

```
example(soc.mca)
map.ca.base() + add.ind(result, mapping = aes(color = sup$Gender)) + add.ellipse(result, sup$Gender)
map.ca.base() + add.ind(result, mapping = aes(color = sup$Age == "65+")) + add.ellipse(result, sup$Age == "65+", dr
```

`add.quadrant.labels` *Annotate labels to the quadrants of an MCA or any ggplot2 based quadrant plot.*

Description

This function is a convience function that uses annotate to easily create labels for the quadrants.

Usage

```
add.quadrant.labels(
  quadrant.labels = c("A", "B", "C", "D"),
  distance = "npc",
  geom = "label",
  color = "black",
  ...
)
```

Arguments

`quadrant.labels`

`distance`	if equal to "npc" labels are positioned dynamically at the edges of the plot. see annotate. If a numeric vector it is interpreted as the distance to 0 on both X and Y.
`geom`	controls the annotation geom; usually you would use "text" or "label".
`color`	either a single value or 4 values that control the color of the labels
`...`	further arguments are passed onto annotate

Value

a ggplot2 layer that can be added to an existing ggplot object.

Examples

```
example(soc.mca)
map.ind(result, point.size = 1) + add.quadrant.labels()
labels <- c("Dominant:\nCultural fraction", "Dominant:\nEconomic fraction", "Dominated:\nEconomic fraction", "Dom
map.ca.base() + add.quadrant.labels(labels, geom = "text")
map.ca.base() + add.ind(result, color = "grey80") + add.quadrant.labels(labels, geom = "text", distance = 1)
map.ca.base() + add.categories(result, color = "grey50", check_overlap = TRUE) + add.quadrant.labels(labels, geom
```

`add.to.label` *Add values to label*

Description

Adds values to the end of the label of each modality.

Usage

```
add.to.label(object, value = "freq", prefix = "default", suffix = ")", dim = 1)
```

Arguments

`object`	is a soc.ca object
`value`	the type of values added to the labels. "freq" adds frequencies, "mass" adds mass values to the active modalities, "ctr" adds contribution values to the active modalities, "cor" adds correlation values. value also accepts any vector with the length of the number of active modalities. "linebreak" adds a linebreak `\n` after the first ":" in the label.
`prefix`	if "default" an appropriate prefix is used
`suffix`	the suffix
`dim`	the dimension from which values are retrieved

Value

a soc.ca object with altered labels in names.mod and names.sup

Examples

```
example(soc.ca)
result.label  <- add.to.label(result)
result.label$names.mod
result.label  <- add.to.label(result, value = "ctr", dim = 2)
result.label$names.mod
result.label  <- add.to.label(result, value = result$variable, prefix = " - ", suffix = "")
result.label$names.mod
result.label  <- add.to.label(result, value = "linebreak")
result.label$names.mod
map.ctr(result.label)
```

assign.label *Assign new labels*

Description

Assigns new labels to a soc.ca object. The input labels are defined in a .csv file created by the export.label function.

Usage

```
assign.label(object, file = FALSE, encoding = "UTF-8", sep = ",")
```

Arguments

object	is a soc.ca object
file	is the path of the .csv file with the new labels. The file is preferably created by the export.label function
encoding	is the encoding of the imported file
sep	is the seperator used to create the imported .csv file

Details

To use this function first export the labels from your soc.mca analysis with the export.label function. Then open and edit the created file with your favorite spreadsheet editor, like LibreOffice Calc. Change labels in the "new.label" column to the desired values and save. Use the assign.label function but remember to assign the results into a new object or overwrite the existing object.

Value

a soc.ca object with altered labels in `object$names.mod`, `object$names.ind` and `object$names.sup`

See Also

export.label, add.to.label

average.coord *Average coordinates*

Description

Find the average coordinates for each category in a variable on two dimensions.

Usage

```
average.coord(object, x, dim = c(1, 2))
```

Arguments

object	is soc.ca result object
x	is a variable of the same length and order as the active variables used to construct the soc.ca object
dim	is the two dimensions used

Value

a matrix with the mean points and frequencies of the given variable

Examples

```
example(soc.ca)
average.coord(result, sup$Income)
```

---

balance *Contribution balance*

Description

Calculates the balance of the contribution of each dimension. This measure indicates whether too much of a dimensions contribution is placed on either the + or - side of the dimension.

Usage

```
balance(object, act.dim = object$nd)
```

Arguments

object	is a soc.ca class object
act.dim	is the number of active dimensions to be measured

Value

A matrix with the share of contribution on each side of 0 and their balance (+/-)

See Also

soc.mca, contribution

Examples

```
example(soc.ca)
balance(result)
balance(result, act.dim = 3)
```

breakdown.variance *Breakdown of variance by group*

Description

Defining a partition of the cloud of individuals into groups, one can calculate the midpoints of the various groups. The total variance of the cloud of individuals can then be broken down to between–within variances, i.e. variance between the groups partitioning the cloud, and variance within the groups The ratio of the between-variance to the total variance is denoted by n2 (eta-square), and accounts for the percentage of variance 'explained' by the group-variable. (see Le Roux & Rouanet 2010, p. 20ff, 69, 114)

Usage

```
breakdown.variance(object, dim = 1:3, variable)
```

Arguments

object	is a soc.ca class object
dim	the dimensions
variable	a factor in the same length and order as the active variables

Value

a matrix

References

Le Roux, Brigitte, and Henry Rouanet. 2010. Multiple Correspondence Analysis. Thousand Oaks, Calif.: Sage Publications.

Examples

```
example(soc.ca)
breakdown.variance(result, dim = 1:3, variable = sup$Gender)
```

contribution *Summaries of contribution values*

Description

Different forms of contribution summaries for soc.ca objects. Results are presented according to the specified mode

Usage

```
contribution(
  object,
  dim = 1,
  all = FALSE,
  indices = FALSE,
  mode = "sort",
  matrix.output = FALSE
)
```

Arguments

`object`	a soc.ca object
`dim`	the included dimensions
`all`	If TRUE returns all modalities instead of just those that contribute above average
`indices`	If TRUE; returns a vector with the row indices of the modalities or individuals
`mode`	indicates which form of output. Possible values: `"sort"`, `"mod"`, `"ind"`, `"variable"`. If the mode is `"variable"`, `dim` can be a sequence of dimensions: `1:5`
`matrix.output`	if TRUE; returns output as a matrix instead of as printed output.

Value

Each mode prints different results:

`"mod"`	Ranks all modalities according to their contribution
`"sort"`	Ranks all modalities according to their contribution and then sorts them according to their coordinates
`"ind"`	Ranks all individuals according to their contribution
`"variable"`	Sorts all modalities according to their variable and sums the contributions per variable

The values reported:

`Ctr`	Contribution values in percentage. Contribution values for individuals are reported in permille
`Coord`	Principal coordinates
`Cor`	The correlation with the dimension

See Also

map.ctr

Examples

```
example(soc.ca)
contribution(result)
contribution(result, 2)
contribution(result, dim = 3, all = TRUE)
contribution(result, indices = TRUE)
contribution(result, 1:2, mode = "variable")
```

`cowboy_cut` *Cut ordinal variables*

Description

If we are in a hurry and need to cut a lot of likert-scale or similar type of variables into MCA-friendly ordered factors this function comes in handy. cowboy_cut will try its best to create approx 3-5 categories, where the top and the bottom are smaller than the middle. Missing or other unwanted categories are recoded but still influence the categorization. So that when cowboy_cut tries to part the top of a variable with a threshold around 10 Make sure that levels are in the right order before cutting.

Usage

```
cowboy_cut(x, top.share = 0.1, bottom.share = 0.1, missing = "Missing")
```

Arguments

`x`	a factor
`top.share`	approximate share in top category
`bottom.share`	approximate share in bottom category
`missing`	a character vector with all the missing or unwanted categories.

Value

a recoded factor

`create.quadrant` *Create categories according to the quadrant position of each individual*

Description

Creates a vector from two dimensions from a soc.ca object. Labels are the cardinal directions with the first designated dimension running East - West. The center category is a circle defined by `cut.radius`.

Usage

```
create.quadrant(
  object,
  dim = c(1, 2),
  cut.min = -0.125,
  cut.max = 0.125,
  cut.radius = 0.25
)
```

Arguments

object	a soc.ca class object
dim	the dimensions
cut.min	Minimum cut value
cut.max	Maximum cut value
cut.radius	Radius of the center category

Value

Returns a character vector with category memberships

See Also

soc.mca

Examples

```
example(soc.ca)
create.quadrant(result, dim = c(2, 1))
table(create.quadrant(result, dim = c(1, 3), cut.radius = 0.5))
```

---

csa.all *Multiple Class Specific Correspondence Analysis on all values in a factor*

Description

csa.all performs a class specific correspondence analysis for each level in a factor variable. Returns a list with soc.csa objects and a list of measures defined by csa.measures

Usage

```
csa.all(object, variable, dim = 1:5, ...)
```

Arguments

object	is a soc.ca class object created with soc.mca
variable	a factor with the same length and order as the active variables that created the soc.ca object
dim	is the dimension analyzed
...	further arguments are directed to csa.measures

Value

results	a list of soc.csa result objects
cor	a list of correlation matrixes
cosines	a list of matrixes with cosine values
angles	a list of matrixes with cosine angles between dimensions

See Also

soc.csa, cor, csa.measures

Examples

```
example(soc.ca)
csa.all(result, taste$Age)
csa.all(result, taste$Age)$measures
```

---

csa.measures *CSA measures*

Description

Several measures for the evaluation of the relations between the dimensions of the CSA and the dimensions the of original MCA

Usage

```
csa.measures(
  csa.object,
  correlations = FALSE,
  cosines = TRUE,
  cosine.angles = TRUE,
  dim.mca = 1:5,
  dim.csa = 1:5,
  format = TRUE,
  ...
)
```

Arguments

csa.object	is a "soc.csa" class object created by the soc.csa function
correlations	if TRUE correlations calculated by the cor function is returned
cosines	if TRUE cosine similarities are returned
cosine.angles	if TRUE angles are calculated in the basis of the cosine values
dim.mca	the dimensions included from the original mca
dim.csa	the dimensions included from the csa
format	if TRUE results are formatted, rounded and printed for screen reading, if FALSE the raw numbers are returned
...	furhter arguments are send to the cor function

Value

A list of measures in either formatted or raw form.

Examples

```
example(soc.csa)
csa.measures(res.csa)
csa.measures(res.csa, correlations = FALSE, cosine.angles = FALSE, dim.mca = 1:5, format = FALSE)
```

---

directors *Directors dataset*

Description

Prosopographical data on the top 100 CEO's from the 82 largest Danish corporations.

Details

The directors dataset is prosopographical data collected from a wide array of sources on biographic and corporate information. Sources include the Danish variant of Who's Who (Blaa Bog), a private business information database (Greens Erhvervsinformation), journalistic portrait articles, article search engines, bibliographic databases and financial reports. CEOs from 82 corporations were selected according to their position as CEO in December 2007. 18 executives are included on other criteria, taking into account the magnitude of the corporations and issues regarding ownership and control, resulting in a final population of 100 CEOs. The 82 corporations have formal ownership and management located in Denmark and were selected through either financial capital, measured as having a turnover of over five billion DKK (650 million Eur.), or organizational capital, defined as having at least 5000 employees; 34 corporations were included on both criteria, 45 on financial capital and three on organizational capital alone. To avoid including investors, rather than executives, a minimum of 500 employees was also required, excluding 12 firms. Companies acting only as subsidiaries were also excluded. Data is for public use and no author permission is needed, but we would love to hear from you if you find the data useful. The following example is based on the analysis from the article: "A Very Economic Elite: The Case of the Danish Top CEOs".

Author(s)

Christoph Ellersgaard

Anton Grau Larsen

References

Ellersgaard, Christoph, Anton Grau Larsen, og Martin D. Munk. 2012. "A Very Economic Elite: The Case of the Danish Top CEOs". Sociology.

Ellersgaard, Christoph Houman, og Anton Grau Larsen. 2010. "Firmaets Maend". Master Thesis, Copenhagen: University of Copenhagen.

Ellersgaard, Christoph Houman, og Anton Grau Larsen. 2011. "Kulturel kapital blandt topdirektoerer i Danmark - En domineret kapitalform?" Dansk Sociologi 22(3):9-29.

Larsen, Anton Grau, og Christoph Houman Ellersgaard. 2012. "Status og integration paa magtens felt for danske topdirektoerer". Praktiske Grunde. Nordisk tidsskrift for kultur- og samfundsvidenskab 2012(2-3).

Examples

```
## Not run:
data(directors)
attach(directors)


active      <- data.frame(careerprofile_maclean_cat, careerfoundation_maclean_cat,
                          years_between_edu_dir_cat, time_in_corp_before_ceo_cat,
                          age_as_ceo_cat, career_changes_cat2, mba, abroad, hd, phd,
                          education, author, placeofbirth, familyclass_bourdieu,
                          partnersfamily_in_whoswho, family_in_whoswho)

sup        <- data.frame(size_prestige, ownership_cat_2, sector, location)

id          <- navn

options(passive = c("MISSING", "Missing", "Irrelevant", "residence_value_cat2: Udlandet"))

result      <- soc.mca(active, sup, id)

result

# Contribution
contribution(result, 1)
contribution(result, 2)
contribution(result, 3)
contribution(result, 1, all = TRUE)
contribution(result, 1, indices = TRUE)
contribution(result, 1, mode = "mod")
contribution(result, mode = "variable")

# Individuals
contribution(result, 1, mode = "ind")
```

```
contribution(result, 2, mode = "ind")


# Table of variance
variance(result)

# Invert
result      <- invert(result, c(1, 2, 3))

# Export and assign label
# export.label(result)

# result      <- assign.label(result,
#  file = "https://raw.github.com/Rsoc/soc.ca/master/extra/director_labels.csv")



# Add.n
result      <- add.to.label(result)
contribution(result, 2)


# The result object or "soc.ca" object
str(result)
dim1 <- result$coord.ind[, 1]
qplot(dim1)

# Quadrant
quad      <- create.quadrant(result)
table(quad)
quad      <- create.quadrant(result, cut.min = 0, cut.max = 0)
table(quad)


# Map of individuals
map.ind(result)
map.ind(result, dim = c(2, 1), label = TRUE)
map.ind(result, dim = c(2, 1), point.size = 3, point.shape = 2)
map.ind(result, dim = c(2, 1), map.title = "The top 100 Danish CEO's",
point.color = quad)
# Map of the individuals colored by contribution
map.ind(result, point.color = result$ctr.ind[, 1],
 point.shape = 18) + scale_color_continuous(low = "white", high = "red")


# Map of contributing modalities
map.ctr(result, dim = c(2, 1))
map.ctr(result, dim = c(2, 1), ctr.dim = 2)
map.ctr(result, point.size = 3)

map.active(result, dim = c(2, 1))
map.sup(result, dim = c(2, 1))
```

```
# Plot.list

# Selecting specific active modalities
select      <- c("Career start: Corporation (n:57)", "No Phd (n:92)")
boo.select  <- match(select, result$names.mod)
map.select(result, list.mod = boo.select)

highcor     <- which(result$cor.mod[, 1] >= 0.2)
map.select(result, list.mod = highcor)

# Selecting specific supplementary modalities

highdim3    <- which(sqrt(result$coord.sup[, 3]^2) >= 0.5)
map.select(result, list.sup = highdim3)

# Selecting specific individuals based on a certain criteria

forfatter   <- author == "Forfatter"
map.select(result, list.ind = forfatter)

# Combining it all
map.select(result, list.mod = highcor, list.sup = highdim3, list.ind = forfatter)

# Add points to an existing plot
ctrplot     <- map.ctr(result, ctr.dim = 1, point.color = "red")
map.add(result, ctrplot, data.type = "ctr", ctr.dim = 2, point.color = "blue")

# Using the list option in add.points
forfatter    <- author == "Forfatter"
map.add(result, ctrplot, data.type = "select", list.ind = forfatter, colour = "purple")

# Using the list option in add.points to add labels to only a part of the cloud of individuals
forfatter     <- author == "Forfatter"
notforfatter  <- author != "Forfatter"
map.forfatter <- map.select(result, list.ind = notforfatter, label = FALSE)
map.forfatter
map.forfatter <- map.add(result, map.forfatter, data.type = "select", list.ind = forfatter)
map.forfatter

# Plotting all the modalities of one individual
result2       <- soc.ca(active, sup, id)
individual    <- which(id == "Lars Larsen")
ind.mat       <- indicator(active)
modalities    <- names(which(ind.mat[individual, ] == 1))
mod.ind       <- match(modalities, result2$names.mod)

lars          <- map.select(result2, list.mod = mod.ind)
map.add(result2, lars, data.type = "select", list.ind = individual, colour = "red")

# Adding concentration ellipses to an existing plot
el.forfatter  <- map.ellipse(result, map.forfatter, author)
el.forfatter
```

```
## End(Not run)
```

`ellipses` *Calculate concentraion ellipses*

Description

Calculate concentraion ellipses

Usage

```
ellipses(object, var, dim = c(1, 2), kappa = 2, npoints = 1000)
```

Arguments

`object`

`var`

`dim`

`kappa`

`npoints`

Examples

```
example(soc.mca)
ellipses(result, active[,1])
```

`export` *Export results from soc.ca*

Description

Export objects from the soc.ca package to csv files.

Usage

```
export(object, file = "export.csv", dim = 1:5)
```

Arguments

`object`	is a soc.ca class object
`file`	is the path and name of the .csv values are to be exported to
`dim`	is the dimensions to be exported

Value

A .csv file with various values in UTF-8 encoding

See Also

soc.mca, contribution

`export.label` *Exports the labels of a soc.ca object into a csv file.*

Description

This function allows easy translation and renaming of modalities by exporting the labels into a .csv file that is easier to work with.

Usage

```
export.label(object, file = FALSE, encoding = "UTF-8", overwrite = FALSE)
```

Arguments

`object`	is a soc.ca object
`file`	is the name and path of the exported file
`encoding`	is the character encoding of the exported file
`overwrite`	decides whether to overwrite already existing files

Details

Two columns are created within the .csv: 'New label' and 'Old label'. In the 'New label' column you write the new labels. Remember to leave 'Old label' unchanged as this column is used for matching.

If you want to add frequencies to the labels with the add.to.label function you should do this after exporting and assigning labels with the assign.label function. Otherwise the matching of the labels is likely to fail.

Value

A .csv with two columns and preferably UTF-8 encoding.

extract_cases *Extract individuals*

Description

Extract individuals

Usage

```
extract_cases(result, dim = 1:3)

extract_ind(result, dim = 1:3)
```

Arguments

result	a soc.ca object or a PCA
dim	the dimensions

Value

a data.frame with coordinates and frequences

Examples

```
example(soc.mca)
extract_cases(result)
```

extract_cats *Extract coordinates for the categories from an soc.mca*

Description

Extract coordinates for the categories from an soc.mca

Usage

```
extract_cats(result, dim = 1:3)

extract_mod(result, dim = 1:3)
```

Arguments

result	a soc.mca object or a PCA
dim	the dimensions

Value

a data.frame with coordinates and frequences

Examples

```
example(soc.mca)
extract_cats(result)
```

---

extract_sup *Extract supplementary categories from an soc.mca*

Description

Extract supplementary categories from an soc.mca

Usage

```
extract_sup(result, dim = 1:3)
```

Arguments

result	a soc.mca object
dim	the dimensions

Value

a data.frame with coordinates and frequences

Examples

```
example(soc.mca)
extract_sup(result)
```

---

get.category.relations *Get and calculate the relationships and oppositions between each pair of categories*

Description

Use this function to calculate PEM pem values, chisq, distance and coordinates for each pair of categories in either an indicator matrix or the categories from an soc.mca result object. These relationship are usefull for both diagnostics, analysis, interpretation and plotting. For plotting combine with add.category.relations to build your plot.

Usage

```
get.category.relations(
  r,
  ind = r$indicator.matrix.active,
  dim = c(1, 2),
  rel = t(combn(colnames(ind), 2))
)
```

Arguments

r	an soc.mca result object
ind	an indicator matrix, see indicator
dim	a numeric vector with the dimensions for the coordinates. This is only sent to extract_mod.
rel	a matrix with pairs of categories
variable	a character vector with the variable where each category in ind came from. If ind was created directly with indicator you can use names(colnames(ind)).
coords	a data.frame with coordinates - similar to those produced by extract_mod

Value

a tibble

Examples

```
example(soc.mca)
get.category.relations(result)
```

---

headings *Calculate contributions per heading*

Description

Calculate contributions per heading

Usage

```
headings(object, dim = 1:3)
```

Arguments

object	a soc.ca object with headings
dim	a numeric vector with the dimensions

Value

a matrix

Examples

```
data(taste)
active.headings <- list()
active.headings$Consumption <- na.omit(taste)[, c("TV", "Film", "Art", "Eat")]
active.headings$Background  <- na.omit(taste)[, c("Gender", "Age", "Income")]
result.headings <- soc.mca(active.headings)
headings(result.headings)
```

---

ind.explorer *Explore the cloud of individuals*

Description

Explore the cloud of individuals

Usage

```
ind.explorer(object, active, sup = NULL)
```

Arguments

object	a a soc.ca class object as created by soc.mca and soc.csa
active	Defines the active modalities in a data.frame with rows of individuals and columns of factors, without NA's'
sup	Defines the supplementary modalities in a data.frame with rows of individuals and columns of factors, without NA's

Value

an html application

Examples

```
## Not run:
example(soc.mca)
ind.explorer(result, active, sup)

## End(Not run)
```

indicator *Indicator matrix*

Description

Creates an indicator matrix from a data.frame with the categories of the questions as columns and individuals as rows.

Usage

```
indicator(x, id = NULL, ps = ": ")
```

Arguments

x	a data.frame of factors
id	a vector defining the labels for the individuals. If id = NULL row number is used.
ps	the seperator used in the creation of the names of the columns.

Value

Returns a indicator matrix

See Also

soc.mca

Examples

```
a  <- rep(c("A","B"), 5)
b  <- rep(c("C", "D"), 5)
indicator(data.frame(a,b))
```

indicator.to.long *Pivot the indicator matrix from an MCA to long format*

Description

Sometimes we want the indicator matrix from an mca in long format and this function delivers just that. The results contain both active and passive categories.

Usage

```
indicator.to.long(r)
```

Arguments

r	an result object of class soc.mca

Value

a tibble in long format

Examples

```
example(soc.mca)
result |> indicator.to.long()
```

---

invert *Invert the direction of coordinates*

Description

Invert one or more axes of a correspondence analysis. The principal coordinates of the analysis are multiplied by -1.

Usage

```
invert(x, dim = 1)
```

Arguments

x	is a soc.ca object
dim	is the dimensions to be inverted

Details

This is a convieniency function as you would have to modify coord.mod, coord.ind and coord.sup in the soc.ca object.

Value

a soc.ca object with inverted coordinates on the specified dimensions

See Also

soc.mca, add.to.label

Examples

```
example(soc.ca)
inverted.result  <- invert(result, 1:2)
result$coord.ind[1, 1:2]
inverted.result$coord.ind[1, 1:2]
```

`map.active` *Map the active modalities*

Description

Creates a map of the active modalities on two selected dimensions.

Usage

```
map.active(
  object,
  dim = c(1, 2),
  point.shape = "variable",
  point.alpha = 0.8,
  point.fill = "whitesmoke",
  point.color = "black",
  point.size = "freq",
  label = TRUE,
  label.repel = FALSE,
  label.alpha = 0.8,
  label.color = "black",
  label.size = 4,
  label.fill = NULL,
  map.title = "active",
  labelx = "default",
  labely = "default",
  legend = NULL
)
```

Arguments

`object`	a soc.ca class object as created by soc.mca and soc.csa
`dim`	the dimensions in the order they are to be plotted. The first number defines the horizontal axis and the second number defines the vertical axis.
`point.shape`	a numerical value defining the shape of the points. If set to its default, the default scale is used. It may be mapped to a variable with a suitable length and order.
`point.alpha`	defines the alpha of the points. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
`point.fill`	defines the fill color of the points. It may be mapped to a variable with a suitable length and order.
`point.color`	defines the color of the points. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
`point.size`	a numerical value defining the size of the points. If set to its default, the size is determined by the frequency of each modality. It may be defined by a variable with a suitable length.

label	if TRUE each point is assigned its label, defined in the soc.ca object. See assign.label and add.to.label for ways to alter the labels.
label.repel	if TRUE overlapping labels are rearranged, see geom_text_repel or geom_label_repel.
label.alpha	defines the alpha of the labels. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
label.color	defines the color of the labels. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
label.size	defines the size of the labels. It may be mapped to a variable with a suitable length and order.
label.fill	defines the color of the box behind the labels. It may be mapped to a variable with a suitable length and order. This only works if label.repel is TRUE. See geom_label_repel.
map.title	the title of the map. If set to its default the standard title is used.
labelx	the label of the horizontal axis. If set to NULL a standard label is used.
labely	the label of the vertical axis. If set to NULL a standard label is used.
legend	if set to TRUE a legend is provided. Change the legend with the guides, theme and linkguide_legend functions from the ggplot2 package.

Examples

```
example(soc.ca)
map.active(result)
map.active(result, dim = c(2, 1))
map.active(result, point.size = result$ctr.mod[, 1],
 map.title = "All active modalities with size according to contribution")
```

---

map.add *Add points to an existing map created by one of the soc.ca mapping functions.*

Description

Add points to an existing map created by one of the soc.ca mapping functions.

Usage

```
map.add(
  object,
  ca.map,
  plot.type = NULL,
  ctr.dim = 1,
  list.mod = NULL,
  list.sup = NULL,
  list.ind = NULL,
  point.shape = "variable",
```

```
  point.alpha = 0.8,
  point.fill = "whitesmoke",
  point.color = "black",
  point.size = "freq",
  label = TRUE,
  label.repel = TRUE,
  label.alpha = 0.8,
  label.color = "black",
  label.size = 4,
  label.fill = NULL,
  labelx = "default",
  labely = "default",
  legend = NULL
)
```

Arguments

object	a soc.ca class object as created by soc.mca and soc.csa
ca.map	a map created using one of the soc.ca map functions
plot.type	defines which type of points to add to the map. Accepted values are: "mod", "sup", "ind", "ctr". These values correspond to the different forms of
ctr.dim	the dimensions of the contribution values
list.mod	a numerical vector indicating which active modalities to plot. It may also be a logical vector of the same length and order as the modalities in object$names.mod.
list.sup	a numerical vector indicating which supplementary modalities to plot. It may also be a logical vector of the same length and order as the modalities in object$names.sup.
list.ind	a numerical vector indicating which individuals to plot. It may also be a logical vector of the same length and order as the modalities in object$names.ind.
point.shape	a numerical value defining the shape of the points. If set to its default, the default scale is used. It may be mapped to a variable with a suitable length and order.
point.alpha	defines the alpha of the points. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
point.fill	defines the fill color of the points. It may be mapped to a variable with a suitable length and order.
point.color	defines the color of the points. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
point.size	a numerical value defining the size of the points. If set to its default, the size is determined by the frequency of each modality. It may be defined by a variable with a suitable length.
label	if TRUE each point is assigned its label, defined in the soc.ca object. See assign.label and add.to.label for ways to alter the labels.
label.repel	if TRUE overlapping labels are rearranged, see geom_text_repel or geom_label_repel.
label.alpha	defines the alpha of the labels. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.

label.color	defines the color of the labels. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
label.size	defines the size of the labels. It may be mapped to a variable with a suitable length and order.
label.fill	defines the color of the box behind the labels. It may be mapped to a variable with a suitable length and order. This only works if label.repel is TRUE. See geom_label_repel.
labelx	the label of the horizontal axis. If set to NULL a standard label is used.
labely	the label of the vertical axis. If set to NULL a standard label is used.
legend	if set to TRUE a legend is provided. Change the legend with the guides, theme and linkguide_legend functions from the ggplot2 package.
dim	the dimensions in the order they are to be plotted. The first number defines the horizontal axis and the second number defines the vertical axis.

Examples

```
example(soc.ca)
original.map    <- map.sup(result)
map.add(result, original.map, plot.type = "ctr", ctr.dim = 2)
map.add(result, map.ind(result), plot.type = "select",list.ind = 1:50,
 point.color = "red", label = FALSE, point.size = result$ctr.ind[1:50, 1]*2000)
```

---

map.array *Array of maps*

Description

This function takes a list of map objects and arranges them into an array.

Usage

```
map.array(x, ncol = 1, title = "", fixed.coord = TRUE, padding = 0.15)
```

Arguments

x	a list of objects created by one of the mapping functions in the soc.ca package or any other ggplot2 plot
ncol	the number of columns the plots are arranged into
title	the main title of the array
fixed.coord	if TRUE the limits of all plots are set to the same as the largest plot
padding	the distance between the most extreme position and the axis limit

Examples

```
## Not run:
example(soc.ca)
map.array(list(map.ind(result), map.mod(result)), ncol = 2)

## End(Not run)
```

`map.ca.base` *Create the base of a soc.ca map*

Description

Create the base of a soc.ca map

Usage

```
map.ca.base(
  up = NULL,
  down = NULL,
  right = NULL,
  left = NULL,
  base_size = 15,
  ...
)
```

Arguments

`up`	the name of + pole on the vertical axis - "North"
`down`	the name of the - pole on the vertical axis - "South"
`right`	the name of the + pole on horizontal axis - "East"
`left`	the name of the - pole on the horizontal axis - "West"
`base_size`	controls the text size of themed labels
`...`	further arguments are passed onto ggplot()

Value

a ggplot2 object

`map.csa.all` *Array of several CSA maps*

Description

Creates an array of Class Specific Mulitple Correspondence analysises

Usage

```
map.csa.all(
  object,
  variable,
  dim = c(1, 2),
  ncol = 2,
  FUN = map.ind,
  fixed.coord = TRUE,
  main.title = "",
  titles = levels(variable),
  ...
)
```

Arguments

`object`	a soc.ca result object
`variable`	a factor with the same order and length as those used for the active modalities in object
`dim`	indicates what dimensions to map and in which order to plot them
`ncol`	the number of columns the maps are arranged into
`FUN`	the mapping function used for the plots; map.active, map.ctr, map.ind, map.select or map.sup
`fixed.coord`	if TRUE the limits of all plots are set to the same as the largest plot
`main.title`	the main title for all the maps
`titles`	a vector of the same length as the number of levels in `variable`. These are the titles given to each subplot
`...`	sends any further arguments to the mapping functions

Examples

```
## Not run:
example(soc.csa)
map.csa.all(result, active[, 1])
map.csa.all(result, active[, 1], FUN = map.ctr, ctr.dim = 1)

## End(Not run)
```

map.csa.mca *Map the coordinates of the individuals in a CSA and its MCA*

Description

Map the coordinates of the individuals in a CSA and its MCA

Usage

```
map.csa.mca(
  csa.object,
  mca.dim = 1,
  csa.dim = 1,
  smooth = TRUE,
  method = "auto"
)
```

Arguments

csa.object	a result object created by the soc.csa function
mca.dim	the dimension from the original MCA
csa.dim	the dimension from the CSA
smooth	if TRUE a line is added to the plot
method	the method used by ggplot to set the line see geom_smooth

See Also

soc.csa, map.csa.all, linkmap.csa.mca.array

Examples

```
example(soc.csa)
csa.res  <- soc.csa(result, class.age)
map.csa.mca(csa.res, mca.dim = 2, csa.dim = 1)
```

map.csa.mca.array *CSA-MCA array*

Description

Create an array of map.csa.mca maps

Usage

```
map.csa.mca.array(csa.object, ndim = 3, fixed.coord = TRUE, ...)
```

Arguments

csa.object	a result object created by the soc.csa function
ndim	the number of dimensions to include in the array, starting from 1
fixed.coord	if TRUE the limits of all plots are set to the same as the largest plot
...	for further arguments see map.csa.mca

Examples

```
example(soc.csa)
csa.res <- soc.csa(result, class.age)
map.csa.mca.array(csa.res, ndim = 3)
```

---

map.ctr *Map the most contributing modalities*

Description

Creates a map of the modalities contributing above average to one or more dimensions on two selected dimension.

Usage

```
map.ctr(
  object,
  dim = c(1, 2),
  ctr.dim = 1,
  point.shape = "variable",
  point.alpha = 0.8,
  point.fill = "whitesmoke",
  point.color = "black",
  point.size = "freq",
  label = TRUE,
  label.repel = TRUE,
  label.alpha = 0.8,
  label.color = "black",
  label.size = 4,
  label.fill = NULL,
  map.title = "ctr",
  labelx = "default",
  labely = "default",
  legend = NULL
)
```

Arguments

object	a soc.ca class object as created by soc.mca and soc.csa
dim	the dimensions in the order they are to be plotted. The first number defines the horizontal axis and the second number defines the vertical axis.
ctr.dim	the dimensions of the contribution values
point.shape	a numerical value defining the shape of the points. If set to its default, the default scale is used. It may be mapped to a variable with a suitable length and order.
point.alpha	defines the alpha of the points. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
point.fill	defines the fill color of the points. It may be mapped to a variable with a suitable length and order.
point.color	defines the color of the points. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
point.size	a numerical value defining the size of the points. If set to its default, the size is determined by the frequency of each modality. It may be defined by a variable with a suitable length.
label	if TRUE each point is assigned its label, defined in the soc.ca object. See assign.label and add.to.label for ways to alter the labels.
label.repel	if TRUE overlapping labels are rearranged, see geom_text_repel or geom_label_repel.
label.alpha	defines the alpha of the labels. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
label.color	defines the color of the labels. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
label.size	defines the size of the labels. It may be mapped to a variable with a suitable length and order.
label.fill	defines the color of the box behind the labels. It may be mapped to a variable with a suitable length and order. This only works if label.repel is TRUE. See geom_label_repel.
map.title	the title of the map. If set to its default the standard title is used.
labelx	the label of the horizontal axis. If set to NULL a standard label is used.
labely	the label of the vertical axis. If set to NULL a standard label is used.
legend	if set to TRUE a legend is provided. Change the legend with the guides, theme and linkguide_legend functions from the ggplot2 package.

Examples

```
example(soc.ca)
map.ctr(result)
map.ctr(result, ctr.dim = c(1, 2))
```

`map.density` *Density plot for the cloud of individuals*

Description

Draws a 2d density plot on top of an existing soc.ca map. The density is calculated by the kde2d function from MASS and plotted by geom_density2d from `ggplot2` `map.density` uses the coordinates of the individuals as a basis for the density calculation. Borders are arbitrary.

Usage

```
map.density(
  object,
  map = map.ind(object),
  group = NULL,
  color = "red",
  alpha = 0.8,
  size = 0.5,
  linetype = "solid"
)
```

Arguments

`object`	a soc.ca class object
`map`	a soc.ca map object created by one of the soc.ca mapping functions
`group`	a factor determining group membership. Density is mapped for each group individually.
`color`	a single value or vector determining the color. See the scale functions of `ggplot2` for ways to alter the scales.
`alpha`	a single value or vector determining the alpha.
`size`	a single value or vector determining the size of the lines.
`linetype`	a single value or vector determining the linetype

Examples

```
example(soc.ca)
map.density(result, map.ind(result, dim = 2:3, point.alpha = 0.2))
map.density(result, map.ind(result, legend = TRUE, point.alpha = 0.2),
 group = duplicated(active), color = duplicated(active),
 linetype = duplicated(active))
map.density(result, map.ctr(result))
```

`map.ellipse` *Concentration ellipses*

Description

Add ellipses for each level in a factor to a plot made from a soc.ca object.

Usage

```
map.ellipse(
  object,
  ca.plot = map.ind(object),
  variable,
  ellipse.label = TRUE,
  ellipse.color = "default",
  label.size = 4,
  draw.levels = 1:nlevels(variable),
  ellipse.line = "solid"
)
```

Arguments

`object`	is a soc.ca class object.
`ca.plot`	is a plot made from a soc.ca object.
`variable`	is a factor of the same length and in the same order as the active varibles used for the soc.ca object.
`ellipse.label`	if TRUE the labels are included in the map.
`ellipse.color`	defines the color of the ellipses. If "default" the globally defined default colors are used. Ellipse.color can be either length of 1 or equal to the number of drawn levels.
`label.size`	defines the size of the labels.
`draw.levels`	indicates the levels in the variable for which a ellipse is drawn.
`ellipse.line`	defines the type of line used for the ellipses.

