Package 'RDigram'

July 30, 2020

Type Package

Title Analysis of Items from Tests and Surveys

Version 0.1.0

Date 2019-12-06

Author Jeppe Bundsgaard & Svend Kreiner

Maintainer Jeppe Bundsgaard <jebu@edu.au.dk>

Description Functions for analysis categorical data from achievement tests and surveys

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Roxygen list(markdown = TRUE,roclets = c(``rd'', ``collate'',``namespace'',``vignette''))

VignetteBuilder knitr

Imports car,
MESS,
DescTools,
stringr,
stringi,
ggraph,
ggplot2,
knitr,
kableExtra,
iarm,
TAM,
eRm

Depends tidygraph,
ggplot2

R topics documented:

`RDigram-package` *Functions for item analysis*

Description

RDigram provides tools for analysis of items from achievement tests and surveys. RDigram is an implementation of a subset of the functions in the DIGRAM statistical program by Svend Kreiner.

`ADLtired` *Physical Activities of Daily Living*

Description

Measure of functional ability of healthy elderly (Avlund et.al., 1993)

Usage

```
data(ADLtired)
```

Format

A data frame with 734 rows and 97 variables of which 19 are used. The data was collected from 734 70-year old in the County of Copenhagen, Denmark. The PADL scale consisted of a total of 16 items covering three different domains.

Mobility function

A: Are you able to walk indoors?

B: Are you able to walk out of doors in nice weather?

C: Are you able to walk out of doors in nice weather?

D: Are you able to manage stairs?

E: Are you able to get outdoors?

F: Are you able to get up from a chair or bed? Lower limb function

G: Are you able to wash thelower part of the body?

H: Are you able to cut your toenails?

I: Are you able to go to the toilet yourself?

J: Are you able to dress the lower part of the body?

K: Are you able to take shoes/stockings on/off? Upper limb function

L: Are you able to wash the upper part of the body?

M: Are you able to cut your fingernails?

N: Are you able to comb your hair?

O: Are you able to wash your hair?

P: Are You able to dress the upper part of the body?

Value

ADLtired is an object of class digram.object.

Source

The `ADLtired` data were obtained from Avlund, K., Schultz-Larsen, K., Kreiner, S. (1993) Construct validation and the Rasch model: Functional ability of healthy elderly people. *Scand. J. Soc. Med.*, 21: 233-245.

`all.patterns` *Create all possible response patterns*

Description

Create all possible response patterns

Usage

```
## S3 method for class 'patterns'
all(maxscores = c(), target = NULL)
```

Arguments

`maxscores`	vector of max scores
`target`	target total score

Details

This function quickly gets very resource intensive. On a 8 core, 16 GB computer, 25 items are too heavy.

Value

Returns a matrix of all possible patterns. If target is set, only patterns with at total score of target is returned

Author(s)

Jeppe Bundsgaard & Svend Kreiner

References

Jeppe Bundsgaard & Svend Kreiner (2019). *Undersøgelse af De Nationale Tests måleegenskaber*. 2nd Ed. Copenhagen: DPU, Aarhus University.

Examples

```
maxscores<-c(1,2,3,1,2,3)
all.patterns(maxscores,target=5)
```

as_tbl_graph.digram.object *Create a tidygraph object from a digram.object*

Description

Create a tidygraph object from a digram.object

Usage

```
as_tbl_graph(do)
```

Arguments

do	A digram.object.
LD	A data.frame with columns for item1, item2 and gamma coefficient. Items can be variable.names, variable.columns or item numbers.
DIF	A data.frame with columns for item, exogenous variable and gamma coefficient. Items and exogenous variables can be variable.names, variable.columns or item numbers.

Value

Returns a tbl_graph

See Also

tidygraph::as_tbl_graph()

Examples

```
library(ggraph)
library(tidygraph)
dograph<-as_tbl_graph(DHP)
ggraph(dograph,layout="fr")+geom_edge_link()+geom_node_label(mapping = aes(label=label))
ggraph(dograph,layout="fr")+geom_edge_link()+geom_node_label(mapping = aes(label=name))

# Show arrows
ggraph(dograph,layout="fr")+geom_edge_link(end_cap = square(.5, 'cm'),arrow = arrow(angle=10,length=unit(.2,"cm"

# A digram.object with a testlet
dograph<-as_tbl_graph(code.testlet(DHP,"a b c"))
ggraph(dograph,layout="fr")+geom_edge_link(end_cap = square(.5, 'cm'),arrow = arrow(angle=10,length=unit(.2,"cm"

# Local dependecy and DIF
dograph<-as_tbl_graph(DHP,LD=data.frame(item1=c(5,3),item2=c(6,5),gamma=c(.53,-.38)),DIF=data.frame(item=c(4),
ggraph(dograph,layout="fr")+geom_edge_link(mapping=aes(label=ifelse(!is.na(gamma),abs(gamma),""),alpha=ifelse(
```

CHIPS *CHIPS: Children's Problem Solving*

Description

914 responses to CHIPS: Children's Problem Solving for 6-12 year old children.

Usage

```
data(CHIPS)
```

Format

A data frame with 914 rows and 58 variables of which 42 are used.

