Package 'cdcatR'

May 12, 2020

Type Package

Title Cognitive Diagnostic Computerized Adaptive Testing in R

Version 1.0.0

Date 2020-04-29

Description
A set of functions for conducting cognitive diagnostic computerized adaptive testing applications.

License GPL-3

LazyData TRUE

Depends R (>= 3.5.0)

Imports GDINA (>= 2.2.0), ggplot2 (>= 3.3.0), cowplot, doParallel (>= 1.0.15), foreach, CDM (>= 7.4.19), NPCD, stats, parallel, snow, doSNOW

URL https://github.com/miguel-sorrel/cdcatR

BugReports https://github.com/miguel-sorrel/cdcatR/issues

RoxygenNote 7.1.0

Encoding UTF-8

Author Miguel A. Sorrel [aut, cre, cph],
Pablo Nájera [aut, cph],
Francisco J. Abad [aut, cph]

Maintainer Miguel A. Sorrel <miguel.sorrel@uam.es>

R topics documented:

att.plot *Plots for attribute mastery estimates*

Description

This function generates a plot monitoring the attribute mastery estimates (*x-axis*: Item position, *y-axis*: Mastery posterior probability estimate). If a parametric CD-CAT has been conducted, posterior probabilites (with confident intervals) of mastering each attribute are plotted. If a nonparametric CD-CAT has been conducted (and pseudo-probabilites have been computed), both nonparametric classification and pseudo-posterior probabilites (with confident intervals) of mastering each attribute are plotted. Pseudo-posterior probabilities is a method in progress. Caution in the interpretation is advised. Colors are used in the plots to indicate mastery (green), non-mastery (red), or uncertainty (blue).

Usage

```
att.plot(cdcat.obj, i, k = NULL)
```

Arguments

cdcat.obj	An object of class cdcat
i	Scalar numeric. It specifies the examinee to be plotted
k	Numeric vector. It specifies the attribute/s to be plotted. Default is NULL, which plots all attributes

Value

att.plot returns a plot of class ggplot.

cdcat *Cognitively based computerized adaptive test application*

Description

cdcat conducts a CD-CAT application for a given dataset. Different item selection rules can be used: the general discrimination index (GDI; de la Torre & Chiu, 2016; Kaplan et al., 2015), the Jensen-Shannon divergence index (JSD; Kang et al., 2017; Minchen & de la Torre, 2016; Yigit et al., 2018), the posterior-weighted Kullback-Leibler index (PWKL; Cheng, 2009), the modified PWKL index (MPWKL; Kaplan et al., 2015), the nonparametric item selection method (NPS; Chang et al., 2019), or random selection. Fixed length or fixed precision CD-CAT can be applied. Fixed precision CD-CAT with NPS is available, by using the pseudo-posterior probability of each student mastering each attribute (experimental).

Usage

```
cdcat(
  fit = NULL,
  dat = NULL,
  itemSelect = "GDI",
  MAXJ = 20,
  FIXED.LENGTH = TRUE,
  att.prior = NULL,
  initial.distr = NULL,
  precision.cut = 0.8,
 NPS.args = list(Q = NULL, gate = NULL, pseudo.prob = T, w.type = 1, seed = NULL),
  n.cores = 2,
  print.progress = TRUE
)
```