Value

a plot with a concentration ellipse containing 80% of the individuals for each modality.

See Also

map.ind, map.ctr

Examples

```
example(soc.ca)
map <- map.ind(result)
map.ellipse(result, map, active[,2])
```

`map.ellipse.array` *Ellipse array*

Description

Create seperate maps with ellipses for each level in a factor arranged in an array.

Usage

```
map.ellipse.array(
  object,
  variable,
  dim = c(1, 2),
  draw.ellipses = TRUE,
  ncol = 2,
  titles = levels(variable),
  main.title = "",
  ...
)
```

Arguments

`object`	a soc.ca class object
`variable`	a factor of the same length as the data.frame used to create object
`dim`	the dimensions in the order they are to be plotted. The first number defines the horizontal axis and the second number defines the vertical axis.
`draw.ellipses`	if TRUE ellipses are drawn
`ncol`	the number of columns the plots are arranged into
`titles`	a vector of the same length as the number of levels in variable. These are the titles given to each subplot
`main.title`	the main title for all the plots
`...`	sends any further arguments to map.select and map.ellipse.

Examples

```
## Not run:
example(soc.ca)
map.ellipse.array(result, active[, 1])

## End(Not run)
```

`map.ind` *Map the individuals of a soc.ca analysis*

Description

Creates a map of the individuals on two selected dimension.

Usage

```
map.ind(
  object,
  dim = c(1, 2),
  point.shape = 21,
  point.alpha = 0.8,
  point.fill = "whitesmoke",
  point.color = "black",
  point.size = 3,
  label = FALSE,
  label.repel = FALSE,
  label.alpha = 0.8,
  label.color = "black",
  label.size = 4,
  label.fill = NULL,
  map.title = "ind",
  labelx = "default",
  labely = "default",
  legend = NULL
)
```

Arguments

`object`	a soc.ca class object as created by soc.mca and soc.csa
`dim`	the dimensions in the order they are to be plotted. The first number defines the horizontal axis and the second number defines the vertical axis.
`point.shape`	a numerical value defining the shape of the points. It may be mapped to a variable with a suitable length and order.
`point.alpha`	defines the alpha of the points. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
`point.fill`	defines the fill color of the points. It may be mapped to a variable with a suitable length and order.
`point.color`	defines the color of the points. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
`point.size`	a numerical value defining the size of the points. It may be defined by a variable with a suitable length.

label	if TRUE each point is assigned its label, defined in the soc.ca object. See assign.label and add.to.label for ways to alter the labels.
label.repel	if TRUE overlapping labels are rearranged, see geom_text_repel or geom_label_repel.
label.alpha	defines the alpha of the labels. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
label.color	defines the color of the labels. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
label.size	defines the size of the labels. It may be mapped to a variable with a suitable length and order.
label.fill	defines the color of the box behind the labels. It may be mapped to a variable with a suitable length and order. This only works if label.repel is TRUE. See geom_label_repel.
map.title	the title of the map. If set to its default the standard title is used.
labelx	the label of the horizontal axis. If set to NULL a standard label is used.
labely	the label of the vertical axis. If set to NULL a standard label is used.
legend	if set to TRUE a legend is provided. Change the legend with the guides, theme and linkguide_legend functions from the ggplot2 package.

Examples

```
example(soc.ca)
map.ind(result)
map.ind(result, map.title = "Each individual is given its shape according to a value in a factor",
 point.shape = active[, 1], legend = TRUE)
map  <- map.ind(result, map.title = "The contribution of the individuals with new scale",
 point.color = result$ctr.ind[, 1], point.shape = 18)
map + scale_color_continuous(low = "white", high = "red")
quad   <- create.quadrant(result)
map.ind(result, map.title = "Individuals in the space given shape and color by their quadrant",
 point.shape = quad, point.color = quad)
```

---

map.mod *Map all modalities*

Description

Creates a map of all active and supplementary modalities on two selected dimension.

Usage

```
map.mod(
  object,
  dim = c(1, 2),
  point.shape = "variable",
  point.alpha = 0.8,
```

```
  point.fill = "whitesmoke",
  point.color = "black",
  point.size = "freq",
  label = TRUE,
  label.repel = FALSE,
  label.alpha = 0.8,
  label.color = "black",
  label.size = 4,
  label.fill = NULL,
  map.title = "mod",
  labelx = "default",
  labely = "default",
  legend = NULL
)
```

Arguments

object	a soc.ca class object as created by soc.mca and soc.csa
dim	the dimensions in the order they are to be plotted. The first number defines the horizontal axis and the second number defines the vertical axis.
point.shape	a numerical value defining the shape of the points. If set to its default, the default scale is used. It may be mapped to a variable with a suitable length and order.
point.alpha	defines the alpha of the points. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
point.fill	defines the fill color of the points. It may be mapped to a variable with a suitable length and order.
point.color	defines the color of the points. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
point.size	a numerical value defining the size of the points. If set to its default, the size is determined by the frequency of each modality. It may be defined by a variable with a suitable length.
label	if TRUE each point is assigned its label, defined in the soc.ca object. See assign.label and add.to.label for ways to alter the labels.
label.repel	if TRUE overlapping labels are rearranged, see geom_text_repel or geom_label_repel.
label.alpha	defines the alpha of the labels. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
label.color	defines the color of the labels. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
label.size	defines the size of the labels. It may be mapped to a variable with a suitable length and order.
label.fill	defines the color of the box behind the labels. It may be mapped to a variable with a suitable length and order. This only works if label.repel is TRUE. See geom_label_repel.
map.title	the title of the map. If set to its default the standard title is used.
labelx	the label of the horizontal axis. If set to NULL a standard label is used.

labely	the label of the vertical axis. If set to NULL a standard label is used.
legend	if set to TRUE a legend is provided. Change the legend with the guides, theme and linkguide_legend functions from the ggplot2 package.

Examples

```
example(soc.ca)
map.mod(result)
map.mod(result, dim = c(3, 2), point.size = 2)
```

`map.path` *Map path along an ordered variable*

Description

Plot a path along an ordered variable. If the variable is numerical it is cut into groups by the min_cut function.

Usage

```
map.path(
  object,
  x,
  map = map.ind(object, dim),
  dim = c(1, 2),
  label = TRUE,
  min.size = length(x)/10,
  ...
)
```

Arguments

object	is a soc.ca result object
x	is an ordered vector, either numerical or factor
map	is a plot object created with one of the mapping functions in the soc.ca package
dim	the dimensions in the order they are to be plotted. The first number defines the horizontal axis and the second number defines the vertical axis.
label	if TRUE the label of the points are shown
min.size	is the minimum size given to the groups of a numerical variable, see min_cut.
...	further arguments are passed onto geom_path, geom_point and geom_text from the ggplot2 package

Examples

```
example(soc.ca)
map <- map.ind(result, point.color = as.numeric(sup$Age))
map <- map + scale_color_continuous(high = "red", low = "yellow")
map.path(result, sup$Age, map)
```

`map.select` *Map select modalities and individuals*

Description

Creates a map of selected modalities or individuals

Usage

```
map.select(
  object,
  dim = c(1, 2),
  ctr.dim = 1,
  list.mod = NULL,
  list.sup = NULL,
  list.ind = NULL,
  point.shape = "variable",
  point.alpha = 0.8,
  point.fill = "whitesmoke",
  point.color = "black",
  point.size = "freq",
  label = TRUE,
  label.repel = FALSE,
  label.alpha = 0.8,
  label.color = "black",
  label.size = 4,
  label.fill = NULL,
  map.title = "select",
  labelx = "default",
  labely = "default",
  legend = NULL,
  ...
)
```

Arguments

`object`	a soc.ca class object as created by soc.mca and soc.csa
`dim`	the dimensions in the order they are to be plotted. The first number defines the horizontal axis and the second number defines the vertical axis.
`ctr.dim`	the dimensions of the contribution values
`list.mod`	a numerical vector indicating which active modalities to plot. It may also be a logical vector of the same length and order as the modalities in object$names.mod.
`list.sup`	a numerical vector indicating which supplementary modalities to plot. It may also be a logical vector of the same length and order as the modalities in object$names.sup.

list.ind	a numerical vector indicating which individuals to plot. It may also be a logical vector of the same length and order as the modalities in object$names.ind.
point.shape	a numerical value defining the shape of the points. If set to its default, the default scale is used. It may be mapped to a variable with a suitable length and order.
point.alpha	defines the alpha of the points. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
point.fill	defines the fill color of the points. It may be mapped to a variable with a suitable length and order.
point.color	defines the color of the points. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
point.size	a numerical value defining the size of the points. If set to its default, the size is determined by the frequency of each modality. It may be defined by a variable with a suitable length.
label	if TRUE each point is assigned its label, defined in the soc.ca object. See assign.label and add.to.label for ways to alter the labels.
label.repel	if TRUE overlapping labels are rearranged, see geom_text_repel or geom_label_repel.
label.alpha	defines the alpha of the labels. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
label.color	defines the color of the labels. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
label.size	defines the size of the labels. It may be mapped to a variable with a suitable length and order.
label.fill	defines the color of the box behind the labels. It may be mapped to a variable with a suitable length and order. This only works if label.repel is TRUE. See geom_label_repel.
map.title	the title of the map. If set to its default the standard title is used.
labelx	the label of the horizontal axis. If set to NULL a standard label is used.
labely	the label of the vertical axis. If set to NULL a standard label is used.
legend	if set to TRUE a legend is provided. Change the legend with the guides, theme and guide_legend functions from the ggplot2 package.
...	further arguments are currently ignored.

