psychotools - Infrastructure for Psychometric Modeling: Version 0.1-1

Zeileis, A; Strobl, Carolin; Wickelmaier, F

Abstract: Infrastructure for psychometric modeling such as data classes (e.g., for paired comparisons) and basic model fitting functions (e.g., for Rasch and Bradley-Terry models). Intended especially as a common building block for fitting psychometric mixture models in package "psychomix" and psychometric tree models in package "psychotree". License: GPL-2

Posted at the Zurich Open Repository and Archive, University of Zurich
ZORA URL: https://doi.org/10.5167/uzh-51108
Supplemental Material

Originally published at:
Package ‘psychotools’

October 5, 2011

Title Infrastructure for Psychometric Modeling

Version 0.1-1

Date 2011-10-05

Authors@R c(person(given = "Achim", family = "Zeileis", role =
  c("aut", "cre"), email = "Achim.Zeileis@R-project.org"),
person(given = "Carolin", family = "Strobl", role = "aut", email = "Carolin.Strobl@psychologie.uzh.ch"),
person(given = "Florian", family = "Wickelmaier", role = "aut", email =
  "Florian.Wickelmaier@uni-tuebingen.de"))

Author Achim Zeileis [aut, cre], Carolin Strobl [aut], Florian Wickelmaier [aut]

Maintainer Achim Zeileis <Achim.Zeileis@R-project.org>

Depends R (>= 2.10.0), stats

Imports graphics, stats

Description Infrastructure for psychometric modeling such as data
  classes (e.g., for paired comparisons) and basic model fitting
  functions (e.g., for Rasch and Bradley-Terry models). Intended
  especially as a common building block for fitting psychometric
  mixture models in package ‘psychomix’ and psychometric tree
  models in package ‘psychotree’.

License GPL-2

Repository CRAN

Date/Publication 2011-10-05 11:07:31

R topics documented:

- btReg.fit ............................................................ 2
- covariates .......................................................... 3
- FirstNames .......................................................... 4
- GermanParties2009 ................................................ 6

1
btReg.fit

Description

btReg.fit is a basic fitting function for simple Bradley-Terry models.

Usage

btReg.fit(y, weights = NULL, type = c("loglin", "logit"), ref = NULL,
undecided = NULL, position = NULL, ...)

Arguments

- **y**: paircomp object with the response.
- **weights**: an optional vector of weights, interpreted as case weights (integer only).
- **type**: character. Should an auxiliary log-linear Poisson model or logistic binomial be employed for estimation? The latter is only available if not undecided effects are estimated.
- **ref**: character or numeric. Which object parameter should be the reference category, i.e., constrained to zero?
- **undecided**: logical. Should an undecided parameter be estimated?
- **position**: logical. Should a position effect be estimated?
- **...**: further arguments passed to functions.

Details

btReg.fit provides a basic fitting function for simple Rasch models, intended as a building block for fitting Rasch trees and Rasch mixtures in the psychotree package, respectively.

btReg.fit returns an object of class "btReg" for which several basic methods are available, including print, plot, summary, coef, vcov, logLik, and worth.
Value

btReg.fit returns an S3 object of class "btReg", i.e., a list with components as follows.

- **coefficients**: estimated parameters on log-scale (without the first parameter which is always constrained to be 0),
- **vcov**: covariance matrix of the parameters in the model,
- **loglik**: log-likelihood of the fitted model,
- **df**: number of estimated parameters,
- **estfun**: empirical estimating function (also known as scores or gradient contributions),
- **weights**: the weights used (if any),
- **n**: number of observations (with non-zero weights),
- **type**: character for model type (see above),
- **ref**: character for reference category (see above),
- **undecided**: logical for estimation of undecided parameter (see above),
- **position**: logical for estimation of position effect (see above),
- **labels**: character labels of the objects compared.

See Also

RaschModel.fit

Examples

```r
## data
data("GermanParties2/zero.noslash/zero.noslash9", package = "psychotools")

## Bradley-Terry model
bt <- btReg.fit(GermanParties2/zero.noslash/zero.noslash9$preference)
summary(bt)
plot(bt)
```

---

### covariates

**Extract/Set Covariates**

**Description**

A generic function for extracting/setting covariates for an object.

**Usage**

```r
covariates(object, ...)
covariates(object) <- value
```
Arguments

- **object**: an object.
- ... arguments passed to methods.
- **value**: an object.