CHIPS (Children's Problem Solving; Hansen, Kreiner, & Hansen, 1992) is an instrument for measuring cognitive function meant for children from 6 to 12 years of age. The theory of cognitive development behind CHIPS describes cognitive function in terms of the person's ability to draw on qualitatively different types of strategies for solving abstract problems. According to the theory, the cognitive function of children develops over three stages, called global (G), analytical/ synthetic (A/S), and comprehensive (C). At the global stage, the child registers likenesses more than anything else. At the analytic/synthetic stage, the child is able to cope with both likenesses and differences and to synthesize them to wholes. At these two stages in the cognitive development, the child is not yet ready to deal with abstract mental images but has to see or handle the physical objects. At the comprehensive stage, the child is able to use abstract principles and rules when it is required for problem solving. CHIPS provides some possibility for evaluating the level of cognitive function in quantitative terms. The main purpose of CHIPS, however, is to classify pupils according to the three stages of cognitive development.

CHIPS consists of three sets of items: \item 11 G items requiring global cognition, \item 14 A/S items requiring analytic/synthetic cognition, and \item 15 C items requiring comprehensive cognition.

Value

CHIPS is an object of class digram.object.

Source

Kreiner, S., Hansen, M. and Hansen, C. R. (2006). On Local Homogeneity and Stochastically Ordered Mixed Rasch Models. In: *Applied Psychological Measurement* 30; 271. DOI: 10.1177/0146621605287909 Hansen, M., Kreiner, S., & Hansen, C. R. (1992). CHIPS—Children's Problem Solving: Manual. Copenhagen: Dansk psykologisk forlag.

code.split *Code items to be split (as having DIF)*

Description

Code items to be split (as having DIF)

Usage

```
code.split(do, split.var, split.on, append = F)
```

Arguments

do	A digram.object
split.var	String. The variables to split (having DIF). A comma separated list of variable numbers, labels or names.
split.on	String. The exogenous variables to split on (causing DIF). A comma separated list of exogenous variable numbers, labels or names.
append	Logical.

Details

If more variables and exogenous variables are given, all possible combinations of these are split.

Examples

```
data(DHP)
do<-code.split(DHP,"a,b","under60")
```

code.testlet *Code items as a testlet/local dependant*

Description

Code items as a testlet/local dependant

Usage

```
code.testlet(do, testlet = NULL, names = NULL, labels = NULL, append = F)
```

Arguments

do	A digram.object
testlet	String. The items that are part of a testlet/are local dependant. Give as a list of comma separated variable numbers, variable labels or variable names. If there is spaces in the variable names, they can be delimited by ".
names	A vector of strings naming the testlets. If names are not given, they are composed of the testlet item names.
append	Logical. Append new testlet variables to the existing ones.

Details

Local dependence is often caused by items sharing a common stimulus. This is called testlets or item bundles (Wang & Wilson 2006. Coding for Local Dependence is the same as identifying a testlet or an item bundle.

Value

Returns a digram.object with the revised testlet-data.frame.

References

Wang, W.-C., & Wilson, M. (2005). The Rasch Testlet Model. *Applied Psychological Measurement*, 29(2), 126–149. https://doi.org/10.1177/0146621604271053

Examples

```
data(DHP)
do<-code.testlet(do=DHP,testlet=c("ab,dhp36 dhp37,5 6"))
```

DHP *Diabetes Health profile (DHP)*

Description

The DHP is a multidimensional patient self-completion diabetes-specific inventory designed to identify psychosocial dysfunction among adult insulin dependent and insulin requiring patients. Factor analyses have suggested that responses to DHP items depend on three latent variables representing Psychological distress, Barriers to Activity and Disinhibited eating. Chwalow et.al (2007) describe a randomized study of the quality of life of type 2 diabetic patients.

Usage

```
data(DHP)
```

Format

A data frame with 185 rows and 8 variables.

The Disinhibited eating (DE) subscale summarizing responses to the following five questions with four ordinal response categories that were coded in such a way that 0 represents no dysfunction and 3 represents a high degree of dysfunction:

A: DHP32 Do you wish there were not so many things to eat? Responses: a) "Not at all", b) "A little", c) "A lot", d) "Very much"

B: DHP34 How likely are you to eat something extra when you feel bored or fed up? Responses: a) "Not at all likely", b) "Not very likely", c) "Quite likely", d) "Very likely"

C: DHP36 When you start eating, how easy do you find it to stop? Responses: a) "Very easy", b) "Quite easy", c) "Not very easy", d) "Not at all easy"

D: DHP38 Do you have problems keeping to you diet because you eat to cheer yourself up? Responses: a) "Never", b) "Sometimes", c) "Usually", d) "Always"

E: DHP39 Do you have problems keeping to your diet because you find it hard saying no to food you like? Responses: a) "Never", b) "Sometimes", c) "Usually", d) "Always" In addition to the items, the DHP project also includes information on sex and age.

Value

DHP is an object of class digram.object.

