Package ‘GDINA’

May 31, 2021

Type Package

Title The Generalized DINA Model Framework

Version 2.8.7

Date 2021-05-28

Description A set of psychometric tools for cognitive diagnosis modeling based on the generalized deterministic inputs, noisy and gate (G-DINA) model by de la Torre (2011) <DOI:10.1007/s11336-011-9207-7> and its extensions, including the sequential G-DINA model by Ma and de la Torre (2016) <DOI:10.1111/bmsp.12070> for polynomous responses, and the polynomous G-DINA model by Chen and de la Torre <DOI:10.1177/0146621613479818> for polynomous attributes. Joint attribute distribution can be independent, saturated, higher-order, loglinear smoothed or structured. Q-matrix validation, item and model fit statistics, model comparison at test and item level and differential item functioning can also be conducted. A graphical user interface is also provided. For tutorials, please check Ma and de la Torre (2020) <DOI:10.18637/jss.v093.i14>, Ma and de la Torre (2019) <DOI:10.1111/emip.12262>, Ma (2019) <DOI:10.1007/978-3-030-05584-4_29> and de la Torre and Akbay (2019) <DOI:10.14689/ejer.2019.80.9>.

License GPL-3

LazyData TRUE

Depends R (>= 3.1.0)

Imports alabama, graphics, ggplot2, MASS, nloptr, numDeriv, Rcpp (>= 0.12.1), Rsolnp, stats, shiny, shinydashboard, utils

Suggests CDM, Matrix, testthat, poLCA, stringr, knitr, rmarkdown

LinkingTo Rcpp, RcppArmadillo

URL https://github.com/Wenchao-Ma/GDINA,
    https://wenchao-ma.github.io/GDINA/

BugReports https://github.com/Wenchao-Ma/GDINA/issues

RoxygenNote 7.1.1

Encoding UTF-8

Collate 'CA.R' 'CR.R' 'DTM.R' 'ExportedFuncs.R' 'GDINA.R' 'GDI.R' 'GDINA-package.R' 'GMSCDM.R' 'HO.R' 'ILCA.R' 'M2.R' 'MCmodel.R'
topics documented:

'Mstep.R' 'Mstep_DTM.R' 'MultipleGroup_Estimation.R'
'RcppExports.R' 'SingleGroup_Estimation.R' 'anova.GDINA.R'
'autoGDINA.R' 'bootSE.R' 'coef.R' 'dif.R' 'ecpe.R' 'simGDINA.R'
'itemfit.R' 'modelcomp.R' 'extract.R' 'frac20.R'
'itemparm.GDINA.R' 'monocheck.R' 'personparm.GDINA.R'
'plotRF.GDINA.R' 's3GDINA.R' 'print.GDINA.R' 'score.R'
'sim10GDINA.R' 'sim10MCDINA.R' 'sim10MCDINA2.R'
'sim20seqGDINA.R' 'sim21seqDINA.R' 'sim30DINA.R' 'sim30GDINA.R'
'sim30pGDINA.R' 'startGDINA.R' 'structuralparm.R'
'summary.GDINA.R' 'utils.R' 'zzz.R'

NeedsCompilation: yes

Author: Wenchao Ma [aut, cre, cph],
        Jimmy de la Torre [aut, cph],
        Miguel Sorrel [ctb],
        Zhehan Jiang [ctb]

Maintainer: Wenchao Ma <wenchao.ma@ua.edu>

VignetteBuilder: knitr

Repository: CRAN

Date/Publication: 2021-05-31 04:40:05 UTC

R topics documented:

GDINA-package ......................................................... 3
att.structure .......................................................... 6
attributepattern ....................................................... 7
autoGDINA .............................................................. 8
bdiagMatrix ............................................................ 12
bootSE ................................................................. 13
CA ................................................................. 14
cjoint ................................................................. 15
ClassRate ............................................................ 16
designmatrix .......................................................... 17
dif ................................................................. 18
DTM ................................................................. 20
ecpe ................................................................. 22
extract ............................................................... 23
frac20 ............................................................... 25
GDINA ............................................................... 26
GMSCDM ............................................................. 46
ILCA ............................................................... 49
indlogLik ............................................................ 50
indlogPost ........................................................... 51
itemfit ............................................................. 52
itemparm ............................................................ 54
LC2LG ............................................................ 55
MCmodel ............................................................ 56
Description

For conducting CDM analysis within the G-DINA model framework.

Details

This package (Ma & de la Torre, 2020a) provides a framework for a series of cognitively diagnostic analyses for dichotomous and polytomous responses.

Various cognitive diagnosis models (CDMs) can be calibrated using the GDINA function, including the G-DINA model (de la Torre, 2011), the deterministic inputs, noisy and gate (DINA; de la Torre, 2009; Junker & Sijtsma, 2001) model, the deterministic inputs, noisy or gate (DINO; Templin & Henson, 2006) model, the reduced reparametrized unified model (R-RUM; Hartz, 2002), the additive CDM (A-CDM; de la Torre, 2011), and the linear logistic model (LLM; Maris, 1999), the multiple-strategy DINA model (de la Torre, & Douglas, 2008) and models defined by users under the G-DINA framework using different link functions and design matrices (de la Torre, 2011). Note that the LLM is also called compensatory RUM and the RRUM is equivalent to the generalized NIDA model.
For ordinal and nominal responses, the sequential G-DINA model (Ma, & de la Torre, 2016) can be fitted and most of the aforementioned CDMs can be used as the processing functions (Ma, & de la Torre, 2016) at the category level. Different CDMs can be assigned to different items within a single assessment. Item parameters are estimated using the MMLE/EM algorithm. Details about the estimation algorithm can be found in Ma and de la Torre (2020). The joint attribute distribution can be modeled using an independent model, a higher-order IRT model (de la Torre, & Douglas, 2004), a loglinear model (Xu & von Davier, 2008), a saturated model or a hierarchical structures (e.g., linear, divergent). Monotonicity constraints for item/category success probabilities can also be specified.

In addition, to handle multiple strategies, generalized multiple-strategy CDMs for dichotomous response (Ma & Guo, 2019) can be fitted using GMSCDM function and diagnostic tree model (Ma, 2019) can also be estimated using DTM function for polytomous responses. Note that these functions are experimental, and are expected to be further extended in the future. Other diagnostic approaches include the multiple-choice model (de la Torre, 2009) and an iterative latent class analysis (ILCA; Jiang, 2019).

Various Q-matrix validation methods (de la Torre, & Chiu, 2016; de la Torre & Ma, 2016; Ma & de la Torre, 2020b; Najera, Sorrel, & Abad, 2019; see Qval), model-data fit statistics (Chen, de la Torre, & Zhang, 2013; Hansen, Cai, Monroe, & Li, 2016; Liu, Tian, & Xin, 2016; Ma, 2020; see modelfit and itemfit), model comparison at test and item level (de la Torre, 2011; de la Torre, & Lee, 2013; Ma, Iaconangelo, & de la Torre, 2016; Ma & de la Torre, 2019; Sorrel, Abad, Olea, de la Torre, & Barrada, 2017; Sorrel, de la Torre, Abad, & Olea, 2017; see modelcomp), and differential item functioning (Hou, de la Torre, & Nandakumar, 2014; Ma, Terzi, Lee, & de la Torre, 2017; see dif) can also be conducted.

To use the graphical user interface, check startGDINA.

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

References


Generate hierarchical attribute structures

This function can be used to generate hierarchical attributes structures, and to provide prior joint attribute distribution with hierarchical structures.

Usage

att.structure(hierarchy.list = NULL, K, Q, att.prob = "uniform")

Arguments

hierarchy.list a list specifying the hierarchical structure between attributes. Each element in this list specifies a DIRECT prerequisite relation between two or more attributes. See example for more information.

K the number of attributes involved in the assessment

Q Q-matrix

att.prob How are the probabilities for latent classes simulated? It can be "random" or "uniform".

Value

att.str reduced latent classes under the specified hierarchical structure
impossible.latentclass impossible latent classes under the specified hierarchical structure
att.prob probabilities for all latent classes; 0 for impossible latent classes

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

See Also

GDINA, autoGDINA


See Also

CDM for estimating G-DINA model and a set of other CDMs; ACTCD and NPCD for nonparametric CDMs; dina for DINA model in Bayesian framework
Examples

```r
## Not run:
###########
# Leighton et al. (2004, p.210)
#
###########
# linear structure A1->A2->A3->A4->A5->A6
K <- 6
linear=list(c(1,2),c(2,3),c(3,4),c(4,5),c(5,6))
att.structure(linear,K)

# convergent structure A1->A2->A3->A5->A6;A1->A2->A4->A5->A6
K <- 6
converg <- list(c(1,2),c(2,3),c(2,4),
c(3,4,5), #this is how to show that either A3 or A4 is a prerequisite to A5
   c(5,6))
at.attrstructure(converg,K)

# convergent structure [the difference between this one and the previous one is that
# A3 and A4 are both needed in order to master A5]
K <- 6
converg2 <- list(c(1,2),c(2,3),c(2,4),
c(3,5), #this is how to specify that both A3 and A4 are needed for A5
   c(4,5), #this is how to specify that both A3 and A4 are needed for A5
   c(5,6))
at.attrstructure(converg2,K)

# divergent structure A1->A2->A3;A1->A4->A5;A1->A4->A6
diverg <- list(c(1,2),
c(2,3),
c(1,4),
c(4,5),
c(4,6))
at.attrstructure(diverg,K)

# unstructured A1->A2;A1->A3;A1->A4;A1->A5;A1->A6
unstru <- list(c(1,2),c(1,3),c(1,4),c(1,5),c(1,6))
at.attrstructure(unstru,K)

## See Example 4 and 5 in GDINA function
## End(Not run)
```

attributepattern

Generate all possible attribute patterns

Description

This function generates all possible attribute patterns. The Q-matrix needs to be specified for polytomous attributes.
Usage

\texttt{attributepattern(K, Q)}

Arguments

\begin{itemize}
  \item \texttt{K} \hspace{1cm} \text{number of attributes}
  \item \texttt{Q} \hspace{1cm} \text{Q-matrix; required when Q-matrix is polytomous}
\end{itemize}

Value

A $2^K \times K$ matrix consisting of attribute profiles for $2^K$ latent classes

Author(s)

Wenchao Ma, The University of Alabama, \texttt{<wenchao.ma@ua.edu>}
Jimmy de la Torre, The University of Hong Kong

Examples

\begin{verbatim}
attributepattern(3)
q <- matrix(scan(text = "0 1 2 1 0 1 1 2 0"),ncol = 3)
q
attributepattern(Q=q)
q <- matrix(scan(text = "0 1 1 0 1 1 1 0"),ncol = 3)
q
attributepattern(K=ncol(q),Q=q)
\end{verbatim}

\hline
autoGDINA & \textit{Q-matrix validation, model selection and calibration in one run} \\
\hline

Description

autoGDINA conducts a series of CDM analyses within the G-DINA framework. Particularly, the GDINA model is fitted to the data first using the \texttt{GDINA} function; then, the Q-matrix is validated using the function \texttt{Qval}. Based on the suggested Q-matrix, the data is fitted by the G-DINA model again, followed by an item level model selection via the Wald test using \texttt{modelcomp}. Lastly, the selected models are calibrated based on the suggested Q-matrix using the \texttt{GDINA} function. The Q-matrix validation and item-level model selection can be disabled by the users. Possible reduced CDMs for Wald test include the DINA model, the DINO model, A-CDM, LLM and RRUM. See \texttt{Details} for the rules of item-level model selection.
autoGDINA

Usage

autoGDINA(
  dat,
  Q,
  modelselection = TRUE,
  modelselectionrule = "simpler",
  alpha.level = 0.05,
  modelselection.args = list(),
  Qvalid = TRUE,
  Qvalid.args = list(),
  GDINA1.args = list(),
  GDINA2.args = list(),
  CDM.args = list()
)

## S3 method for class 'autoGDINA'
summary(object, ...)

Arguments

dat A required $N \times J$ matrix or data.frame consisting of the responses of $N$
  individuals to $J$ items. Missing values need to be coded as NA.

Q A required matrix; The number of rows occupied by a single-strategy
dichotomous item is 1, by a polytomous item is the number of nonzero categories, and
by a multiple-strategy dichotomous item is the number of strategies. The
number of column is equal to the number of attributes if all items are single-strategy
dichotomous items, but the number of attributes + 2 if any items are polytomous
or have multiple strategies. For a polytomous item, the first column represents
the item number and the second column indicates the nonzero category number.
For a multiple-strategy dichotomous item, the first column represents the
item number and the second column indicates the strategy number. For binary
attributes, 1 denotes the attributes are measured by the items and 0 means the
attributes are not measured. For polytomous attributes, non-zero elements indi-
cate which level of attributes are needed (see Chen, & de la Torre, 2013). See
Examples.

modelselection logical; conducting model selection or not?

modelselectionrule how to conducted model selection? Possible options include simpler, largestp
and DS. See Details.

alpha.level nominal level for the Wald test. The default is 0.05.

modelselection.args arguments passed to modelcomp

Qvalid logical; validate Q-matrix or not? TRUE is the default.

Qvalid.args arguments passed to Qval

GDINA1.args arguments passed to GDINA function for initial G-DINA calibration

GDINA2.args arguments passed to GDINA function for the second G-DINA calibration
Cdm.args arguments passed to GDINA function for final calibration
object GDINA object for various S3 methods
... additional arguments

Details

After the Wald statistics for each reduced CDM were calculated for each item, the reduced models with p values less than the pre-specified alpha level were rejected. If all reduced models were rejected for an item, the G-DINA model was used as the best model; if at least one reduced model was retained, three different rules can be implemented for selecting the best model:

When modelselectionrule is simpler:
If (a) the DINA or DINO model was one of the retained models, then the DINA or DINO model with the larger p value was selected as the best model; but if (b) both DINA and DINO were rejected, the reduced model with the largest p value was selected as the best model for this item. Note that when the p-values of several reduced models were greater than 0.05, the DINA and DINO models were preferred over the A-CDM, LLM, and R-RUM because of their simplicity. This procedure is originally proposed by Ma, Iaconangelo, and de la Torre (2016).

When modelselectionrule is largestp:
The reduced model with the largest p-values is selected as the most appropriate model.

When modelselectionrule is DS:
The reduced model with non-significant p-values but the smallest dissimilarity index is selected as the most appropriate model. Dissimilarity index can be viewed as an effect size measure, which quantifies how dissimilar the reduced model is from the G-DINA model (See Ma, Iaconangelo, and de la Torre, 2016 for details).

Value

a list consisting of the following elements:

GDINA1.obj initial GDINA calibration of class GDINA
GDINA2.obj second GDINA calibration of class GDINA
Qval.obj Q validation object of class Qval
Wald.obj model comparison object of class modelcomp
CDM.obj Final CDM calibration of class GDINA

Methods (by generic)

• summary: print summary information

Note

Returned GDINA1.obj, GDINA2.obj and CDM.obj are objects of class GDINA, and all S3 methods suitable for GDINA objects can be applied. See GDINA and extract. Similarly, returned Qval.obj and Wald.obj are objects of class Qval and modelcomp.
Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References


See Also

GDINA, modelcomp, Qval

Examples

```r
## Not run:
# simulated responses
Q <- sim10GDINA$simQ
dat <- sim10GDINA$simdat

# misspecified Q
misQ <- Q
misQ[10,] <- c(0,1,0)
out1 <- autoGDINA(dat,misQ,modelselectionrule="largestp")
summary(out1)
AIC(out1$CDM.obj)

# simulated responses
Q <- sim30GDINA$simQ
dat <- sim30GDINA$simdat

# misspecified Q
misQ <- Q
misQ[1,] <- c(1,1,0,1,0)
auto <- autoGDINA(dat,misQ,Qvalid = TRUE, Qvalid.args = list(method = "wald"),
    modelselectionrule="simpler")
summary(auto)
AIC(auto$CDM.obj)

# using the other selection rule
out11 <- autoGDINA(dat,misQ,modelselectionrule="simpler",
    modelselection.args = list(models = c("DINO","DINA")))
summary(out11)

# disable model selection function
out12 <- autoGDINA(dat,misQ,modelselection=FALSE)
out12
```
summary(out12)

# Disable Q-matrix validation
out3 <- autoGDINA(dat = dat, Q = misQ, Qvalid = FALSE)
out3
summary(out3)

## End(Not run)

---

**bdiagMatrix**

Create a block diagonal matrix

**Description**

Create a block diagonal matrix

**Usage**

bdiagMatrix(mlist, fill = 0)

**Arguments**

- `mlist`: a list of matrices
- `fill`: value to fill the non-diagonal elements

**Value**

a block diagonal matrix

**See Also**

bdiag in Matrix

**Examples**

m1 <- bdiagMatrix(list(matrix(1:4, 2), diag(3)))
m2 <- bdiagMatrix(list(matrix(1:4, 2), diag(3)), fill = NA)
Description

This function conducts nonparametric and parametric bootstrap to calculate standard errors of model parameters. Parametric bootstrap is only applicable to single group models.

Usage

bootSE(GDINA.obj, bootsample = 50, type = "nonparametric", randomseed = 12345)

Arguments

GDINA.obj an object of class GDINA
bootsample the number of bootstrap samples
type type of bootstrap method. Can be parametric or nonparametric
randomseed random seed for resampling

Value

itemparm.se standard errors for item probability of success in list format
delta.se standard errors for delta parameters in list format
lambda.se standard errors for structural parameters of joint attribute distribution
boot.est resample estimates

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References


Examples

## Not run:
# For illustration, only 5 resamples are run
# results are definitely not reliable

dat <- sim30GDINA$simdat
Q <- sim30GDINA$simQ
fit <- GDINA(dat = dat, Q = Q, model = "GDINA", att.dist = "higher.order")
boot.fit <- bootSE(fit,bootsample = 5,randomseed=123)
boot.fit$delta.se
Description
This function calculates test-, pattern- and attribute-level classification accuracy indices based on GDINA estimates from the GDINA function using approaches in Iaconangelo (2017) and Wang, Song, Chen, Meng, and Ding (2015). It is only applicable for dichotomous attributes.

Usage
CA(GDINA.obj, what = "MAP")

Arguments
GDINA.obj estimated GDINA object returned from GDINA
what what attribute estimates are used? Default is "MAP".

Value
a list with elements
  tau estimated test-level classification accuracy, see Iaconangelo (2017, Eq 2.2)
  tau_l estimated pattern-level classification accuracy, see Iaconangelo (2017, p. 13)
  tau_k estimated attribute-level classification accuracy, see Wang, et al (2015, p. 461 Eq 6)
  CCM Conditional classification matrix, see Iaconangelo (2017, p. 13)

Author(s)
Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References


Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
fit
CA(fit)

## End(Not run)
```

---

cjoint

**Combine R Objects by Columns**

Description

Combine a sequence of vector, matrix or data-frame arguments by columns. Vector is treated as a column matrix.

Usage

```r
cjoint(..., fill = NA)
```

Arguments

- `...`: vectors or matrices
- `fill`: a scalar used when these objects have different number of rows.

Value

a data frame

See Also

cbind

Examples

```r
cjoint(2,c(1,2,3,4),matrix(1:6,2,3))
cjoint(v1 = 2, v2 = c(3,2), v3 = matrix(1:6,3,2),
      v4 = data.frame(c(3,4,5,6,7),rep("x",5)),fill = 99)
```
ClassRate

Classification Rate Evaluation

Description

This function evaluates the classification rates for two sets of attribute profiles.

Usage

ClassRate(att1, att2)

Arguments

att1 a matrix or data frame of attribute profiles
att2 a matrix or data frame of attribute profiles

Value

a list with the following components:

PCA the proportion of correctly classified attributes (i.e., attribute level classification rate)
PCV a vector giving the proportions of correctly classified attribute vectors (i.e., vector level classification rate). The first element is the proportion of at least one attribute in the vector are correctly identified; the second element is the proportion of at least two attributes in the vector are correctly identified; and so forth. The last element is the proportion of all elements in the vector are correctly identified.

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References


Examples

```r
## Not run:
N <- 2000
# model does not matter if item parameter is probability of success
Q <- sim30GDINA$simQ
J <- nrow(Q)
gs <- matrix(0.1,J,2)
set.seed(12345)
sim <- simGDINA(N,Q,gs.parm = gs)
GDINA.est <- GDINA(sim$dat,Q)
```
designmatrix

CR <- ClassRate(sim$attribute,personparm(GDINA.est))
CR

## End(Not run)

designmatrix  Generate design matrix

Description
This function generates the design matrix for an item

Usage
designmatrix(Kj = NULL, model = "GDINA", Qj = NULL)

Arguments

Kj  Required except for the MS-DINA model; The number of attributes required for
item j

model the model associated with the design matrix; It can be "GDINA","DINA","DINO",  "ACDM" or "MSDINA". The default is "GDINA". Note that models "LLM" and "RRUM" have the same design matrix as the ACDM.

Qj  the Q-matrix for item j; This is required for "MSDINA" model; The number of
rows is equal to the number of strategies and the number of columns is equal to
the number of attributes.

Value
a design matrix (Mj). See de la Torre (2011) for details.

References

Examples

## Not run:
designmatrix(Kj = 2, model = "GDINA")
designmatrix(Kj = 3, model = "DINA")
msQj <- matrix(c(1,0,0,1, 1,1,0,0),nrow=2,byrow=TRUE)
designmatrix(model = "MSDINA",Qj = msQj)

## End(Not run)
Differential item functioning for cognitive diagnosis models

Description

This function is used to detect differential item functioning using the Wald test (Hou, de la Torre, & Nandakumar, 2014; Ma, Terzi, & de la Torre, 2021) and the likelihood ratio test (Ma, Terzi, & de la Torre, 2021). The forward anchor item search procedure developed in Ma, Terzi, and de la Torre (2021) was implemented. Note that it can only detect DIF for two groups currently.

Usage

dif(
    dat,
    Q,
    group,
    model = "GDINA",
    method = "wald",
    anchor.items = NULL,
    dif.items = "all",
    p.adjust.methods = "holm",
    approx = FALSE,
    SE.type = 2,
    FS.args = list(on = FALSE, alpha.level = 0.05, maxit = 10, verbose = FALSE),
    ...
)

# S3 method for class 'dif'
summary(object, ...)

Arguments

dat item responses from two groups; missing data need to be coded as NA
Q Q-matrix specifying the association between items and attributes
group a factor or a vector indicating the group each individual belongs to. Its length must be equal to the number of individuals.
model model for each item.
method DIF detection method; It can be "wald" for Hou, de la Torre, and Nandakumar’s (2014) Wald test method, and "LR" for likelihood ratio test (Ma, Terzi, Lee, & de la Torre, 2017).
anchor.items which items will be used as anchors? Default is NULL, which means none of the items are used as anchors. For LR method, it can also be an integer vector giving the item numbers for anchors or "all", which means all items are treated as anchor items.
dif.items which items are subject to DIF detection? Default is "all". It can also be an integer vector giving the item numbers.

p.adjust.methods adjusted p-values for multiple hypothesis tests. This is conducted using p.adjust function in stats, and therefore all adjustment methods supported by p.adjust can be used, including "holm", "hochberg", "hommel", "bonferroni", "BH" and "BY". See p.adjust for more details. "holm" is the default.

approx Whether an approximated LR test is implemented? If TRUE, parameters of items except the studied one will not be re-estimated.

SE.type Type of standard error estimation methods for the Wald test.

FS.args arguments for the forward anchor item search procedure developed in Ma, Terzi, and de la Torre (2021). A list with the following elements:

- on - logical; TRUE if activate the forward anchor item search procedure. Default = FALSE.
- alpha.level - nominal level for Wald or LR test. Default = .05.
- maxit - maximum number of iterations allowed. Default = 10.
- verbose - logical; print information for each iteration or not? Default = FALSE.

... arguments passed to GDINA function for model calibration

object dif object for S3 method

Value
A data frame giving the Wald statistics and associated p-values.

Methods (by generic)
- summary: print summary information

Author(s)
Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

References


See Also
GDINA
Examples

```r
## Not run:
set.seed(123456)
N <- 3000
Q <- sim30GDINA$simQ
gs <- matrix(.2, ncol = 2, nrow = nrow(Q))
# By default, individuals are simulated from uniform distribution
# and deltas are simulated randomly
sim1 <- simGDINA(N, Q, gs.parm = gs, model = "DINA")
sim2 <- simGDINA(N, Q, gs.parm = gs, model = c(rep("DINA", nrow(Q) - 1), "DINO"))
dat <- rbind(extract(sim1, "dat"), extract(sim2, "dat"))
gr <- rep(c("G1", "G2"), each = N)

# DIF using Wald test
dif.wald <- dif(dat, Q, group = gr, method = "Wald")
dif.wald
# DIF using LR test
dif.LR <- dif(dat, Q, group = gr, method = "LR")
dif.LR
# DIF using Wald test + forward search algorithm
dif.wald.FS <- dif(dat, Q, group = gr, method = "Wald", FS.args = list(on = TRUE, verbose = TRUE))
dif.wald.FS
# DIF using LR test + forward search algorithm
dif.LR.FS <- dif(dat, Q, group = gr, method = "LR", FS.args = list(on = TRUE, verbose = TRUE))
dif.LR.FS
## End(Not run)
```

---

**DTM**

*Experimental function for diagnostic multiple-strategy CDMs*

**Description**

This function estimates the diagnostic tree model (Ma, 2018) for polytomous responses with multiple strategies. It is an experimental function, and will be further optimized.

**Usage**

```r
DTM(
  dat,
  Qc,
  delta = NULL,
  Tmatrix = NULL,
  conv.crit = 0.001,
  conv.type = "pr",
  maxitr = 1000
)
```
Arguments

**dat**  
A required $N \times J$ data matrix of $N$ examinees to $J$ items. Missing values are currently not allowed.

**Qc**  
A required $J \times K + 2$ category and attribute association matrix, where $J$ represents the number of items or nonzero categories and $K$ represents the number of attributes. Entry 1 indicates that the attribute is measured by the item, and 0 otherwise. The first column gives the item number, which must be numeric and match the number of column in the data. The second column indicates the category number.

**delta**  
initial item parameters

**Tmatrix**  
The mapping matrix showing the relation between the OBSERVED responses (rows) and the PSEUDO items (columns); The first column gives the observed responses.

**conv.crit**  
The convergence criterion for max absolute change in item parameters.

**conv.type**  
convergence criteria; Can be pr, LL and delta, indicating category response function, log-likelihood and delta parameters, respectively.

**maxitr**  
The maximum iterations allowed.

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References


See Also

GDINA for MS-DINA model and single strategy CDMs, and GMSCDM for generalized multiple strategies CDMs for dichotomous response data

Examples

```r
## Not run:
K=5
g=0.2
item.no <- rep(1:6,each=4)
# the first node has three response categories: 0, 1 and 2
node.no <- rep(c(1,1,2,3),6)
Q1 <- matrix(0,length(item.no),K)
Q2 <- cbind(7:(7+K-1),rep(1,K),diag(K))
for(j in 1:length(item.no)) {
  Q1[,sample(1:K,sample(3,1))] <- 1
}
Qc <- rbind(cbind(item.no,node.no,Q1),Q2)
Tmatrix.set <- list(cbind(c(0,1,2,3,3),c(0,1,2,1,2),c(NA,0,NA,1,NA),c(NA,NA,0,NA,1)),
cbind(c(0,1,2,3,4),c(0,1,2,1,2),c(NA,0,NA,1,NA),c(NA,NA,0,NA,1)))
```
Examination for the Certificate of Proficiency in English (ECPE) data

Description

Examination for the Certificate of Proficiency in English (ECPE) data (the grammar section) has been used in Henson and Templin (2007), Templin and Hoffman (2013), Feng, Habing, and Huebner (2014), and Templin and Bradshaw (2014), among others.

Usage

ecope

Format

A list of responses and Q-matrix with components:

dat Responses of 2922 examinees to 28 items.
Q The 28 × 3 Q-matrix.

Details

The data consists of responses of 2922 examinees to 28 items involving 3 attributes. Attribute 1 is morphosyntactic rules, Attribute 2 is cohesive rules and Attribute 3 is lexical rules.

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References

## Examples

```r
## Not run:
mod1 <- GDINA(ecpe$dat,ecpe$Q)
mod1
summary(mod1)

mod2 <- GDINA(ecpe$dat,ecpe$Q,model="RRUM")
mod2
anova(mod1,mod2)
# You may compare the following results with Feng, Habing, and Huebner (2014)
coef(mod2,"rrum")

# G-DINA with hierarchical structure
# see Templin & Bradshaw, 2014
ast <- att.structure(list(c(3,2),c(2,1)),K=3)
est.gdina2 <- GDINA(ecpe$dat,ecpe$Q,model = "GDINA",
control = list(conv.crit = 1e-6),
att.str = list(c(3,2),c(2,1)))
# see Table 7 in Templin & Bradshaw, 2014
summary(est.gdina2)

## End(Not run)
```

### Description

A generic function to extract elements from objects of class GDINA, itemfit, modelcomp, Qval or simGDINA. This page gives the elements that can be extracted from the class GDINA. To see what can be extracted from itemfit, modelcomp, and Qval, go to the corresponding function help page.

Objects which can be extracted from GDINA objects include:

- **AIC**: AIC
- **att.prior**: attribute prior weights for calculating marginalized likelihood in the last EM iteration
- **attributepattern**: all attribute patterns involved in the current calibration
- **BIC**: BIC
- **CAIC**: Consistent AIC
- **catprob.cov**: covariance matrix of item probability parameter estimates; Need to specify `SE.type`
- **catprob.parm**: item parameter estimates
- **catprob.se**: standard error of item probability parameter estimates; Need to specify `SE.type`
- **convergence**: TRUE if the calibration is converged.
- **dat**: raw data
del.ind  deleted observation number  
delta.cov  covariance matrix of delta parameter estimates; Need to specify SE.type  
delta.parm  delta parameter estimates  
delta.se  standard error of delta parameter estimates; Need to specify SE.type  
designmatrix  A list of design matrices for each item/category  
deviance  deviance, or negative two times observed marginal log likelihood  
discrim  GDINA discrimination index  
expectedCorrect  expected # of examinees in each latent group answering item correctly  
expectedTotal  expected # of examinees in each latent group  
higher.order  higher-order model specifications  
LCprob.parm  success probabilities for all latent classes  
logLik  observed marginal log likelihood  
linkfunc  link functions for each item  
initial.catprob  initial item category probability parameters  
natt  number of attributes  
ncat  number of categories  
group  number of groups  
nitem  number of items  
nitr  number of EM iterations  
nobs  number of observations, or sample size  
nLC  number of latent classes  
prevalence  prevalence of each attribute  
posterior.prob  posterior weights for each latent class  
reduced.LG  Reduced latent group for each item  
SABIC  Sample size Adjusted BIC  
sequential  is a sequential model fitted?

Usage

extract(object, what, ...)  

Arguments

object  objects from class GDINA.itemfit, modelcomp, Qval or simGDINA  
what  what to extract  
...  additional arguments
Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
extract(fit,"discrim")
extract(fit,"designmatrix")
## End(Not run)
```

frac20  

### Tatsuoka's fraction subtraction data

Description

Fraction Subtraction data (Tatsuoka, 2002) consists of responses of 536 examinees to 20 items measuring 8 attributes.

Usage

frac20

Format

A list of responses and Q-matrix with components:

- `dat`: responses of 536 examinees to 20 items
- `Q`: The $20 \times 8$ Q-matrix

Author(s)

Wenchao Ma, The University of Alabama. <wenchao.ma@ua.edu>

References


Examples

```r
## Not run:
mod1 <- GDINA(frac20$dat,frac20$Q,model="DINA")
mod1
summary(mod1)
# Higher order model
mod2 <- GDINA(frac20$dat,frac20$Q,model="DINA",att.dist="higher.order")
mod2
anova(mod1,mod2)
## End(Not run)
```

GDINA

*CDM calibration under the G-DINA model framework*

Description

GDINA calibrates the generalized deterministic inputs, noisy and gate (G-DINA; de la Torre, 2011) model for dichotomous responses, and its extension, the sequential G-DINA model (Ma, & de la Torre, 2016a; Ma, 2017) for ordinal and nominal responses. By setting appropriate constraints, the deterministic inputs, noisy and gate (DINA; de la Torre, 2009; Junker & Sijtsma, 2001) model, the deterministic inputs, noisy or gate (DINO; Templin & Henson, 2006) model, the reduced reparametrized unified model (R-RUM; Hartz, 2002), the additive CDM (A-CDM; de la Torre, 2011), the linear logistic model (LLM; Maris, 1999), and the multiple-strategy DINA model (MS-DINA; de la Torre & Douglas, 2008; Huo & de la Torre, 2014) can also be calibrated. Note that the LLM is equivalent to the C-RUM (Hartz, 2002), a special case of the GDM (von Davier, 2008), and that the R-RUM is also known as a special case of the generalized NIDA model (de la Torre, 2011).

In addition, users are allowed to specify design matrix and link function for each item, and distinct models may be used in a single test for different items. The attributes can be either dichotomous or polytomous (Chen & de la Torre, 2013). Joint attribute distribution may be modelled using independent or saturated model, structured model, higher-order model (de la Torre & Douglas, 2004), or loglinear model (Xu & von Davier, 2008). Marginal maximum likelihood method with Expectation-Maximization (MMLE/EM) algorithm is used for item parameter estimation.

To compare two or more GDINA objects, use method `anova`.

To calculate structural parameters for item and joint attribute distributions, use method `coef`.

To calculate lower-order incidental (person) parameters use method `personparm`. To extract other components returned, use `extract`. To plot item/category response function, use `plot`. To check whether monotonicity is violated, use `monocheck`. To conduct analysis in graphical user interface, use `startGDINA`.

Usage

```r
GDINA(
  dat,  
  Q,  
  model = "GDINA",
```
sequential = FALSE,
att.dist = "saturated",
mono.constraint = FALSE,
group = NULL,
linkfunc = NULL,
design.matrix = NULL,
latent.var = "att",
att.prior = NULL,
att.str = NULL,
verbose = 1,
higher.order = list(),
loglinear = 2,
catprob.parm = NULL,
control = list(),
item.names = NULL,
solver = NULL,
nloptr.args = list(),
auglag.args = list(),
solnp.args = list(),
...)

## S3 method for class 'GDINA'
anova(object, ...)

## S3 method for class 'GDINA'
coef(
 object,
 what = c("catprob", "delta", "gs", "itemprob", "LCprob", "rrum", "lambda"),
 withSE = FALSE,
 SE.type = 2,
 digits = 4,
 ...)

## S3 method for class 'GDINA'
extract(object, what, SE.type = 2, ...)

## S3 method for class 'GDINA'
personparm(object, what = c("EAP", "MAP", "MLE", "mp", "HO"), digits = 4, ...)

## S3 method for class 'GDINA'
logLik(object, ...)

## S3 method for class 'GDINA'
deviance(object, ...)

## S3 method for class 'GDINA'
nobs(object, ...)  
## S3 method for class 'GDINA'
vcov(object, ...)  
## S3 method for class 'GDINA'
npar(object, ...)  
## S3 method for class 'GDINA'
indlogLik(object, ...)  
## S3 method for class 'GDINA'
indlogPost(object, ...)  
## S3 method for class 'GDINA'
summary(object, ...)

Arguments

dat  A required $N \times J$ matrix or data.frame consisting of the responses of $N$ individuals to $J$ items. Missing values need to be coded as NA.

Q  A required matrix; The number of rows occupied by a single-strategy dichotomous item is 1, by a polytomous item is the number of nonzero categories, and by a multiple-strategy dichotomous item is the number of strategies. The number of column is equal to the number of attributes if all items are single-strategy dichotomous items, but the number of attributes + 2 if any items are polytomous or have multiple strategies. For a polytomous item, the first column represents the item number and the second column indicates the nonzero category number. For a multiple-strategy dichotomous item, the first column represents the item number and the second column indicates the strategy number. For binary attributes, 1 denotes the attributes are measured by the items and 0 means the attributes are not measured. For polytomous attributes, non-zero elements indicate which level of attributes are needed (see Chen, & de la Torre, 2013). See Examples.

model  A vector for each item or nonzero category, or a scalar which will be used for all items or nonzero categories to specify the CDMs fitted. The possible options include "GDINA","DINA","DINO","ACDM","LLM","RRUM","MSDINA" and "UDF". Note that model can also be "logitGDINA" and "logGDINA", indicating the saturated G-DINA model in logit and log link functions. They are equivalent to the identity link saturated G-DINA model. The logit G-DINA model is identical to the log-linear CDM. When "UDF", indicating user defined function, is specified for any item, arguments design.matrix and linkfunc need to be defined.

sequential  logical; TRUE if the sequential model is fitted for polytomous responses.

att.dist  How is the joint attribute distribution estimated? It can be (1) saturated, which is the default, indicating that the proportion parameter for each permissible latent class is estimated separately; (2) higher.order, indicating that a higher-order joint attribute distribution is assumed (higher-order model can be specified in higher.order argument); (3) fixed, indicating that the weights speci-
A vector of length $2^K$ for single group model, or a matrix of dimension $2^K \times$ no. of groups to specify attribute prior distribution for $2^K$ latent classes for all groups under a multiple group model. Only applicable for dichotomous attributes. The sum of all elements does not have to be equal to 1; however, it will be normalized so that the sum is equal to 1 before calibration. The label for each latent class can be obtained by calling `attributepattern(K)`. See examples for more info.

**att.str**
Specify attribute structures. NULL, by default, means there is no structure. Attribute structure needs be specified as a list - which will be internally handled by `att.structure` function. See examples. It can also be a matrix giving all permissible attribute profiles.

**verbose**
How to print calibration information after each EM iteration? Can be 0, 1 or 2, indicating to print no information, information for current iteration, or information for all iterations.

**higher.order**
A list specifying the higher-order joint attribute distribution with the following components:

- **model** - a character indicating the IRT model for higher-order joint attribute distribution. Can be "2PL", "1PL" or "Rasch", representing two parameter logistic IRT model, one parameter logistic IRT model and Rasch model, respectively. For "1PL" model, a common slope parameter is estimated. "Rasch" is the default model when `att.dist = "higher.order"`. Note that slope-intercept form is used for parameterizing the higher-order IRT model (see Details).
• nquad - a scalar specifying the number of integral nodes. Default = 25.
• SlopeRange - a vector of length two specifying the range of slope parameters. Default = [0.1, 5].
• InterceptRange - a vector of length two specifying the range of intercept parameters. Default = [-4, 4].
• SlopePrior - a vector of length two specifying the mean and variance of log(slope) parameters, which are assumed normally distributed. Default: mean = 0 and sd = 0.25.
• InterceptPrior - a vector of length two specifying the mean and variance of intercept parameters, which are assumed normally distributed. Default: mean = 0 and sd = 1.
• Prior - logical; indicating whether prior distributions should be imposed to slope and intercept parameters. Default is FALSE.

loglinear the order of loglinear smooth for attribute space. It can be either 1 or 2 indicating the loglinear model with main effect only and with main effect and first-order interaction.

catprob.parm A list of initial success probability parameters for each nonzero category.

control A list of control parameters with elements:
• maxitr A vector for each item or nonzero category, or a scalar which will be used for all items or nonzero categories to specify the maximum number of EM cycles allowed. Default = 2000.
• conv.crit The convergence criterion. Default = 0.0001.
• conv.type How is the convergence criterion evaluated? A vector with possible elements: "ip", indicating the maximum absolute change in item success probabilities, "mp", representing the maximum absolute change in mixing proportion parameters, "delta", indicating the maximum absolute change in delta parameters, neg2LL indicating the absolute change in negative two times loglikelihood, or neg2LL indicating the relative absolute change in negative two times loglikelihood (i.e., the absolute change divided by -2LL of the previous iteration). Multiple criteria can be specified. If so, all criteria need to be met. Default = c("ip", "mp").
• nstarts how many sets of starting values? Default = 3.
• lower.p A vector for each item or nonzero category, or a scalar which will be used for all items or nonzero categories to specify the lower bound for success probabilities. Default = .0001.
• upper.p A vector for each item or nonzero category, or a scalar which will be used for all items or nonzero categories to specify the upper bound for success probabilities. Default = .9999.
• lower.prior The lower bound for mixing proportion parameters (latent class sizes). Default = .Machine$double.eps.
• randomseed Random seed for generating initial item parameters. Default = 123456.
• smallNcorrection A numeric vector with two elements specifying the corrections applied when the expected number of individuals in some latent groups are too small. If the expected no. of examinees is less than
the second element, the first element and two times the first element will be added to the numerator and denominator of the closed-form solution of probabilities of success. Only applicable for the G-DINA, DINA and DINO model estimation without monotonic constraints.

- **MstepMessage**: Integer; Larger number prints more information from Mstep optimizer. Default = 1.

**item.names** A vector giving the item names. By default, items are named as "Item 1", "Item 2", etc.

**solver** A string indicating which solver should be used in M-step. By default, the solver is automatically chosen according to the models specified. Possible options include slsqp, nloptr, solnp and auglag.

**nloptr.args** a list of control parameters to be passed to opts argument of nloptr function.

**auglag.args** a list of control parameters to be passed to the alabama::auglag() function. It can contain two elements: control.outer and control.optim. See auglag.

**solnp.args** a list of control parameters to be passed to control argument of solnp function.

**...** additional arguments

**object** GDINA object for various S3 methods

**what** argument for various S3 methods; For calculating structural parameters using coef, what can be

- **itemprob** - item success probabilities of each reduced attribute pattern.
- **catprob** - category success probabilities of each reduced attribute pattern; the same as itemprob for dichotomous response data.
- **LCprob** - item success probabilities of each attribute pattern.
- **gs** - guessing and slip parameters of each item/category.
- **delta** - delta parameters of each item/category, see G-DINA formula in details.
- **rrum** - RRUM parameters when items are estimated using RRUM.
- **lambda** - structural parameters for joint attribute distribution.

For calculating incidental parameters using personparm, what can be

- **EAP** - EAP estimates of attribute pattern.
- **MAP** - MAP estimates of attribute pattern.
- **MLE** - MLE estimates of attribute pattern.
- **mp** - marginal mastery probabilities.
- **HO** - EAP estimates of higher-order ability if a higher-order is fitted.

**withSE** argument for method coef; estimate standard errors or not?

**SE.type** type of standard errors. For now, SEs are calculated based on outer-product of gradient. It can be 1 based on item-wise information, 2 based on incomplete information and 3 based on complete information.

**digits** How many decimal places in each number? The default is 4.
Value

GDINA returns an object of class GDINA. Methods for GDINA objects include extract for extracting various components, coef for extracting structural parameters, personparm for calculating incidental (person) parameters, summary for summary information. AIC, BIC, logLik, deviance and npar can also be used to calculate AIC, BIC, observed log-likelihood, deviance and number of parameters.

Methods (by generic)

- anova: Model comparison using likelihood ratio test
- coef: extract structural parameter estimates
- extract: extract various elements of GDINA estimates
- personparm: calculate person attribute patterns and higher-order ability
- logLik: calculate log-likelihood
- deviance: calculate deviance
- nobs: calculate number of observations
- vcov: calculate covariance-matrix for delta parameters
- npar: calculate the number of parameters
- indlogLik: extract log-likelihood for each individual
- indlogPost: extract log posterior for each individual
- summary: print summary information

The G-DINA model

The generalized DINA model (G-DINA; de la Torre, 2011) is an extension of the DINA model. Unlike the DINA model, which collapses all latent classes into two latent groups for each item, if item \( j \) requires \( K_j^* \) attributes, the G-DINA model collapses \( 2^K \) latent classes into \( 2^{K_j^*} \) latent groups with unique success probabilities on item \( j \), where \( K_j^* = \sum_{k=1}^{K} q_{jk} \).

Let \( \alpha_{lj}^* \) be the reduced attribute pattern consisting of the columns of the attributes required by item \( j \), where \( l = 1, \ldots, 2^{K_j^*} \). For example, if only the first and the last attributes are required, \( \alpha_{lj}^* = (\alpha_{l1}, \alpha_{lK}) \). For notational convenience, the first \( K_j^* \) attributes can be assumed to be the required attributes for item \( j \) as in de la Torre (2011). The probability of success \( P(X_j = 1|\alpha_{lj}^*) \) is denoted by \( P(\alpha_{lj}^*) \). To model this probability of success, different link functions as in the generalized linear models are used in the G-DINA model. The item response function of the G-DINA model using the identity link can be written as

\[
f[P(\alpha_{lj}^*)] = \delta_{j0} + \sum_{k=1}^{K_j^*} \delta_{jk} \alpha_{lk} + \sum_{k'=k+1}^{K_j^*} \sum_{k=1}^{K_j^*-1} \delta_{jkk'} \alpha_{lk} \alpha_{lk'} + \cdots + \delta_{j12\cdots K_j^*} \prod_{k=1}^{K_j^*} \alpha_{lk},
\]

or in matrix form,

\[
f[P_j] = M_j \delta_j,
\]

where \( \delta_{j0} \) is the intercept for item \( j \), \( \delta_{jk} \) is the main effect due to \( \alpha_{lk} \), \( \delta_{jkk'} \) is the interaction effect due to \( \alpha_{lk} \) and \( \alpha_{lk'} \), \( \delta_{j12\cdots K_j^*} \) is the interaction effect due to \( \alpha_{l1}, \cdots, \alpha_{lK_j^*} \). The log and logit links can also be employed.
Other CDMs as special cases

Several widely used CDMs can be obtained by setting appropriate constraints to the G-DINA model. This section introduces the parameterization of different CDMs within the G-DINA model framework very briefly. Readers interested in this please refer to de la Torre (2011) for details.

DINA model In DINA model, each item has two item parameters - guessing ($g$) and slip ($s$). In traditional parameterization of the DINA model, a latent variable $\eta$ for person $i$ and item $j$ is defined as

$$
\eta_{ij} = \prod_{k=1}^{K} \alpha_{ik}^{q_{ik}}
$$

Briefly speaking, if individual $i$ master all attributes required by item $j$, $\eta_{ij} = 1$; otherwise, $\eta_{ij} = 0$. Item response function of the DINA model can be written by

$$
P(X_{ij} = 1 | \eta_{ij}) = (1 - s_j)^{\eta_{ij}} g_j^{1 - \eta_{ij}}
$$

To obtain the DINA model from the G-DINA model, all terms in identity link G-DINA model except $\delta_{j0}$ and $\delta_{j12\cdots K_j^*}$ need to be fixed to zero, that is,

$$
P(\alpha_{ij}^*) = \delta_{j0} + \delta_{j12\cdots K_j^*} \prod_{k=1}^{K_j^*} \alpha_{lk}
$$

In this parameterization, $\delta_{j0} = g_j$ and $\delta_{j0} + \delta_{j12\cdots K_j^*} = 1 - s_j$.

DINO model The DINO model can be given by

$$
P(\alpha_{ij}^*) = \delta_{j0} + \delta_{j1} I(\alpha_{ij}^* \neq 0)
$$

where $I(\cdot)$ is an indicator variable. The DINO model is also a constrained identity link G-DINA model. As shown by de la Torre (2011), the appropriate constraint is

$$
\delta_{jk} = -\delta_{jk'} \delta_{k''} = \cdots = (-1)^{K_j^*+1} \delta_{j12\cdots K_j^*},
$$

for $k = 1, \cdots, K_j^*, k' = 1, \cdots, K_j^* - 1$, and $k'' > k', \cdots, K_j^*$.