Examples

```r
## method for "paircomp" data
pc <- paircomp(rbind(
    c(1, 1, 1), # a > b, a > c, b > c
    c(1, 1, -1), # a > b, a > c, b < c
    c(1, -1, -1), # a > b, a < c, b < c
    c(1, 1, 1)))
covariates(pc)
covariates(pc) <- data.frame(foo = factor(c(1, 2, 2), labels = c("foo", "bar")))
covariates(pc)
```

---

**FirstNames**

**Popularity of First Names**

Description

Preferences of 192 respondents choosing among six boys names with respect to their popularity.

Usage

```r
data("FirstNames")
```

Format

A data frame containing 192 observations on 11 variables.

- **preference**: Paired comparison of class `paircomp`. All 15 pairwise choices among six boys names: Tim, Lucas, Michael, Robin, Benedikt, and Julius.
- **ordered.prefs**: Ordered paired comparison of class `paircomp`. Same as `preference`, but within-pair order is recognized.
- **gender**: Factor coding gender.
- **age**: Integer. Age of the respondents in years.
- **education**: Ordered factor. Level of education: 1 Hauptschule with degree (Secondary General School), 2 and 3 Realschule without and with degree (Intermediate Secondary School), 4 and 5 Gymnasium without and with degree (High School), 6 and 7 Studium without and with degree (University).
- **children**: Integer. Number of children.
- **state**: Factor. State of Germany where participant grew up.
- **state.reg**: Factor. The region (south, north-west, east) each state belongs to.
FirstNames

fname  Factor. Participant’s first name(s). (Umlaute in Jörg and Jürgen have been transliterated to Joerg and Juergen for portability of the data.)

interviewer  Factor. Interviewer id.

gender.int  Factor coding interviewer’s gender.

Details

A survey was conducted at the Department of Psychology, Universität Tübingen, in June 2009. The sample was stratified by gender and age (younger versus older than 30 years) with 48 participants in each group. The interviewers were Psychology Master’s students who collected the data for course credits.

Participants were presented with 15 pairs of boys names in random order. On each trial, their task was to choose the name they would rather give to their own child. The pairs of boys names were read to the participants one at a time. A given participant compared each pair in one order only, hence the NA’s in ordered.pref.

The names were selected to fall within the upper (Tim, Lucas), mid (Michael, Robin) and lower (Benedikt, Julius) range of the top 100 of the most popular boys names in Germany in the years from 1990 to 1999 (http://www.beliebte-vornamen.de/3778-1990er-jahre.htm). The names have either front (e, i) or back (o, u) vowels in the stressed syllables. Phonology of the name and attractiveness of a person have been shown to be related (Perfors, 2004; Hartung et al., 2009).

References


See Also

car

table

casecontrol

Examples

data("FirstNames", package = "psychotools")
summary(FirstNames$preference)
covariates(FirstNames$preference)
Description
Preferences of 192 respondents choosing among five German political parties and abstention from voting.

Usage
data("GermanParties2009")

Format
A data frame containing 192 observations on 6 variables.

- **preference**: Paired comparison of class `paircomp`. All 15 pairwise choices among five German parties and abstention from voting.
- **ordered.pref**: Ordered paired comparison of class `paircomp`. Same as `preference`, but within-pair order is recognized.
- **gender**: Factor coding gender.
- **age**: Integer. Age of the respondents in years.
- **education**: Ordered factor. Level of education: 1 no degree, 2 Hauptschule (Secondary General School), 3 Realschule (Intermediate Secondary School), 4 Gymnasium (High School), 5 Studium (University)
- **crisis**: Factor. Do you feel affected by the economic crisis?
- **interviewer**: Factor. Interviewer id.

Details
A survey was conducted at the Department of Psychology, Universität Tübingen, in June 2009, three months before the German election. The sample was stratified by gender and age (younger versus older than 30 years) with 48 participants in each group.

The parties to be compared were Die Linke (socialists), Die Grünen (ecologists), SPD (social democrats), CDU/CSU (conservatives), and FDP (liberals). In addition, there was the option of abstaining from voting (coded as `none`).

Participants were presented with 15 pairs of options in random order. On each trial, their task was to choose the party they would rather vote for at an election for the German parliament. A given participant compared each pair in one order only, hence the NA's in `ordered.pref`.