Source

The `DHP` data were obtained from Chwalow J., Meadows K., Mesbah M., Coliche V., Mollett E., (2007) Empirical validation of a quality of life instrument: empirical internal validation and analysi of a quality of life instrument in French diabetic patients during an educational intervention. In C. Huber, N. Limnios, M. Mesbah, N. Nikulin (eds). *Mathematical Methods in Survival Analysis, Reliability and Quality of Life.* London: Hernes.

`digram.estimate` *Estimate RDigram object using TAM*

Description

Estimate RDigram object using TAM

Usage

```
digram.estimate(
  do,
  items = NULL,
  groups = NULL,
  ncases = 0,
  constraint = "cases",
```

```
  use.package = c("TAM", "eRm"),
  collapse.testlets = F,
  init.model = NULL,
  tam.control = list(),
  sum0 = T,
  verbose = T,
  ...
)
```

Arguments

do	A digram.object
items	The items to include in the analysis
groups	Names or column numbers of exogenous variables to use for grouping in TAM. If more names are given, all combinations of values are calculated and used as grouping variables.
ncases	Number of cases to sample for the estimation (0 uses all cases)
constraint	Constraint on "cases" or "items"
use.package	Which R package to use for the estimation. TAM and eRm are implemented.
collapse.testlets	Testlets are estimated using a bifactorial model in TAM and a data matrix in eRm. Setting collapse.testlets to TRUE calculates super-items instead and estimate a normal polytomous model.
init.model	In TAM, the model that was output from an earlier estimation can be used to set sensible init-values for the estimation.
tam.control	Use this to set control parameters in TAM estimation.
sum0	Set to TRUE if you want eRm to sum the parameters to 0. If FALSE the first parameter is set to 0.
verbose	Set to TRUE to get information about the estimation progress.

Details

Uses either the package TAM or eRm to estimate the model. If items have been coded as testlets, a bifactorial model is used in TAM (`tam.fa()`). Otherwise `tam.mml()` is used for estimation. In eRm, testlets are managed by creating an interaction parameter between the testlet items. In this case `LPCM()` is used for estimation. This is also the case, if groups are provided. Otherwise `PCM()` is used for estimation.

Value

Returns a TAM result object

References

Wang, W.-C., & Wilson, M. (2005). The Rasch Testlet Model. *Applied Psychological Measurement*, 29(2), 126–149. https://doi.org/10.1177/0146621604271053 Rijmen, F. (2009). *Three multidimensional models for testlet-based tests: Formal relations and an empirical comparison.* ETS Research Report Series, 2009(2), i–13. https://doi.org/10.1002/j.2333-8504.2009.tb02194.x

See Also

tam.mml(), tam.fa(), PCM(), LPCM()

Examples

```
data(DHP)
do<-DHP
mod1<-digram.estimate(do)
summary(mod1)
do2<-code.LD(do,"ef")
mod2<-digram.estimate(do2)
summary(mod2)
mod1$deviance
mod2$deviance
mod1$deviance-mod2$deviance
```

---

digram.object *Create DIGRAM Object*

Description

Create a digram.object.

Usage

```
digram.object(project=NULL,data=data.frame(),variables=colnames(data),filter.conditions=data.frame(
```

Arguments

project	The name of the DIGRAM project
data	The data.frame or matrix with the data
variables	A vector of column names or numbers from the dataset to include in the recoded dataset (the order matters in the recoded data) *or* a list of variables each element in the form list(variable.name="",variable.label="",ncat=0,category.names=c
filter.conditions	A data.frame with three columns: variable.number, max, and min. Only cases, for which all values of filter variables belong to the intervals defined by the corresponding minimum and maximum values (both included), will be used in the analysis.

`recursive.structure` A vector of cutpoints to define the recursive blocks

`comments` A string

Value

Returns a digram.object

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. (2003). *Introduction to DIGRAM*. Dept. of Biostatistics, University of Copenhagen.

Examples

```
library(iarm)
do<-digram.object(project = "desc2",data = desc2,variables = c(5:14,2:4,1),recursive.structure = c(10,13))
```

`digram.wrightmap` *Draw a Wrightmap*

Description

Draw a Wrightmap

Usage

```
digram.wrightmap(
  mod,
  do = NULL,
  dimname = NULL,
  cols = collist,
  colscheme = "(.*)_.*",
  verbose = T
)
```

Arguments

`mod`	Output from tam.mml or tam.fa.
`do`	A digram.object. Used to create title. Alternative to dimname
`dimname`	A string. Used to create title. Alternative to do.
`cols`	A vector of colors.
`colscheme`	A regular expression as string. Used to give items of the same type, same colors.
`verbose`	Boolean. Set to TRUE to get information about progress.

