Package ‘GDINA’

April 13, 2017

Type Package

Title The Generalized DINA Model Framework

Version 1.4.2

Date 2017-4-12

Description A set of psychometric tools for cognitive diagnostic analyses for both dichotomous and polytomous responses. