Package ‘corregp’

October 12, 2022

Title Functions and Methods for Correspondence Regression

Version 2.0.2

Description A collection of tools for correspondence regression, i.e. the correspondence analysis of the crosstabulation of a categorical variable Y in function of another one X, where X can in turn be made up of the combination of various categorical variables. Consequently, correspondence regression can be used to analyze the effects for a polytomous or multinomial outcome variable.

Depends R (>= 3.2.0), graphics, diagram, rgl, stats

Imports data.table, ellipse, gplots, utils

Suggests datasets, grDevices, MASS, knitr

License GPL-3

LazyData true

VignetteBuilder knitr

Date 2018-03-11

RoxygenNote 6.0.1

NeedsCompilation no

Author Koen Plevoets [aut, cre]

Maintainer Koen Plevoets <`koen.plevoets@ugent.be`>

Repository CRAN

Date/Publication 2018-03-14 10:20:10 UTC

R topics documented:

`corregp-package` *Functions and Methods for Correspondence Regression*

Description

This package provides functions and methods for performing correspondence regression, i.e. the correspondence analysis of the crosstabulation of a categorical variable Y in function of another one X, where X can in turn be made up of the combination of various categorical variables. Consequently, correspondence regression can be used to analyze the effects for a polytomous or multinomial outcome variable. The central function in the package is `corregp`, which enables methods for printing, summarizing and plotting the output. Additionally, there are functions for computing confidence intervals, ellipses or 3D ellipsoids (by means of bootstrapping).

Contents

This package consists of the following datasets, functions, generics and methods (some internal functions are no longer exported in version 2):

Datasets:

- `HairEye` Hair and eye color of statistics students (data frame).
- `COMURE` The use of linguistic variants in translations vs. non-translations and in six different registers.
- `AVT` The use of linguistic variants in audio-visual translation (subtitles).
- `TSS` The use of inflected or uninflected determiners in vernacular Belgian Dutch.

Functions:

- `ci` A helper function to compute confidence intervals on the basis of a numeric vector.
- `corregp` The basic function to perform correspondence regression. Typically, one starts here, and then one uses `print`, `summary`, `anova`, `screeplot` or `plot` methods.
- `corregplicate` A function for repeated correspondence regressions with bootstrapping in order to handle large data sets.

Generics:

- `cint` Compute confidence intervals.
- `cell` Compute confidence ellipses.
- `cell3d` Compute 3D confidence ellipsoids.
- `ciplot` Plot confidence intervals.
- `pcplot` Plot parallel coordinates.
- `agplot` Plot an association graph.
- `plotag` Plot an association graph.

Methods:

- `print.corregp` Print the output of a correspondence regression.
- `summary.corregp` Give a `summary` of a correspondence regression.
- `print.summary.corregp` Print the `summary` of a correspondence regression.
- `screeplot.corregp` Make a scree plot on the basis of the output of a correspondence regression.
- `anova.corregp` Give an `anova` table on the basis of a correspondence regression.
- `print.anova.corregp` Print an `anova` table on the basis of a correspondence regression.
- `coef.corregp` Give the coefficients on the basis of a correspondence regression.
- `coefficients.corregp` Give the coefficients on the basis of a correspondence regression.
- `fitted.corregp` Give the fitted values on the basis of a correspondence regression.
- `fitted.values.corregp` Give the fitted values on the basis of a correspondence regression.
- `residuals.corregp` Give the residuals on the basis of a correspondence regression.
- `resid.corregp` Give the residuals on the basis of a correspondence regression.
- `cint.corregp` Compute confidence intervals on the basis of the output of a correspondence regression. Typically, this function is not so much used directly as it is called by a `ciplot.corregp` command.
- `ciplot.corregp` Plot confidence intervals on the basis of the output of a correspondence regression.
- `pcplot.corregp` Make a parallel coordinate plot on the basis of the output of a correspondence regression.
- `cell.corregp` Compute confidence ellipses on the basis of the output of a correspondence regression. Typically, this function is not so much used directly as it is called by a `plot.corregp` command.
- `plot.corregp` Plot the output (and the optional confidence ellipses) of a correspondence regression.
- `cell3d.corregp` Compute 3D confidence ellipsoids on the basis of a correspondence regression. Typically, this function is not so much used directly as it is called by a `plot3d.corregp` command.

- `plot3d.corregp` Plot the 3D output (and the optional confidence ellipsoids) of a correspondence regression.
- `agplot.corregp` Make an association graph on the basis of the output of a correspondence regression.
- `plotag.corregp` Make an association graph on the basis of the output of a correspondence regression.

Future prospects

- Specify a `predict` method for a.o. supplementary points.
- Specify a `plot` method for an anova table.
- Enable scale transformations for all plots (and corresponding confidence regions).
- Provide the possibility for so-called "calibration lines".

Author

Koen Plevoets, <`koen.plevoets@ugent.be`>

Acknowledgements

This package has benefited greatly from the helpful comments of Isabelle Delaere and Gert De Sutter. Thanks to Kurt Hornik and Uwe Ligges for proofing this package.

`agplot.corregp` *Plotting an Association Graph for Correspondence Regression*

Description

Function to make an association graph of the (significant) coordinate scores in correspondence regression.

Usage

```
## S3 method for class 'corregp'
agplot(x, axes = NULL, ysub = NULL, xsub = NULL,
  sort = NULL, na.rm = FALSE, col = "black", cex = par("cex"),
  font = par("font"), family = par("family"), lwd = par("lwd"),
  lty = par("lty"), ycol = col, xcol = col, ncol = c("white",
  "lightgray"), nwid = lwd, lcol = col, lwid = lwd, pcol = lcol,
  ppos = NULL, ptyp = "simple", zoom = 1, hshft = 0, vshft = 0,
  main = NULL, cl = 0.95, nq = TRUE, digits = 2, ...)

## S3 method for class 'corregp'
plotag(x, axes = NULL, ysub = NULL, xsub = NULL,
  sort = NULL, na.rm = FALSE, col = "black", cex = par("cex"),
  font = par("font"), family = par("family"), lwd = par("lwd"),
```

```
  lty = par("lty"), ycol = col, xcol = col, ncol = c("white",
  "lightgray"), nwid = lwd, lcol = col, lwid = lwd, pcol = lcol,
  ppos = NULL, ptyp = "simple", zoom = 1, hshft = 0, vshft = 0,
  main = NULL, cl = 0.95, nq = TRUE, digits = 2, ...)

agplot(x, ...)

plotag(x, ...)
```

Arguments

`x`	The output of a call to `corregp` (i.e. an object of class "corregp").
`axes`	The axes for which to plot the association graph: a vector of indices. Defaults to all the axes.
`ysub`	Vector of indices to select a subset of the Y levels.
`xsub`	Vector of indices to select a subset of the X levels. Can also be `"all"` or `"both"` (or abbreviations).
`sort`	Vector of axes for which to sort the coordinate scores. The default (`NULL`) plots all levels in the order in which they appear in the correspondence regression `x`.
`na.rm`	Logical specifying whether to omit `NA` coordinates from the plot. Defaults to `FALSE`.
`col`	Color of the association graph: either `numeric` or see `colors`.
`cex`	Character expansion factor: a number to specify the size of the text labels.
`font`	Font of the text labels (levels): `1` for plain, `2` for bold, `3` for italic, and `4` for bold italic. Defaults to `1`.
`family`	Font family of the text labels (levels): can be `"serif"`, `"sans"`, `"mono"` or one of the `Hershey` fonts.
`lwd`	Line width of the association graph: a number to specify the line width.
`lty`	Line type of the association graph (i.e. linking edges): `0` or `"blank"`, `1` or `"solid"`, `2` or `"dashed"`, `3` or `"dotted"`, `4` or `"dotdash"`, `5` or `"longdash"`, `6` or `"twodash"`. Defaults to `1`.
`ycol`	Color of the levels in Y: either `numeric` or see `colors`.
`xcol`	Color of the levels in X: either `numeric` or see `colors`.
`ncol`	Fill color of the nodes: either `numeric` or see `colors`. Defaults to `c("white","lightgray")`: the first value is for the nodes of the axes and the second value is for the nodes of the X and Y levels.
`nwid`	Line width of the nodes: a number to specify the line width. If a vector of two values is specified, then the first width is for the nodes of the axes and the second width is for the nodes of the X and Y levels.
`lcol`	Color of the links (edges): either `numeric` or see `colors`. If a vector of two values is specified, then the first color is for the scores > `0` and the second color is for the scores < `0`.

`lwid`	Line width of the links (edges): a number to specify the line width. If a vector of two values is specified, then the first width is for the scores > `0` and the second width is for the scores < `0`.
`pcol`	Color of the pointer (arrow head): either `numeric` or see `colors`. If a vector of two values is specified, then the first color is for the scores > `0` and the second color is for the scores < `0`.
`ppos`	Relative position of the pointer (arrow head): a vector of values between 0 and 1 for each axis.
`ptyp`	Type of of the pointer (arrow head): can be `"simple"`, `"curved"`, `"triangle"`, `"circle"`, `"ellipse"` or `"T"`. Defaults to `"simple"`.
`zoom`	Zoom factor of the association graph. Defaults to `1`.
`hshft`	Horizontal shift of the association graph. Defaults to `0`.
`vshft`	Vertical shift of the association graph. Defaults to `0`.
`main`	The main title of the association graph.
`cl`	The confidence level for the confidence intervals. Defaults to `0.95`.
`nq`	Logical specifying whether to use a normal quantile (i.e. apply `qnorm`) in the computation of the confidence interval. Defaults to `TRUE`. If `FALSE`, then the confidence interval is computed directly with the `quantile` function.
`digits`	Integer specifying the number of decimals for the scores as labels of the links (edges). Defauls to `2`.
`...`	Further arguments passed to or from other methods.

Details

Association graphs (of a `corregp` output) in the **corregp** package make use of various functionalities of the package **diagram**.

Value

A plot window containing the association graph.

See Also

`corregp`, `cint.corregp`, `pcplot.corregp`, `plot3d.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
agplot(haireye.crg, axes = 1:2, xsub = c("Hair", "Sex"))
plotag(haireye.crg, axes = 1:2, xsub = c("Hair", "Sex"))
```

`anova.corregp` *Building an ANOVA Table for Correspondence Regression*

Description

Method to construct an ANOVA table for correspondence regression, i.e. a table with the Chi-squared deviation for each term in the `formula` of the `corregp` call (or of each individual level in X in case `xep = FALSE`).

Usage

```
## S3 method for class 'corregp'
anova(object, nf = NULL, cl = 0.95, nq = TRUE, ...)
```

Arguments

`object`	The output of a call to `corregp` (i.e. an object of class "corregp").
`nf`	The number of dimensions to be retained in the reduced space. Defaults to all dimensions (no reduction).
`cl`	The confidence level for the confidence intervals. Defaults to `0.95`.
`nq`	Logical specifying whether to use a normal quantile (i.e. apply `qnorm`) in the computation of the confidence interval. Defaults to `TRUE`. If `FALSE`, then the confidence interval is computed directly with the `quantile` function.
`...`	Further arguments passed to or from other methods.

Details

If `object` was made with bootstrap replications, then `anova.corregp` will automatically compute confidence intervals for the Chi-squared deviations by means of the `ci` function.

Value

A matrix with the Chi-squared deviations for all the terms in the `formula` of `object`, based on the selected number of dimensions. If `object` was made with the argument `xep = FALSE`, then the output contains the Chi-squared deviation for every individual level in X.

See Also

`print.anova.corregp`, `ci`, `summary.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
anova(haireye.crg, nf = 2)
```

`AVT` *The Use of Linguistic Variants in Audio-Visual Translation (Subtitles)*

Description

This data set was a follow-up study to the `COMURE` project and was conducted at the Department of Translation, Interpreting and Communication of Ghent University between 2014 and 2018.

Format

A data frame with 3302 rows and 7 variables.

- `Variant` The linguistic variant used in a set of alternatives (27 levels).
- `Variable` The linguistic variable specifying a set of alternatives (13 levels).
- `Variety` The dichotomization of `Variant` into standard and non-standard.
- `Speaker` The role of the speaker in the data (2 levels).
- `Language` The language (and source language) of the data (3 levels).
- `Language2` The same as `Language` but with the observations of level `intra.nl` set to `NA`.
- `Genre` The genre or register of the data (2 levels).

Source

Prieels, L., I. Delaere, K. Plevoets and G. De Sutter (2015) A corpus-based multivariate analysis of linguistic norm-adherence in audiovisual and written translation. *Across Languages and Cultures* **16** (2), 209–231.

Examples

```
data(AVT)
# The execution of corregp may be slow, due to bootstrapping:
avt.crg <- corregp(Variant ~ Speaker * Language * Genre, data = AVT, part = "Variable", b = 3000)
avt.crg
summary(avt.crg, parm = "b", add_ci = TRUE)
screeplot(avt.crg, add_ci = TRUE)
anova(avt.crg, nf = 2)
avt.col <- ifelse( xtabs(~ Variant + Variety, data = AVT)[, "Standard"] > 0, "blue", "red")
plot(avt.crg, x_ell = TRUE, xsub = c("Speaker", "Language", "Genre"), col_btm = avt.col,
  col_top = "black")
```

`cell.corregp` *Confidence Ellipses for Correspondence Regression*

Description

Method to compute confidence ellipses for coordinates in correspondence regression.

Usage

```
## S3 method for class 'corregp'
cell(object, parm = "x", axes = 1:2, cl = 0.95,
  np = 100, ...)

cell(object, ...)
```

Arguments

`object`	The output of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to compute the confidence ellipses. Can be either `"y"`, `"x"`, or any vector of term names in X, level names in X or level names in Y. Defaults to `"x"`.
`axes`	The axes for which to compute the confidence ellipses: a vector of two values. Defaults to the first two axes.
`cl`	The confidence level for the confidence ellipses. Defaults to `0.95`.
`np`	The number of points to represent the confidence ellipses. Defaults to `100`.
`...`	Further arguments passed to or from other methods.

Details

`cell` (of a `corregp` output) makes use of `ellipse` from the package **ellipse**.