Value

Returns a list of item difficulties as thresholds and person abillities as thetas

Examples

```
mod<-digram.estimate(DHP)
digram.wrightmap(mod=mod,do=DHP)
```

---

draw.plausible.response

Draw plausible responses based on item parameters

Description

Draw plausible responses based on item parameters

Usage

```
draw.plausible.response(
  item.params,
  param.type = c("pcm", "log.item.score", "multiplicative", "xsi"),
  R = 0,
  num.responses = 1
)
```

Arguments

`item.params`	a matrix of item parameters (items in rows, thresholds in columns)
`param.type`	Type of item parameters. One of pcm (RUMM2030), log.item.score (?), multiplicative (DIGRAM or RDigram, xsi (Conquest or TAM))
`R`	Total score of the response
`num.responses`	The number of response patterns to return

Value

Returns a matrix of plausible response patterns

Examples

```
item.params<-matrix(c(1,.5,1,1,1,2,1,4),nrow=4)
draw.plausible.response(item.params=item.params,param.type="multiplicative",R=2,num.responses=10)
```

gamma.matrix *Gammas of item.parameters given total score using matrix calculations*

Description

Gammas of item.parameters given total score using matrix calculations

Usage

```
## S3 method for class 'matrix'
gamma(
  item.params = NULL,
  param.type = c("pcm", "log.item.score", "multiplicative", "xsi"),
  R = NULL
)
```

Arguments

item.params	a matrix of item parameters (using the PCM parametrisation). Items in rows, threshold values in columns
param.type	Type of item parameters given. One of pcm (RUMM2030), log.item.score (?), multiplicative (DIGRAM or RDigram, xsi (Conquest or TAM))
R	The total score for which to calculate the gamma parameter (set to NULL to get gammas for all possible total scores)

Details

Always use gamma.matrix instead of gamma.pattern. It is far more efficient.

Value

Returns gamma for the given total score.

Author(s)

Jeppe Bundsgaard & Svend Kreiner

References

Jeppe Bundsgaard & Svend Kreiner (2019). *Undersøgelse af De Nationale Tests måleegenskaber*. 2nd Ed. Copenhagen: DPU, Aarhus University.

Examples

```
item.params<-matrix(c(1,.5,1,1,1,2,1,4),nrow=4)
gamma.matrix(item.params,"multiplicative",3)
```

gamma.pattern *Gammas of item.parameters given total score using patterns - inefficient*

Description

Gammas of item.parameters given total score using patterns - inefficient

Usage

```
## S3 method for class 'pattern'
gamma(
  item.params,
  param.type = c("pcm", "log.item.score", "multiplicative", "xsi"),
  R = 0
)
```

Arguments

item.params	a matrix of item parameters (using the PCM parametrisation). Items in rows, threshold values in columns
param.type	Type of item parameters given. One of pcm (RUMM2030), log.item.score (?), multiplicative (DIGRAM or RDigram, xsi (Conquest or TAM))
R	The total score for which to calculate the gamma parameter

Details

Always use gamma.matrix instead of gamma.pattern. It is far more efficient.

Value

Returns gamma for the given total score.

Author(s)

Jeppe Bundsgaard & Svend Kreiner

References

Jeppe Bundsgaard & Svend Kreiner (2019). *Undersøgelse af De Nationale Tests måleegenskaber*. 2nd Ed. Copenhagen: DPU, Aarhus University.

Examples

```
item.params<-matrix(c(1,.5,1,1,1,2,1,4),nrow=4)
gamma.pattern(item.params,"multiplicative",3)
```

`get.column.no` *Get column number(s) of a variable in the recoded data*

Description

Get column number(s) of a variable in the recoded data

Usage

```
get.column.no(do, variable.name.num)
```

Arguments

`do`	A DIGRAM object
`variable.name.num`	The number, label or name of the variable(s) to search for

Value

Returns the column.number(s)

Examples

```
data(DHP)
get.column.no(DHP,"dhp36")
```

`get.item.params` *Get item parameters and standard errors as a data.frame from a TAM or eRm object*

Description

Get item parameters and standard errors as a data.frame from a TAM or eRm object

Usage

```
get.item.params(obj, type = c("andersen", "conquest"), include.se = F)
```

Arguments

`obj`	A TAM or eRm-object
`type`	For TAM-objects, return Andersen or Conquest ("IRT") parameters.

Details

Andersen parameters are given as easiness parameters as is the convention. Difficulty can be calculated as the negated version of the parameters.

Value

Returns a matrix of item parameters and standard errors, one row for each item

See Also

item.params.convert()

Examples

```
data(DHP)
tamobj<-digram.estimate(DHP)
get.item.params(tamobj)
```

---

get.labels *Get variable labels*

Description

Get variable labels

Usage

```
get.labels(do, items = NULL)
```

Arguments

do	A digram.object
items	The variable.numbers of the items or exogeneous variables to provide the labels for

Value

Returns the variable.labels of the items/exogeneous variables .

Examples

```
get.labels(DHP,1:2)
```

get.variable.names *Get variable names*

Description

Get variable names

Usage

```
get.variable.names(do, items = NULL)
```

Arguments

do	A digram.object
items	The variable.numbers of the items or exogeneous variables to provide the names for

Value

Returns the variable.names of the items/exogeneous variables .