Additive models with different link functions The A-CDM, LLM and R-RUM can be obtained by setting all interactions to be zero in identity, logit and log link G-DINA model, respectively. Specifically, the A-CDM can be formulated as

$$
P(\alpha_{ij}^*) = \delta_{j0} + \sum_{k=1}^{K_j^*} \delta_{jk} \alpha_{lk}.
$$

The item response function for LLM can be given by

$$
\text{logit}[P(\alpha_{ij}^*)] = \delta_{j0} + \sum_{k=1}^{K_j^*} \delta_{jk} \alpha_{lk},
$$

and lastly, the RRUM, can be written as

$$
\text{log}[P(\alpha_{ij}^*)] = \delta_{j0} + \sum_{k=1}^{K_j^*} \delta_{jk} \alpha_{lk}.
$$

It should be noted that the LLM is equivalent to the compensatory RUM, which is subsumed by the GDM, and that the RRUM is a special case of the generalized noisy inputs, deterministic “And” gate model (G-NIDA).
Joint Attribute Distribution

The joint attribute distribution can be modeled using various methods. This section mainly focuses on the so-called higher-order approach, which was originally proposed by de la Torre and Douglas (2004) for the DINA model. It has been extended in this package for all condensation rules. Particularly, three IRT models are available for the higher-order attribute structure: Rasch model (Rasch), one parameter logistic model (1PL) and two parameter logistic model (2PL). For the Rasch model, the probability of mastering attribute $k$ for individual $i$ is defined as

$$P(\alpha_k = 1|\theta_i, \lambda_{0k}) = \frac{\exp(\theta_i + \lambda_{0k})}{1 + \exp(\theta_i + \lambda_{0k})}$$

For the 1PL model, the probability of mastering attribute $k$ for individual $i$ is defined as

$$P(\alpha_k = 1|\theta_i, \lambda_{0k}, \lambda_1) = \frac{\exp(\lambda_1 \theta_i + \lambda_{0k})}{1 + \exp(\lambda_1 \theta_i + \lambda_{0k})}$$

For the 2PL model, the probability of mastering attribute $k$ for individual $i$ is defined as

$$P(\alpha_k = 1|\theta_i, \lambda_{0k}, \lambda_{1k}) = \frac{\exp(\lambda_{1k} \theta_i + \lambda_{0k})}{1 + \exp(\lambda_{1k} \theta_i + \lambda_{0k})}$$

where $\theta_i$ is the ability of examinee $i$. $\lambda_{0k}$ and $\lambda_{1k}$ are the intercept and slope parameters for attribute $k$, respectively. In the Rasch model, $\lambda_{1k} = 1\forall k$; whereas in the 1PL model, a common slope parameter $\lambda_1$ is estimated. The probability of joint attributes can be written as

$$P(\alpha|\theta, \lambda) = \prod_k P(\alpha_k|\theta, \lambda)$$

Model Estimation

The MMLE/EM algorithm is implemented in this package. For G-DINA, DINA and DINO models, closed-form solutions exist. See de la Torre (2009) and de la Torre (2011) for details. For ACDM, LLM and RRUM, closed-form solutions do not exist, and therefore some general optimization techniques are adopted in M-step (Ma, Iaconangelo & de la Torre, 2016). The selection of optimization techniques mainly depends on whether some specific constraints need to be added.

The sequential G-DINA model is a special case of the diagnostic tree model (DTM; Ma, 2019) and estimated using the mapping matrix accordingly (See Tutz, 1997; Ma, 2019).

The Number of Parameters

For dichotomous response models: Assume a test measures $K$ attributes and item $j$ requires $K_j^*$ attributes: The DINA and DINO model has 2 item parameters for each item; if item $j$ is ACDM, LLM or RRUM, it has $K_j^* + 1$ item parameters; if it is G-DINA model, it has $2K_j^*$ item parameters. Apart from item parameters, the parameters involved in the estimation of joint attribute distribution need to be estimated as well. When using the saturated attribute structure, there are $2^K - 1$ parameters for joint attribute distribution estimation; when using a higher-order attribute structure, there are $K$, $K + 1$, and $2 \times K$ parameters for the Rasch model, 1PL model and 2PL model, respectively. For polytomous response data using the sequential G-DINA model, the number of item parameters are counted at category level.
Note
anova function does NOT check whether models compared are nested or not.

Author(s)
Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

References


### See Also

See `autoGDINA` for Q-matrix validation, item-level model comparison and model calibration in one run; See `modelfit` and `itemfit` for model and item fit analysis, `Qval` for Q-matrix validation, `modelcomp` for item level model comparison and `simGDINA` for data simulation. `GMSCDM` for a series of multiple strategy CDMs for dichotomous data, and `DTM` for diagnostic tree model for multiple strategies in polytomous response data Also see `gdina` in `CDM` package for the G-DINA model estimation.

### Examples

```r
## Not run:
####################################
# Example 1. #
# GDINA, DINA, DINO #
# ACDM, LLM and RRUM #
# estimation and comparison #
#
# GDINA model --------#

dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ

#---------GDINA model --------#
mod1 <- GDINA(dat = dat, Q = Q, model = "GDINA")
mod1
# summary information
summary(mod1)

AIC(mod1) #AIC
BIC(mod1) #BIC
logLik(mod1) #log-likelihood value
```

---


### See Also

See `autoGDINA` for Q-matrix validation, item-level model comparison and model calibration in one run; See `modelfit` and `itemfit` for model and item fit analysis, `Qval` for Q-matrix validation, `modelcomp` for item level model comparison and `simGDINA` for data simulation. `GMSCDM` for a series of multiple strategy CDMs for dichotomous data, and `DTM` for diagnostic tree model for multiple strategies in polytomous response data Also see `gdina` in `CDM` package for the G-DINA model estimation.

### Examples

```r
## Not run:
####################################
# Example 1. #
# GDINA, DINA, DINO #
# ACDM, LLM and RRUM #
# estimation and comparison #
#
# GDINA model --------#

dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ

#---------GDINA model --------#
mod1 <- GDINA(dat = dat, Q = Q, model = "GDINA")
mod1
# summary information
summary(mod1)

AIC(mod1) #AIC
BIC(mod1) #BIC
logLik(mod1) #log-likelihood value
```
deviance(mod1) # deviance: -2 log-likelihood
npar(mod1) # number of parameters

head(indlogLik(mod1)) # individual log-likelihood
head(indlogPost(mod1)) # individual log-posterior

# structural parameters
# see ?coef
coef(mod1) # item probabilities of success for each latent group
coef(mod1, what = "delta") # delta parameters
coef(mod1, what = "gs", withSE = TRUE) # guessing and slip parameters

# person parameters
# see ?personparm
personparm(mod1) # EAP estimates of attribute profiles
personparm(mod1, what = "MAP") # MAP estimates of attribute profiles
personparm(mod1, what = "MLE") # MLE estimates of attribute profiles

#plot item response functions for item 10
plot(mod1, item = 10)
plot(mod1, item = 10, withSE = TRUE) # with error bars

#plot mastery probability for individuals 1, 20 and 50
plot(mod1, what = "mp", person = c(1, 20, 50))

# Use extract function to extract more components
# See ?extract

# ------- DINA model -------#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod2 <- GDINA(dat = dat, Q = Q, model = "DINA")

coef(mod2, what = "gs") # guess and slip parameters
coef(mod2, what = "gs", withSE = TRUE) # guess and slip parameters + standard errors

# Model comparison at the test level via likelihood ratio test
anova(mod1, mod2)

# ------- DINO model -------#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod3 <- GDINA(dat = dat, Q = Q, model = "DINO")

coef(mod3, what = "gs") # guess and slip parameters
coef(mod3, what = "gs", withSE = TRUE) # guess and slip parameters + standard errors

# Model comparison at the test level via likelihood ratio test
anova(mod1, mod2, mod3)
# --------- ACDM model -------#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod4 <- GDINA(dat = dat, Q = Q, model = "ACDM")
mod4

# --------- LLM model -------#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod4b <- GDINA(dat = dat, Q = Q, model = "LLM")
mod4b

# --------- RRUM model -------#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod4c <- GDINA(dat = dat, Q = Q, model = "RRUM")
mod4c

# --- Different CDMs for different items --- #
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
models <- c(rep("GDINA",3),"LLM","DINA","DINO","ACDM","RRUM","LLM","RRUM")
mod5 <- GDINA(dat = dat, Q = Q, model = models)
anova(mod1,mod2,mod3,mod4,mod4b,mod4c,mod5)

# Example 2. 
# Model estimations 
# With monotonocity constraints 
#******************************************************************************
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
# for item 10 only
mod11 <- GDINA(dat = dat, Q = Q, model = "GDINA",mono.constraint = c(rep(FALSE,9),TRUE))
mod11
mod11a <- GDINA(dat = dat, Q = Q, model = "DINA",mono.constraint = TRUE)
mod11a
mod11b <- GDINA(dat = dat, Q = Q, model = "ACDM",mono.constraint = TRUE)
mod11b
mod11c <- GDINA(dat = dat, Q = Q, model = "LLM",mono.constraint = TRUE)
mod11c
mod11d <- GDINA(dat = dat, Q = Q, model = "RRUM",mono.constraint = TRUE)
mod11d
ccoef(mod11d,"delta")
ccoef(mod11d,"rrum")

# Example 3a. 
# Model estimations 
# With Higher-order att structure 
#******************************************************************************
GDINA

```
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
# --- Higher order G-DINA model ---#
mod12 <- GDINA(dat = dat, Q = Q, model = "DINA",
               att.dist="higher.order",higher.order=list(nquad=31,model = "2PL"))
personparm(mod12,"HO") # higher-order ability
# structural parameters
# first column is slope and the second column is intercept
coef(mod12,"lambda")
# --- Higher order DINA model ---#
mod22 <- GDINA(dat = dat, Q = Q, model = "DINA", att.dist="higher.order",
               higher.order=list(model = "2PL",Prior=TRUE))

# Example 3b.
# Model estimations
# With log-linear att structure
# Example 3c.
# With independent att structure
# --- GDINA model with independent attribute space ---#
mod33 <- GDINA(dat = dat, Q = Q, att.dist="independent")
coef(mod33,"lambda") # mastery probability for each attribute

# Example 4.
# With fixed att structure
# --- User-specified attribute priors ----#
prior <- c(0.1,0.1,0.1,0.1,0.15,0.15,0.15,0.15)
fit GDINA model with fixed prior dist.
```

```
modp1 <- GDINA(dat = dat, Q = Q, att.prior = prior, att.dist = "fixed")
extract(modp1, what = "att.prior")

####################################
# Example 5a. #
# G-DINA #
# with hierarchical att structure #
####################################

# --- User-specified attribute structure ----#
Q <- sim30GDINA$simQ
K <- ncol(Q)
# divergent structure A1->A2->A3;A1->A4->A5
diverg <- list(c(1,2),
c(2,3),
c(1,4),
c(4,5))
struc <- att.structure(diverg,K)
set.seed(123)
# data simulation
N <- 1000
true.lc <- sample(c(1:2^K),N,replace=TRUE,prob=struc$att.prob)
table(true.lc) #check the sample
true.att <- attributepattern(K)[true.lc,]
gs <- matrix(rep(0.1,2*nrow(Q)),ncol=2)
# data simulation
simD <- simGDINA(N,Q,gs.parm = gs, model = "GDINA",attribute = true.att)
dat <- extract(simD,"dat")
modp1 <- GDINA(dat = dat, Q = Q, att.str = diverg, att.dist = "saturated")
modp1
c coef(modp1,"lambda")

####################################
# Example 5b. #
# Reduced model (e.g.,ACDM) #
# with hierarchical att structure #
####################################

# --- User-specified attribute structure ----#
Q <- sim30GDINA$simQ
K <- ncol(Q)
# linear structure A1->A2->A3->A4->A5
linear <- list(c(1,2),
c(2,3),
c(3,4),
c(4,5))
struc <- att.structure(linear,K)
set.seed(123)
# data simulation
N <- 1000
true.lc <- sample(c(1:2^K),N,replace=TRUE,prob=struc$att.prob)
table(true.lc) #check the sample
true.att <- attributepattern(K)[true.lc,]
gs <- matrix(rep(0.1,2*nrow(Q)),ncol=2)
  # data simulation
  simD <- simGDINA(N,Q,gs.parm = gs, model = "ACDM",attribute = true.att)
dat <- extract(simD,"dat")

modp1 <- GDINA(dat = dat, Q = Q, model = "ACDM",
               att.str = linear, att.dist = "saturated")
coef(modp1)
coef(modp1,"lambda")

# Example 6.
# Specify initial values for item parameters
#
# check initials to see the format for initial item parameters
initials <- sim10GDINA$simItempar
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod.initial <- GDINA(dat,Q,catprob.parm = initials)

# compare initial item parameters
Map(rbind, initials,extract(mod.initial,"initial.catprob"))

# Example 7a.
# Fix item and structure parameters
# Estimate person attribute profile

# check initials to see the format for initial item parameters
initials <- sim10GDINA$simItempar
prior <- c(0.1,0.1,0.1,0.1,0.1,0.15,0.15,0.15)
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod.ini <- GDINA(dat,Q,catprob.parm = initials,att.prior = prior,
                 att.dist = "fixed",control=list(maxitr = 0))

personparm(mod.ini)
  # compare item parameters
  Map(rbind, initials,coef(mod.ini))

# Example 7b.
# Fix parameters for some items
# Estimate person attribute profile

# check initials to see the format for initial item parameters
initials <- sim10GDINA$simItempar
prior <- c(0.1,0.1,0.1,0.1,0.1,0.15,0.15,0.15)
```r
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
# fix parameters of the first 5 items; do not fix mixing proportion parameters
mod.ini <- GDINA(dat, Q, catprob.parm = initials,
        att.dist = "saturated", control = list(maxitr = c(rep(0, 5), rep(2000, 5))))
personparm(mod.ini)
# compare item parameters
Map(rbind, initials, coef(mod.ini))

# Example 8.
# polytomous attribute
# model estimation
# see Chen, de la Torre 2013
#
# --- polytomous attribute G-DINA model --- #
dat <- sim30pGDINA$simdat
Q <- sim30pGDINA$simQ
# polytomous G-DINA model
pout <- GDINA(dat, Q)

# ----- polytomous DINa model ------#
pout2 <- GDINA(dat, Q, model = "DINA")
anova(pout, pout2)

# Example 9.
# Sequential G-DINA model
# see Ma, & de la Torre 2016
#
# --- polytomous attribute G-DINA model --- #
dat <- sim20seqGDINA$simdat
Q <- sim20seqGDINA$simQ
Q
# Item Cat A1 A2 A3 A4 A5
# 1 1 1 0 0 0 0
# 1 2 0 1 0 0 0
# 2 1 0 0 1 0 0
# 2 2 0 0 0 1 0
# 3 1 0 0 0 1
# 3 2 1 0 0 0
# 4 1 0 0 0 1
# ...