In order to minimize response biases, the pairs of options were read to the participants one at a time. Participants made their choices by crossing either “First Option” or “Second Option” on an anonymous response sheet.

The interviewers were Psychology Master’s students who collected the data for course credits. Since they mainly interviewed people they knew, the results are not representative of the political opinions
in Germany. As far as the winner of the survey (Die Grünen) is concerned, however, the results agree with the outcome of the election for the Tübingen voters.

The results of the election on September 27, 2009 (number of so-called Zweitstimmen in percent) were:

<table>
<thead>
<tr>
<th></th>
<th>Germany</th>
<th>Tübingen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Die Linke</td>
<td>11.9</td>
<td>8.5</td>
</tr>
<tr>
<td>Die Grünen</td>
<td>10.7</td>
<td>27.9</td>
</tr>
<tr>
<td>SPD</td>
<td>23.0</td>
<td>21.1</td>
</tr>
<tr>
<td>CDU/CSU</td>
<td>33.8</td>
<td>23.0</td>
</tr>
<tr>
<td>FDP</td>
<td>14.6</td>
<td>13.9</td>
</tr>
<tr>
<td>Others</td>
<td>6.0</td>
<td>5.7</td>
</tr>
</tbody>
</table>

The voter turnout was 70.8 percent in Germany and 80.5 percent in Tübingen.

See Also

paircomp

Examples

```r
data("GermanParties2/zero.noslash/zero.noslash9", package = "psychotools")
summary(GermanParties2/zero.noslash/zero.noslash9$preference)
```

labels<-  

Set Labels

Description

A generic function for setting labels for an object.

Usage

```r
labels(object) <- value
```

Arguments

- **object**: an object.
- **value**: an object.
Examples

```r
## method for "paircomp" data
pc <- paircomp(rbind(
   c(1, 1, 1), # a > b, a > c, b > c
   c(1, 1, -1), # a > b, a > c, b < c
   c(1, -1, -1), # a > b, a < c, b < c
   c(1, 1, 1)))
labels(pc)
labels(pc) <- c("ah", "be", "ce")
pc
```

---

**mscale**  
*Extract/Replace Measurement Scale*

### Description

Generic functions for extracting and replacing the measurement scale from an object.

### Usage

```r
mscale(object, ...)
mscale(object) <- value
```

### Arguments

- `object`  
an object.
- `...`  
arguments passed to methods.
- `value`  
an object describing the measurement scale.

### Examples

```r
## methods for "paircomp" data
pc <- paircomp(rbind(
   c(2, 1, 0),
   c(1, 1, -1),
   c(1, -2, -1),
   c(0, 0, 0)))
pca

## extract
mscale(pc)

## replace (collapse to >/=/< scale)
mscale(pc) <- sign(mscale(pc))
pca
```
Description

A class for representing data from paired comparison experiments along with methods for many generic functions.

Usage

```r
paircomp(data,
  labels = NULL, mscale = NULL, ordered = FALSE, covariates = NULL)
```

Arguments

data: matrix. A matrix with integer values where the rows correspond to subjects and the columns to paired comparisons between objects. See below for details.

labels: character. A vector of character labels for the objects. By default a suitable number of letters is used.

mscale: integer. A vector of integers giving the measurement scale. See below for details. By default guessed from data.

ordered: logical. Does data contain both orderings of each comparison?

covariates: data.frame. An optional data.frame with object covariates, i.e., it must have the same number of rows as the length of labels. May be NULL (default).

Details

paircomp is designed for holding paired comparisons of \( k \) objects measured for \( n \) subjects. The comparisons should be coded in an integer matrix data with \( n \) rows (subjects) and \( \binom{k}{2} \) columns (unless ordered = TRUE, see below). The columns must be ordered so that objects are sequentially compared with all previous objects, i.e.: 1:2, 1:3, 2:3, 1:4, 2:4, 3:4, etc. Each column represents the results of a comparison for two particular objects. Positive values signal that the first object was preferred, negative values that the second was preferred, zero signals no preference. Larger absolute values signal stronger preference.

mscale provides the underlying measurement scale. It must be a symmetric sequence of integers of type \((-i):i\) where \( i \) must be at least 1. However, it may exclude 0 (i.e., forced choice).

If ordered = TRUE, the order of comparison matters and thus data is assumed to have twice as many columns. The second half of columns then corresponds to the comparisons 2:1, 3:1, 3:2, 4:1, 4:2, 4:3, etc.