Examples

```
example(soc.ca)
map.select(result, map.title = "Map of the first ten modalities", list.mod = 1:10)
select   <- active[, 3]
select   <- select == levels(select)[2]
map.select(result, map.title = "Map of all individuals sharing a particular value",
 list.ind = select, point.size = 3)
map.select(result, map.title = "Map of both select individuals and modalities",
 list.ind = select, list.mod = 1:10)
```

`map.sup` *Map the supplementary modalities*

Description

Creates a map of the supplementary modalities on two selected dimension.

Usage

```
map.sup(
  object,
  dim = c(1, 2),
  point.shape = "variable",
  point.alpha = 0.8,
  point.fill = "whitesmoke",
  point.color = "black",
  point.size = "freq",
  label = TRUE,
  label.repel = TRUE,
  label.alpha = 0.8,
  label.color = "black",
  label.size = 4,
  label.fill = NULL,
  map.title = "sup",
  labelx = "default",
  labely = "default",
  legend = NULL
)
```

Arguments

`object`	a soc.ca class object as created by soc.mca and soc.csa
`dim`	the dimensions in the order they are to be plotted. The first number defines the horizontal axis and the second number defines the vertical axis.
`point.shape`	a numerical value defining the shape of the points. If set to its default, the default scale is used. It may be mapped to a variable with a suitable length and order.
`point.alpha`	defines the alpha of the points. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
`point.fill`	defines the fill color of the points. It may be mapped to a variable with a suitable length and order.
`point.color`	defines the color of the points. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
`point.size`	a numerical value defining the size of the points. If set to its default, the size is determined by the frequency of each modality. It may be defined by a variable with a suitable length.

label	if TRUE each point is assigned its label, defined in the soc.ca object. See assign.label and add.to.label for ways to alter the labels.
label.repel	if TRUE overlapping labels are rearranged, see geom_text_repel or geom_label_repel.
label.alpha	defines the alpha of the labels. Values range from 0 to 1. It may be mapped to a variable with a suitable length and order.
label.color	defines the color of the labels. It may be mapped to a variable with a suitable length and order. See colors for some of the valid values.
label.size	defines the size of the labels. It may be mapped to a variable with a suitable length and order.
label.fill	defines the color of the box behind the labels. It may be mapped to a variable with a suitable length and order. This only works if label.repel is TRUE. See geom_label_repel.
map.title	the title of the map. If set to its default the standard title is used.
labelx	the label of the horizontal axis. If set to NULL a standard label is used.
labely	the label of the vertical axis. If set to NULL a standard label is used.
legend	if set to TRUE a legend is provided. Change the legend with the guides, theme and linkguide_legend functions from the ggplot2 package.

Examples

```
example(soc.ca)
map.sup(result)
map.sup(result, dim = c(2, 1))
map.sup(result, point.size = result$coord.sup[, 4],
 map.title = "All supplementary modalities with size according to coordinate on the 4th dimension")
```

---

mca.eigen.check *MCA Eigenvalue check*

Description

Two variables that have perfectly or almost perfectly overlapping sets of categories will skew an mca analysis. This function tries to find the variables that do that so that we may remove them from the analysis or set some of the categories as passive. An MCA is run on all pairs of variables in the active dataset and we take first and strongest eigenvalue for each pair. Values range from 0.5 to 1, where 1 signifies a perfect or near perfect overlap between sets of categories while 0.5 is the opposite - a near orthogonal relationship between the two variables. While a eigenvalue of 1 is a strong candidate for intervention, probably exclusion of one of the variables, it is less clear what the lower bound is. But values around 0.8 are also strong candidates for further inspection.

Usage

```
mca.eigen.check(x, passive = "Missing")
```

Arguments

x	a data.frame of factors or a result object from soc.mca
passive	a character vector with the full or partial names of categories to be set as passive. Each element in passive is passed to a grep function.

Value

a tibble

Examples

```
example(soc.mca)
mca.eigen.check(active)
mca.eigen.check(result)
```

---

mca.triads *Compare MCA's with triads*

Description

Compare MCA's with triads

Usage

```
mca.triads(l.mca, l.triads, dim = c(1, 2), fix.mca = 1)
```

Arguments

l.mca	a list of soc.mca objects
l.triads	a list of triads
dim	the dimensions of the plane
fix.mca	the indice of the mca that is used as a fixpoint for the axis across mca's

Value

a triad object

min_cut *Cut a continuous variable into categories with a specified minimum*

Description

Many continuous variables are very unequally distributed, often with many individuals in the lower categories and fewer in the top. As a result it is often difficult to create groups of equal size, with unique cut-points. By defining the wanted minimum of individuals in each category, but still allowing this minimum to be surpassed, it is easy to create ordinal variables from continuous variables. The last category will not neccessarily have the minimum number of individuals.

Usage

```
min_cut(x, min.size = length(x)/10)
```

Arguments

x	is a continuous numerical variable
min.size	is the minimum number of individuals in each category

Value

a numerical vector with the number of each category

Examples

```
a <- 1:1000
table(min_cut(a))
b <- c(rep(0, 50), 1:500)
table(min_cut(b, min.size = 20))
```

moschidis *Moschidis example*

Description

The example dataset used by Odysseas E. Moschidis (2009):

Author(s)

Odysseas E. Moschidis

References

Moschidis, Odysseas E. “A Different Approach to Multiple Correspondence Analysis (MCA) than That of Specific MCA.” Mathématiques et Sciences Humaines / Mathematics and Social Sciences 47, no. 186 (October 15, 2009): 77–88. https://doi.org/10.4000/msh.11091.

Examples

```
# The moschidis example
#data(moschidis)
#active    <- moschidis[, c("E1","E2", "E3")]
#id        <- moschidis[, c("ID")]
# result    <- soc.mca(active, identifier = id, Moschidis = FALSE)

# Compare output to Moschidis (2009, p. 85)
#result$inertia_full
# In the analysis of the 'real' data the modality
#'E1: 1' with a low mass (fr/Q) has a very high contribution to the fourth axis
#result$ctr.mod[, 4]

# Using the transformed model suggested by Moschidis (2009) that takes into
# account the number of modalities per question in order to balance the
# contribution of the modalities
#result_trans    <- soc.mca(active, identifier = id, Moschidis = TRUE)
#result_trans$inertia_full
#result_trans$ctr.mod[, 4]
```

---

pe13 *The Field of the Danish Power Elite*

Description

This dataset was used to construct a field of the Danish Power Elite from 2013

Author(s)

Jacob Lunding, Anton Grau Larsen and Christoph Ellersgaard

---

political_space97 *French Political Space example*

Description

The example dataset used by Brigitte Le Roux & Henry Rouanet (2004):

Author(s)

Brigitte Le Roux

References

Perrineau, Pascal, Jean Chiche, Brigitte Le Roux, and Henry Rouanet. "L'espace politique des électeurs français à la fin des années 1990: nouveaux et anciens clivages, hétérogénéité des électorats." Revue Francaise de Science Politique, no. 3 (June 2000): 463–88.

Le Roux, Brigitte, and Henry Rouanet. Multiple Correspondence Analysis. Thousand Oaks, Calif.: Sage Publications, 2010.

Examples

```
# French Political Space example
data(political_space97)
#Recoding
political_space97$Democracy <- ifelse(political_space97$Democracy %in% 1:2, "1_2",
                                      political_space97$Democracy)
political_space97$Politicians <- ifelse(political_space97$Politicians %in% 1:2, "1_2",
                                        political_space97$Politicians)

#Assigning questions to themes
ethno   <- data.frame(Immigrants = political_space97$Immigrants,
                      "North-Africans" = political_space97$NorthAfricans,
                      Races = political_space97$Races,
                       "At home" = political_space97$AtHome, check.names = FALSE)

autho   <- data.frame("Death Penalty" = political_space97$DeathPenalty,
                       School = political_space97$School, check.names = FALSE)

social  <- data.frame("Strike Effectiveness" = political_space97$StrikeEffectivness,
                       "Strike 95" = political_space97$Strike95,
                      "Unions" = political_space97$Unions,
                "Public services" = political_space97$PublicServices, check.names = FALSE)

economy <- data.frame(Liberalism = political_space97$Liberalism,
                      Profit = political_space97$Profit,
                      Privatization = political_space97$Privatization,
                    Globalization = political_space97$Globalization, check.names = FALSE)

politics <- data.frame(Democracy = political_space97$Democracy,
                       Politicians = political_space97$Politicians, check.names = FALSE)

supranat <- data.frame(Euro = political_space97$Euro, "EU Power" = political_space97$EUpower,
                       "End EU" = political_space97$EndEU,
                   "EU protection" = political_space97$EUprotection, check.names = FALSE)

# Creating and naming list of headings
active <- list(ethno, autho, social, economy, politics, supranat)
names(active) <- c("Ethnocentrism", "Authoritarianism",
                    "Social", "Economy", "Politics", "Supranationality")
sup    <- data.frame(political_space97$Vote)

result <- soc.mca(active, sup = sup, passive = ": 5")
headings(result)
```

```
map.active(result, point.color = result$headings,
           point.shape = result$headings, label.color = result$headings)
```

`print.soc.mca` *Print soc.ca objects*

Description

Prints commonly used measures used in the analysis of multiple correspondence analysis

Usage

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

Arguments

`x`	is a soc.ca class object
`...`	further arguments are ignored

Value

Active dimensions is the number of dimensions remaining after the reduction of the dimensionality of the analysis.