# sequential G-DINA model
sGDINA <- GDINA(dat, Q, sequential = TRUE)
sDINA <- GDINA(dat, Q, sequential = TRUE, model = "DINA")
anova(sGDINA, sDINA)
coef(sDINA) # processing function
coef(sDINA, "itemprob") # success probabilities for each item
```
coef(sDINA,"LCprob") # success probabilities for each category for all latent classes

# Example 10a. #
# Multiple-Group G-DINA model #

Q <- sim10GDINA$simQ
K <- ncol(Q)
# parameter simulation
# Group 1 - female
N1 <- 3000
gs1 <- matrix(rep(0.1,2*nrow(Q)),ncol=2)
# Group 2 - male
N2 <- 3000
gs2 <- matrix(rep(0.2,2*nrow(Q)),ncol=2)

# data simulation for each group
sim1 <- simGDINA(N1,Q,gs.parm = gs1,model = "DINA",att.dist = "higher.order",
                 higher.order.parm = list(theta = rnorm(N1),
                                 lambda = data.frame(a=rep(1.5,K),b=seq(-1,1,length.out=K))))
sim2 <- simGDINA(N2,Q,gs.parm = gs2,model = "DINO",att.dist = "higher.order",
                 higher.order.parm = list(theta = rnorm(N2),
                                 lambda = data.frame(a=rep(1,K),b=seq(-2,2,length.out=K))))

data <- rbind(extract(sim1,"dat"),extract(sim2,"dat"))
gr <- rep(c(1,2),c(3000,3000))

# Fit G-DINA model
mg.est <- GDINA(dat = data,Q = Q,group = gr)
summary(mg.est)
extract(mg.est,"posterior.prob")
coef(mg.est,"lambda")

# Example 10b. #
# Multiple-Group G-DINA model #

Q <- sim30GDINA$simQ
K <- ncol(Q)
# parameter simulation
N1 <- 3000
gs1 <- matrix(rep(0.1,2*nrow(Q)),ncol=2)
N2 <- 3000
gs2 <- matrix(rep(0.2,2*nrow(Q)),ncol=2)

# data simulation for each group
# two groups have different theta distributions
sim1 <- simGDINA(N1,Q,gs.parm = gs1,model = "DINA",att.dist = "higher.order",
                 higher.order.parm = list(theta = rnorm(N1),
                                 lambda = data.frame(a=rep(1,K),b=seq(-2,2,length.out=K))))
sim2 <- simGDINA(N2,Q,gs.parm = gs2,model = "DINO",att.dist = "higher.order",
    higher.order.parm = list(theta = rnorm(N2,1,1), lambda = data.frame(a=rep(1,K),b=seq(-2,2,length.out=K))))

# combine data - different groups have distinct item parameters
# see ?bdiagMatrix
dat <- bdiagMatrix(list(extract(sim1,"dat"),extract(sim2,"dat")),fill=NA)
Q <- rbind(Q,Q)
gr <- rep(c(1,2),c(3000,3000))
mg.est <- GDINA(dat = dat,Q = Q,group = gr)
# Fit G-DINA model
mg.est <- GDINA(dat = dat,Q = Q,group = gr,att.dist="higher.order",
    higher.order=list(model = "Rasch"))
summary(mg.est)
coef(mg.est,"lambda")
personparm(mg.est)
personparm(mg.est,"HO")
extract(mg.est,"posterior.prob")

########################################
# Example 11.  
# Bug DINO model
########################################

c <- rbind(c(0.8,0.2),
c(0.7,0.1),
c(0.9,0.2),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1))
sim <- simGDINA(N=1000,Q=Q,catprob.parm = ip,model = "DINO")
dat <- extract(sim,"dat")
# use latent.var to specify a bug model
est <- GDINA(dat=dat,Q=Q,latent.var="bugs",model="DINO")
coef(est)

########################################
# Example 12.  
# Bug DINA model
########################################

c <- rbind(c(0.8,0.2),
c(0.9,0.2),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1,0.1,0.1,0.1,0.1))
sim <- simGDINA(N=1000,Q=Q,catprob.parm = ip,model = "DINO")
dat <- extract(sim,"dat")
# use latent.var to specify a bug model
est <- GDINA(dat=dat,Q=Q,latent.var="bugs",model="DINO")
coef(est)
c(0.7,0.1),
c(0.9,0.2),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.9,0.9,0.9,0.9,0.1))
sim <- simGDINA(N=1000,Q=Q,catprob.parm = ip,model="DINA")
dat <- extract(sim,"dat")
# use latent.var to specify a bug model
est <- GDINA(dat=dat,Q=Q,latent.var="bugs",model="DINA")
coef(est)

####################################
# Example 13a. #
# user specified design matrix #
# LCDM (logit G-DINA) #
####################################

dat <- sim30GDINA$simdat
Q <- sim30GDINA$simQ

# LCDM
lcdm <- GDINA(dat = dat, Q = Q, model = "logitGDINA", control=list(conv.type="neg2LL"))

# Another way is to find design matrix for each item first => must be a list
D <- lapply(rowSums(Q),designmatrix,model="GDINA")
# for comparison, use change in -2LL as convergence criterion
# LCDM
lcdm2 <- GDINA(dat = dat, Q = Q, model = "UDF", design.matrix = D,
linkfunc = "logit", control=list(conv.type="neg2LL"),solver="slsqp")

# identity link GDINA
iGDINA <- GDINA(dat = dat, Q = Q, model = "GDINA",
control=list(conv.type="neg2LL"),solver="slsqp")

# compare all three models => identical
anova(lcdm,lcdm2,iGDINA)

####################################
# Example 13b. #
# user specified design matrix #
# RRUM #
####################################

dat <- sim30GDINA$simdat
Q <- sim30GDINA$simQ

# specify design matrix for each item => must be a list
# D can be defined by the user
D <- lapply(rowSums(Q),designmatrix,model="ACDM")
# for comparison, use change in -2LL as convergence criterion
# RRUM
logACDM <- GDINA(dat = dat, Q = Q, model = "UDF", design.matrix = D,
linkfunc = "log", control=list(conv.type="neg2LL"),solver="slsqp")

# identity link GDINA
RRUM <- GDINA(dat = dat, Q = Q, model = "RRUM",
control=list(conv.type="neg2LL"),solver="slsqp")

# compare two models => identical
anova(logACDM,RRUM)

#########################################################################
# Example 14.  #
# Multiple-strategy DINA model #
#########################################################################
Q <- matrix(c(1,1,1,1,0,
1,1,1,0,0,
1,1,1,0,1,
1,1,0,0,1,
1,0,0,0,0),ncol = 5,byrow = TRUE)
d <- list(
  item1=c(0.2,0.7),
  item2=c(0.1,0.6),
  item3=c(0.2,0.6),
  item4=c(0.2,0.7),
  item5=c(0.1,0.8))

set.seed(12345)
sim <- simGDINA(N=1000,Q = Q, delta.parm = d,
model = c("MSDINA","MSDINA","DINA",
"DINA","DINA","MSDINA","MSDINA"))

# simulated data
dat <- extract(sim,what = "dat")
# estimation
# MSDINA need to be specified for each strategy
est <- GDINA(dat,Q,model = c("MSDINA","MSDINA","DINA",
"DINA","DINA","MSDINA","MSDINA"))
coef(est,"delta")

## End(Not run)
Description

An (experimental) function for calibrating the multiple-strategy CDMs for dichotomous response data (Ma & Guo, 2019)

Usage

```r
GMSCDM(
  dat,
  msQ,
  model = "ACDM",
  s = 1,
  att.prior = NULL,
  delta = NULL,
  control = list()
)
```

Arguments

dat A required binary item response matrix

msQ A multiple-strategy Q-matrix; the first column gives item numbers and the second column gives the strategy number. See examples.

model CDM used; can be "DINA", "DINO", "ACDM", "LLM", and "RRUM", representing the GMS-DINA, GMS-DINO, GMS-ACDM, GMS-LLM and GMS-RRUM in Ma & Guo (2019), respectively. It can also be "rDINA" and "rDINO", representing restricted GMS-DINA and GMS-DINO models where delta_jm1 are equal for all strategies. Note that only a single model can be used for the whole test.

s strategy selection parameter. It is equal to 1 by default.

att.prior mixing proportion parameters.

delta delta parameters in list format.

control a list of control arguments

Value

an object of class GMSCDM with the following components:

IRF A matrix of success probabilities for each latent class on each item (IRF)

delta A list of delta parameters

attribute A list of estimated attribute profiles including EAP, MLE and MAP estimates.

testfit A list of test fit statistics including deviance, number of parameters, AIC and BIC

sIRF strategy-specific item response function

pjmc Probability of adopting each strategy on each item for each latent class

sprv Strategy pravelence

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
References


See Also

GDINA for MS-DINA model and single strategy CDMs, and DTM for diagnostic tree model for multiple strategies in polytomous response data

Examples

```r
## Not run:
#########################################################################
# data simulation
#########################################################################
set.seed(123)
msQ <- matrix(
  c(1,1,0,1,
   1,2,1,0,
   2,1,1,0,
   3,1,0,1,
   4,1,1,1,
   5,1,1,1),6,4,byrow = T)
# J x L - 00,10,01,11
LC.prob <- matrix(c(
  0.2,0.7727,0.5889,0.8125,
  0.1,0.9,0.1,0.9,
  0.1,0.1,0.8,0.8,
  0.2,0.5,0.4,0.7,
  0.2,0.4,0.7,0.9),5,4,byrow=TRUE)
N <- 10000
att <- sample(1:4,N,replace=TRUE)
dat <- 1*(t(LC.prob[,att])>matrix(runif(N*5),N,5))
est <- GMSCDM(dat,msQ)
# item response function
est$IRF
# strategy specific IRF
est$sIRF
```

```r
#########################################################################
# Example 14 from GDINA function
#########################################################################
```
ILCA

Iterative latent-class analysis

Description

This function implements an iterative latent class analysis (ILCA; Jiang, 2019) approach to estimating attributes for cognitive diagnosis.

Usage

ILCA(dat, Q, seed.num = 5)
indlogLik

Arguments

- dat: A required binary item response matrix.
- Q: A required binary item and attribute association matrix.
- seed.num: seed number; Default = 5.

Value

Estimated attribute profiles.

Author(s)

Zhehan Jiang, The University of Alabama

References


Examples

```r
## Not run:
ILCA(sim10GDINA$simdat, sim10GDINA$simQ)
## End(Not run)
```

indlogLik

Extract log-likelihood for each individual

Description

Extract individual log-likelihood.

Usage

```r
indlogLik(object, ...)
```

Arguments

- object: GDINA object
- ...: additional arguments
Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ

fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
iL <- indlogLik(fit)
iL[1:6,]

## End(Not run)
```

indlogPost

Extract log posterior for each individual

Description

Extract individual log posterior.

Usage

```r
indlogPost(object, ...)
```

Arguments

- `object`: GDINA object
- `...`: additional arguments

Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
iP <- indlogPost(fit)
iP[1:6,]

## End(Not run)
```
itemfit

Item fit statistics

Description

Calculate item fit statistics (Chen, de la Torre, & Zhang, 2013) and draw heatmap plot for item pairs

Usage

```r
itemfit(
  GDINA.obj,
  person.sim = "post",
  p.adjust.methods = "holm",
  cor.use = "pairwise.complete.obs",
  digits = 4,
  N.resampling = NULL,
  randomseed = 123456
)
```

## S3 method for class 'itemfit'
extract(object, what, ...)

## S3 method for class 'itemfit'
summary(object, ...)

Arguments

- **GDINA.obj**: An estimated model object of class GDINA
- **person.sim**: Simulate expected responses from the posterior or based on EAP, MAP and MLE estimates.
- **p.adjust.methods**: p-values for the proportion correct, transformed correlation, and log-odds ratio can be adjusted for multiple comparisons at test and item level. This is conducted using `p.adjust` function in `stats`, and therefore all adjustment methods supported by `p.adjust` can be used, including "holm", "hochberg", "hommel", "bonferroni", "BH" and "BY". See `p.adjust` for more details. "holm" is the default.
- **cor.use**: how to deal with missing values when calculating correlations? This argument will be passed to use when calling `stats::cor`.
- **digits**: How many decimal places in each number? The default is 4.
- **N.resampling**: the sample size of resampling. By default, it is the maximum of 1e+5 and ten times of current sample size.
- **randomseed**: random seed; This is used to make sure the results are replicable. The default random seed is 123456.
- **object**: objects of class itemfit for various S3 methods
what argument for S3 method extract indicating what to extract; It can be "p" for proportion correct statistics, "r" for transformed correlations, logOR for log odds ratios and "maxitemfit" for maximum statistics for each item.

... additional arguments

Value

an object of class itemfit consisting of several elements that can be extracted using method extract. Components that can be extracted include:

- **p** the proportion correct statistics, adjusted and unadjusted p values for each item
- **r** the transformed correlations, adjusted and unadjusted p values for each item pair
- **logOR** the log odds ratios, adjusted and unadjusted p values for each item pair
- **maxitemfit** the maximum proportion correct, transformed correlation, and log-odds ratio for each item with associated item-level adjusted p-values

Methods (by generic)

- **extract**: extract various elements from itemfit objects
- **summary**: print summary information

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

References


Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ

mod1 <- GDINA(dat = dat, Q = Q, model = "GDINA")
mod1
itmfit <- itemfit(mod1)

# Print "test-level" item fit statistics
# p-values are adjusted for multiple comparisons
# for proportion correct, there are J comparisons
# for log odds ratio and transformed correlation,
# there are J*(J-1)/2 comparisons
```
itmfit

# The following gives maximum item fit statistics for
# each item with item level p-value adjustment
# For each item, there are J-1 comparisons for each of
# log odds ratio and transformed correlation
summary(itmfit)

# use extract to extract various components
extract(itmfit,"r")

mod2 <- GDINA(dat,Q,model="DINA")
itmfit2 <- itemfit(mod2)
#misfit heatmap
plot(itmfit2)
itmfit2

## End(Not run)

---

**itemparm**

extract item parameters (deprecated)

### Description

This function has been deprecated; use coef instead.

### Usage

```r
itemparm(
  object,
  what = c("catprob", "gs", "delta", "rrum", "itemprob", "LCprob"),
  withSE = FALSE,
  SE.type = 2,
  digits = 4,
  ...
)
```

```r
## S3 method for class 'GDINA'
itemparm(
  object,
  what = c("catprob", "gs", "delta", "rrum", "itemprob", "LCprob"),
  withSE = FALSE,
  SE.type = 2,
  digits = 4,
  ...
)
```
**LC2LG**

*Transformation between latent classes and latent groups*

**Description**

This function gives the equivalent latent classes which have the same category success probabilities for each category or item.

**Usage**

```r
LC2LG(Q, sequential = FALSE, att.str = NULL)
```

**Arguments**

- **Q**: A required $J \times K$ binary Q-matrix. $J$ represents test length and $K$ represents the number of attributes of this test. Entry 1 at row $j$ and column $k$ represents the $k^{th}$ attribute is measured by item $j$, and 0 means item $j$ does not measure attribute $k$.
- **sequential**: logical; whether the Q-matrix is a Qc-matrix for sequential models?
- **att.str**: attribute structure. See GDINA for details.
Value

An item or category by latent class matrix. In the G-DINA model, if item j measures $K_j$ attributes, $2^K$ latent classes can be combined into $2^{K_j}$ latent groups. This matrix gives which latent group each of $2^K$ latent classes belongs to for each item.

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

Examples

```r
attributepattern(3)
q <- matrix(scan(text = "0 1 0 1 0 1 1 0"), ncol = 3)
q
LC2LG(Q = q)
```

MCmodel

Multiple-choice models

Description

This function estimates the multiple-choice DINA model (de la Torre, 2009).

Usage

```r
MCmodel(
  dat,
  Qc,
  model = "MCDINA",
  key = NULL,
  conv.crit = 0.001,
  maxitr = 2000,
  conv.type = "pr"
)
```

Arguments

dat A required $N \times J$ data matrix of $N$ examinees to $J$ items. Values must be 1, 2,... representing nominal categories. Missing values are currently not allowed.

Qc A required category and attribute association matrix. The first column gives the item number, which must be numeric (i.e., 1,2,...) and match the number of column in the data. The second column indicates the coded category of each item. The number in the second column must match with the number in the data, but if a category is not coded, it should not be included in the Q-matrix. Entry
1 indicates that the attribute is measured by the category, and 0 otherwise. Note that the MC-DINA model assumes that the category with the largest number of 1s is the key and that the coded distractors should allow to assign examinees uniquely.

model MCDINA only currently. Other MC models may be incorporated.
key a numeric vector giving the key of each item. See Examples. NULL by default indicating the coded category requiring the largest number of 1s is the key.
conv.crit The convergence criterion for max absolute change in conv.type for two consecutive iterations.
maxitr The maximum iterations allowed.
conv.type convergence criteria; Can be pr or LL, indicating category response function, or -2 times log-likelihood, respectively.