Examples

```
get.variable.names(DHP,1:2)
```

header.format *Not exported*

Description

Not exported

Usage

```
header.format(t = "", margin = 0)
```

`item.correlations` *Item correlations*

Description

Calculate item correlations, item-rest correlations and correlations between items and exogenous variables

Usage

```
item.correlations(do=NULL,resp=NULL,items=1:do$recursive.structure[1],exo=(do$recursive.structure[1
```

Arguments

`do`	an object of class `digram.object`
`resp`	A data.frame or matrix of recoded data (only used if `do` is `NULL`)
`items`	A vector of columns from the recoded data to include as items in the analysis *or* a character vector of variable labels
`exo`	A vector of columns from the recoded data to include as exogenous variables in the analysis *or* a character vector of variable labels
`max.name.length`	Maximum length of item names (to be printed in tables)
`accept.na`	A boolean. Include cases with missing values in responses
`verbose`	Print results

Details

First step in item screening: Analysis of consistency (Positive correlations)

M1 Y_i and Y_j are positively correlated for all pairs of items

M2 Y_a is positively monotonically related to the rest-score R_a and all subscores S_B where $Y_a \notin$ B

M3 If X is positively related to θ, then X will also be positively related to S, to all subscores, S_A, and all item responses Y_i

Value

Returns a list of correlations

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. & Christensen, K.B. (2011). Item Screening in Graphical Loglinear Rasch Models. *Psychometrika*, vol. 76, no. 2, pp. 228-256. DOI: 10.1007/s11336-9203-Y

Examples

```
library(iarm)
do<-digram.object(project = "desc2",data = desc2,variables = c(5:14,2:4,1),recursive.structure = c(10,13))
item.correlations(do)
```

`item.DIF` *Detect Differential Item Functioning*

Description

Detect Differential Item Functioning (DIF)

Usage

```
item.DIF(do=NULL,resp=NULL,items=1:do$recursive.structure[1],exo=(do$recursive.structure[1]+1):do$r
```

Arguments

`do`	an object of class `digram.object`
`resp`	A data.frame or matrix of recoded data (only used if `do` is `NULL`)
`items`	A vector of columns from the recoded data to include as items in the analysis *or* a character vector of variable labels
`exo`	A vector of columns from the recoded data to include as exogenous variables in the analysis *or* a character vector of variable labels
`p.adj`	the kind of multiple p-value testing adjustment to be used (one of "BH","holm", "hochberg", "hommel", "bonferroni", "BY", "none").
`max.name.length`	Maximum length of item names (to be printed in tables)
`digits`	Number of digits in table
`only.significant`	Only list fit values significantly different from 1
`verbose`	Print results
`saved.result`	To avoid repeated calculation, you can provide a saved version of the analysis (returned from item.DIF())

Details

Second step in item screening: Analysis of DIF and local dependency

C2 $Y_i \perp X_j \mid S$ for all $i = 1 \ldots k$ and $j = 1 \ldots m$

C4 $Y_a \perp Y_b \mid R_a$ and $Y_a \perp Y_b \mid R_b$ Conditional independence of A and B given C is denoted as $A \perp B \mid C$.

Use `local.independence()` to detect local dependency

If you want to use this function in R Markdown or Bookdown, you need to use xelatex as latex engine, and you need to force dev to use cairo_pdf or png. Add this in you yaml header:

```
output:
pdf_document:
latex_engine: xelatex
```

Add this in your setup chunck:

```
knitr::opts_chunk$set(echo = TRUE,dev = "cairo_pdf",dpi = 300)
```

Value

Returns a list of DIF-information

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. & Christensen, K.B. (2011). Item Screening in Graphical Loglinear Rasch Models. *Psychometrika*, vol. 76, no. 2, pp. 228-256. DOI: 10.1007/s11336-9203-Y

See Also

`partgam_LD()`, `local.independence()`

Examples

```
item.DIF(DHP)
```

---

item.params.convert *Convert item parameters from one parametrization to another*

Description

Convert item parameters from one parametrization to another

Usage

```
item.params.convert(
  from.model = NULL,
  item.params = c(),
  from = c("pcm", "pcm.cent", "andersen", "psd", "conquest"),
  to = c("pcm", "pcm.cent", "andersen", "psd", "conquest"),
  return.vector = F
)
```

Arguments

`from.model`	A model of class TAM or eRm.
`item.params`	A matrix of item parameters (items in rows, thresholds in columns) (not needed if from.model is given)
`from, to`	Type of item parameters. One of pcm, pcm.cent, andersen, psd, conquest (from not needed if from.model is given).
`return.vector`	Get result as a vector of category values instead of the default data.frame

Details

The Rasch model can be parameterized in multiple ways. This function translates parameters from one to the other. The following models are supported:

- Power series distribution parametrization (psd)
  $\frac{\xi^x \gamma_x}{G(\xi,\gamma_1 \ldots \gamma_k)}$, where ξ is the person parameter (ability), and γ_x are the item parameters. Related through $\xi = exp(\theta)$ and $\gamma_x = exp(\delta_x)$ to the Andersen parametrization
- Andersen parametrization (andersen)
  $\frac{exp(x\theta+\delta_x)}{G(\theta,\delta_1 \ldots \delta_k)}$, where θ is the person parameter (ability), and δ_x are the item (easiness) parameters
- Partial Credit Model (PCM)/Masters' parametrization (pcm)
  $\frac{exp(x\theta - \sum_{i=1}^{x} \tau_i)}{G(\theta,\tau_1 \ldots \tau_k)}$, where θ is the person parameter (ability), and τ_x are the item step parameters
- Partial Credit Model (PCM)/Masters' parametrization with centralized item step parameters (pcm.cent)
  $\frac{exp(x(\theta-\beta) - \sum_{i=1}^{x} \beta_i)}{G(\theta,\beta_1 \ldots \beta_k)}$, where θ is the person parameter (ability), $\beta = \frac{1}{k} \sum_{i=1}^{k} \tau_i$ is the avarage of the τ parameters from the PCM parametrization, and $\beta_x = \tau_x - \frac{1}{k} \sum_{i=1}^{k} \tau_i$ are the centralized item step parameters
- Conquest parametrization (conquest)
  $\frac{exp(x(\theta-\psi - \sum x\psi_x)}{G(\theta,\psi_1 \ldots \psi_k)}$, where $\sum_l imitsi = 1^x \psi_i \equiv 0$. And where θ is the person parameter (ability), $\psi = \frac{1}{k} \sum_{i=1}^{k} \tau_i$ is the avarage of the τ parameters from the PCM parametrization, and $\psi_x = \tau_x - \frac{1}{k} \sum_{i=1}^{k} \tau_i$ are the centralized item step parameters

TAM uses Conquest parametrization. eRm uses Andersen parametrization. RUMM 2030 uses Partial Credit parametrization with centralized item step parameters. DIGRAM uses Power Series Distribution parametrization.

Value

Returns item parameters in the parametrization specified in *to*.

Author(s)

Jeppe Bundsgaard & Svend Kreiner

References

Andersen, B. E. (1970). Asymptotic properties of conditional likelihood estimators. *Journal of the Royal Statistical Society*, Series B, 32,283-301.

Brown, N. J. S. (2004). Interpreting Ordered Partition Model Parameters from ConQuest. https://bearcenter.berkeley.edu/sites/default/files/report%20-%20opm_parameters.pdf

Hatzinger, R., & Rusch, T. (2009). IRT models with relaxed assumptions in eRm: A manual-like instruction. *Psychology Science Quarterly*, 51(1), 87–120.

Kreiner, S. (n.d.). Om beregning af item-parametre I TAM.

Kreiner, s. (n.d.). Parameterization of graphical loglinear Rasch models.

Bundsgaard, J. & Kreiner, S. (2019). *Undersøgelse af De Nationale Tests måleegenskaber*. 2nd Ed. Copenhagen: DPU, Aarhus University.

Masters, G. N. (1982). A Rasch model for partial credit scoring. *Psychometrika*, 47(2), 149–174.

Examples

```
item.params<-matrix(c(0,1,2,3,1,2,3,4),nrow=4)
item.params.convert(item.params=item.params,from="conquest",to="psd")
```

---

local.independence *Detect local dependence*

Description

Investigate items for local independence

Usage

```
local.independence(do=NULL,resp=NULL,items=1:do$recursive.structure[1],digits=2,verbose=T)
```

Arguments

do	an object of class digram.object
resp	A data.frame or matrix of recoded data (only used if do is NULL)
items	A vector of columns from the recoded data to include as items in the analysis *or* a character vector of variable labels
p.adj	the kind of multiple p-value testing adjustment to be used (one of "holm", "BH","hochberg", "hommel", "bonferroni", "BY", "none"), see p.adjust().
digits	Number of digits in table

`max.name.length`
: Maximum length of item names (to be printed in tables)

`only.significant`
: Only list fit values significantly different from 1

`use.names`
: Use item names instead of item labes as node labels

`verbose`
: Print results

`saved.result`
: To avoid repeated calculation, you can provide a saved version of the analysis (returned from local.independence())

Details

Second step in item screening: Analysis of DIF and local dependence

C4 $Y_a \perp Y_b \mid R_a$ and $Y_a \perp Y_b \mid R_b$ Conditional independence of A and B given C is denoted as $A \perp B \mid C$.

Use `item.DIF()` for detection of Differential Item Functioning

If you want to use this function in R Markdown or Bookdown, you need to use xelatex as latex engine, and you need to force dev to use cairo_pdf or png. Add this in you yaml header:

```
output:
pdf_document:
latex_engine: xelatex
```

Add this in your setup chunck:

```
knitr::opts_chunk$set(echo = TRUE,dev = "cairo_pdf",dpi = 300)
```

Value

Returns a list of local dependencies

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. & Christensen, K.B. (2011). Item Screening in Graphical Loglinear Rasch Models. *Psychometrika*, vol. 76, no. 2, pp. 228-256. DOI: 10.1007/s11336-9203-Y

See Also

`partgam_LD()`, `item.DIF()`

Examples

```
local.independence(DHP)
```

`person.fit.pattern` *Person fit based on response pattern*

Description

Person fit based on response pattern

Usage

```
person.fit.pattern(
  do = NULL,
  resp = NULL,
  items = NULL,
  item.params = matrix(),
  param.type = c("pcm", "log.item.score", "multiplicative", "xsi"),
  num.montecarlo = 0,
  verbose = T
)
```

Arguments

`do`	a DIGRAM object
`resp`	a matix of responses (if no DIGRAM object is supplied)
`items`	a vector of items to use
`item.params`	a matrix of item parameters. Items in rows, thresholds in columns
`param.type`	Type of item parameters given. One of pcm (RUMM2030), log.item.score (?), multiplicative (DIGRAM or RDigram, xsi (Conquest or TAM))
`num.montecarlo`	the number of iterations if the calculation of alternative patterns should be done using a montecarlo solution. 0 to do all patterns.
`verbose`	set to TRUE if you want to follow the progression

Value

Returns a list of results for each respondent, consisting of response pattern probability and the p-value of getting this pattern or a pattern of lower probability.