Typically, `cell` is not so much used directly as it is called by a `plot.corregp` command.

Value

A list containing `np` points for each confidence ellipse of interest.

See Also

`plot.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
cell(haireye.crg, parm = "y")
cell(haireye.crg, parm = c("Hair", "Sex"))
```

`cell3d.corregp` *3D Confidence Ellipsoids for Correspondence Regression*

Description

Method to compute 3D confidence ellipsoids for coordinates in correspondence regression.

Usage

```
## S3 method for class 'corregp'
cell3d(object, parm = "x", axes = 1:3, cl = 0.95, ...)

cell3d(object, ...)
```

Arguments

`object`	The output of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to compute the confidence ellipsoids. Can be either `"y"`, `"x"`, or any vector of term names in X, level names in X or level names in Y. Defaults to `"x"`.
`axes`	The axes for which to compute the confidence ellipsoids: a vector of three values. Defaults to the first three axes.
`cl`	The confidence level for the confidence ellipsoids. Defaults to `0.95`.
`...`	Further arguments passed to or from other methods.

Details

`cell3d` (of a `corregp` output) makes use of `ellipse3d` from the package **rgl**.

Typically, `cell3d` is not so much used directly as it is called by a `plot3d.corregp` command.

Value

A list containing coordinate points for each confidence ellipsoid of interest.

See Also

`plot3d.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
cell3d(haireye.crg, parm = "y")
cell3d(haireye.crg, parm = c("Hair", "Sex"))
```

`ci` *Confidence Interval*

Description

This is the basic function for computing a confidence interval on the basis of a sample of data values.

Usage

```
ci(x, cl = 0.95, nq = TRUE)
```

Arguments

`x`	A numeric vector.
`cl`	The confidence level for the confidence interval. Defaults to `0.95`.
`nq`	Logical specifying whether to use a normal quantile (i.e. apply `qnorm`) in the computation of the confidence interval. Defaults to `TRUE`. If `FALSE`, then the confidence interval is computed directly with the `quantile` function.

Value

A vector with two components `Lower` and `Upper` giving the lower and upper confidence limits respectively.

See Also

`ciplot.corregp`, `anova.corregp`, `agplot.corregp`, `confint`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
ci(haireye.crg$conf$eigen[, 1])
ci(haireye.crg$conf$eigen[, 2])
```

`cint.corregp` *Confidence Intervals for Correspondence Regression*

Description

Method to compute confidence intervals for coordinates in correspondence regression.

Usage

```
## S3 method for class 'corregp'
cint(object, parm = "x", axis, cl = 0.95, nq = TRUE,
  ...)

cint(object, ...)
```

Arguments

`object`	The output of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to compute the confidence intervals. Can be either `"y"`, `"x"`, or any vector of term names in X, level names in X or level names in Y. Defaults to `"x"`.
`axis`	The axis for which to compute the confidence intervals.
`cl`	The confidence level for the confidence interval. Defaults to `0.95`.
`nq`	Logical specifying whether to use a normal quantile (i.e. apply `qnorm`) in the computation of the confidence intervals. Defaults to `TRUE`. If `FALSE`, then the confidence intervals are computed directly with the `quantile` function.
`...`	Further arguments passed to or from other methods.

Details

`cint` (of a `corregp` output) makes use of `ci`.

Typically, `cint` is not so much used directly as it is called by a `ciplot.corregp` command.

Value

A matrix with `Lower` and `Upper` confidence limits for the coordinates of interest.

See Also

`ci`, `ciplot.corregp`, `agplot.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
cint(haireye.crg, parm = "y", axis = 1)
cint(haireye.crg, parm = c("Hair", "Sex"), axis = 1)
```

`ciplot.corregp` *Plotting Confidence Intervals for Correspondence Regression*

Description

Method to plot confidence intervals for coordinates in correspondence regression.

Usage

```
## S3 method for class 'corregp'
ciplot(x, parm = "x", axis, cl = 0.95, nq = TRUE,
  horiz = FALSE, na.rm = FALSE, type = "p", col = "darkgrey",
  cex = par("cex"), font = par("font"), family = par("family"),
  alim = NULL, adir = 1, ecol = "darkgrey", ewid = par("lwd"),
  etyp = par("lty"), psym = 16, pcol = par("col"), pcex = cex,
  pbgc = par("bg"), lwd = ewid, lty = etyp, sfrac = 0.01, gap = 0,
  main = NULL, sub = NULL, ...)

ciplot(x, ...)
```

Arguments

`x`	The output of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to plot the confidence intervals. Can be either `"y"`, `"x"`, or any vector of term names in X, level names in X or level names in Y. Defaults to `"x"`.
`axis`	The axis for which to plot the confidence intervals.
`cl`	The confidence level for the confidence intervals. Defaults to `0.95`.
`nq`	Logical specifying whether to use a normal quantile (i.e. apply `qnorm`) in the computation of the confidence intervals. Defaults to `TRUE`. If `FALSE`, then the confidence intervals are computed directly with the `quantile` function.
`horiz`	Logical specifying whether the confidence intervals should be plotted horizontally or not. Defaults to `FALSE`.
`na.rm`	Logical specifying whether to omit `NA` coordinates from the plot. Defaults to `FALSE`.
`type`	The type of plot: see `plot.default`. For correspondence regression, there is an additional option `"labs"` which plots the text labels at the centers of the confidence intervals. Defaults to `"p"`.

`col`	Color of the text labels: either `numeric` or see `colors`.
`cex`	Character expansion factor: a number to specify the size of the text labels.
`font`	Font of the text labels: `1` for plain, `2` for bold, `3` for italic, and `4` for bold italic. Defaults to `1`.
`family`	Font family of the text labels: can be `"serif"`, `"sans"`, `"mono"` or one of the `Hershey` fonts.
`alim`	Vector of two values specifying the lower and upper limit between which to plot the axis.
`adir`	Reading direction of the text labels on the (horizontal) axis: either a `numeric` value between `0` and `3` (see the `las` argument in the graphical parameters `par`) or a `character` value matching either `"horizontal"` or `"vertical"`. Defaults to `1` (horizontal).
`ecol`	Color of the error bars: either `numeric` or see `colors`.
`ewid`	Width of the error bars: a number to specify the line width.
`etyp`	Line type of the error bars: `0` or `"blank"`, `1` or `"solid"`, `2` or `"dashed"`, `3` or `"dotted"`, `4` or `"dotdash"`, `5` or `"longdash"`, `6` or `"twodash"`. Defaults to `1`.
`psym`	The symbol (or "plotting character") to use for the centers of the confidence intervals.
`pcol`	Color of the center symbol: either `numeric` or see `colors`.
`pcex`	Character expansion factor of the center symbol.
`pbgc`	Background color of the center symbol: either `numeric` or see `colors`.
`lwd`	Width of all lines except for the error bars, e.g. the connecting lines: a number to specify the line width.
`lty`	Line type of all lines except for the error bars, e.g. the connecting lines: `0` or `"blank"`, `1` or `"solid"`, `2` or `"dashed"`, `3` or `"dotted"`, `4` or `"dotdash"`, `5` or `"longdash"`, `6` or `"twodash"`. Defaults to `1`.
`sfrac`	Width of "crossbar" at the end of error bar as a fraction of the x plotting region. Defaults to 0.01.
`gap`	Space left between the center of the error bar and the lines marking the error bar in units of the height (width) of the letter "O". Defaults to 0.
`main`	The main title of the plot.
`sub`	The subtitle of the plot.
`...`	Further arguments passed to or from other methods.

Details

`ciplot` (of a `corregp` output) makes use of `plotCI` from the package **gplots**.

Value

A plot window containing the confidence intervals.

See Also

`ci`, `plotCI`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
ciplot(haireye.crg, parm = "y", axis = 1)
ciplot(haireye.crg, parm = c("Hair", "Sex"), axis = 1)
```

`coef.corregp` *Extracting Coefficients from Correspondence Regression*

Description

Method to extract the coefficients (i.e. scores) of a correspondence regression.

Usage

```
## S3 method for class 'corregp'
coef(object, parm = "x", axes = NULL, ...)

## S3 method for class 'corregp'
coefficients(object, parm = "x", axes = NULL, ...)
```

Arguments

`object`	The output of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to extract the coefficients. Can be either `"y"`, `"x"`, or any vector of term names in X, level names in X or level names in Y. Defaults to `"x"`.
`axes`	The axes for which to extract the coefficients: a vector of indices. Defaults to all the axes.
`...`	Further arguments passed to or from other methods.

Details

The coefficients in correspondence regression are the same as the coordinate scores.

Value

A matrix or vector with coefficients (i.e. scores) for the parameters and axes of interest.

See Also

`fitted.corregp`, `residuals.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
coef(haireye.crg, parm = c("Hair", "Sex"), axes = 1:2)
coefficients(haireye.crg, parm = c("Hair", "Sex"), axes = 1:2)
```

`COMURE` *The Use of Linguistic Variants in Translations vs. Non-translations and in Six Different Registers*

Description

This data set was a case study in the COMURE project ("**co**rpus-based, **mu**ltivariate research of **re**gister variation in translated and non-translated Belgian Dutch") which was conducted at the Department of Translation, Interpreting and Communication of Ghent University between 2010 and 2014.

Format

A data frame with 3762 rows and 5 variables.

- `Variant` The linguistic variant used in a set of alternatives (27 levels).
- `Variable` The linguistic variable specifying a set of alternatives (13 levels).
- `Variety` The dichotomization of `Variant` into standard and non-standard.
- `Register` The register or "Text type" of the data (6 levels).
- `Language` The language (and source language) of the data (3 levels).

Source

Delaere, I., G. De Sutter and K. Plevoets (2012) Is translated language more standardized than non-translated language? *Target* **24** (2), 203–224.

Examples

```
data(COMURE)
# The execution of corregp may be slow, due to bootstrapping:
comure.crg <- corregp(Variant ~ Register * Language, data = COMURE, part = "Variable", b = 3000)
comure.crg
summary(comure.crg, parm = "b", add_ci = TRUE)
screeplot(comure.crg, add_ci = TRUE)
anova(comure.crg, nf = 2)
comure.col <- ifelse( xtabs(~ Variant + Variety, data = COMURE)[, "Standard"] > 0, "blue", "red")
plot(comure.crg, x_ell = TRUE, xsub = c("Register", "Language"), col_btm = comure.col,
  col_top = "black")
```

`confGet` *Getting* `conf` *Components from* `corregp` *Objects*

Description

Internal function for retrieving the `conf` component(s) in a `corregp` object.

Usage

```
confGet(crg, parm)
```

Arguments

`crg`	The output of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to retrieve the `conf` components. Can be either `"y"`, `"x"`, or any vector of term names in X, level names in X or level names in Y.

Details

`confGet` is an internal function to be called by `cint.corregp`, `cell.corregp` or `cell3d.corregp`, but not by users.

Value

A list of components selected with `parm`.

`corregp` *Correspondence Regression*

Description

This is the basic function for *correspondence regression*, i.e. the correspondence analysis of a contingency table formed by the categorical variables Y and X, where X can be in turn made up of the combinations of various categorical variables.

Usage

```
corregp(formula, data, part = NULL, b = 0, xep = TRUE, std = FALSE,
  rel = TRUE, phi = FALSE, chr = ".", b_scheme = "multinomial")
```

Arguments

`formula`	A `formula` specification of which factors to cross with each other. The left-hand (`y`) side must be a single factor. The right-hand side (`x`) can involve all the usual specifications of interactions and/or nested analyses.
`data`	The data frame containing the variables specified in the `formula`.
`part`	Character vector specifying the names of conditional factors (e.g. a factor partioning the levels of the left-hand side `y` into groups). This argument is relevant for analyses in which one wants to remove between-item variation.
`b`	Number of the bootstrap replications (simulations). If `0` (i.e. the default), then the analysis is exploratory.
`xep`	Logical specifying whether to output the separate terms in the right-hand side (`x`) as components in a list. If `FALSE`, then all `x` output is collected in a matrix.
`std`	Logical specifying whether to output the standardized coordinates. Defaults to `FALSE`.
`rel`	Logical specifying whether to divide the coordinates by the `sqrt` of their totals, so that one obtains coordinates for the relative frequencies (as is customary in correspondence analysis). Defaults to `TRUE`.
`phi`	Logical specifying whether to compute the output on the scale of the *Chi-squared* value of the contingency table or of the *Phi-squared* value (which is *Chi-squared* divided by N). Reminiscent of `corresp` in package **MASS**, defaults to `FALSE`.
`chr`	Character specifying the separator string for constructing the interaction terms.
`b_scheme`	Character specifying the sampling scheme for bootstrapping. Must match either `"multinomial"` (the default) or `"product-multinomial"`.

Details

Correspondence regression rests on the idea, described by Gilula and Haberman (1988), of using a correspondence analysis to model a polytomous or multinomial (i.e. 'multi-category') response variable (`Y`) in terms of other (possibly interacting) factors (`X`) (see also 3.2 in Van der Heijden et al. 1989). These are specified in the argument `formula`, which can be constructed in all the usual ways of specifying a model formula: e.g.

- `Y ~ X1 + X2 + X1 : X2` or `Y ~ X1 * X2`
- `Y ~ (X1 + X2 + X3) ^ 2`
- `Y ~ X1 * X2 * X3 - X1 : X2 : X3`
- `...`

Correspondence regression then crosstabulates the `Y` factor with all the combinations in `X`, thus producing a typical contingency table, on which a simple correspondence analysis is performed (see Greenacre 2017: 121-128 for the outline of this approach). The more general effects in `X` are obtained by aggregating the combinations.

Correspondence regression also allows for inferential validation of the effects, which is done by means of the bootstrap (in fact, Monte Carlo simulation). Setting the argument `b` to a number > 0, `b` replicates of the contingency table are generated with multinomial sampling. From these, `b` new values are derived for the coordinates in both `Y` and `X` as well as for the eigenvalues (also called the "principal inertias"). On the basis of the replicate/simulated values, confidence intervals, ellipses or ellipsoids can be computed. CAUTION: bootstrapping/simulation is computationally quite intensive, so it can take a while to reach results, especially with a large `b`.