Active modalities is the number of modalities that are not set as passive.

Share of passive mass is the percentage of the total mass that is represented by the passive modalities.

The values represented in the scree plot are the adjusted inertias, see variance

The active variables are represented with their number of active modalities and their share of the total variance/inertia.

See Also

soc.mca, contribution

Examples

```
example(soc.ca)
print(result)
```

`prune.mca` *Remove unnecessary variables from an MCA*

Description

This function tests and removes variables that have no or too few relations with other variables. In other words variables that only contribute with random noise to the analysis. Removing these variables will tend to increase the strength of the first dimensions and give a wider dispersion of the cloud of cases on the first dimensions. Removing these variables can also give a simpler analysis that is easier to interpret and communicate. The core of the pruning procedure uses the mca.eigen.check to construct a weighted network of relations between variables. Tie strength is measured by the first eigenvalue of an MCA between the two variables. Ties between variables with a weak relationship are removed and variables with few connections to other variables are discarded. With the default values a analysis without irrelevant variables is unchanged. Note that passive categories are inherited from the original analysis and are not included in the mca.eigen.check. This procedure does not help with variables that are too strongly related.

Usage

```
prune.mca(
  r,
  eigen.cut.off = 0.55,
  network.pruning = TRUE,
  average.pruning = FALSE,
  min.degree = 1
)
```

Arguments

`r`	a result object from soc.mca
`eigen.cut.off`	the cut.off for the first eigen value from mca.eigen.check
`network.pruning`	If TRUE variables are pruned on the basis their degree
`average.pruning`	If TRUE variables with a sum of ties below average are discarded. This
`min.degree`	the minimum number of ties a variable has to have to remain in the analysis

Value

A list containing:

`var`	a tibble with the weighted degree of the variables
`mca.eigen.check`	The results from mca.eigen.check
`g`	a network graph - see igraph
`remaining.var`	a character vector with the names of the remaining variables

removed	a character vector with the names of the removed variables
pruned.r	A pruned version of the original soc.mca object

References

Inspired by: Durand, Jean-Luc, and Brigitte Le Roux. 2018. "Linkage Index of Variables and its Relationship with Variance of Eigenvalues in PCA and MCA." Statistica Applicata 29(2):123–35. doi: 10.26398/ijas.0029-006.

Examples

```
example(soc.mca)
pr <- prune.mca(result)
pr$removed              # This example has no irrelevant variables so nothing is removed
```

---

randomize.mca *Create a randomized mca on the basis of an existing mca*

Description

We sample from each of the active variables independently removing the original correlations but retaining the frequencies of the categories. This function is useful to see the extent to which the mca solution reflects the correlations between variables or the frequency distribution between the active categories. Passive categories are inherited from the original analysis.

Usage

```
randomize.mca(r, replace = FALSE)
```

Arguments

r	a result object from soc.mca
replace	

Value

a soc.mca object

Examples

```
example(soc.mca)
randomize.mca(result)
```

soc.ca *Soc.ca a package for specific correspondence analysis*

Description

This package is optimized to the needs of scientists within the social sciences. The soc.ca package produces specific and class specific multiple correspondence analysis on survey-like data. Soc.ca is optimized to only give the most essential statistical output sorted so as to help in analysis. Seperate functions exists for near publication-ready plots and tables.

Details

We are in debt to the work of others, especially Brigitte Le Roux and Henry Rouanet for the mathematical definitions of the method and their examples. Furthermore this package was initially based on code from the ca package written by Michael Greenacre and Oleg Nenadic.

If you are looking for features that are absent in soc.ca, it may be available in some of these packages for correspondence analysis: **ca**, **anacor** and **FactoMineR**.

References

Le Roux, Brigitte, and Henry Rouanet. 2010. Multiple correspondence analysis. Thousand Oaks: Sage.

Le Roux, Brigitte, and Henry Rouanet. 2004. Geometric Data Analysis from Correspondence Analysis to Structured Data Analysis. Dordrecht: Kluwer Academic Publishers.

Examples

```
data(taste)
# Create a data frame of factors containing all the active variables
taste          <- taste[which(taste$Isup == 'Active'), ]

attach(taste)
active         <- data.frame(TV, Film, Art, Eat)
sup            <- data.frame(Gender, Age, Income)
detach(taste)

# Runs the analysis
result         <- soc.mca(active, sup)
```

soc.csa *Class Specific Multiple Correspondence Analysis*

Description

soc.csa performs a class specific multiple correspondence analysis on a data.frame of factors, where cases are rows and columns are variables. Most descriptive and analytical functions that work for soc.mca, also work for soc.csa

Usage

```
soc.csa(object, class.indicator, sup = NULL)
```

Arguments

object	is a soc.ca class object created with soc.mca
class.indicator	the row indices of the class specific individuals
sup	Defines the supplementary modalities in a data.frame with rows of individuals and columns of factors, without NA's

Value

nd	Number of active dimensions
n.ind	The number of active individuals
n.mod	The number of active modalities
eigen	Eigenvectors
total.inertia	The sum of inertia
adj.inertia	A matrix with all active dimensions, adjusted and unadjusted inertias. See variance
freq.mod	Frequencies for the active modalities. See add.to.label
freq.sup	Frequencies for the supplementary modalities. See add.to.label
ctr.mod	A matrix with the contribution values of the active modalities per dimension. See contribution
ctr.ind	A matrix with the contribution values of the individuals per dimension.
cor.mod	The correlation or quality of each modality per dimension.
cor.ind	The correlation or quality of each individual per dimension.
mass.mod	The mass of each modality
coord.mod	A matrix with the principal coordinates of each active modality per dimension.
coord.ind	A matrix with the principal coordinates of each individual per dimension.

`coord.sup` A matrix with the principal coordinates of each supplementary modality per dimension. Notice that the position of the supplementary modalities in class specific analysis is the mean point of the individuals, which is not directly comparable with the cloud of the active modalities.

`indicator.matrix` A indicator matrix. See indicator

`names.mod` The names of the active modalities

`labels.mod` The shorter labels of the active modalities

`names.ind` The names of the individuals

`names.sup` The names of the supplementary modalities

`names.passive` The names of the passive modalities

`modal` A matrix with the number of modalities per variable and their location

`variable` A vector with the name of the variable for each of the active modalities

`variable.sup` A vector with the name of the variable for each of the supplementary modalities

`original.class.indicator` The class indicator

`original.result` The original soc.ca object used for the CSA

Author(s)

Anton Grau Larsen, University of Copenhagen

Stefan Bastholm Andrade, University of Copenhagen

Christoph Ellersgaard, University of Copenhagen

References

Le Roux, B., og H. Rouanet. 2010. Multiple correspondence analysis. Thousand Oaks: Sage.

See Also

add.to.label, contribution

Examples

```
example(soc.ca)
class.age    <- which(taste$Age == '55-64')
res.csa      <- soc.csa(result, class.age)
res.csa
```

soc.mca *soc.mca* soc.mca *performs a specific multiple correspondence analysis on a data.frame of factors, where cases are rows and columns are variables.*

Description

Specific Multiple Correspondence Analysis

Usage

```
soc.mca(
  active,
  sup = NULL,
  identifier = NULL,
  passive = getOption("passive", default = "Missing"),
  weight = NULL,
  Moschidis = FALSE,
  detailed.results = FALSE
)
```

Arguments

active	Defines the active modalities in a data.frame with rows of individuals and columns of factors, without NA's'. Active can also be a named list of data.frames. The data.frames will correspond to the analytical headings.
sup	Defines the supplementary modalities in a data.frame with rows of individuals and columns of factors, without NA's
identifier	A single vector containing a single value for each row/individual in x and sup. Typically a name or an id.number.
passive	A single character vector with the full or partial names of the passive modalities. All names that have a full or partial match will be set as passive.
weight	a numeric vector with the weights for the individual rows. The weight is normalized afterwards.
Moschidis	If TRUE adjusts contribution values for rare modalities. see moschidis.
detailed.results	If FALSE the result object is trimmed to reduce its memory footprint.