Arguments

fit	An object of class GDINA or gdina. Calibrated item bank with the GDINA::GDINA (Ma & de la Torre, 2020) or CDM::gdina (Robitzsch et al., 2020) R packages functions
dat	Numeric matrix of dimensions N number of examinees x J number of items. Dataset to be analyzed. If is.null(dat) the data is taken data from the fit object (i.e., the calibration sample is used)
itemSelect	Scalar character. Item selection rule: GDI, JSD, MPWKL, PWKL, NPS, or random
MAXJ	Scalar numeric. Maximum number of items to be applied regardless of the FIXED.LENGTH argument. Default is 20
FIXED.LENGTH	Scalar logical. Fixed CAT-length (TRUE) or fixed-precision (FALSE) application. Default is TRUE
att.prior	Numeric vector of length 2^K, where K is the number of attributes. Prior distribution for MAP/EAP estimates. Default is uniform
initial.distr	Numeric vector of length 2^K, where K is the number of attributes. Weighting distribution to initialize itemSelect at item position 1. Default is uniform
precision.cut	Scalar numeric. Cutoff for fixed-precision (assigned pattern posterior probability > precision.cut; Hsu, Wang, & Chen, 2013). When itemSelect = "NPS" this is evaluated at the attribute level using the pseudo-posterior probabilities for each attribute (K assigned attribute pseudo-posterior probability > precision.cut). Default is .80. A higher cutoff is recommended when itemSelect = "NPS"
NPS.args	A list of options when itemSelect = "NPS". Q = Q-matrix to be used in the analysis. gate = "AND" or "OR", depending on whether a conjunctive o disjunctive nonparametric CDM is used. pseudo.prob = pseudo-posterior probability of each examinee mastering each attribute (experimental). w.type = weight type used for computing the pseudo-posterior probability (experimental): 1 = Power-of-2 weight; 2 = Exponential weight. seed = Numeric vector of length 1. NPS has a random component, so a seed is required for consistent results.
n.cores	Scalar numeric. Number of cores to be used during parallelization. Default is 2
print.progress	Scalar logical. Prints a progress bar to the console. Default is TRUE

Value

`cdcat` returns an object of class `cdcat`.

est A list that contains for each examinee the mastery posterior probability estimates at each step of the CAT (`est.cat`) and the items applied (`item.usage`)

specifications A list that contains all the specifications

References

Chang, Y.-P., Chiu, C.-Y., & Tsai, R.-C. (2019). Nonparametric CAT for CD in educational settings with small samples. *Applied Psychological Measurement, 43*, 543-561.

Cheng, Y. (2009). When cognitive diagnosis meets computerized adaptive testing: CD-CAT. *Psychometrika, 74*, 619-632.

de la Torre, J., & Chiu, C. Y. (2016). General method of empirical Q-matrix validation. *Psychometrika, 81*, 253-273.

George, A. C., Robitzsch, A., Kiefer, T., Gross, J., & Uenlue, A. (2016). The R Package CDM for cognitive diagnosis models. *Journal of Statistical Software, 74*, 1-24. doi:10.18637/jss.v074.i02

Hsu, C. L., Wang, W. C., & Chen, S. Y. (2013). Variable-length computerized adaptive testing based on cognitive diagnosis models. *Applied Psychological Measurement, 37*, 563-582.

Kang, H.-A., Zhang, S., & Chang, H.-H. (2017). Dual-objective item selection criteria in cognitive diagnostic computerized adaptive testing. *Journal of Educational Measurement, 54*, 165-183.

Kaplan, M., de la Torre, J., & Barrada, J. R. (2015). New item selection methods for cognitive diagnosis computerized adaptive testing. *Applied Psychological Measurement, 39*, 167-188.

Ma, W. & de la Torre, J. (2020). GDINA: The generalized DINA model framework. R package version 2.7.9. Retrived from https://CRAN.R-project.org/package=GDINA

Minchen, N., & de la Torre, J. (2016, July). *The continuous G-DINA model and the Jensen-Shannon divergence*. Paper presented at the International Meeting of the Psychometric Society, Asheville, NC, United States.

Robitzsch, A., Kiefer, T., George, A. C., & Uenlue, A. (2020). CDM: Cognitive Diagnosis Modeling. R package version 7.5-15. https://CRAN.R-project.org/package=CDM

Yigit, H. D., Sorrel, M. A., de la Torre, J. (2018). Computerized adaptive testing for cognitively based multiple-choice data. *Applied Psychological Measurement, 43*, 388-401.