The argument `parm` can be used when one wants to perform a correspondence regression of `Y` onto `X` conditional on other factors. These conditioning factors are therefore equivalent to *random factors*, and `corregp` always conditions on the joint variable of all the specified factors. One such use of conditioning factors is a so-called *lectometric* analysis in linguistics, where the levels of `Y` are grouped/partitioned/nested into clusters and one wants to exclude the heterogeneity between the clusters.

Value

An object of class "corregp", i.e. a list with components:

`eigen`	A vector of eigenvalues of the correpondence regression.
`y`	The coordinates (matrix) of the Y levels.
`x`	The coordinates of the X levels. If `xep` is `TRUE`, then this is a list with a component for each term name.
`freq`	A list of the frequencies of every Y and X level.
`conf`	If $b > 0$. A list of bootstrap replicates for the eigenvalues, the coordinates of Y levels, the coordinates of X levels and the frequencies of both the Y levels and the X levels.
`aux`	A list of auxiliary information (such as the U and V matrices of the SVD, the specified values for all the arguments) to be passed to other functions and methods.

References

Gilula, Z. and S.J. Haberman (1988) The analysis of multivariate contingency tables by restricted canonical and restricted association models. *Journal of the American Statistical Association* **83** (403), 760–771.

Greenacre, M. (2017) *Correspondence analysis in practice, Third edition*. Boca Raton: Chapman and Hall/CRC.

Van der Heijden, P.G.M., A. de Falguerolles and J. de Leeuw (1989) A combined approach to contingency table analysis using correspondence analysis and log-linear analysis. *Applied Statistics* **38** (2), 249–292.

See Also

`print.corregp`, `summary.corregp`, `screeplot.corregp`, `anova.corregp`, `plot.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
haireye.crg
```

`corregplicate` *Repeated Correspondence Regression*

Description

A function for repeated correspondence regressions with bootstrapping in order to handle large data sets. This is essentially a wrapper `replicate(n = r, expr = corregp(...), simplify = FALSE)`, so it may dissappear in the future.

Usage

```
corregplicate(formula, data, part = NULL, b = 100, r = 10, xep = TRUE,
  std = FALSE, rel = TRUE, phi = FALSE, chr = ".",
  b_scheme = "multinomial")
```

Arguments

`formula`	A `formula` specification of which factors to cross with each other. The left-hand (`y`) side must be a single factor. The right-hand side (`x`) can involve all the usual specifications of interactions and/or nested analyses.
`data`	The data frame containing the variables specified in the `formula`.
`part`	Character vector specifying the names of conditional factors (e.g. a factor partioning the levels of the left-hand side `y` into groups). This argument is relevant for analyses in which one wants to remove between-item variation.
`b`	Number of the bootstrap replications (simulations).
`r`	Number of repeated calls to `corregp`.
`xep`	Logical specifying whether to output the separate terms in the right-hand side (`x`) as components in a list. If `FALSE`, then all `x` output is collected in a matrix.
`std`	Logical specifying whether to output the standardized coordinates. Defaults to `FALSE`.
`rel`	Logical specifying whether to divide the coordinates by the `sqrt` of their totals, so that one obtains coordinates for the relative frequencies (as is customary in correspondence analysis). Defaults to `TRUE`.
`phi`	Logical specifying whether to compute the output on the scale of the *Chi-squared* value of the contingency table or of the *Phi-squared* value (which is *Chi-squared* divided by *N*). Reminiscent of `corresp` in package **MASS**, defaults to `FALSE`.
`chr`	Character specifying the separator string for constructing the interaction terms.
`b_scheme`	Character specifying the sampling scheme for bootstrapping. Must match either `"multinomial"` (the default) or `"product-multinomial"`.

Value

An object of class "corregp" in which the bootstrap replications of all the repeated calls to `corregp` are put together.

See Also

`corregp`.

`fitted.corregp` *Extracting Fitted Values from Correspondence Regression*

Description

Method to extract the fitted values of a correspondence regression.

Usage

```
## S3 method for class 'corregp'
fitted(object, parm = "all", nf = NULL, ...)

## S3 method for class 'corregp'
fitted.values(object, parm = "all", nf = NULL, ...)
```

Arguments

`object`	The output of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to extract the fitted values. Can be `"all"`, `"both"` (or abbreviations), `"y"` or `"x"` for the fitted values of every cell in the data, but it can also be any vector of term names in X or level names in X. Defaults to `"all"`.
`nf`	The number of dimensions to be retained in the reduced space. Defaults to all dimensions (no reduction).
`...`	Further arguments passed to or from other methods.

Details

If all dimensions are retained, then the fitted values will only be equal to the observed counts if no conditioning factors were specified with the argument `"part"` in the `corregp` call. This is because the associations with the conditioning factors (in `"part"`) are not taken into account.

Value

A matrix or vector with the fitted values for the parameters of interest, based on the selected number of dimensions.

See Also

`coef.corregp`, `residuals.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
fitted(haireye.crg, parm = c("Hair", "Sex"), nf = 2)
fitted.values(haireye.crg, parm = c("Hair", "Sex"), nf = 2)
```

`HairEye` *Hair and Eye Color of Statistics Students (Data Frame)*

Description

The distribution of hair color, eye color and sex among 592 statistics students (from Snee 1974 and Friendly 1992).

Format

A data frame with 592 rows and 3 variables.

Source

This is simply a data frame version of the in-built data set `HairEyeColor`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
haireye.crg
summary(haireye.crg, parm = "b", add_ci = TRUE)
screeplot(haireye.crg, add_ci = TRUE)
anova(haireye.crg, nf = 2)
plot(haireye.crg, x_ell = TRUE, xsub = c("Hair", "Sex"))
```

`pcplot.corregp` *Parallel Coordinate Plotting for Correspondence Regression*

Description

Method to produce a *parallel coordinate plot* of the output of a correspondence regression.

Usage

```
## S3 method for class 'corregp'
pcplot(x, parm = "x", axes, add_ci = FALSE, cl = 0.95,
  nq = TRUE, col = "darkgrey", cex = par("cex"), font = par("font"),
  family = par("family"), lwd = par("lwd"), lty = par("lty"),
  lcol = col, psym = NULL, pcol = col, pcex = cex, ecol = "red",
  ewid = 1, etyp = 2, acol = "black", awid = 1, atyp = 1,
  acex = cex, afnt = font, adir = 1, add_scale = FALSE, main = NULL,
  sub = NULL, ...)