Value

paircomp returns an object of class "paircomp" which is a matrix (essentially data) with all remaining arguments of paircomp as attributes (after being checked and potentially suitably coerced or transformed).
See Also

subset.paircomp, print.paircomp

Examples

## a simple paired comparison
pc <- paircomp(rbind(
  c(1, 1, 1), # a > b, a > c, b > c
  c(1, 1, -1), # a > b, a > c, b < c
  c(1, -1, -1), # a > b, a < c, b < c
  c(1, 1, 1))))

## basic methods
pc
str(pc)
summary(pc)
pc[2:3]
c(pc[2], pc[c(1, 4)])

## methods to extract/set attributes
labels(pc)
labels(pc) <- c("ah", "be", "ce")
pc
mscale(pc)
covariates(pc)
covariates(pc) <- data.frame(foo = factor(c(1, 2, 2), labels = c("foo", "bar")))
covariates(pc)
names(pc)
names(pc) <- LETTERS[1:4]

## reorder() and subset() both select a subset of
## objects and/or reorders the objects
reorder(pc, c("ce", "ah"))

## include paircomp object in a data.frame
## (i.e., with subject covariates)
dat <- data.frame(
  x = rnorm(4),
  y = factor(c(1, 2, 1, 1), labels = c("hansi", "beppi")))
dat$pc <- pc
dat

## formatting with long(er) labels and extended scale
pc2 <- paircomp(rbind(
  c(4, 1, 0),
  c(1, 2, -1),
  c(1, -2, -1),
  c(0, 0, -3)),
labels = c("Nordrhein-Westfalen", "Schleswig-Holstein", "Baden-Wuerttemberg"))
## default: abbreviate
print(pc2)
print(pc2, abbreviate = FALSE)
print(pc2, abbreviate = FALSE, width = FALSE)

## paired comparisons with object covariates
pc3 <- paircomp(rbind(
  c(2, 1, 0),
  c(1, 1, -1),
  c(1, -2, -1),
  c(0, 0, 0)),
labels = c("New York", "Rio", "Tokyo"),
covariates = data.frame(hemisphere = factor(c(1, 2, 1), labels = c("North", "South"))))
covariates(pc3)

---

### Description

Base graphics plotting function for Rasch models.

### Usage

```r
## S3 method for class 'btReg'
plot(x, worth = TRUE, index = TRUE, names = TRUE,
     ref = TRUE, abbreviate = FALSE, type = NULL, lty = NULL,
     xlab = "Objects", ylab = NULL, ...)
```

### Arguments

- `x` an object of class "btReg".
- `worth` logical. Should worth parameters (or alternatively coefficients on log-scale) be displayed?
- `index` logical. Should different indexes for different items be used?
- `names` logical. Should the names for the objects be displayed?
- `ref` logical. Should a horizontal line for the reference level be drawn?
- `abbreviate` logical or numeric. Should object names be abbreviated? If numeric this controls the length of the abbreviation.
- `type` plot type. Default is "b" if index is TRUE.
- `lty` line type.
- `xlab`, `ylab` x and y axis labels.
- `...` further arguments passed to `plot`. 
plot.paircomp

### Description

Plotting the frequency table from "paircomp" data.

### Usage

```r
## S3 method for class 'paircomp'
plot(x, off = 0.05,
     xlab = "Proportion of comparisons", ylab = "", tol.xlab = 0.05,
     abbreviate = TRUE, hue = NULL, chroma = 40, luminance = 80,
     xlim = c(0, 1), ylim = NULL, xaxs = "i", yaxs = "i", ...)
```

### Arguments

- **x**: an object of class "paircomp".
- **off**: numeric. Offset between segments on the y-axis.
- **xlab, ylab**: character. Axis labels.
- **tol.xlab**: numeric. convenience tolerance parameter for x-axis annotation. If the distance between two labels drops under this threshold, they are plotted equidistantly.
- **abbreviate**: logical or integer. Should object labels be abbreviated? Alternative an integer with the desired abbreviation length. The default is some heuristic based on the length of the labels.
- **hue**: numeric. A vector of hues in [0, 360], recycled to the number of objects compared in x. A sequential palette is computed for each hue, see below.
- **chroma**: numeric. Maximum chroma in the palette.
- **luminance**: numeric. Minimum (and maximum) luminance in the palette. If omitted, the maximum is set to 95.
- **xlim, ylim, xaxs, yaxs, ...**: graphical arguments passed to `plot`.