Examples

```
#######################################
# Example 1.                          #
# CD-CAT simulation for a GDINA obj  #
#######################################

#-----------Data----------#
Q <- sim180GDINA$simQ
K <- ncol(Q)
dat <- sim180GDINA$simdat
att <- sim180GDINA$simalpha

#----------Model estimation----------#
fit <- GDINA::GDINA(dat = dat, Q = Q, verbose = 0) # GDINA package
#fit <- CDM::gdina(data = dat, q.matrix = Q, progress = 0) # CDM package
```

```
#---------------CD-CAT---------------#
res.FIXJ <- cdcat(fit = fit, dat = dat, FIXED.LENGTH = TRUE,
                  MAXJ = 20, n.cores = 2)
res.VARJ <- cdcat(fit = fit, dat = dat, FIXED.LENGTH = FALSE,
                  MAXJ = 20, precision.cut = .80, n.cores = 2)

#---------------Results--------------#
res.FIXJ$est[[1]] # estimates for the first examinee (fixed-length)
res.VARJ$est[[1]] # estimates for the first examinee (fixed-precision)
att.plot(cdcat.obj = res.FIXJ, i = 1) # plot for the first examinee (fixed-length)
att.plot(cdcat.obj = res.VARJ, i = 1) # plot  for the first examinee (fixed-precision)
# FIXJ summary
res.FIXJ.sum.real <- cdcat.summary(cdcat.obj = res.FIXJ, alpha = att) # vs. real accuracy
res.FIXJ.sum.real$recovery$plotPCV
res.FIXJ.sum.real$recovery$plotPCA
res.FIXJ.sum.real$item.exposure$plot
# VARJ summary
res.VARJ.sum.real <- cdcat.summary(cdcat.obj = res.VARJ, alpha = att)
res.VARJ.sum.real$recovery
res.VARJ.sum.real$CATlength$stats
res.VARJ.sum.real$CATlength$plot
res.VARJ.sum.real$item.exposure$plot
# vs. maximum observable accuracy
att.J <- GDINA::personparm(fit, "MAP")[, -(K+1)] # GDINA package
# att.J <- t(sapply(strsplit(as.character(fit$pattern$map.est), ""), as.numeric)) # CDM package
class.J <- GDINA::ClassRate(att, att.J) # upper-limit for accuracy
res.FIXJ.sum.obse <- cdcat.summary(cdcat.obj = res.FIXJ, alpha = att.J)
res.FIXJ.sum.obse$recovery$plotPCV + ggplot2::geom_hline(yintercept = class.J$PCV[K],
                                                         color = "firebrick3")
res.FIXJ.sum.obse$recovery$plotPCA + ggplot2::geom_hline(yintercept = class.J$PCA,
                                                         color = "firebrick3")

#######################################
# Example 2.                          #
# CD-CAT simulation for multiple      #
# GDINA objs and comparison of        #
# performance on a validation sample  #
#######################################

#---------------Data---------------#
Q <- sim180combination$simQ
K <- ncol(Q)
parm <- sim180combination$specifications$item.bank$simcatprob.parm
dat.c <- sim180combination$simdat[,,1]
att.c <- sim180combination$simalpha[,,1]
dat.v <- sim180combination$simdat[,,2]
att.v <- sim180combination$simalpha[,,2]

#-----(multiple) Model estimation----#
fitTRUE <- GDINA::GDINA(dat = dat.c, Q = Q, catprob.parm = parm,
           control = list(maxitr = 0), verbose = 0)

fitGDINA <- GDINA::GDINA(dat = dat.c, Q = Q, verbose = 0)
fitDINA <- GDINA::GDINA(dat = dat.c, Q = Q, model = "DINA", verbose = 0)
LR2step <- LR.2step(fitGDINA)
models <- LR2step$models.adj.pvalues
fitLR2 <- GDINA::GDINA(dat = dat.c, Q = Q, model = models, verbose = 0)
```

```
#---------------CD-CAT---------------#
fit.l <- list(fitTRUE, fitLR2, fitGDINA, fitDINA)
res.FIXJ.l <- lapply(fit.l, function(x)  cdcat(dat = dat.v,fit = x,
                                               FIXED.LENGTH = TRUE, n.cores = 2))