pcplot(x, ...)
```

Arguments

`x`	The output of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to plot the coordinates. Can be either `"y"`, `"x"`, or any vector of term names in X, level names in X or level names in Y. Defaults to `"x"`.
`axes`	The axes to plot.
`add_ci`	Logical specifying whether to include the confidence intervals. Defaults to `FALSE`.
`cl`	The confidence level for the confidence intervals. Defaults to `0.95`.
`nq`	Logical specifying whether to use a normal quantile (i.e. apply `qnorm`) in the computation of the confidence intervals. Defaults to `TRUE`. If `FALSE`, then the confidence intervals are computed directly with the `quantile` function.
`col`	Color of the text labels: either `numeric` or see `colors`.
`cex`	Character expansion factor: a number to specify the size of the text labels.
`font`	Font of the text labels: `1` for plain, `2` for bold, `3` for italic, and `4` for bold italic. Defaults to `1`.
`family`	Font family of the text labels: can be `"serif"`, `"sans"`, `"mono"` or one of the `Hershey` fonts.
`lwd`	Width of the connecting lines: a number to specify the line width.
`lty`	Line type of the connecting lines: `0` or `"blank"`, `1` or `"solid"`, `2` or `"dashed"`, `3` or `"dotted"`, `4` or `"dotdash"`, `5` or `"longdash"`, `6` or `"twodash"`. Defaults to `1`.
`lcol`	Color of the connecting lines: either `numeric` or see `colors`.
`psym`	The symbol (or "plotting character") for the values of the coordinates on the axes.
`pcol`	Color of the symbol for the values on the axes: either `numeric` or see `colors`.
`pcex`	Character expansion factor of the symbol for the values on the axes.
`ecol`	Color of the error lines (connecting the confidence intervals on each axis): either `numeric` or see `colors`.

`ewid`	Width of the error lines (connecting the confidence intervals on each axis): a number to specify the line width.
`etyp`	Line type of the error lines (connecting the confidence intervals on each axis): `0` or `"blank"`, `1` or `"solid"`, `2` or `"dashed"`, `3` or `"dotted"`, `4` or `"dotdash"`, `5` or `"longdash"`, `6` or `"twodash"`. Defaults to `2`.
`acol`	Color of the parallel axes: either `numeric` or see `colors`.
`awid`	Width of the parallel axes: a number to specify the line width.
`atyp`	Line type of the parallel axes: `0` or `"blank"`, `1` or `"solid"`, `2` or `"dashed"`, `3` or `"dotted"`, `4` or `"dotdash"`, `5` or `"longdash"`, `6` or `"twodash"`. Defaults to `1`.
`acex`	Character expansion factor for the labels of the parallel axes.
`afnt`	Font for the labels of the parallel axes: `1` for plain, `2` for bold, `3` for italic, and `4` for bold italic.
`adir`	Reading direction of the labels on the parallel axes: either a `numeric` value between `0` and `3` (see the `las` argument in the graphical parameters `par`) or a `character` value matching either `"horizontal"` or `"vertical"`. Defaults to `1` (horizontal).
`add_scale`	Logical specifying whether to add a scale for the parallel axes (which are normalised).
`main`	The main title of the plot.
`sub`	The subtitle of the plot.
`...`	Further arguments passed to or from other methods.

Details

Although adding lines for confidence intervals is possible, it is not recommended, as it typically leads to an unreadable plot.

Value

A parallel coordinate plot containing the output of a correspondence regression.

See Also

`ciplot.corregp`, `plot.corregp`, `plot3d.corregp`, `agplot.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
pcplot(haireye.crg, parm = "y", axes = 1:3)
pcplot(haireye.crg, parm = c("Hair", "Sex"), axes = 1:3)
```

`plot.corregp` *Plotting Correspondence Regression*

Description

Basic method to plot the output of a correspondence regression.

Usage

```
## S3 method for class 'corregp'
plot(x, axes = 1:2, y_btm = TRUE, y_ell = FALSE,
  x_ell = FALSE, ysub = NULL, xsub = NULL, hlim = NULL, vlim = NULL,
  expa_btm = 1, expa_top = 1, asp = 1, asp_btm = asp, asp_top = asp,
  col_btm = "darkgrey", col_top = "red", cex_btm = par("cex"),
  cex_top = cex_btm, font_btm = par("font"), font_top = font_btm,
  fam_btm = par("family"), fam_top = fam_btm, col_ell = par("col"),
  lwd_ell = par("lwd"), lty_ell = par("lty"), col_ori = par("col"),
  lwd_ori = par("lwd"), lty_ori = 1, main = NULL, sub = NULL,
  hlab = NULL, vlab = NULL, cl = 0.95, np = 100, add_ori = TRUE, ...)
```

Arguments

`x`	The output of a call to `corregp` (i.e. an object of class "corregp").
`axes`	The axes to plot: a vector of two values. Defaults to the first two axes.
`y_btm`	Logical specifying whether the Y levels should be plotted first ("at the bottom") and then be overlaid by the X levels. Defaults to `TRUE`.
`y_ell`	Logical specifying whether the confidence ellipses of the Y levels should be plotted. Defaults to `FALSE`.
`x_ell`	Logical specifying whether the confidence ellipses of the X levels should be plotted. Defaults to `FALSE`.
`ysub`	Vector of indices to select a subset of the Y levels.
`xsub`	Vector of indices to select a subset of the X levels.
`hlim`	Vector of two values specifying the lower and upper limit between which to plot the horizontal axis.
`vlim`	Vector of two values specifying the lower and upper limit between which to plot the vertical axis.
`expa_btm`	Expansion factor for the bottom coordinates: a number to rescale the axes.
`expa_top`	Expansion factor for the top coordinates: a number to rescale the axes.
`asp`	The aspect ratio for the whole plot. See `plot.window`.
`asp_btm`	The aspect ratio for the bottom coordinates. See `plot.window`.
`asp_top`	The aspect ratio for the top coordinates. See `plot.window`.
`col_btm`	Color of the bottom levels: either `numeric` or see `colors`. Defaults to `"darkgrey"`.

`col_top`	Color of the top levels: either `numeric` or see `colors`. Defaults to `"red"`.
`cex_btm`	Character expansion factor of the bottom levels: a number to specify the size of the text labels.
`cex_top`	Character expansion factor of the top levels: a number to specify the size of the text labels.
`font_btm`	Font of the bottom levels: `1` for plain, `2` for bold, `3` for italic, and `4` for bold italic. Defaults to `1`.
`font_top`	Font of the top levels: `1` for plain, `2` for bold, `3` for italic, and `4` for bold italic. Defaults to `1`.
`fam_btm`	Font family of the bottom levels: can be `"serif"`, `"sans"`, `"mono"` or one of the `Hershey` fonts.
`fam_top`	Font family of the top levels: can be `"serif"`, `"sans"`, `"mono"` or one of the `Hershey` fonts.
`col_ell`	Color of the confidence ellipses: either a number or see `colors`.
`lwd_ell`	Width of the confidence ellipses: a number to specify the line width.
`lty_ell`	Line type of the confidence ellipses: `0` or `"blank"`, `1` or `"solid"`, `2` or `"dashed"`, `3` or `"dotted"`, `4` or `"dotdash"`, `5` or `"longdash"`, `6` or `"twodash"`. Defaults to `1`.
`col_ori`	Color of the lines through the origin: either `numeric` or see `colors`.
`lwd_ori`	Width of the lines through the origin: a number to specify the line width.
`lty_ori`	Line type of the lines through the origin: `0` or `"blank"`, `1` or `"solid"`, `2` or `"dashed"`, `3` or `"dotted"`, `4` or `"dotdash"`, `5` or `"longdash"`, `6` or `"twodash"`. Defaults to `1`.
`main`	The main title of the plot.
`sub`	The subtitle of the plot.
`hlab`	The title of the horizontal axis.
`vlab`	The title of the vertical axis.
`cl`	The confidence level for the confidence ellipses. Defaults to `0.95`.
`np`	The number of points to represent the confidence ellipses. Defaults to `100`.
`add_ori`	Logical specifying whether to add lines through the origin. Defaults to `TRUE`.
`...`	Further arguments passed to or from other methods.