### See Also

- `btReg.fit`
The `plot.RaschModel` method creates a frequency table (using `summary`) and visualizes this using a sort of spine plot with HCL-based diverging palettes. See Zeileis, Hornik, Murrell (2009) for the underlying ideas.


See Also

paircomp

**Examples**

data("GermanParties2009", package = "psychotools")
par(mar = c(5, 6, 3, 6))
plot(GermanParties2009$preference, abbreviate = FALSE)

---

**plot.RaschModel**  
**Visualizing Simple Rasch Models**

**Description**

Base graphics plotting function for Rasch models.

**Usage**

```r
## S3 method for class 'RaschModel'
plot(x, difficulty = TRUE, center = TRUE, index = TRUE, names = NULL, abbreviate = FALSE, ref = TRUE, col = cbind("lightgray", "black"), refcol = "lightgray", linecol = "black", lty = 2, cex = 1, pch = cbind(19, 1), type = NULL, ylim = NULL, xlab = "Items", ylab = NULL, ...)
```

**Arguments**

- `x`: an object of class "RaschModel".
- `difficulty`: logical. Should item difficulty (or alternatively: easiness) parameters be displayed?
- `center`: logical. Should the item parameters be centered?
- `index`: logical. Should different indexes for different items be used?
- `names`: logical. Should the names for the objects be displayed?
abbreviate logical or numeric. Should object names be abbreviated? If numeric this controls the length of the abbreviation.

ref logical. Should a horizontal line for the reference level be drawn?

col, pch, cex graphical appearance of plotting symbols. Can be of the same length as number of items. Additionally col and pch can be matrices with two columns resulting in two symbols being overplotted.

refcol line color for reference line (if ref).

linecol line color.

lty line type.

type plot type. Default is "b" if index is TRUE.

ylim y axis limits.

xlab, ylab x and y axis labels.

... further arguments passed to plot.

See Also

RaschModel.fit

print.paircomp Formatting Paired Comparison Data

Description

Fine control for formatting and printing objects of "paircomp" data.

Usage

## S3 method for class 'paircomp'
format(x, sep =", ", brackets = TRUE, abbreviate = NULL, width = getOption("width") - 7, ...)

## S3 method for class 'paircomp'
print(x, quote = FALSE, ...)

Arguments

x an object of class "paircomp".

sep character. A character for separating comparisons within subjects.

brackets logical or character. Either a logical (Should brackets be wrapped around all comparisons for a single subject?) or a character of length two with opening and ending symbol.

abbreviate logical or integer. Should object labels be abbreviated? Alternative an integer with the desired abbreviation length. The default is some heuristic based on the length of the labels.
width integer or logical. Maximal width of the string for a subject. If FALSE no maximal width is set.

... arguments passed to other functions.

quote logical. Should quotes be printed?

Details

The print method just calls format (passing on all further arguments) and then prints the resulting string.

See Also

paircomp

Examples

pc2 <- paircomp(rbind(
  c(4, 1, 0),
  c(1, 2, -1),
  c(1, -2, -1),
  c(0, 0, -3)),
labels = c("New York", "Rio", "Tokyo"))

print(pc2)
print(pc2, abbreviate = FALSE)
print(pc2, abbreviate = FALSE, width = 10)
RaschModel.fit

hessian logical. Should the Hessian of the final model be computed? If set to FALSE, the vcov method can only return NAs and consequently no standard errors or tests are available in the summary.

graddtol, ... further arguments passed to nlm.

Details

RaschModel.fit provides a basic fitting function for simple Rasch models, intended as a building block for fitting Rasch trees and Rasch mixtures in the psychotree package, respectively.

RaschModel.fit returns an object of class "RaschModel" for which several basic methods are available, including print, plot, summary, coef, vcov, logLik, and worth.