res.VARJ.l <- lapply(fit.l, function(x)  cdcat(dat = dat.v,fit = x,
                                               FIXED.LENGTH = FALSE, n.cores = 2))

#---------------Results--------------#
fitbest <- GDINA::GDINA(dat = dat.v, Q = Q, catprob.parm = parm,
          control = list(maxitr = 1), verbose = 0)
fitbest.acc <- GDINA::personparm(fitbest, "MAP")[, -(K+1)]
class.J <- GDINA::ClassRate(att.v, fitbest.acc) # upper-limit for accuracy
# FIXJ comparison
res.FIXJ.sum <- cdcat.summary(cdcat.obj = res.FIXJ.l, alpha = att.v)
res.FIXJ.sum$recovery$PCVcomp + ggplot2::geom_hline(yintercept = class.J$PCV[K],
                                                    color = "firebrick3")
res.FIXJ.sum$recovery$PCAmcomp + ggplot2::geom_hline(yintercept = class.J$PCA,
                                                    color = "firebrick3")
res.FIXJ.sum$item.exposure$stats
res.FIXJ.sum$item.exposure$plot
# VARJ comparison
res.VARJ.sum <- cdcat.summary(cdcat.obj = res.VARJ.l, alpha = att.v)
res.VARJ.sum$recovery
res.VARJ.sum$item.exposure$stats
res.VARJ.sum$item.exposure$plot
res.VARJ.sum$CATlength$stats
res.VARJ.sum$CATlength$plot

#####################################
# Example 3.                        #
# Nonparametric CD-CAT for          #
# small-scale assessment            #
#####################################

#-----------Data----------#
Q <- sim180DINA$simQ
K <- ncol(Q)
N <- 50
dat <- sim180DINA$simdat[1:N,]
att <- sim180DINA$simalpha[1:N,]

#--------Nonparametric CD-CAT--------#
res.NPS.FIXJ <- cdcat(dat = dat, itemSelect = "NPS", FIXED.LENGTH = TRUE,
                      MAXJ = 25, n.cores = 2,
                      NPS.args = list(Q = Q, gate = "AND", pseudo.prob = TRUE, w.type = 1,
                      seed = 12345))
res.NPS.VARJ <- cdcat(dat = dat, itemSelect = "NPS", FIXED.LENGTH = FALSE,
                      MAXJ = 25, precision.cut = 0.90, n.cores = 2,
                      NPS.args = list(Q = Q, gate = "AND", pseudo.prob = TRUE, w.type = 1,
                      seed = 12345))

#---------------Results--------------#
res.NPS.FIXJ$est[[1]] # estimates for the first examinee (fixed-length)
res.NPS.VARJ$est[[1]] # estimates for the first examinee (fixed-precision)
att.plot(res.NPS.FIXJ, i = 1) # plot for estimates for the first examinee (fixed-length)
att.plot(res.NPS.VARJ, i = 1) # plot for estimates for the first examinee (fixed-precision)
```

```
# FIXJ summary
res.NPS.FIXJ.sum.real <- cdcat.summary(cdcat.obj = res.NPS.FIXJ, alpha = att) # vs. real accuracy
res.NPS.FIXJ.sum.real$recovery$plotPCV
res.NPS.FIXJ.sum.real$recovery$plotPCA
res.NPS.FIXJ.sum.real$item.exposure$plot
# VARJ summary
res.NPS.VARJ.sum.real <- cdcat.summary(cdcat.obj = res.NPS.VARJ, alpha = att)
res.NPS.VARJ.sum.real$recovery
res.NPS.VARJ.sum.real$CATlength$stats
res.NPS.VARJ.sum.real$CATlength$plot
res.NPS.VARJ.sum.real$item.exposure$plot
# vs. maximum observable accuracy
fit <- NPCD::AlphaNP(Y = dat, Q = Q, gate = "AND")
att.J <- fit$alpha.est
class.J <- GDINA::ClassRate(att, att.J) # upper-limit for accuracy
res.NPS.FIXJ.sum.obse <- cdcat.summary(cdcat.obj = res.NPS.FIXJ, alpha = att.J)
res.NPS.FIXJ.sum.obse$recovery$plotPCV + ggplot2::geom_hline(yintercept = class.J$PCV[K],
                                                             color = "firebrick3")
res.NPS.FIXJ.sum.obse$recovery$plotPCA + ggplot2::geom_hline(yintercept = class.J$PCA,
                                                             color = "firebrick3")
```