Details

The plot of a correspondence regression is by definition a `biplot`.

Value

A plot window containing the output of a correspondence regression.

References

Gower, J., S. Lubbe and N. Le Roux (2011) *Understanding biplots*. Chichester: Wiley.

Greenacre, M. (2010) *Biplots in practice*. Bilbao: Fundacion BBVA.

See Also

`corregp`, `summary.corregp`, `screeplot.corregp`, `anova.corregp`, `biplot`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
plot(haireye.crg, x_ell = TRUE, xsub = c("Hair", "Sex"))
```

`plot3d.corregp` *3D Plotting for Correspondence Regression*

Description

Method to produce a 3D plot for a correspondence regression.

Usage

```
## S3 method for class 'corregp'
plot3d(x, axes = 1:3, y_btm = TRUE, y_ell = FALSE,
  x_ell = FALSE, ysub = NULL, xsub = NULL, hlim = NULL, vlim = NULL,
  dlim = NULL, asp = par3d("scale"), col_btm = "darkgrey",
  col_top = "red", cex_btm = par3d("cex"), cex_top = cex_btm,
  font_btm = par3d("font"), font_top = font_btm,
  fam_btm = par3d("family"), fam_top = fam_btm, col_ell = "black",
  lwd_ell = 1, lty_ell = "shade", opa_ell = 0.2, col_ori = "grey",
  lwd_ori = 1, main = NULL, sub = NULL, hlab = NULL, vlab = NULL,
  dlab = NULL, cl = 0.95, add_ori = TRUE, ...)
```

Arguments

`x`	The output of a call to `corregp` (i.e. an object of class "corregp").
`axes`	The axes to plot: a vector of three values. Defaults to the first three axes.
`y_btm`	Logical specifying whether the Y levels should be plotted first ("at the bottom") and then be overlaid by the X levels. Defaults to `TRUE`.
`y_ell`	Logical specifying whether the confidence ellipsoids of the Y levels should be plotted. Defaults to `FALSE`.
`x_ell`	Logical specifying whether the confidence ellipsoids of the X levels should be plotted. Defaults to `FALSE`.
`ysub`	Vector of indices to select a subset of the Y levels.
`xsub`	Vector of indices to select a subset of the X levels.
`hlim`	Vector of two values specifying the lower and upper limit between which to plot the horizontal axis.

`vlim`	Vector of two values specifying the lower and upper limit between which to plot the vertical axis.
`dlim`	Vector of two values specifying the lower and upper limit between which to plot the "depth" axis.
`asp`	The aspect ratio for the whole plot. See `aspect3d`.
`col_btm`	Color of the bottom levels: either `numeric` or see `colors`. Defaults to `"darkgrey"`.
`col_top`	Color of the top levels: either `numeric` or see `colors`. Defaults to `"red"`.
`cex_btm`	Character expansion factor of the bottom levels: a number to specify the size of the text labels.
`cex_top`	Character expansion factor of the top levels: a number to specify the size of the text labels.
`font_btm`	Font of the bottom levels: `1` for plain, `2` for bold, `3` for italic, and `4` for bold italic.
`font_top`	Font of the top levels: `1` for plain, `2` for bold, `3` for italic, and `4` for bold italic.
`fam_btm`	Font family of the bottom levels: can be `"serif"`, `"sans"`, `"mono"` or code"symbol".
`fam_top`	Font family of the top levels: can be `"serif"`, `"sans"`, `"mono"` or `"symbol"`.
`col_ell`	Color of the confidence ellipsoids: either a number or see `colors`. Defaults to `"black"`.
`lwd_ell`	Width of the confidence ellipsoids: a number to specify the line width.
`lty_ell`	Line type of the confidence ellipsoids: either `"shade"`, `"wire"`, or `"dots"`. Defaults to `"shade"`.
`opa_ell`	Opaqueness of the confidence ellipsoids: a number between `0` for fully transparent and `1` for fully opaque. Defaults to `0.2`.
`col_ori`	Color of the lines through the origin: either a number or see `colors`. Defaults to `"grey"`.
`lwd_ori`	Width of the lines through the origin: a number to specify the line width. Defaults to `1`.
`main`	The main title of the plot.
`sub`	The subtitle of the plot.
`hlab`	The title of the horizontal axis.
`vlab`	The title of the vertical axis.
`dlab`	The title of the "depth" axis.
`cl`	The confidence level for the confidence ellipsoids. Defaults to `0.95`.
`add_ori`	Logical specifying whether to add lines through the origin. Defaults to `TRUE`.
`...`	Further arguments passed to or from other methods.

Details

`plot3d` (of a `corregp` output) makes use of `plot3d` (and `text3d` and `abclines3d`) from the package **rgl**.

Value

A 3D plot window containing the output of a correspondence regression.

See Also

`corregp`, `pcplot.corregp`, `agplot.corregp`, `plot3d`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
plot3d(haireye.crg, x_ell = TRUE, xsub = c("Hair", "Sex"))
```

`print.anova.corregp` *Printing the ANOVA Table of Correspondence Regression*

Description

Method to print the output of `anova.corregp`.

Usage

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

Arguments

`x`	The output of a call to `anova` on a "corregp" object (i.e. an object of class "anova.corregp").
`...`	Further arguments passed to or from other methods.

Value

The output of a call to `anova` on a "corregp" object.

See Also

`anova.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
anova(haireye.crg, nf = 2)
```

`print.corregp` *Printing Correspondence Regression*

Description

Method to print the output of `corregp`.

Usage

```
## S3 method for class 'corregp'
print(x, nf = 2, ...)
```

Arguments

`x`	The output of a call to `corregp` (i.e. an object of class "corregp").
`nf`	The number of dimensions to print. Defaults to the first two dimensions.
`...`	Further arguments passed to or from other methods.

Value

The output of a call to `corregp`.

See Also

`corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
haireye.crg
print(haireye.crg, nf = 3)
```

`print.summary.corregp` *Printing the Summary of Correspondence Regression*

Description

Method to print the output of `summary.corregp`.

Usage

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

Arguments

`x`	The output of a call to `summary` on a "corregp" object (i.e. an object of class "summary.corregp").
`...`	Further arguments passed to or from other methods.

Value

The output of a call to `summary` on a "corregp" object. The eigenvalues and contributions are printed with `TOTAL`s.

See Also

`summary.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
summary(haireye.crg, add_ci = TRUE)
summary(haireye.crg, parm = "y", contrib = "pts_axs", nf = 2)
```

`residuals.corregp` *Extracting Residuals from Correspondence Regression*

Description

Method to extract the residuals of a correspondence regression.

Usage

```
## S3 method for class 'corregp'
residuals(object, parm = "all", nf = NULL, ...)