Value

RaschModel.fit returns an S3 object of class "RaschModel", i.e., a list with components as follows.

coefficients estimated item difficulty parameters (without first item parameter which is always constrained to be 0),
vcov covariance matrix of the parameters in the model,
loglik log-likelihood of the fitted model,
df number of estimated parameters,
data the original data supplied (excluding columns without variance),
weights the weights used (if any),
n number of observations (with non-zero weights),
items status indicator (0, 0/1, 1) of all original items,
na logical indicating whether the data contains NAs,
elementary_symmetric_functions List of elementary symmetric functions for estimated parameters (up to order 2; or 1 in case of numeric derivatives),
nlm_code convergence code from nlm,
iterations number of iterations used by nlm,
graddtol tolerance passed to nlm.

See Also

btReg.fit

Examples

## Verbal aggression data
data("VerbalAggression", package = "psychotools")

## Rasch model for the self-to-blame situations
m <- RaschModel.fit(VerbalAggression$resp2[, 1:12])
summary(m)
plot(m)
Description

Paired comparison judgments of 40 selected listeners with respect to eight audio reproduction modes and for types of music.

Usage

data("SoundQuality")

Format

A data frame containing 783 observations on 6 variables.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Factor. Listener ID.</td>
</tr>
<tr>
<td>time</td>
<td>Factor. Listening experiment before or after elicitation and scaling of more specific auditory attributes.</td>
</tr>
<tr>
<td>progmat</td>
<td>Factor. The Program material: Beethoven, Rachmaninov, Steely Dan, Sting.</td>
</tr>
<tr>
<td>repet</td>
<td>The repetition within each time point.</td>
</tr>
<tr>
<td>session</td>
<td>The experimental session coding the presentation order of the program material.</td>
</tr>
<tr>
<td>preference</td>
<td>Paired comparison of class paircomp. Preferences for all 28 paired comparisons from 8 audio reproduction modes: Mono, Phantom Mono, Stereo, Wide-Angle Stereo, 4-channel Matrix, 5-channel Upmix 1, 5-channel Upmix 2, and 5-channel Original.</td>
</tr>
</tbody>
</table>

Details

The data were collected within a series of experiments conducted at the Sound Quality Research Unit (SQRU), Department of Acoustics, Aalborg University, Denmark, between September 2004 and March 2005.

The results of scaling listener preference and spatial and timbral auditory attributes are reported in Choisel and Wickelmaier (2007).

Details about the loudspeaker setup and calibration are given in Choisel and Wickelmaier (2006).

The attribute elicitation procedure is described in Wickelmaier and Ellermeier (2007) and in Choisel and Wickelmaier (2006).

The selection of listeners for the experiments is described in Wickelmaier and Choisel (2005).

References


See Also

paircomp

Examples

data("SoundQuality", package = "psychotools")
summary(SoundQuality$preference)
ftable(xtabs(~ time + repet + progrmat, data = SoundQuality))

subset.paircomp     Subsetting/Reordering Paired Comparison Data

Description

Selection of subsets of objects to be compared and/or reordering of objects in "paircomp" data.

Usage

## S3 method for class 'paircomp'
reorder(x, labels, ...)
## S3 method for class 'paircomp'
subset(x, subset, select, ...)

Arguments

x an object of class "paircomp".
labels, select character or integer. Either a vector of (at least two) elements of labels(x) or an integer with their position. Partial string matching is enabled.
subset currently not implemented. (Should be a specification of subsets of subjects.)
... currently not used.

Details

The subset method currently just calls the reorder method.
See Also

paircomp

Examples

```r
c <- paircomp(rbind(
  c(1, 1, 1), # a > b, a > c, b > c
  c(1, 1, -1), # a > b, a > c, b < c
  c(1, -1, -1), # a > b, a < c, b < c
  c(1, 1, 1)))
reorder(pc, c("c", "a")))
```

Description

Responses of 316 subjects to 24 items describing possible reactions to 4 different frustrating situations.

Usage

```r
data("VerbalAggression")
```

Format

A data frame containing 316 observations on 4 variables.

- **resp** Item response matrix with values 0/1/2 coding no/perhaps/yes, respectively.
- **resp2** Dichotomized item response matrix with perhaps/yes merged to 1.
- **gender** Factor coding gender.
- **anger** Trait anger, assessed by the Dutch adaptation of the state-trait anger scale (STAS).

Details

The 24 items are constructed by factorial combination of four different frustrating situations (see below), three possible verbally aggressive responses (curse, scold, shout), and two behavioural models (want, do). The four situations are

- **S1:** A bus fails to stop for me.
- **S2:** I miss a train because a clerk gave me faulty information.
- **S3:** The grocery store closes just as I am about to enter.
- **S4:** The operator disconnects me when I used up my last 10 cents for a call.

