Package 'cdcatR'

May 8, 2026

Type Package

Title Cognitive Diagnostic Computerized Adaptive Testing

Version 1.0.7

Date 2026-02-11

Description Provides a set of functions for conducting cognitive diagnostic computerized adaptive testing applications (Chen, 2009) <DOI:10.1007/s11336-009-9123-2>). It includes different item selection rules such us the global discrimination index (Kaplan, de la Torre, and Barrada (2015) <DOI:10.1177/0146621614554650>) and the nonparametric selection method (Chang, Chiu, and Tsai (2019) <DOI:10.1177/0146621618813113>), as well as several stopping rules. Functions for generating item banks and responses are also provided. To guide item bank calibration, model comparison at the item level can be conducted using the two-step likelihood ratio test statistic by Sorrel, de la Torre, Abad and Olea (2017) <DOI:10.1027/1614-2241/a000131>.

License GPL-3

LazyData TRUE

Depends R (>= 3.5.0)

Imports cdmTools (>= 1.0.1), GDINA (>= 2.2.0), ggplot2 (>= 3.3.0), cowplot, foreach, doSNOW, NPCD, stats

Suggests CDM

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

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

RoxygenNote 7.3.3.9000

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>

NeedsCompilation no

Repository CRAN

Date/Publication 2026-02-11 09:30:02 UTC

Contents

`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), the general nonparametric item selection method (GNPS; Chiu & Chang, 2021), or random selection. Fixed length or fixed precision CD-CAT can be applied. Fixed precision CD-CAT with NPS and GNPS 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,
  startRule = "random",
  startK = FALSE,
  att.prior = NULL,
  initial.distr = NULL,
  precision.cut = 0.8,
  NP.args = list(Q = NULL, gate = NULL, PPP = TRUE, w = 2),
  itemExposurecontrol = NULL,
  b = 0,
  maxr = 1,
  itemConstraint = NULL,
  constraint.args = list(ATTRIBUTEc = NULL),
  n.cores = 2,
  seed = NULL,
  print.progress = TRUE
)
```

Arguments

fit	An object of class GDINA, gdina (parametric CD-CAT), or GNPC (non-parametric CD-CAT based on GNPS). Calibrated item bank with the GDINA::GDINA (Ma & de la Torre, 2020), CDM::gdina (Robitzsch et al., 2020), or cdmTools::GNPC (Najera et al., 2022) 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`, `GNPS`, 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`

startRule — Scalar character. Starting rule: first item is selected at random with `random` and first item is selected using `itemSelect` with `max`. Default is `random`. Seed for random is `NPS.args$seed`

startK — Scalar logical. Start the CAT with an identity matrix (`TRUE`) or not proceed with startRule from the first item (`FALSE`). Default is `FALSE`

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"`

NP.args — A list of options when `itemSelect = "NPS" or "GNPS"`. `Q` = Q-matrix to be used in the analysis. `gate` = "AND" or "OR", depending on whether a conjunctive o disjunctive nonparametric CDM is used. `PPP` = pseudo-posterior probability of each examinee mastering each attribute (experimental). `w` = weight type used for computing the pseudo-posterior probability (experimental)

itemExposurecontrol — Scalar character. Item exposure control: `NULL` or progressive method (Barrada, Olea, Ponsoda, & Abad, 2008) with `"progressive"`. Default is `NULL`. Seed for the random component is `NPS.args$seed`

b — Scalar numeric. Acceleration parameter for the item exposure method. Only applies if `itemExposurecontrol = "progressive"`. In the progressive method the first item is selected at random and the last item (i.e., `MAXJ`) is selected purely based on `itemSelect`. The rest of the items are selected combining both a random and information components. The loss of importance of the random component will be linear with `b = 0`, inverse exponential with `b < 0`, or exponential with `b > 0`. Thus, `b` allows to optimize accuracy (`b < 0`) or item security (`b > 0`). Default is 0

maxr — Scalar numeric. Value should be in the range 0-1. Maximum item exposure rate that is tolerated. Default is 1. Note that for `maxr < 1` parallel computing cannot be implemented

`itemConstraint` Scalar character. Constraints that must be satisfied by the set of items applied: `NULL` or attribute constraint (Henson & Douglas, 2005) with `"attribute"`. If `"attribute"` is chosen, then each attribute must be measured at least a specific number of times indicated in the `constraint.args$ATTRIBUTEc` argument. Default is `NULL`

`constraint.args`