Value
an object of class MCmodel with the following components:

prob.parm A list of success probabilities for each reduced latent class on each item (IRF)
prob.se A list of standard errors of item parameters
attribute A list of estimated attribute profiles including EAP, MLE and MAP estimates.
testfit A list of test fit statistics including deviance, number of parameters, AIC and BIC
R expected # of individuals in each latent group choosing each option
lik posterior probability
itr Total # of iterations

Author(s)
Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References

See Also
GDINA for G-DINA model

Examples
```r
## Not run:
# check the format of the data
# Entry 0 is not allowed
head(sim@MCDINA$simdat)
```
# check the format of the Q-matrix
# Take item 1 as an example:
# category 2 has a q-vector (1,0,0)
# category 1 has a q-vector (0,1,0)
# category 4 has a q-vector (1,1,0)
# category 3 is not included in the Q-matrix because it is not coded
# the order of the coded categories in the Q-matrix doesn't matter

take item 1 as an example:

category 2 has a q-vector (1,0,0)
category 1 has a q-vector (0,1,0)
category 4 has a q-vector (1,1,0)
category 3 is not included in the Q-matrix because it is not coded
the order of the coded categories in the Q-matrix doesn't matter

# Distractors involving more attributes
# some distractors may involve attributes that are not involved by the key option
# this is not allowed by the "original" MC-DINA (de la Torre, 2009) but is allowed
# in the current implementation

# Users need to specify the key for each item to appropriately handle such an issue
# Note item 1 below: category 1 is the key (as indicated in the key argument below)
# The distractor (category 4) involves an attribute not included by the key option

Qc <- matrix(c(1,1,1,0,
               1,2,0,1,0,
               1,3,1,0,0,
               1,4,1,0,1,
               2,1,1,0,0,
               2,3,1,1,0,
               2,2,1,1,1,
               3,4,1,1,1,
               3,2,1,1,0,
               3,3,0,1,1,
               4,1,0,1,1,
               4,2,0,0,1,
               5,1,1,0,0,
               6,3,0,1,0,
               7,2,0,0,1,
               8,4,1,0,0,
               9,1,0,1,0,
               10, 4,0,0,1),ncol = 5,byrow = TRUE)
est2 <- MCmodel(sim10MCDINA2$simdat,Qc, key = c(1,2,4,1,1,3,2,4,1,4))
modelcomp  

**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 the Wald test, likelihood ratio (LR) test or Lagrange multiplier (LM) test. For Wald test, see de la Torre (2011), de la Torre and Lee (2013), Ma, Iaconangelo and de la Torre (2016) and Ma & de la Torre (2018) for details. For LR test and a two-step LR approximation procedure, see Sorrel, de la Torre, Abad, and Olea (2017), Ma (2017) and Ma & de la Torre (2019). For LM test, which is only applicable for DINA, DINO and ACDM, see Sorrel, Abad, Olea, de la Torre, and Barrada (2017). This function also calculates the dissimilarity between the reduced models and the G-DINA model, which can be viewed as a measure of effect size (Ma, Iaconangelo & de la Torre, 2016).

**Usage**

```r
modelcomp(
  GDINA.obj = NULL,
  method = "Wald",
  items = "all",
  p.adjust.methods = "holm",
  models = c("DINA", "DINO", "ACDM", "LLM", "RRUM"),
  decision.args = list(rule = "simpler", alpha.level = 0.05, adjusted = FALSE),
  DS = FALSE,
  Wald.args = list(SE.type = 2, varcov = NULL),
  LR.args = list(LR.approx = FALSE),
  LM.args = list(reducedMDINA = NULL, reducedMDINO = NULL, reducedMACDM = NULL, SE.type = 2)
)
```

## S3 method for class 'modelcomp'

```r
extract(
  object,
  what = c("stats", "pvalues", "adj.pvalues", "df", "DS", "selected.model"),
  digits = 4,
  ...
)
```

## S3 method for class 'modelcomp'

```r
summary(object, ...)
```
Arguments

**GDINA.obj**  An estimated model object of class GDINA:

**method**  method for item level model comparison; can be `wald`, `LR` or `LM`.

**items**  a vector of items to specify the items for model comparison.

**p.adjust.methods**  adjusted p-values for multiple hypothesis tests. This is conducted using `p.adjust` function in `stats`, and therefore all adjustment methods supported by `p.adjust` can be used, including "holm", "hochberg", "hommel", "bonferroni", "BH" and "BY". See `p.adjust` for more details. "holm" is the default, indicating the Holm method.

**models**  a vector specifying which reduced CDMs are possible reduced CDMs for each item. The default is "DINA","DINO","ACDM","LLM", and "RRUM".

**decision.args**  a list of options for determining the most appropriate models including (1) rule can be either "simpler" or "largestp". See details; (2) alpha.level for the nominal level of decision; and (3) adjusted can be either TRUE or FALSE indicating whether the decision is based on p value (adjusted = FALSE) or adjusted p values.

**DS**  whether dissimilarity index should be calculated? FALSE is the default.

**Wald.args**  a list of options for Wald test including (1) `SE.type` giving the type of covariance matrix for the Wald test; by default, it uses outer product of gradient based on incomplete information matrix; (2) `varcov` for user specified variance-covariance matrix. If supplied, it must be a list, giving the variance covariance matrix of success probability for each item or category. The default is `NULL`, in which case, the estimated variance-covariance matrix from the GDINA function is used.

**LR.args**  a list of options for LR test including for now only `LR.approx`, which is either TRUE or FALSE, indicating whether a two-step LR approximation is implemented or not.

**LM.args**  a list of options for LM test including `reducedMDINA`, `reducedMDINO`, and `reducedMACDM` for DINA, DINO and ACDM estimates from the GDINA function; `SE.type` specifies the type of covariance matrix.

**object**  object of class `modelcomp` for various S3 methods.

**what**  argument for S3 method `extract` indicating what to extract; It can be "wald" for wald statistics, "wald.p" for associated p-values, "df" for degrees of freedom, and "DS" for dissimilarity between G-DINA and other CDMs.

**digits**  How many decimal places in each number? The default is 4.

**...**  additional arguments

Details

After the test statistics for each reduced CDM were calculated for each item, the reduced models with p values less than the pre-specified alpha level were rejected. If all reduced models were rejected for an item, the G-DINA model was used as the best model; if at least one reduced model was retained, two different rules can be implemented for selecting the best model specified in argument `decision.args`:
(1) when rule="simpler", which is the default, 
If (a) the DINA or DINO model was one of the retained models, then the DINA or DINO model with 
the larger p value was selected as the best model; but if (b) both DINA and DINO were rejected, 
the reduced model with the largest p value was selected as the best model for this item. Note that 
when the p-values of several reduced models were greater than 0.05, the DINA and DINO models 
were preferred over the A-CDM, LLM, and R-RUM because of their simplicity. This procedure is 
originally proposed by Ma, Iaconangelo, and de la Torre (2016).

(2) When rule="largestp":
The reduced model with the largest p-values is selected as the most appropriate model.

Value
an object of class modelcomp. Elements that can be extracted using extract method include 

stats  Wald or LR statistics 
pvalues  p-values associated with the test statistics 
adj.pvalues  adjusted p-values 
df  degrees of freedom 
DS  dissimilarity between G-DINA and other CDMs

Methods (by generic)
• extract: extract various elements from modelcomp objects 
• summary: print summary information

Author(s)
Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
Miguel A. Sorrel, Universidad Autonoma de Madrid
Jimmy de la Torre, The University of Hong Kong

References
de la Torre, J., & Lee, Y. S. (2013). Evaluating the wald test for item-level comparison of saturated 
Ma, W., Iaconangelo, C., & de la Torre, J. (2016). Model similarity, model selection and attribute 
Ma, W. (2017). A Sequential Cognitive Diagnosis Model for Graded Response: Model Develop-
ment, Q-Matrix Validation,and Model Comparison. Unpublished doctoral dissertation. New 
Brunswick, NJ: Rutgers University.
Ma, W., & de la Torre, J. (2019). Category-Level Model Selection for the Sequential G-DINA 

**See Also**

GDINA, autoGDINA

**Examples**

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
# --- GDINA model ---#
fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
fit

# Wald test
w <- modelcomp(fit)
w
# LR and Two-step LR test
lr <- modelcomp(fit, method = "LR")
lr
TwostepLR <- modelcomp(fit, items = c(6:10), method = "LR", LR.args = list(LR.approx = TRUE))
TwostepLR

# LM test
dina <- GDINA(dat = dat, Q = Q, model = "DINA")
dino <- GDINA(dat = dat, Q = Q, model = "DINO")
acdm <- GDINA(dat = dat, Q = Q, model = "ACDM")
lm <- modelcomp(method = "LM", LM.args = list(reducedMDINA = dina,
processedMDINO = dino, reducedMACDM = acdm))
lm
```
modelfit

## End(Not run)

---

**Description**

Calculate various absolute model-data fit statistics

**Usage**

```r
modelfit(GDINA.obj, CI = 0.9, ItemOnly = FALSE)
```

**Arguments**

- **GDINA.obj**: An estimated model object of class `GDINA`
- **CI**: numeric value from 0 to 1 indicating the range of the confidence interval for RMSEA. Default returns the 90% interval.
- **ItemOnly**: should joint attribute distribution parameters be considered? Default = FALSE. See Ma (2019).

**Details**

Various model-data fit statistics including M2 statistic for G-DINA model with dichotmous responses (Liu, Tian, & Xin, 2016; Hansen, Cai, Monroe, & Li, 2016) and for sequential G-DINA model with graded responses (Ma, 2020). It also calculates SRMSR and RMSEA2.

**Author(s)**

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

**References**


Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod1 <- GDINA(dat = dat, Q = Q, model = "DINA")
modelfit(mod1)

## End(Not run)
```

### monocheck

This function checks if monotonicity is violated

**Description**

If mastering an additional attribute lead to a lower probabilities of success, the monotonicity is violated.

**Usage**

`monocheck(object, strict = FALSE)`

**Arguments**

- `object`: object of class `GDINA`
- `strict`: whether a strict monotonicity is checked?

**Value**

a logical vector for each item or category indicating whether the monotonicity is violated (TRUE) or not (FALSE)

**Examples**

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod1 <- GDINA(dat = dat, Q = Q, model = "DINA")
check <- monocheck(mod1)
check
mod2 <- GDINA(dat = dat, Q = Q, model = "DINA", mono.constraint = check)
check2 <- monocheck(mod2)
check2

## End(Not run)
```
npar

Calculate the number of parameters

Description

Calculate the number of parameters for GDINA estimates. Returned the total number of parameters, the number of item parameters and the number parameters of joint attribute distribution.

Usage

npar(object, ...)

Arguments

object GDINA object
...
additional arguments

Examples

## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ

fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
npar(fit)

## End(Not run)

personparm

calculate person (incidental) parameters

Description

Function to calculate various person attribute parameters, including "EAP", "MAP", and "MLE", for EAP, MAP and MLE estimates of attribute patterns (see Huebner & Wang, 2011), "mp" for marginal mastery probabilities, and "HO" for higher-order ability estimates if a higher-order model is fitted. See GDINA for examples.

Usage

personparm(object, what = c("EAP", "MAP", "MLE", "mp", "HO"), digits = 4, ...)

Arguments

- **object**: estimated GDINA object returned from `GDINA`
- **what**: what to extract; It can be "EAP", "MAP", and "MLE", for EAP, MAP and MLE estimates of attribute patterns, and "mp" for marginal mastery probabilities, and "HO" for higher-order ability estimates if a higher-order model is fitted.
- **digits**: number of decimal places.
- **...**: additional arguments

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

References


Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
# EAP
head(personparm(fit))
# MAP
head(personparm(fit, what = "MAP"))
## End(Not run)
```

plot.GDINA

Create plots for GDINA estimates

Description

Create various plots for GDINA estimates

Usage

```r
## S3 method for class 'GDINA'
plot(
  x,
  what = "IRF",
  item = "all",
  withSE = FALSE,
)```
plot.itemfit

SE.type = 2,
person = 1,
att.names = NULL,
...
)

Arguments

x model object of class GDINA
what type of plot. Can be "IRF" for item/category response function plot, or "mp" for mastery probabilities for individuals.
item A scalar or vector specifying the item(s) for IRF plots.
withSE logical; Add error bar (estimate - SE, estimate + SE) to the IRF plots?
SE.type How is SE estimated. By default, it's based on OPG using incomplete information.
person A scalar or vector specifying the number of individuals for mastery plots.
att.names Optional; a vector for attribute names.
... additional arguments

See Also

GDINA, autoGDINA

Examples

## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod1 <- GDINA(dat = dat, Q = Q, model = "GDINA")
# plot item response functions for item 10
plot(mod1, item = 10)
plot(mod1, what = "IRF", item = 10, withSE = TRUE)

# plot mastery probabilities for individuals 4 and 10
plot(mod1, what = "mp", person = c(4,10))
plot(mod1, what = "mp", person = c(4,10,15),
att.names = c("addition","subtraction","multiplication"))

## End(Not run)

---

plot.itemfit Item fit plots

Description

Create plots of bivariate heatmap for item fit
Usage

```r
## S3 method for class 'itemfit'
plot(x, type = "all", adjusted = TRUE, ...)
```

Arguments

- `x`: model object of class `itemfit`
- `type`: type of heatmap plot
- `adjusted`: logical; plot adjusted or unadjusted p-values?
- `...`: additional arguments

See Also

- `GDINA`, `itemfit`

Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
ift <- itemfit(fit)
# plot the adjusted p values for log odds or transformed correlation
plot(ift)
# plot unadjusted p values for log odds
plot(ift, adjusted = FALSE, type = "logOR")
## End(Not run)
```

---

**plot.Qval**  
*Mesa plot for Q-matrix validation*

Description

The mesa plot was first proposed by de la Torre and Ma (2016) for graphically illustrating the best q-vector(s) for each item. The q-vector on the edge of the mesa is likely to be the best q-vector.