## S3 method for class 'corregp'
resid(object, parm = "all", nf = NULL, ...)
```

Arguments

`object`	The output of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to extract the residuals. Can be `"all"`, `"both"` (or abbreviations), `"y"` or `"x"` for the residuals of every cell in the data, but it can also be any vector of term names in X or level names in X. Defaults to `"all"`.
`nf`	The number of dimensions to be retained in the reduced space. Defaults to all dimensions (no reduction).
`...`	Further arguments passed to or from other methods.

Details

If all dimensions are retained, then the residuals will only be exactly zero to the observed counts if no conditioning factors were specified with the argument `"part"` in the `corregp` call. This is because the associations with the conditioning factors (in `"part"`) are not taken into account.

Value

A matrix or vector with the residuals for the parameters of interest, based on the selected number of dimensions.

See Also

`coef.corregp`, `fitted.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
residuals(haireye.crg, parm = c("Hair", "Sex"), nf = 2)
resid(haireye.crg, parm = c("Hair", "Sex"), nf = 2)
```

`screeplot.corregp` *Scree Plotting*

Description

Method to produce a *scree plot*, i.e. a bar chart of the eigenvalues.

Usage

```
## S3 method for class 'corregp'
screeplot(x, type = "value", add_ci = FALSE, cl = 0.95,
  nq = TRUE, ...)
```

Arguments

`x`	The output of a call to `corregp` (i.e. an object of class "corregp").
`type`	A character specification of which type of values to plot: either `"value"` for the *actual eigenvalues*, `"%"` for *percentages* or `"cum_%"` for *cumulative percentages*. Defaults to `"value"`.
`add_ci`	Logical specifying whether to include the confidence intervals. Defaults to `FALSE`.
`cl`	The confidence level for the confidence intervals. Defaults to `0.95`.

`nq`	Logical specifying whether to use a normal quantile (i.e. apply `qnorm`) in the computation of the confidence intervals. Defaults to `TRUE`. If `FALSE`, then the confidence intervals are computed directly with the `quantile` function.
`...`	Further arguments passed to or from other methods.

Details

`screeplot` (of a `corregp` output) makes use of `barplot2` from the package **gplots**.

Value

A plot window containing the scree plot.

See Also

`corregp`, `summary.corregp`, `anova.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
screeplot(haireye.crg, add_ci = TRUE)
```

`summary.corregp` *Summarizing Correspondence Regression*

Description

Method to produce a summary of a correspondence regression.

Usage

```
## S3 method for class 'corregp'
summary(object, parm = NULL, contrib = NULL, nf = NULL,
  add_ci = FALSE, cl = 0.95, nq = TRUE, ...)
```

Arguments

`object`	The outout of a call to `corregp` (i.e. an object of class "corregp").
`parm`	The parameter for which to compute the contributions `contrib`. Can be either `"y"` for the Y contributions, `"x"` for the X contributions, `"both"` which can be abbreviated to `"b"`, or a vector of term names in X. Defaults to `"b"`.
`contrib`	The type of contributions to be computed: either *from points to axes* (absolute contributions) or *from axes to points* (squared correlations). The specification can be `"pnts_to_axes"` or `"axes_to_pnts"`, `"pts2axs"` or `"axs2pts"`, `"p_a"` or `"a_p"`, or any other reasonable abbreviation.

`nf`	The number of dimensions to be retained in the reduced space. Defaults to all dimensions (no reduction).
`add_ci`	Logical specifying whether to compute confidence intervals for the eigenvalues (and eigenvalues only). Defaults to `FALSE`.
`cl`	The confidence level for the confidence intervals. Defaults to `0.95`.
`nq`	Logical specifying whether to use a normal quantile (i.e. apply `qnorm`) in the computation of the confidence intervals. Defaults to `TRUE`. If `FALSE`, then the confidence intervals are computed directly with the `quantile` function.
`...`	Further arguments passed to or from other methods.

Value

An object of class "summary.corregp", providing a summary of a correspondence regression, i.e. a list with components:

`formula`	The `formula` specified to the `formula` argument in the call to `corregp`.
`data`	The name of the data frame specified to the `data` argument in the call to `corregp`.
`part`	The name of the factor specified to the `part` argument in the call to `corregp`.
`chi_squared`	The chi-squared value of the correspondence regression.
`phi_squared`	The phi-squared value of the correspondence regression, i.e. the chi-squared value divided by `N`.
`N`	The total number of observations.
`eigen`	Depending on `add_ci`: if `FALSE`, a matrix of the actual eigenvalues, their percentages and cumulative percentages; if `TRUE`, a list of the actual eigenvalues, their percentages and cumulative percentages together with the lower and upper confidence limits for each.
`y`	If `parm` is `"y"` or `"b"`. A list of components `p_a` for the absolute contributions and//or `a_p` for the squared correlations, depending on `contrib`.
`x`	If `parm` is `"y"`, `"b"` or any of the term names in X. A list of components `p_a` for the absolute contributions and/or `a_p` for the squared correlations, depending in `contrib`.

See Also

`corregp`, `print.summary.corregp`, `anova.corregp`.

Examples

```
data(HairEye)
haireye.crg <- corregp(Eye ~ Hair * Sex, data = HairEye, b = 3000)
summary(haireye.crg, add_ci = TRUE)
summary(haireye.crg, parm = "y", contrib = "pts_axs", nf = 2)
```

`TSS` *The Use of Inflected or Uninflected Determiners in the Belgian Dutch Vernacular*

Description

The distribution of the Belgian Dutch *-e(n)*-suffix with 14 determiners in 14 registers and for several speaker characteristics.

Format

A data frame with 40778 rows and 13 variables.

- `Variant` The linguistic variant used in a set of alternatives (35 levels).
- `Variable` The linguistic variable specifying a set of alternatives (14 levels).
- `Inflected` Numeric variable specifying whether the linguistic variant is inflected (`1`) or not (`0`).
- `Register` The register of the data in the Spoken Dutch Corpus (14 levels: see here for their definition).
- `Register2` The dichotomization of `Register` into private and public.
- `SpeakerID` The ID of the speaker in the Spoken Dutch Corpus (1144 levels).
- `Region` The region in which the speaker lived until the age of 18 (4 levels).
- `Sex` The sex of the speaker (2 levels).
- `BirthYear` The year in which the speaker was born (63 levels).
- `Decade` The decade in which the speaker was born (7 levels).
- `Generation` The generation cohort in which the speaker was born (5 levels).
- `Education` The level of education of the speaker (3 levels).
- `Occupation` The level of occupation of the speaker (10 levels: see here for their definition).

Source

Plevoets, K. (2008) *Tussen spreek- en standaardtaal*. Leuven, Doctoral dissertation. Available online here.

Examples

```
data(TSS)
# The execution of corregp may be slow, due to bootstrapping:
tss.crg <- corregp(Variant ~ Register2 * Region, data = TSS, part = "Variable", b = 3000)
tss.crg
summary(tss.crg, parm = "b", add_ci = TRUE)
screeplot(tss.crg, add_ci = TRUE)
anova(tss.crg, nf = 2)
tss.col <- ifelse( xtabs(~ Variant + Inflected, data = TSS)[, 1] > 0, "blue", "red")
plot(tss.crg, x_ell = TRUE, xsub = c("Register2", "Region"), col_btm = tss.col, col_top = "black")
```

Index