A list of options when `itemConstraint != "NULL"`. At the moment it only includes the argument `ATTRIBUTEc` which must be a numeric vector of length `ncol(Q)` indicating the minimum number of items per attribute to be administered. Default is 3

`n.cores` Scalar numeric. Number of cores to be used during parallelization. Default is 2

`seed` Numeric vector of length 1. Some methods have a random component, so a seed is required for consistent results

`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

Barrada, J. R., Olea, J., Ponsoda, V., & Abad, F. J. (2008). Incorporating randomness in the Fisher information for improving item-exposure control in CATs.*British Journal of Mathematical and Statistical Psychology, 61*, 493-513.

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.

Chiu, C. Y., & Chang, Y. P. (2021). Advances in CD-CAT: The general nonparametric item selection method. *Psychometrika, 86*, 1039-1057.

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

Henson, R., & Douglas, J. (2005). Test construction for cognitive diagnosis. *Applied Psychological Measurement, 29*, 262-277.

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.

Nájera, P., Sorrel, M. A., & Abad, F. J. (2022). cdmTools: Useful Tools for Cognitive Diagnosis Modeling. R package version 1.0.1. https://CRAN.R-project.org/package=cdmTools

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$alpha.recovery$PCV.plot
res.FIXJ.sum.real$item.exposure$exp.plot
# VARJ summary
res.VARJ.sum.real <- cdcat.summary(cdcat.obj = res.VARJ, alpha = att)
res.VARJ.sum.real$alpha.recovery$PCV
res.VARJ.sum.real$item.exposure$stats
res.VARJ.sum.real$item.exposure$length.plot
res.VARJ.sum.real$item.exposure$exp.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$alpha.recovery$PCV.plot + ggplot2::geom_hline(yintercept = class.J$PCV[K],
                                                    color = "firebrick3")
res.FIXJ.sum.obse$alpha.recovery$PCA.plot + ggplot2::geom_hline(yintercept = class.J$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 (NPS)      #
#####################################

#-----------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,
                     NP.args = list(Q = Q, gate = "AND", pseudo.prob = TRUE, w.type = 2),
                      seed = 12345)
res.NPS.VARJ <- cdcat(dat = dat, itemSelect = "NPS", FIXED.LENGTH = FALSE,
                      MAXJ = 25, precision.cut = 0.90, n.cores = 2,
                     NP.args = list(Q = Q, gate = "AND", pseudo.prob = TRUE, w.type = 2),
                      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$alpha.recovery$PCV.plot
res.NPS.FIXJ.sum.real$item.exposure$exp.plot
# VARJ summary
res.NPS.VARJ.sum.real <- cdcat.summary(cdcat.obj = res.NPS.VARJ, alpha = att)
res.NPS.VARJ.sum.real$alpha.recovery$PCV.plot
res.NPS.VARJ.sum.real$item.exposure$stats
res.NPS.VARJ.sum.real$item.exposure$length.plot
res.NPS.VARJ.sum.real$item.exposure$exp.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$alpha.recovery$PCV.plot + ggplot2::geom_hline(yintercept = class.J$PCV[K],
                                                    color = "firebrick3")
res.NPS.FIXJ.sum.obse$alpha.recovery$PCA.plot + ggplot2::geom_hline(yintercept = class.J$PCA,
                                                    color = "firebrick3")

#####################################
# Example 4.                        #
# Nonparametric CD-CAT for          #
# small-scale assessment (GNPS)     #
#####################################

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

#----------Model calibration----------#
gnpc <- cdmTools::GNPC(dat = dat, Q = Q, verbose = 0)

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

#---------------Results--------------#
res.GNPS.FIXJ$est[[1]] # estimates for the first examinee (fixed-length)
res.GNPS.VARJ$est[[1]] # estimates for the first examinee (fixed-precision)
att.plot(res.GNPS.FIXJ, i = 1) # plot for estimates for the first examinee (fixed-length)
att.plot(res.GNPS.VARJ, i = 1) # plot for estimates for the first examinee (fixed-precision)
# FIXJ summary
res.GNPS.FIXJ.sum.real <- cdcat.summary(cdcat.obj = res.GNPS.FIXJ, alpha = att) # vs. real accuracy
res.GNPS.FIXJ.sum.real$alpha.recovery$PCV.plot
res.GNPS.FIXJ.sum.real$item.exposure$exp.plot
# VARJ summary
res.GNPS.VARJ.sum.real <- cdcat.summary(cdcat.obj = res.GNPS.VARJ, alpha = att)
res.GNPS.VARJ.sum.real$alpha.recovery$PCV.plot
res.GNPS.VARJ.sum.real$item.exposure$exp.plot
res.GNPS.VARJ.sum.real$item.exposure$length.plot
```