`cdcat.summary` *Summary information for a* `cdcat` *object*

Description

This function provides classification accuracy, item exposure, and CAT length results for `cdcat` object. If a list of `cdcat` objects is included, these objects are compared through different tables and plots.

Usage

```
cdcat.summary(cdcat.obj, alpha, label = NULL)
```

Arguments

`cdcat.obj`	An object or list of objects of class `cdcat`
`alpha`	Numeric matrix of dimensions N x K with the reference attribute patterns used to compute attribute classification accuracy. It is expected that it will contain the true, generating alpha pattern or those estimated with the entire item bank. It is a guideline to evaluate the `cdcat` results
`label`	Character vector that contains the labels for the `cdcat` object(s). If `NULL` (by default), the models are used as labels

Value

`cdcat.summary` returns an object of class `cdcat.summary`.

recovery A list that contains the attribute classification accuracy results calculated at the pattern- (*PCV*) and attribute-levels (*PCA*). Two plots monitoring these variables are provided when `FIXED.LENGTH = TRUE`

item.exposure A list that contains the item exposure rates results: descriptive statistics (`stats`) and a plot representing the item exposure rates (`plot`). Note that when `FIXED.LENGTH = FALSE` the overlap rate is calculated based on the average CAT length

CATlength If the object or list of objects of class `cdcat` are fixed-precision applications (i.e., `FIXED.LENGTH = FALSE`), this additional list is included. It contains descriptive statistics (`stats`) and a plot (`plot`) describing the CAT length

---

`gen.data` *Data generation*

Description

This function can be used to generate datasets based on an object of class `gen.itembank`. The user can manipulate the examinees' attribute distribution or provide a matrix of attribute profiles. Data are simulated using the `GDINA::simGDINA` function (Ma & de la Torre, 2020).

Usage

```
gen.data(
  N = NULL,
  R = 1,
  item.bank = NULL,
  att.profiles = NULL,
  att.dist = "uniform",
  mvnorm.parm = list(mean = NULL, sigma = NULL, cutoffs = NULL),
  higher.order.parm = list(theta = NULL, lambda = NULL),
  categorical.parm = list(att.prior = NULL),
  seed = NULL
)
```

Arguments

`N`	Scalar numeric. Sample size for the datasets
`R`	Scalar numeric. Number of datasets replications. Default is 1
`item.bank`	An object of class `gen.itembank`
`att.profiles`	Numeric matrix indicating the true attribute profile for each examinee (*N* examinees x *K* attributes). If `NULL` (by default), `att.dist` must be specified
`att.dist`	Numeric vector of length 2^*K*, where *K* is the number of attributes. Distribution for attribute simulation. It can be `"uniform"` (by default), `"higher.order"`, `"mvnorm"`, or `"categorical"`. See `simGDINA` function of package `GDINA` for more information. Only used when `att.profiles = NULL`
`mvnorm.parm`	A list of arguments for multivariate normal attribute distribution (`att.dist = "mvnorm"`). See `simGDINA` function of package `GDINA` for more information
`higher.order.parm`	A list of arguments for higher-order attribute distribution (`att.dist = "higher.order"`). See `simGDINA` function of package `GDINA` for more information
`categorical.parm`	A list of arguments for categorical attribute distribution (`att.dist = "categorical"`). See `simGDINA` function of package `GDINA` for more information
`seed`	Scalar numeric. A scalar to use with `set.seed`

Value

gen.data returns an object of class gen.data.