Value

nd	Number of active dimensions
n.ind	The number of active individuals
n.mod	The number of active modalities
eigen	Eigenvectors

total.inertia	The sum of inertia
adj.inertia	A matrix with all active dimensions, adjusted and unadjusted inertias. See variance
freq.mod	Frequencies for the active modalities. See add.to.label
freq.sup	Frequencies for the supplementary modalities. See add.to.label
ctr.mod	A matrix with the contribution values of the active modalities per dimension. See contribution
ctr.ind	A matrix with the contribution values of the individuals per dimension.
cor.mod	The correlation or quality of each modality per dimension.
cor.ind	The correlation or quality of each individual per dimension.
mass.mod	The mass of each modality
coord.mod	A matrix with the principal coordinates of each active modality per dimension.
coord.ind	A matrix with the principal coordinates of each individual per dimension.
coord.sup	A matrix with the principal coordinates of each supplementary modality per dimension.
names.mod	The names of the active modalities
labels.mod	The shorter labels of the active modalities
names.ind	The names of the individuals
names.sup	The names of the supplementary modalities
names.passive	The names of the passive modalities
modal	A matrix with the number of modalities per variable and their location
variable	A character vector with the name of the variable of the active modalities
Rosenlund.tresh	A numeric vector with the contribution values adjusted with the Rosenlund threshold, see: see p 92 in: Rosenlund, Lennart. Exploring the City with Bourdieu: Applying Pierre Bourdieu's Theories and Methods to Study the Community. Saarbrücken: VDM Verlag Dr. Müller, 2009.
t.test.sup	A matrix with a the student t-test of the coordinates of the supplementary variables
Share.of.var	A matrix the share of variance for each variable

Author(s)

Anton Grau Larsen

Jacob Lunding

Stefan Bastholm Andrade

Christoph Ellersgaard

References

Le Roux, B., og H. Rouanet. 2010. Multiple correspondence analysis. Thousand Oaks: Sage.

See Also

soc.csa, contribution

Examples

```
# Loads the "taste" dataset included in this package
data(taste)
# Create a data frame of factors containing all the active variables
taste          <- taste[which(taste$Isup == 'Active'), ]

attach(taste)
active         <- data.frame(TV, Film, Art, Eat)
sup            <- data.frame(Gender, Age, Income)
detach(taste)

# Runs the analysis
result         <- soc.mca(active, sup)

# Prints the results
result

# A specific multiple correspondence analysis
# options defines what words or phrases that are looked for in the labels of the active modalities.
options(passive = c("Film: CostumeDrama", "TV: Tv-Sport"))
soc.mca(active, sup)
options(passive = NULL)
```

---

supplementary.categories

Supplementary coordinates for a data.frame of factors

Description

Calculate the average coordinates in the category cloud of a soc.mca analysis.

Usage

```
supplementary.categories(object, sup, dim = 1:2)
```

Arguments

object	a soc.mca result object
sup	a data.frame of factors or an indicator matrix
dim	a numeric vector with the two dimensions calculated

Value

a data.frame with coordinates and labels

Examples

```
example(soc.mca)
supplementary.categories(result, sup)
```

`supplementary.individuals` *Add supplementary individuals to a result object*

Description

Add supplementary individuals to a result object

Usage

```
supplementary.individuals(object, sup.indicator, replace = FALSE)
```

Arguments

`object`	is a soc.ca class object created with soc.mca
`sup.indicator`	is a indicator matrix for the supplementary individuals with the same columns as the active variables in object.
`replace`	if TRUE the coordinates of the active individuals are discarded. If FALSE the coordinates of the supplementary and active individuals are combined. The factor `object$supplementary.individuals` marks the supplementary individuals.

Value

a soc.ca class object created with soc.mca

Examples

```
example(soc.mca)
res.pas   <- soc.mca(active, passive = "Costume")
res.sup   <- supplementary.individuals(res.pas, sup.indicator = indicator(active))
a         <- res.sup$coord.ind[res.sup$supplementary.individuals == "Supplementary",]
b         <- res.pas$coord.ind
all.equal(as.vector(a), as.vector(b))
map.ind(res.sup)
```

taste *Taste dataset*

Description

The taste example dataset used by Le Roux & Rouanet(2010):

Value

The variables included in the dataset:

`Preferred TV program`
: (8 categories): news, comedy, police, nature, sport, films, drama, soap operas

`Preferred Film`
: (8 categories): action, comedy, costume drama, documentary, horror, musical, romance, SciFi

`Preferred type of Art`
: (7 categories): performance, landscape, renaissance, still life, portrait, modern, impressionsism

`Preferred place to Eat out`
: (6 categories): fish & chips, pub, Indian restuarant, Italian restaurant, French restaurant, steak house

Author(s)

Brigitte Le Roux

References

Le Roux, Brigitte, Henry Rouanet, Mike Savage, og Alan Warde. 2008. "Class and Cultural Division in the UK". Sociology 42(6):1049-1071.

Le Roux, B., og H. Rouanet. 2010. Multiple correspondence analysis. Thousand Oaks: Sage.

Examples

```
## Not run:
# The taste example
data(taste)
data_taste           <- taste[which(taste$Isup == 'Active'), ]
active          <- data.frame(data_taste$TV, data_taste$Film, data_taste$Art, data_taste$Eat)
sup                  <- data.frame(data_taste$Gender, data_taste$Age, data_taste$Income)

# Multiple Correspondence Analysis
result.mca     <- soc.mca(active, sup)
str(result.mca)
result.mca

variance(result.mca) # See p.46 in Le Roux(2010)
```

```
contribution(result.mca, 1)
contribution(result.mca, 2)
contribution(result.mca, 1:3, mode = "variable")

map.active(result.mca, point.fill = result.mca$variable)
map.active(result.mca,
 map.title="Map of active modalities with size of contribution to 1. dimension",
 point.size=result.mca$ctr.mod[, 1])
map.active(result.mca,
 map.title="Map of active modalities with size of contribution to 2. dimension",
 point.size=result.mca$ctr.mod[, 2])

map.ind(result.mca)
map.ind(result.mca, dim=c(1, 2), point.color=result.mca$ctr.ind[, 1],
 point.shape=18) + scale_color_continuous(low="white", high="black")

# Plot of all dublets
map.ind(result.mca, map.title="Map of all unique individuals", point.color=duplicated(active))
map.ind(result.mca, map.title="Map with individuals colored by the TV variable",
 point.color=active$TV)

# Ellipse
map             <- map.ind(result.mca)
map.ellipse(result.mca, map, as.factor(data_taste$Age == '55-64'))

##### Specific Multiple Correspondence Analysis
options(passive= c("Film: CostumeDrama", "TV: Tv-Sport"))
result.smca     <- soc.mca(active, sup)
result.smca
result.smca$names.passive

##### Class Specific Correspondence Analysis
options(passive=NULL)

class.age     <- which(data_taste$Age == '55-64')
result.csca   <- soc.csa(result.mca, class.age, sup)
str(result.csca)
# Correlations
csa.measures(result.csca)
variance(result.csca)
contribution(result.csca, 1)
contribution(result.csca, 2)
contribution(result.csca, 1:3, mode = "variable")

# Plots
map.ind(result.csca)
map.csa.mca(result.csca)
map.csa.mca.array(result.csca)

## End(Not run)
```

to.MCA *Convert to MCA class from FactoMineR*

Description

Convert to MCA class from FactoMineR

Usage

```
to.MCA(object, active, dim = 1:5)
```

Arguments

object	is a soc.ca object
active	the active variables
dim	a numeric vector

Value

an FactoMineR class object

variance *Variance table*

Description

variance returns a table of variance for the selected dimensions.

Usage

```
variance(object, dim = NULL)
```

Arguments

object	is a soc.ca object
dim	is the included dimensions, if set to NULL, then only the dimensions explaining approx. more than 0.90 of the adjusted variance are included

Value

If assigned variance returns a matrix version of the table of variance.

See Also

soc.mca, print.soc.mca

Examples

```
example(soc.ca)
variance(result)
variance(result, dim = 1:4)
```

---

what.is.x *Check if data is valid for soc.mca*

Description

Performs tests on what has been passed on to soc.mca by the user.

Usage

```
what.is.x(x)
```

Arguments

x	the active variables sent to soc.mca

Value

a character vector with an evaluation of whether x is data.frame, a list of data.frames, an indicator or a list of indicators.

Examples

```
## Not run:
  # Valid scenarios ----

# X is a valid data.frame
x <- taste[, 2:7]
what.is.x(x)

# X is a valid indicator
x <- indicator(taste[, 2:7])
what.is.x(x)

# X is a valid list of data.frames with names
x <- list(nif = taste[, 2:3], hurma = taste[, 4:5])
what.is.x(x)

# X is a valid list of indicators
x <- list(nif = indicator(taste[, 2:3]), hurma = indicator(taste[, 4:5]))
what.is.x(x)

# Invalid scenarios ----

# X is a matrix - but not numeric
```

```
x <- as.matrix(taste[, 2:7])
what.is.x(x)

# X is a of data.frames list but does not have names
x <- list(taste[, 1:3], taste[, 4:5])
what.is.x(x)

# X is a list of indicators but does not have names
x <- list(indicator(taste[, 2:3]), indicator(taste[, 4:5]))
what.is.x(x)

# X is a data.frame and contains NA
x <- taste[, 2:7]
x[1,1] <- NA
what.is.x(x)

# X is a list of indicators and contains NA
x <- list(nif = indicator(taste[, 2:3]), hurma = indicator(taste[, 4:5]))
x[[1]][1,1] <- NA
what.is.x(x)

# X contains elements that are neither a matrix nor a data.frame
x <- list(nif = 1:10, taste[, 1:3], taste[, 4:7])
what.is.x(x)

# X contains both indicators and matrixes
x <- list(nif = taste[, 2:3], hurma = indicator(taste[, 5:6]))
what.is.x(x)

## End(Not run)
```

Index