Author(s)

Jeppe Bundsgaard & Svend Kreiner

References

Jeppe Bundsgaard & Svend Kreiner (2019). *Undersøgelse af De Nationale Tests måleegenskaber.* 2nd Ed. Copenhagen: DPU, Aarhus University.

Examples

```
data(DHP)
item.params<-matrix(c(1.000,2.283,1.150,0.509,
                      1.000,1.117,2.630,6.082,
                      1.000,1.380,4.105,2.058,
                      1.000,0.276,0.127,0.070,
                      1.000,2.141,0.330,0.472,
                      1.000,10.304,2.963,4.784),byrow=T,nrow=6)
person.fit.pattern(do=DHP, item.params=item.params,param.type="multiplicative")
```

---

pf3 *Physical functioning subscale*

Description

The dataset originated in a Danish Health survey. This is the SF36 subscale measuring physical functioning.

Usage

```
data(pf3)
```

Format

A data frame with 1049 rows and 25 variables of which 12 are used. The scale summarizes responses to the following ten items:

Does your health now limit you in these activities? If so, how much?

A PF1: Vigorous activities

B PF2: Moderate activities

C PF3: Lifting or carrying groceries

D PF4: Climbing several flights of stairs

E PF5: Climbing one flight of stairs

F PF6: Bending, kneeling, or stooping

G PF7: Walking more than a mile

H PF8: Walking several blocks

I PF9: Walking one block

J PF10: Bathing or dressing yourself The responses to these questions were coded in the following way:

0 Limited a lot

1 Limited a little

2 Not limited so that a low score indicates physical impairment. Gender and Age are also included in this project.

Value

pf3 is an object of class digram.object.

Source

The `pf3` data were obtained from Kreiner, S. (2003). *Introduction to DIGRAM*. Dept. of Biostatistics, University of Copenhagen.

`print.digram.object` *Print DIGRAM object*

Description

Prints information about a DIGRAM object

Usage

```
## S3 method for class 'digram.object'
print(do = NULL)
```

Arguments

`do`	a DIGRAM object

Details

Prints

Value

Returns nothing

Note

Any notes about the operation of the function

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

References concerning the methodology employed by function

Examples

```
print(DHP)
```

read.digram *Read DIGRAM files into digram.object*

Description

Reads data from DIGRAM files and create a digram.object.

Usage

```
read.digram(project=NULL,path="")
```

Arguments

project	The name of the DIGRAM project from which to load data
path	Path to the project files

Value

A digram.object

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. (2003). *Introduction to DIGRAM*. Dept. of Biostatistics, University of Copenhagen.

Examples

```
do<-read.digram("DHP",path = "DHP")
```

score.information *Print score information*

Description

Prints information about the recoded data.

Usage

```
score.information(do=NULL,resp=NULL,items=1:length(do$variables),accept.na = F)
```

Arguments

do	An object of class digram.object
resp	A data.frame or matrix of recoded data (only used if do is NULL)
items	A vector of columns from the recoded data to include in the analysis *or* a character vector of variable labels
accept.na	A boolean. Include cases with missing values in responses

Value

Returns NULL. Prints Average item scores and score distribution and Score groups for tests of Rasch models

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. (2003). *Introduction to DIGRAM*. Dept. of Biostatistics, University of Copenhagen.

Examples

```
score.information(do,items="abcdef")
```

---

summary.digram.object *Summarize DIGRAM object*

Description

Summarizes information about a DIGRAM object

Usage

```
## S3 method for class 'digram.object'
summary(do = NULL)
```

Arguments

do	a DIGRAM object

Value

Returns nothing

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. (2003). *Introduction to DIGRAM*. Dept. of Biostatistics, University of Copenhagen.

Examples

```
summary(DHP)
```

`TIMSSTaiAus` *TIMSS data for Taiwan and Austrailia*

Description

TIMSS 2011 mathematics tests, released items, blocks 1 and 14, for Taiwan and Australian students. From TAM tutorial https://www.edmeasurementsurveys.com/TAM/Tutorials/5PartialCredit.htm. Coded according to the tutorial.

Usage

```
data(TIMSSTaiAus)
```

Format

A data frame with 1773 rows and 14 variables. 11 response variables, 3 context variables. The data was collected in IEA's TIMSS 2011 mathematics tests, released items, blocks 1 and 14, for Taiwan and Australian students.

The booklet can be downloaded here: https://www.edmeasurementsurveys.com/TAM/Tutorials/data/TIMSS

Value

TIMSSTaiAus is an object of class digram.object.