simdat An array containing the simulated responses (dimensions N examinees x J items x R replicates). If R = 1, a matrix is provided

simalpha An array containing the simulated attribute profiles (dimensions N examinees x K attributes x R replicates). If R = 1, a matrix is provided

specifications A list that contains all the specifications

References

Ma, W. & de la Torre, J. (2020). GDINA: The generalized DINA model framework. R package version 2.7.9. Retrived from https://CRAN.R-project.org/package=GDINA

Examples

```
####################################
# Example 1.                       #
# Generate dataset (GDINA item     #
# parameters and uniform attribute #
# distribution)                    #
####################################

Q <- sim180GDINA$simQ
bank <- gen.itembank(Q = Q, mean.IQ = .70, range.IQ = .20, model = "GDINA")

simdata <- gen.data(N = 1000, item.bank = bank)

####################################
# Example 2.                       #
# Generate multiple datasets (DINA #
# model and multivariate normal    #
# attribute distribution)          #
####################################

Q <- sim180GDINA$simQ
K <- ncol(Q)
bank <- gen.itembank(Q = Q, mean.IQ = .70, range.IQ = .20, model = "DINA")

cutoffs <- qnorm(c(1:K)/(K+1))
m <- rep(0,K)
vcov <- matrix(0.5,K,K)
diag(vcov) <- 1
simdata <- gen.data(N = 1000, R = 20, item.bank = bank, att.dist = "mvnorm",
                  mvnorm.parm = list(mean = m, sigma = vcov, cutoffs = cutoffs))

####################################
# Example 3.                       #
# Generate dataset (multiple       #
# models and higher-order          #
# attribute distribution)          #
####################################

Q <- sim180GDINA$simQ
```

```
K <- ncol(Q)
model <- sample(c("DINA", "DINO", "ACDM"), size = nrow(Q), replace = TRUE)
bank <- gen.itembank(Q = Q, mean.IQ = .70, range.IQ = .20, model = model)

N <- 1000
theta <- rnorm(N)
lambda <- data.frame(a = runif(K, 0.7, 1.3), b = seq( -2, 2, length.out = K))
simdata <- gen.data(N = N, item.bank = bank, att.dist = "higher.order",
                    higher.order.parm = list(theta = theta,lambda = lambda))

###################################
# Example 4.                      #
# Generate dataset (GDINA model   #
# and given attribute profiles)   #
###################################

Q <- sim180GDINA$simQ
K <- ncol(Q)
bank <- gen.itembank(Q = Q, mean.IQ = .70, range.IQ = .20, model = "GDINA")

att.profiles <- matrix(data = c(1,0,0,0,0,
                                1,1,0,0,0,
                                1,1,1,0,0,
                                1,1,1,1,1), ncol = K, byrow = TRUE)
simdata <- gen.data(item.bank = bank, att.profiles = att.profiles)
```

`gen.itembank` *Item bank generation*

Description

This function can be used to generate an item bank. The user can provide a Q-matrix or create one defining a set of arguments. Item quality is sampled from a uniform distribution with mean = *mean.IQ* and range = *range.IQ*. Item parameters are generated so that the monotonicity constraint is satisfied.

Usage

```
gen.itembank(
  Q = NULL,
  gen.Q = list(J = NULL, K = NULL, propK.J = NULL, nI = 1, minJ.K = NULL, max.Kcor = 1),
  mean.IQ,
  range.IQ,
  model = "GDINA",
  min.param = 0,
  seed = NULL
)
```