Usage

```r
## S3 method for class 'Qval'
plot(
  x,
  item,
  type = "best",
  no.qvector = 10,
  data.label = TRUE,
)```
eps = "auto",
original.q.label = FALSE,
auto.ylim = TRUE,
...)

Arguments

x model object of class Qvalidation
item a vector specifying which item(s) the plots are drawn for
type types of the plot. It can be "best" or "all". If "best", for all q-vectors requiring the same number of attributes, only the one with the largest PVAF is plotted, which means $K_j$ q-vectors are plotted; If "all", all q-vectors will be plotted.
no.qvector the number of q vectors that need to be plotted when type="all". The default is 10, which means the 10 q vectors with the largest PVAFs are plotted.
data.label logical; To show data label or not?
eps the cutoff for PVAF. If not NULL, it must be a value between 0 and 1. A horizontal line will be drawn accordingly.
original.q.label logical; print the label showing the original q-vector or not?
auto.ylim logical; create y range automatically or not?
... additional arguments passed to plot function

References


See Also

Qval, autoGDINA

Examples

## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
Q[1,] <- c(0,1,0)
mod1 <- GDINA(dat = dat, Q = Q, model = "GDINA")
out <- Qval(mod1,eps = 0.9)
item <- c(1,2,10)
plot(out,item=item,data.label=FALSE,type="all")
plot(out,item=10,type="best",eps=0.95)
plot(out,item=10,type="all",no.qvector=6)

## End(Not run)
Qval

Q-matrix validation

Description

Q-matrix validation for the (sequential) G-DINA model based on PVAF (de la Torre & Chiu, 2016; Najera, Sorrel, & Abad, 2019; Najera et al., 2020), stepwise Wald test (Ma & de la Torre, 2020) or mesa plot (de la Torre & Ma, 2016). All these methods are suitable for dichotomous and ordinal response data. If too many modifications are suggested based on the default PVAF method, you are suggested to try the stepwise Wald test method, iterative procedures or predicted cutoffs. You should always check the mesa plots for further examination.

Usage

```r
Qval(
  GDINA.obj, 
  method = "PVAF", 
  iter = "none", 
  eps = 0.95, 
  digits = 4, 
  wald.args = list(), 
  iter.args = list(empty.att = FALSE, max.iter = 150, verbose = FALSE)
)
```

## S3 method for class 'Qval'
extract(object, what = c("sug.Q", "varsigma", "PVAF", "eps", "Q"), ...)

## S3 method for class 'Qval'
summary(object, ...)

Arguments

- **GDINA.obj** an estimated model object of class GDINA
- **method** which Q-matrix validation method is used? Can be either "PVAF" or "wald".
- **iter** implement the method iteratively? Can be "none" for non-iterative validation (by default), "test", "test.att", or "item" (Najera et al., 2020).
- **eps** cutoff value for PVAF from 0 to 1. Default = 0.95. Note that it can also be -1, indicating the predicted cutoff based on Najera, Sorrel, and Abad (2019).
- **digits** how many decimal places in each number? The default is 4.
- **wald.args** a list of arguments for the stepwise Wald test method.
- **SE.type** type of covariance matrix for the Wald test
- **alpha.level** alpha level for the Wald test
- **GDI** it can be 0, 1 or 2; 0 means GDI is not used to choose the attribute - when more than one attributes are significant, the one with the largest p-value will be selected; GDI=1 means the attribute with the largest GDI will be selected; GDI=2 means the q-vector with the largest GDI will be selected.
**Verbose** print detailed information or not?

**Stepwise** TRUE for stepwise approach and FALSE for forward approach

**Iter.args** a list of arguments for the iterative implementation.

**Empty.att** can a Q-matrix with an empty attribute (i.e., measured by no items) be provided? Default is FALSE

**Max.iter** maximum number of iterations. Default is 150

**Verbose** print information after each iteration? Default is FALSE

**Object** Qval objects for S3 methods

**What** argument for S3 method extract indicating what to extract; It can be "sug.Q" for suggested Q-matrix, "Q" for original Q-matrix, "varsigma" for varsigma index, and "PVAF" for PVAF.

... additional arguments

**Value**

An object of class Qval. Elements that can be extracted using extract method include:

- **sug.Q** suggested Q-matrix
- **Q** original Q-matrix
- **Varsigma** varsigma index
- **PVAF** PVAF

**Methods (by generic)**

- **extract**: extract various elements from Qval objects
- **summary**: print summary information

**Author(s)**

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>,
Miguel A. Sorrel, Universidad Autónoma de Madrid,
Jimmy de la Torre, The University of Hong Kong

**References**


See Also

GDINA

Examples

## Not run:

```r
# Binary response

dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
Q[10,] <- c(0,1,0)

# Fit the G-DINA model
mod1 <- GDINA(dat = dat, Q = Q, model = "GDINA")

# Q-validation using de la Torre and Chiu's method
pvaf <- Qval(mod1, method = "PVAF", eps = 0.95)
extract(pvaf, what = "PVAF")
# See also:
extract(pvaf, what = "varsigma")
extract(pvaf, what = "sug.Q")

# Draw mesa plots using the function plot
plot(pvaf, item = 10)

# The stepwise Wald test
stepwise <- Qval(mod1, method = "wald")
extract(stepwise, what = "PVAF")
# See also:
extract(stepwise, what = "varsigma")
extract(stepwise, what = "sug.Q")

# Set eps = -1 to determine the cutoff empirically
pvaf2 <- Qval(mod1, method = "PVAF", eps = -1)

# Iterative procedure (test-attribute level)
pvaf3 <- Qval(mod1, method = "PVAF", eps = -1,
             iter = "test.att", iter.args = list( verbose = 1))

```

```
seq.est <- GDINA(sim20seqGDINA$simdat, sim20seqGDINA$simQ, sequential = TRUE)
stepwise <- Qval(seq.est, method = "wald")
## End(Not run)

---

text

### rowMatch

*Count the frequency of a row vector in a data frame*

**Description**

Count the frequency of a row vector in a data frame

**Usage**

```r
rowMatch(df, vec = NULL)
```

**Arguments**

- `df`: a data frame or matrix
- `vec`: the vector for matching

**Value**

- count the number of vector vec in the data frame
- row.no: row numbers of the vector vec in the data frame

**Examples**

```r
df <- data.frame(V1=c(1L,2L), V2=LETTERS[1:3], V3=rep(1,12))
rowMatch(df, c(2,"B",1))
```

---

### score

*Score function*

**Description**

Calculate score function for each dichotomous item or each nonzero category for polytomous items

Only applicable to saturated model of joint attribute distribution

**Usage**

```r
score(object, parm = "delta")
```
Arguments

object  an object of class GDINA
parm    Either delta or prob indicating score function for delta parameters and success probability parameters

Value

a list where elements give the score functions for each item or category

Examples

## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
score(fit)
## End(Not run)

---

**sim10GDINA**

*Simulated data (10 items, G-DINA model)*

Description

Simulated data, Q-matrix and item parameters for a 10-item test with 3 attributes.

Usage

sim10GDINA

Format

A list with components:

- simdat  simulated responses of 1000 examinees
- simQ    artificial Q-matrix
- simItempar  artificial item parameters (probability of success for each latent group)

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References

### sim10MCDINA

**Simulated data (10 items, MC-DINA model)**

**Description**

Simulated data, Q-matrix and item parameters for a 10-item test measuring 3 attributes.

**Usage**

sim10MCDINA

**Format**

A list with components:

- simdat: simulated responses of 3000 examinees
- simQ: artificial Q-matrix

**Author(s)**

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

**References**


---

### sim10MCDINA2

**Simulated data (10 items, MC-DINA model)**

**Description**

Simulated data, Q-matrix and item parameters for a 10-item test measuring 5 attributes.

**Usage**

sim10MCDINA2

**Format**

A list with components:

- simdat: simulated responses of 3000 examinees
- simQ: artificial Q-matrix
sim20seqGDINA

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References


---

**sim20seqGDINA**

*Simulated data (20 items, sequential G-DINA model)*

Description

Simulated data, Qc-matrix and item parameters for a 20-item test measuring 5 attributes.

Usage

sim20seqGDINA

Format

A list with components:

- simdat  simulated polytomous responses of 2000 examinees
- simQ    artificial Qc-matrix
- simItempar artificial item parameters (category level probability of success for each latent group)

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References

**sim21seqDINA**

**Simulated data (21 items, sequential DINA model)**

**Description**

Simulated data, and Qc-matrix for a 21-item test measuring 5 attributes.

**Usage**

`sim21seqDINA`

**Format**

A list with components:

- `simdat` simulated responses of 2000 examinees
- `simQ` artificial Qc-matrix

**Author(s)**

Wenchao Ma, The University of Alabama. <wenchao.ma@ua.edu>

**References**


---

**sim30DINA**

**Simulated data (30 items, DINA model)**

**Description**

Simulated data, Q-matrix and item parameters for a 30-item test measuring 5 attributes.

**Usage**

`sim30DINA`

**Format**

A list with components:

- `simdat` simulated responses of 1000 examinees
- `simQ` artificial Q-matrix
- `simItempar` artificial item parameters (probability of success for each latent group)
sim30GDINA

Author(s)
Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References

---

**sim30GDINA**

*Simulated data (30 items, G-DINA model)*

---

**Description**

Simulated data, Q-matrix and item parameters for a 30-item test measuring 5 attributes.

**Usage**

`sim30GDINA`

**Format**

A list with components:

- `simdat` simulated responses of 1000 examinees
- `simQ` artificial $30 \times 5$ Q-matrix
- `simItempar` artificial item parameters(probability of success for each latent group)

**Author(s)**

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

**References**

Simulated data for diagnostic tree model

Description

Data generation for diagnostic tree model

Usage

`simDTM(N, Qc, gs.parm, Tmatrix, red.delta = NULL, att.gr = NULL)`

Arguments

- `N` : sample size
- `Qc` : Association matrix between attributes (column) and PSEUDO items (row); The first column is item number and the second column is the pseudo item number for each item. If a pseudo item has more than one nonzero categories, more than one rows are needed.
- `gs.parm` : the same as the gs.parm in simGDINA function in the GDINA package. It is a list with the same number of elements as the number of rows in the Qc matrix
- `Tmatrix` : mapping matrix showing the relation between the OBSERVED responses (rows) and the PSEUDO items (columns); The first column gives the observed responses.
- `red.delta` : reduced delta parameters using logit link function
- `att.gr` : attribute group indicator
Examples

```r
## Not run:
K = 5
g = 0.2
item.no <- rep(1:6, each = 4)
# the first node has three response categories: 0, 1 and 2
node.no <- rep(c(1, 1, 2, 3), 6)
Q1 <- matrix(0, length(item.no), K)
Q2 <- cbind(7:(7 + K - 1), rep(1, K), diag(K))
for (j in 1:length(item.no)) {
  Q1[j, sample(1:K, sample(3, 1))] <- 1
}
Qc <- rbind(cbind(item.no, node.no, Q1), Q2)
Tmatrix.set <- list(cbind(c(0, 1, 2, 3, 3), c(0, 1, 2, 1, 2), c(NA, 0, NA, 1, NA), c(NA, NA, 0, NA, 1)),
                    cbind(c(0, 1, 2, 3, 4), c(0, 1, 2, 1, 2), c(NA, 0, NA, 1, NA), c(NA, NA, 0, NA, 1)),
                    cbind(c(0, 1), c(0, 1)))
Tmatrix <- Tmatrix.set[c(1, 1, 1, 1, 1, 1, 1, rep(3, K))]
sim <- simDTM(N = 2000, Qc = Qc, gs.parm = matrix(0.2, nrow(Qc), 2), Tmatrix = Tmatrix)
est <- DTM(dat = sim$dat, Qc = Qc, Tmatrix = Tmatrix)
## End(Not run)
```

simGDINA

Data simulation based on the G-DINA models

Description

Simulate responses based on the G-DINA model (de la Torre, 2011) and sequential G-DINA model (Ma & de la Torre, 2016), or CDMs subsumed by them, including the DINA model, DINO model, ACDM, LLM and R-RUM. Attributes can be simulated from uniform, higher-order or multivariate normal distributions, or be supplied by users. See Examples and Details for how item parameter specifications. See the help page of GDINA for model parameterizations.

Usage

```r
simGDINA(
  N,
  Q,
  gs.parm = NULL,
  delta.parm = NULL,
  catprob.parm = NULL,
  model = "GDINA",
  sequential = FALSE,
  gs.args = list(type = "random", mono.constraint = TRUE),
  delta.args = list(design.matrix = NULL, linkfunc = NULL),
  attribute = NULL,
  att.dist = "uniform",
  item.names = NULL,
```

higher.order.parm = list(theta = NULL, lambda = NULL),
mvnorm.parm = list(mean = NULL, sigma = NULL, cutoffs = NULL),
att.prior = NULL,
digits = 4
)

## S3 method for class 'simGDINA'
extract(
  object,
  what = c("dat", "Q", "attribute", "catprob.parm", "delta.parm", "higher.order.parm",
           "mvnorm.parm", "LCprob.parm"),
  ...
)

Arguments

N Sample size.

Q A required matrix; The number of rows occupied by a single-strategy dichoto-
mous item is 1, by a polytomous item is the number of nonzero categories, and
by a multiple-strategy dichotomous item is the number of strategies. The number
of column is equal to the number of attributes if all items are single-strategy di-
ichotomous items, but the number of attributes + 2 if any items are polytomous or
have multiple strategies. For a polytomous item, the first column represents the
item number and the second column indicates the nonzero category number. For
a multiple-strategy dichotomous item, the first column represents the item num-
ber and the second column indicates the strategy number. For binary attributes,
1 denotes the attributes are measured by the items and 0 means the attributes
are not measured. For polytomous attributes, non-zero elements indicate which
level of attributes are needed. See Examples.

gs.parm A matrix or data frame for guessing and slip parameters. The number of rows
occupied by a dichotomous item is 1, and by a polytomous item is the number of
nonzero categories. The number of columns must be 2, where the first column
represents the guessing parameters (or $P(0)$), and the second column represents
slip parameters (or $1 - P(1)$). This may need to be used in conjunction with the
argument gs.args.

delta.parm A list of delta parameters of each latent group for each item or category. This
may need to be used in conjunction with the argument delta.args.

catprob.parm A list of success probabilities of each latent group for each non-zero category of
each item. See Examples and Details for more information.

model A character vector for each item or nonzero category, or a scalar which will
be used for all items or nonzero categories to specify the CDMs. The possi-
ble options include "GDINA", "DINA", "DINO", "ACDM", "LLM", "RRUM", "MSDINA"
and "UDF". When "UDF", indicating user defined function, is specified for any
item, delta.parm must be specified, as well as options design.matrix and
linkfunc in argument delta.args.

sequential logical; TRUE if the sequential model is used for polytomous responses simula-
tion, and FALSE if there is no polytomously scored items.
gs.args  a list of options when `gs.parm` is specified. It consists of two components:

- **type** How are the delta parameters for ACDM, LLM, RRUM generated? It can be either "random" or "equal". "random" means the delta parameters are simulated randomly, while "equal" means that each required attribute contributes equally to the probability of success (`P`), `logit(P)` or `log(P)` for ACDM, LLM and RRUM, respectively. See `Details` for more information.

- **mono.constraint** A vector for each item/category or a scalar which will be used for all items/categories to specify whether monotonicity constraints should be satisfied if the generating model is the G-DINA model. Note that this is applicable only for the G-DINA model when `gs.parm` is used. For ACDM, LLM and RRUM, monotonicity constraints are always satisfied and therefore this argument is ignored.

delta.args  a list of options when `delta.parm` is specified. It consists of two components:

- **linkfunc** a vector of link functions for each item/category; It can be "identity", "log" or "logit". Only necessary when, for some items, `model="UDF"`.

- **design.matrix** a list of design matrices; Its length must be equal to the number of items (or nonzero categories for sequential models). If CDM for item `j` is specified as "UDF" in argument `model`, the corresponding design matrix must be provided; otherwise, the design matrix can be `NULL`, which will be generated automatically.

attribute  optional user-specified person attributes. It is a `N x K` matrix or data frame. If this is not supplied, attributes are simulated from a distribution specified in `att.dist`.

att.dist  A string indicating the distribution for attribute simulation. It can be "uniform", "higher.order", "mvnorm" or "categorical" for uniform, higher-order, multivariate normal and categorical distributions, respectively. The default is the uniform distribution. To specify structural parameters for the higher-order and multivariate normal distributions, see `higher.order.parm` and `mvnorm.parm`, respectively. To specify the probabilities for the categorical distribution, use `att.prior` argument.

item.names  A vector giving the name of items or categories. If it is `NULL` (default), items are named as "Item 1", "Item 2", etc.

higher.order.parm  A list specifying parameters for higher-order distribution for attributes if `att.dist=higher.order`. Particularly, `theta` is a vector of length `N` representing the higher-order ability for each examinee, and `lambda` is a `K x 2` matrix. Column 1 gives the slopes for the higher-order model and column 2 gives the intercepts. See `GDINA` for the formulations of the higher-order models.

mvnorm.parm  a list of parameters for multivariate normal attribute distribution. `mean` is a vector of length `K` specifying the mean of multivariate normal distribution; and `sigma` is a positive-definite symmetric matrix specifying the variance-covariance matrix. `cutoffs` is a vector giving the cutoff for each attribute. See `Examples`.

att.prior  probability for each attribute pattern. Order is the same as that returned from `attributepattern(Q = Q)`. This is only applicable when `att.dist="categorical"`.

digits  How many decimal places in each number? The default is 4.
object of class simGDINA for method extract
what argument for S3 method extract indicating what to extract
... additional arguments

Details

Item parameter specifications in simGDINA:

Item parameters can be specified in one of three different ways.