---

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 = NULL, label = NULL, plots = TRUE)
```

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. This is required to obtain the `alpha.recovery` output and if a list of objects of class `cdcat` is provided as input.
`label`	Character vector that contains the labels for the `cdcat` object(s). If `NULL` (by default), the models are used as labels
`plots`	Scalar logical. Whether or not the plots should be created. Default is `TRUE`

Value

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

If a list of objects of class `cdcat` is provided:

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 If only one object of class `cdcat` is provided:

alpha.estimates Information about the classifications made by the CD-CAT procedure

item.exposure A list that contains the item exposure rates and CAT length 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

alpha.recovery If `alpha` was provided a list that contains information on attribute classification accuracy is provided

specifications A list that contains all the specifications

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)
```

---

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*. Alternatively, it is possible to provide a matrix with the guessing and slip parameters (*gs.param*) or a list with the success probabilities of each latent group (*catprob.parm*). 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 = 1, max.Kcor = 1),
  mean.IQ = NULL,
  range.IQ = NULL,
```

```
  gs.parm = NULL,
  catprob.parm = NULL,
  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* = $P(\mathbf{1})$ - $P(\mathbf{0})$ (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
`gs.parm`	A matrix or data frame for guessing and slip parameters. The number of columns must be 2, where the first column represents the guessing parameters (or $P(\mathbf{0})$), and the second column represents slip parameters (or 1-$P(\mathbf{1})$)
`catprob.parm`	A list of success probabilities of each latent group for each non-zero category of each item. This argument requires to specify a Q-matrix in `Q`
`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. (2020). Improving robustness in Q-matrix validation using an iterative and dynamic procedure. *Applied Psychological Measurement, 44*, 431-446.

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 with gs.parm            #
####################################

Q <- sim180GDINA$simQ
J <- nrow(Q)
gs <- data.frame(g = runif(J, 0.2, 0.4), s = runif(J, 0, 0.2))
bank <- gen.itembank(Q = Q, gs.parm = gs, model = "GDINA", min.param = 0.05)

####################################
# Example 3.                       #
# Generate item bank providing a   #
# Q-matrix with catprob.parm       #
####################################

Q <- sim180GDINA$simQ[c(1:5, 73:77, 127:131),]
catparm.list <- list(J1 = c(0.2, 0.8),
                     J2 = c(0.1, 0.7),
                     J3 = c(0.2, 0.9),
                     J4 = c(0.3, 0.9),
                     J5 = c(0.3, 0.8),
                     J6 = c(0.2, 0.4, 0.5, 0.8),
                     J7 = c(0.1, 0.7, 0.8, 0.9),
                     J8 = c(0.2, 0.3, 0.3, 0.7),
                     J9 = c(0.2, 0.4, 0.4, 0.6),
                     J10 = c(0.3, 0.5, 0.6, 0.9),
                     J11 = c(0.1, 0.3, 0.3, 0.5, 0.4, 0.5, 0.7, 0.8),
                     J12 = c(0.2, 0.6, 0.7, 0.6, 0.7, 0.8, 0.8, 0.9),
                     J13 = c(0.2, 0.6, 0.2, 0.3, 0.6, 0.7, 0.4, 0.9),
                     J14 = c(0.3, 0.4, 0.3, 0.5, 0.5, 0.6, 0.7, 0.9),
                     J15 = c(0.1, 0.1, 0.2, 0.1, 0.2, 0.3, 0.2, 0.8))
```

```
bank <- gen.itembank(Q = Q, catprob.parm = catparm.list)

###################################
# Example 4.                      #
# 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 5.                      #
# 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. Conducting this type of analysis can facilitate the calibration of the item bank and have implications for the CAT accuracy and item usage (Sorrel, Abad, & Nájera, 2021).

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.

Sorrel, M. A., Abad, F. J., & Nájera, P. (2021). Improving accuracy and usage by correctly selecting: The effects of model selection in cognitive diagnosis computerized adaptive testing. *Applied Psychological Measurement, 45*, 112-129.

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