Arguments

`Q`	Numeric matrix of length J number of items x K number of atributes. Q-matrix

gen.Q	A list of arguments to generate a Q-matrix if Q is not provided. J = number of items (scalar numeric). K = number of attributes (scalar numeric). propK.J = numeric vector summing up to 1 that determines the proportion of 1-attribute, 2-attribute, ..., items. The length of propK.J determines the maximum number of attributes considered for an item (see Examples below). nI = Scalar numeric that sets the minimum number of identity matrices to be included in the Q-matrix. minJ.K = numeric vector of length *K* that sets the minimum number of items measuring each attribute. max.Kcor = scalar numeric that sets the maximum positive correlation allowed between two attributes
mean.IQ	Item discrimination (mean for the uniform distribution). *mean.IQ* = *P*(**1**) - *P*(**0**) (Sorrel et al., 2017; Najera et al., in press). Must be a scalar numeric between 0 and 1
range.IQ	Item discrimination (range for the uniform distribution). Must be a scalar numeric between 0 and 1
model	A character vector of length *J* with one model for each item, or a single value to be used for all items. The possible options include "DINA", "DINO", "ACDM", and "GDINA". One-attribute items will be coded in the output as "GDINA"
min.param	Scalar numeric. Minimum value for the delta parameter of the principal effects of each attribute. Only usable if model = "ACDM" or model = "GDINA"
seed	Scalar numeric. A scalar to use with set.seed

Value

gen.itembank returns an object of class gen.itembank.

simQ Generated Q-matrix (only if gen.Q arguments have been used)

simcatprob.parm A list of success probabilities for each latent group in each item

simdelta.parm A list of delta parameters for each item

check A list that contains the mean.IQ and range.IQ for the item bank so that users can check whether these values match the expected results

specifications A list that contains all the specifications

References

Najera, P., Sorrel, M. A., de la Torre, J., & Abad, F. J. (in press). Improving robustness in Q-matrix validation using an iterative and dynamic procedure. *Applied Psychological Measurement*.

Sorrel, M. A., Abad, F. J., Olea, J., de la Torre, J., & Barrada, J. R. (2017). Inferential item-fit evaluation in cognitive diagnosis modeling. *Applied Psychological Measurement, 41*, 614-631.

Examples

```
###################################
# Example 1.                      #
# Generate item bank providing a  #
# Q-matrix using the G-DINA model #
###################################

Q <- sim180GDINA$simQ
bank <- gen.itembank(Q = Q, mean.IQ = .70, range.IQ = .20, model = "GDINA")

###################################
```

```
# Example 2.                        #
# Generate item bank providing a    #
# Q-matrix using multiple models    #
####################################

Q <- sim180GDINA$simQ
K <- ncol(Q)
model <- sample(c("DINA", "DINO", "ACDM"), size = nrow(Q), replace = TRUE)
bank <- gen.itembank(Q = Q, mean.IQ = .70, range.IQ = .20, model = model)

####################################
# Example 3.                        #
# Generate item bank without        #
# providing a Q-matrix (using       #
# gen.Q arguments)                  #
####################################

bank <- gen.itembank(gen.Q = list(J = 150, K = 5, propK.J = c(0.4, 0.3, 0.2, 0.1),
                     nI = 3, minJ.K = 30, max.Kcor = 1),
                     mean.IQ = .80, range.IQ = .10, min.param = 0.1)
```

`LR.2step` *Item-level model comparison using 2LR test*

Description

This function evaluates whether the saturated G-DINA model can be replaced by reduced CDMs without significant loss in model data fit for each item using two-step likelihood ratio test (2LR). Sorrel, de la Torre, Abad, and Olea (2017) and Ma & de la Torre (2018) can be consulted for details.

Usage

```
LR.2step(fit, p.adjust.method = "holm", alpha.level = 0.05)
```

Arguments

`fit`	Calibrated item bank with the `GDINA::GDINA` (Ma & de la Torre, 2020) or `CDM::gdina` (Robitzsch et al., 2020) R packages functions
`p.adjust.method`	Scalar character. Correction method for *p*-values. Possible values include `"holm"`, `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"`, and `"none"`. See `p.adjust` function from the `stats` R package for additional details. Default is `holm`
`alpha.level`	Scalar numeric. Alpha level for decision. Default is `0.05`