The first and probably the easiest way is to specify the guessing and slip parameters for each item or nonzero category using gs.parm, which is a matrix or data frame for \( P(\alpha_{lj}^* = 0) \) and \( 1 - P(\alpha_{lj}^* = 1) \) for all items for dichotomous items and \( S(\alpha_{ljh}^* = 0) \) and \( 1 - S(\alpha_{ljh}^* = 1) \) for all nonzero categories for polytomous items. Note that \( 1 - P(\alpha_{lj}^* = 0) - P(\alpha_{lj}^* = 1) \) or \( 1 - S(\alpha_{lj}^* = 0) - S(\alpha_{lj}^* = 1) \) must be greater than 0. For generating ACDM, LLM, and RRUM, delta parameters are generated randomly if type="random", or in a way that each required attribute contributes equally, as in Ma, Iaconangelo, & de la Torre (2016) if type="equal". For ACDM, LLM and RRUM, generated delta parameters are always positive, which implies that monotonicity constraints are always satisfied. If the generating model is the G-DINA model, mono.constraint can be used to specify whether monotonicity constraints should be satisfied.

The second way of simulating responses is to specify success probabilities (i.e., \( P(\alpha_{lj}^* = 0) \) or \( S(\alpha_{ljh}^* = 0) \)) for each nonzero category of each item directly using the argument catprob.parm. If an item or category requires \( K_j^* \) attributes, \( 2^{K_j^*} \) success probabilities need to be provided. catprob.parm must be a list, where each element gives the success probabilities for nonzero category of each item. Note that success probabilities cannot be negative or greater than one.

The third way is to specify delta parameters for data simulation. For DINA and DINO model, each nonzero category requires two delta parameters. For ACDM, LLM and RRUM, if a nonzero category requires \( K_j^* \) attributes, \( K_j^* + 1 \) delta parameters need to be specified. For the G-DINA model, a nonzero category requiring \( K_j^* \) attributes has \( 2^{K_j^*} \) delta parameters. It should be noted that specifying delta parameters needs to ascertain the derived success probabilities are within the \([0, 1]\) interval.

Please note that you need to specify item parameters in ONLY one of these three ways. If gs.parm is specified, it will be used regardless of the inputs in catprob.parm and delta.parm. If gs.parm is not specified, simGDINA will check if delta.parm is specified; if yes, it will be used for data generation. if both gs.parm and delta.parm are not specified, catprob.parm is used for data generation.

Value

an object of class simGDINA. Elements that can be extracted using method extract include:

- **dat** simulated item response matrix
- **Q** Q-matrix
- **attribute** A \( N \times K \) matrix for individuals’ attribute patterns
- **catprob.parm** a list of non-zero category success probabilities for each latent group
- **delta.parm** a list of delta parameters
- **higher.order.parm** Higher-order parameters
**mvnorm.parm** multivariate normal distribution parameters

**LCprob.parm** A matrix of item/category success probabilities for each latent class

**Author(s)**

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

**References**


**Examples**

```r
## Not run:
####################################################
# Example 1 #
# Data simulation (DINA) #
####################################################
N <- 500
Q <- simGDINA$simQ
J <- nrow(Q)
gs <- data.frame(guess=rep(0.1,J),slip=rep(0.1,J))

# Simulated DINA model; to simulate G-DINA model
```
# and other CDMs, change model argument accordingly
sim <- simGDINA(N,Q,gs.parm = gs,model = "DINA")

# True item success probabilities
extract(sim,what = "catprob.parm")

# True delta parameters
extract(sim,what = "delta.parm")

# simulated data
extract(sim,what = "dat")

# simulated attributes
extract(sim,what = "attribute")

##########################################################
# Example 2
# Data simulation (RRUM)
##########################################################
N <- 500
Q <- sim30GDINA$simQ
J <- nrow(Q)
gs <- data.frame(guess=rep(0.2,J),slip=rep(0.2,J))
# Simulated RRUM
# deltas except delta0 for each item will be simulated
# randomly subject to the constraints of RRUM
sim <- simGDINA(N,Q,gs.parm = gs,model = "RRUM")

# simulated data
extract(sim,what = "dat")

# simulated attributes
extract(sim,what = "attribute")

##########################################################
# Example 3
# Data simulation (LLM)
##########################################################
N <- 500
Q <- sim30GDINA$simQ
J <- nrow(Q)
gs <- data.frame(guess=rep(0.1,J),slip=rep(0.1,J))
# Simulated LLM
# By specifying type="equal", each required attribute is
# assumed to contribute to logit(P) equally
sim <- simGDINA(N,Q,gs.parm = gs,model = "LLM",gs.args = list (type="equal"))
#check below for what the equal contribution means
extract(sim,what = "delta.parm")

# simulated data
extract(sim,what = "dat")
# Example 4
# Data simulation (all CDMs)
set.seed(12345)
N <- 500
Q <- sim10GDINA$simQ
J <- nrow(Q)
gs <- data.frame(guess=rep(0.1,J),slip=rep(0.1,J))
# Simulated different CDMs for different items
models <- c("GDINA","DINO","DINA","ACDM","LLM","RRUM","GDINA","LLM","RRUM","DINA")
sim <- simGDINA(N,Q,gs.parm = gs,model = models,gs.args = list(type="random"))

# simulated data
extract(sim,what = "dat")

# simulated attributes
extract(sim,what = "attribute")

# Example 5
# Data simulation (all CDMs) using probability of success in list format
# success probabilities for each item need to be provided in list format as follows:
# if item j requires Kj attributes, 2^Kj success probabilities need to be specified
# e.g., item 1 only requires 1 attribute
# therefore P(0) and P(1) should be specified;
# similarly, item 10 requires 3 attributes,
# P(000),P(100),P(010)....P(111) should be specified;
# the latent class represented by each element can be obtained
# by calling attributepattern(Kj)
itemparm.list <- list(item1=c(0.2,0.9),
                      item2=c(0.1,0.8),
                      item3=c(0.1,0.9),
                      item4=c(0.1,0.3,0.5,0.9),
                      item5=c(0.1,0.1,0.1,0.8),
                      item6=c(0.2,0.9,0.9,0.9),
                      item7=c(0.1,0.45,0.45,0.8),
                      item8=c(0.1,0.28,0.28,0.8),
                      item9=c(0.1,0.4,0.4,0.8),
                      item10=c(0.1,0.2,0.3,0.4,0.4,0.5,0.7,0.9))

set.seed(12345)
N <- 500
Q <- sim10GDINA$simQ
# When simulating data using catprob.parm argument,
# it is not necessary to specify model and type
sim <- simGDINA(N,Q,catprob.parm = itemparm.list)

# Example 6
# Data simulation (all CDMs)
# using delta parameters in list format
# ################################################################
delta.list <- list(c(0.2,0.7),
c(0.1,0.7),
c(0.1,0.8),
c(0.1,0.7),
c(0.1,0.8),
c(0.2,0.3,0.2,0.1),
c(0.1,0.35,0.35),
c(-1.386294,0.9808293,1.791759),
c(-1.609438,0.6931472,0.6),
c(0.1,0.1,0.2,0.3,0,0,0,0.0,0.1,0.1))
model <- c("GDINA","GDINA","GDINA","DINA","DINO","GDINA","ACDM","LLM","RRUM","GDINA")
N <- 500
Q <- sim10GDINA$simQ
# When simulating using delta.parm argument, model needs to be
# specified
sim <- simGDINA(N,Q,delta.parm = delta.list, model = model)

# Example 7
# Data simulation (higher order DINA model)
# ################################################################
Q <- sim30GDINA$simQ
gs <- matrix(0.1,nrow(Q),2)
N <- 500
set.seed(12345)
theta <- rnorm(N)
K <- ncol(Q)
lambda <- data.frame(a=rep(1,K),b=seq(-2,2,length.out=K))
sim <- simGDINA(N,Q,gs.parm = gs, model="DINA", att.dist = "higher.order", higher.order.parm = list(theta = theta,lambda = lambda))

# Example 8
# Data simulation (higher-order CDMs)
# ################################################################
Q <- sim30GDINA$simQ
gs <- matrix(0.1,nrow(Q),2)
models <- c(rep("GDINA",5),
rep("DINO",5),
rep("DINA",5))
simGDINA

```r
rep("DINA",5),
rep("ACDM",5),
rep("LLM",5),
rep("RRUM",5))
N <- 500
set.seed(12345)
theta <- rnorm(N)
K <- ncol(Q)
lambda <- data.frame(a=runif(K,0.7,1.3),b=seq(-2,2,length.out=K))
sim <- simGDINA(N,Q,gs.parm = gs, model=models, att.dist = "higher.order",
   higher.order.parm = list(theta = theta,lambda = lambda))

# Example 9
# Data simulation (higher-order model) using the multivariate normal threshold model

# See Chiu et al., (2009)

N <- 500
Q <- sim10GDINA$simQ
K <- ncol(Q)
gs <- matrix(0.1,nrow(Q),2)
cutoffs <- qnorm(c(1:K)/(K+1))
m <- rep(0,K)
v cov <- matrix(0.5,K,K)
diag(v cov) <- 1
simMV <- simGDINA(N,Q,gs.parm = gs, att.dist = "mvnorm",
   mvnorm.parm=list(mean = m, sigma = v cov,cutoffs = cutoffs))

# Example 10
# Simulation using user-specified att structure

# --- User-specified attribute structure ----#
Q <- sim30GDINA$simQ
K <- ncol(Q)
# divergent structure A1->A2->A3;A1->A4->A5;A1->A4->A6
diverg <- list(c(1,2),
   c(2,3),
   c(1,4),
   c(4,5))
struc <- att.structure(diverg,K)
# data simulation
N <- 1000
# data simulation
gs <- matrix(0.1,nrow(Q),2)
```
set.seed(12345)

N <- 500
Q <- sim30GDINA$simQ
J <- nrow(Q)
gs <- data.frame(guess=rep(0.1,J),slip=rep(0.1,J))
# Simulated different CDMs for different items
sim <- simGDINA(N,Q,gs.parm = gs,model = "GDINA",gs.args=list(mono.constraint=TRUE))

# True item success probabilities
extract(sim,what = "catprob.parm")

# True delta parameters
extract(sim,what = "delta.parm")

# simulated data
extract(sim,what = "dat")

# simulated attributes
extract(sim,what = "attribute")

set.seed(12345)

N <- 2000
# restricted Qc matrix
Qc <- sim20seqGDINA$simQ
#total number of categories
J <- nrow(Qc)
gs <- data.frame(guess=rep(0.1,J),slip=rep(0.1,J))
# simulate sequential DINA model
simseq <- simGDINA(N, Qc, sequential = TRUE, gs.parm = gs, model = "GDINA")

# True item success probabilities
extract(simseq,what = "catprob.parm")

# True delta parameters
extract(simseq,what = "delta.parm")
# simulated data
extract(simseq,what = "dat")

# simulated attributes
extract(simseq,what = "attribute")

# Example 13
# DINA model Attribute generated using categorical distribution

Q <- sim10GDINA$simQ
gs <- matrix(0.1, nrow(Q), 2)
N <- 5000
set.seed(12345)
prior <- c(0.1, 0.2, 0, 0, 0.2, 0, 0, 0.5)
sim <- simGDINA(N, Q, gs.parm = gs, model="DINA", att.dist = "categorical", att.prior = prior)
# check latent class sizes
table(sim$att.group)/N

# Example 14
# MS-DINA model

Q <- matrix(c(1, 1, 1, 1, 0,
1, 2, 0, 1, 1,
2, 1, 1, 0, 0,
3, 1, 0, 1, 0,
4, 1, 0, 0, 1,
5, 1, 1, 0, 0,
5, 2, 0, 1, 0), ncol = 5, byrow = TRUE)
d <- list(
  item1 = c(0.2, 0.7),
  item2 = c(0.1, 0.6),
  item3 = c(0.2, 0.6),
  item4 = c(0.2, 0.7),
  item5 = c(0.1, 0.8))

set.seed(12345)
sim <- simGDINA(N=1000, Q = Q, delta.parm = d,
  model = c("MSDINA", "MSDINA", "DINA", "DINA", "DINA", "MSDINA", "MSDINA"))

# simulated data
extract(sim, what = "dat")

# simulated attributes
extract(sim, what = "attribute")
startGDINA

Graphical user interface of the GDINA function

Description

An interactive Shiny application for running GDINA function. See Ma and de la Torre (2019) and de la Torre and Akbay (2019) for tutorials.

Usage

startGDINA()

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>

References


Examples

## Not run:
library(shiny)
library(shinydashboard)
startGDINA()

## End(Not run)
**unique_only**

*Unique values in a vector*

**Description**

Unique values in a vector

**Usage**

`unique_only(vec)`

**Arguments**

- `vec` a vector

**Value**

sorted unique values

**See Also**

`unique`

**Examples**

```r
c(4,2,3,5,4,4,4)
unique_only(vec)
# see the difference from unique
unique(vec)

vec <- letters[1:5]
unique_only(vec)
```

---

**unrestrQ**

*Generate unrestricted Qc matrix from an restricted Qc matrix*

**Description**

Generate unrestricted Qc matrix from an restricted Qc matrix

**Usage**

`unrestrQ(Qc)`
unrestrQ

Arguments

Qc  an restricted Qc matrix

Value

an unrestricted Qc matrix

Examples

Qc <- sim21seqDINA$simQc
Qc
unrestrQ(Qc)
Index

* datasets
  ecpe, 22
  frac20, 25
  sim10GDINA, 74
  sim10MCDINA, 75
  sim10MCDINA2, 75
  sim20seqGDINA, 76
  sim21seqDINA, 77
  sim30DINA, 77
  sim30GDINA, 78
  sim30pGDINA, 79

* package
  GDINA-package, 3

anova, 26
anova.GDINA(GDINA), 26
att.structure, 6
attributepattern, 7
auglag, 31
autoGDINA, 6, 8, 36, 62, 67, 69

bdiagMatrix, 12
bootSE, 13

CA, 14
cbind, 15
cjoint, 15
ClassRate, 16
coeff, 26, 31, 32
coeff.GDINA(GDINA), 26
designmatrix, 17
deviance.GDINA(GDINA), 26
dif, 4, 18
DTM, 4, 20, 36, 48

ecee, 22
extract, 10, 23, 26, 32
extract.GDINA(GDINA), 26
extract.itemFit(itemfit), 52
extract.modelcomp(modelcomp), 59
extract.Qval(Qval), 70
extract.simGDINA(simGDINA), 80
frac20, 25
GDINA, 3, 6, 8, 10, 11, 14, 19, 21, 26, 48, 55,
57, 62, 64–68, 72, 80, 82
GDINA-package, 3
GMSCDM, 4, 21, 36, 46
ILCA, 49
-indlogLik, 50
-indlogLik.GDINA(GDINA), 26
-indlogPost, 51
-indlogPost.GDINA(GDINA), 26
itemfit, 4, 23, 36, 52, 68
itemparm, 54
LC2LG, 55
logLik.GDINA(GDINA), 26

MCmodel, 56
modelcomp, 4, 8, 10, 11, 23, 36, 59
modelfit, 4, 36, 63
monocheck, 26, 64

nloptr, 31
nobs.GDINA(GDINA), 26
npar, 65
npar.GDINA(GDINA), 26

personparm, 26, 31, 32, 65
personparm.GDINA(GDINA), 26
plot, 26
plot.GDINA, 66
plot.itemfit, 67
plot.Qval, 68
Qval, 4, 8, 10, 11, 23, 36, 69, 70
rowMatch, 73

94
INDEX

score, 73
sim10GDINA, 74
sim10MCDINA, 75
sim10MCDINA2, 75
sim20seqGDINA, 76
sim21seqDINA, 77
sim30DINA, 77
sim30GDINA, 78
sim30pGDINA, 79
simDTM, 79
simGDINA, 36, 80
slsqp, 31
solnp, 31
startGDINA, 4, 26, 91
summary.autoGDINA (autoGDINA), 8
summary.dif (dif), 18
summary.GDINA (GDINA), 26
summary.itemfit (itemfit), 52
summary.modelcomp (modelcomp), 59
summary.Qval (Qval), 70
unique, 92
unique_only, 92
unrestrQ, 92

vcov.GDINA (GDINA), 26