Note that the first two situations are other-to-blame situations, and the latter two are self-to-blame situations.
The subjects were 316 first-year psychology students from a university in the Dutch speaking part of Belgium. Participation was a partial fulfillment of the requirement to participate in research. The sample consists of 73 males and 243 females, reflecting the gender proportion among psychology students. The average age was 18.4.

Source


http://bear.soe.berkeley.edu/EIRM/

References


See Also

RaschModel.fit

Examples

data("VerbalAggression", package = "psychotools")

## Rasch model for the self-to-blame situations
m <- RaschModel.fit(VerbalAggression$resp2[, 1:12])
summary(m)
plot(m)

worth

Extract Worth Parameters

Description

Generic functions for extracting worth parameters from paired comparison models.

Usage

worth(object, ...)

Arguments

object an object.
... arguments passed to methods.
worth

See Also

btReg.fit, RaschModel.fit

Examples

## data
data("GermanParties2/zero/noslash/zero.noslash9", package = "psychotools")

## Bradley-Terry model
bt <- btReg.fit(GermanParties2/zero/noslash/zero.noslash9$preference)

## worth parameters
worth(bt)
Index

*Topic **classes**
  - covariates, 3
  - labels<-, 7
  - mscale, 8
  - paircomp, 9
  - plot.paircomp, 12
  - print.paircomp, 14
  - subset.paircomp, 18
  - worth, 20

*Topic **datasets**
  - FirstNames, 4
  - GermanParties2/zero.noslash/zero.noslash9, 6
  - SoundQuality, 17
  - VerbalAggression, 19

*Topic **hplot**
  - plot.btReg, 11
  - plot.RaschModel, 13

*Topic **regression**
  - btReg.fit, 2
  - RaschModel.fit, 15
  - [.paircomp (paircomp), 9

  as.character.paircomp (paircomp), 9
  as.data.frame.paircomp (paircomp), 9
  as.double.paircomp (paircomp), 9
  as.integer.paircomp (paircomp), 9
  as.matrix, 15
  as.matrix.paircomp (paircomp), 9

  btReg.fit, 2, 12, 16, 21
  c.paircomp (paircomp), 9
  coef.btReg (btReg.fit), 2
  coef.RaschModel (RaschModel.fit), 15
  covariates, 3
  covariates.paircomp (paircomp), 9
  covariates<-(covariates), 3
  covariates<-.paircomp (paircomp), 9

  deviance.btReg (btReg.fit), 2

  FirstNames, 4
  format.paircomp (print.paircomp), 14

  GermanParties2009, 6
  is.na.paircomp (paircomp), 9
  labels.paircomp (paircomp), 9
  labels<-, 7
  labels<-.paircomp (paircomp), 9
  length.paircomp (paircomp), 9
  logLik.btReg (btReg.fit), 2
  logLik.RaschModel (RaschModel.fit), 15

  mscale, 8
  mscale.paircomp (paircomp), 9
  mscale<-(mscale), 8
  mscale<-.paircomp (paircomp), 9

  names.paircomp (paircomp), 9
  names<-.paircomp (paircomp), 9
  nlm, 16

  paircomp, 4–7, 9, 13, 15, 17–19
  plot, 11, 12, 14
  plot.btReg, 11
  plot.paircomp, 12
  plot.RaschModel, 13
  print.btReg (btReg.fit), 2
  print.paircomp, 10, 14
  print.RaschModel (RaschModel.fit), 15
  print.summary.btReg (btReg.fit), 2
  print.summary.RaschModel (RaschModel.fit), 15

  RaschModel.fit, 3, 14, 15, 20, 21
  reorder.paircomp (subset.paircomp), 18
  rep.paircomp (paircomp), 9

  SoundQuality, 17
  str.paircomp (paircomp), 9
subset.paircomp, 10, 18
summary.btReg(btReg.fit), 2
summary.paircomp(paircomp), 9
summary.RaschModel(RaschModel.fit), 15
vcov.btReg(btReg.fit), 2
vcov.RaschModel(RaschModel.fit), 15
VerbalAggression, 19
worth, 2, 16, 20
worth.btReg(btReg.fit), 2
worth.RaschModel(RaschModel.fit), 15
xtfrm.paircomp(paircomp), 9