Value

`LR2.step` returns an object of class `LR2.step`

LR2 Numeric matrix. LR2 statistics

pvalues Numeric matrix. *p*-values associated with the 2LR statistics

adj.pvalues Numeric matrix. Adjusted *p*-values associated with the 2LR statistics

df Numeric matrix. Degrees of freedom

models.adj.pvalues Character vector denoting the model selected for each item using the *largestp* rule (Ma et al., 2016). All statistics whose *p*-values are less than `alpha.level` are rejected. All statistics with *p*-value larger than `alpha.level` define the set of candidate reduced models. The G-DINA model is retained if all statistics are rejected. Whenever the set includes more than one model, the model with the largest *p*-value is selected as the best model for that item

References

Ma, W. & de la Torre, J. (2018). Category-level model selection for the sequential G-DINA model. *Journal of Educational and Behavorial Statistic, 44*, 45-77.

Ma, W. & de la Torre, J. (2020). GDINA: The generalized DINA model framework. R package version 2.7.9. Retrived from https://CRAN.R-project.org/package=GDINA

Ma, W., Iaconangelo, C., & de la Torre, J. (2016). Model similarity, model selection and attribute classification. *Applied Psychological Measurement, 40*, 200-217.

Robitzsch, A., Kiefer, T., George, A. C., & Uenlue, A. (2020). CDM: Cognitive Diagnosis Modeling. R package version 7.5-15. https://CRAN.R-project.org/package=CDM

Sorrel, M. A., de la Torre, J., Abad, F. J., & Olea, J. (2017). Two-step likelihood ratio test for item-level model comparison in cognitive diagnosis models. *Methodology, 13*, 39-47.

Examples

```
Q <- sim180DINA$simQ
dat <- sim180DINA$simdat
resGDINA <- GDINA::GDINA(dat = dat, Q = Q, model = "GDINA",verbose = FALSE)
#resCDM <- CDM::gdina(data = dat, q.matrix = Q, rule = "GDINA", progress = FALSE)
LR2.GDINA <- LR.2step(fit = resGDINA) # GDINA package
#LR2.CDM <- LR.2step(fit = resCDM) # CDM package
mean(LR2.GDINA$models.adj.pvalues[which(rowSums(Q) != 1)] ==
      sim180DINA$specifications$item.bank$specifications$model[which(rowSums(Q) != 1)])
#mean(LR2.CDM$models.adj.pvalues[which(rowSums(Q) != 1)] ==
#     sim180DINA$specifications$item.bank$specifications$model[which(rowSums(Q) != 1)])
```

---

sim180combination *Simulated data (180 items, a combination of DINA, DINO, and A-CDM items)*

Description

Simulated data, Q-matrix and item parameters for a 180-item bank with 5 attributes. Data generated using the `gen.data` function.

Usage

```
sim180combination
```

Format

A list with components:

`simdat` Numeric array. Simulated responses of 250 examinees for two replicates

`simQ` Numeric matrix. Simulated Q-matrix

`simalpha` Numeric array. Simulated attribute patterns of 250 examinees for two replicates

`specifications` A list that contains all the specifications that were used in the `gen.itembank` function

---

`sim180DINA` *Simulated data (180 items, DINA model)*

Description

Simulated data, Q-matrix and item parameters for a 180-item bank with 5 attributes. Data generated using the `gen.data` function.

Usage

```
sim180DINA
```

Format

A list with components:

`simdat` Numeric matrix. Simulated responses of 500 examinees

`simQ` Simulated Q-matrix

`simalpha` Numeric matrix. Simulated attribute patterns of 500 examinees

`specifications` A list that contains all the specifications that were used in the `gen.itembank` function

---

`sim180GDINA` *Simulated data (180 items, G-DINA model)*

Description

Simulated data, Q-matrix and item parameters for a 180-item bank with 5 attributes. Data generated using the `gen.data` function.

Usage

```
sim180GDINA
```

Format

A list with components:

`simdat` Numeric matrix. Simulated responses of 500 examinees

`simQ` Simulated Q-matrix

`simalpha` Numeric matrix. Simulated attribute patterns of 500 examinees

`specifications` A list that contains all the specifications that were used in the `gen.itembank` function

Index