Source

https://timss.bc.edu/timss2011/index.html

`variable.delete` *Delete variable*

Description

Delete one or more variable(s) in a DIGRAM Object.

Usage

```
variable.delete(do = NULL, variable.to.delete = NULL)
```

Arguments

`do`	A digram.object
`variable.to.delete`	The number or name of the variable(s) which should be deleted

Details

A set of variables can be deleted by providing a vector of variable names/numbers.

Value

Returns a DIGRAM object with (a) deleted variable(s)

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. (2003). *Introduction to DIGRAM*. Dept. of Biostatistics, University of Copenhagen.

Examples

```
data(DHP)
DHP<-variable.delete(do=DHP,variable.to.delete="dhp36")
```

`variable.update` *Update variable*

Description

Update one or more variable(s) in a DIGRAM Object.

Usage

```
variable.update(
  do = NULL,
  variable.to.update = NULL,
  variable.name = NULL,
  variable.label = NULL,
  category.names = NULL,
  variable.type = NULL,
  minimum = NULL,
  maximum = NULL,
  cutpoints = NULL
)
```

Arguments

`do`	A digram.object
`variable.to.update`	The number or name of the variable(s) which should be updated
`variable.name`	New name for the variable (string)
`variable.label`	New label for the variable (one or more uppercase letters)
`variable.type`	Type of variable ("ordinal" or "nominal")
`minimum`	Smallest value of the variable (lower values are coded NA)
`maximum`	Largest value of the variable (lower values are coded NA)
`cutpoints`	Cutpoints used to recode the variable Category 1: minimum ¡= values ¡= cutpoint(1) Category 2: cutpoint(1) ¡ values ¡= cutpoint(2) ... Category N: cutpoint(n) ¡ values ¡= maximum

Details

A set of variables can be updated by providing a vector of variable names/numbers to update and ordered lists whith their new properties. For variable.type, cutpoints, minimum and maximum you can provide one property to give all variables.

Value

Returns a DIGRAM object with (an) updated variable(s)

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. (2003). *Introduction to DIGRAM*. Dept. of Biostatistics, University of Copenhagen.

Examples

```
data(DHP)
DHP<-variable.update(do=DHP,variable.to.update="dhp36",cutpoints=c(2,3))
DHP<-variable.update(do=DHP,variable.to.update=c(1,3),variable.label=list("32","36"))
```

---

variables.combine *Combine variables*

Description

Combines one or more variable(s) in a DIGRAM Object.

Usage

```
variables.combine(
  do = NULL,
  variables.to.combine = NULL,
  variable.name = NULL,
  variable.label = NULL,
  category.names = NULL,
  combine.type = c("sum", "or", "xor", "and", "onlyif"),
  minimum = NULL,
  maximum = NULL,
  cutpoints = NULL
)
```

Arguments

do	A digram.object
variables.to.combine	The numbers or names of the two or more variables to combine
variable.name	Name of the variable (string)
variable.label	Label of the variable (one or more uppercase letters)

combine.type	How to combine the variables. Values are: "sum": add values of ingoing variables together "or": if one of the variables has a value other than 0, use it (in case of polytomous values, use the highest) "xor": if ONLY one of the variables has a value other than 0, use it "and": if ALL variables has the same value, use it "onlyif" if the first variable has a value, use it. If the second has a value, add it to the first. If the third has a value ... etc.
minimum	Smallest value of the variable (lower values are coded NA)
maximum	Largest value of the variable (lower values are coded NA)
cutpoints	Cutpoints used to recode the variable Category 1: minimum ¡= values ¡= cutpoint(1) Category 2: cutpoint(1) ¡ values ¡= cutpoint(2) ... Category N: cutpoint(n) ¡ values ¡= maximum

Details

Only ordinal variables can be combined. The resulting variable can be manipulated and deleted as ordinary variables using variable.update and variable.delete.

Value

Returns a DIGRAM object with the new combined variable.

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

Examples

```
data(DHP)
DHP<-variables.combine(do=DHP,variables.to.combine=c("dhp36","dhp34"))
```

---

write.digram *Write DIGRAM Object*

Description

Writes a RDigram object to DIGRAM files.

Usage

```
write.digram(do = NULL, path = "", filename = do$project)
```

Arguments

do	The digram.object
path	Path to where to save the files
filename	Optional. Set to project name if not set.

Details

DIGRAM is a Windows based program. Therefore it expects CRLF (\r\n) newlines. If you edit the files in Linux after export, your might need to take care to keep the CRLF newlines.

DIGRAM doesn't accept labels with more than one character. Therefore two-character labels are converted starting from AA-¿a. Labels from BA are special characters, starting from BA-¿ü. DIGRAM might not like RDigram's choice of characters.

DIGRAM doesn't accept more than 58 categories. Therefore variables with more than 58 categories are re-coded to include only 58 categories.

Value

Returns nothing

Author(s)

Jeppe Bundsgaard jebu@edu.au.dk

References

Kreiner, S. (2003). *Introduction to DIGRAM*. Dept. of Biostatistics, University of Copenhagen.

Examples

```
write.digram(do = DHP,path = "DHP")
```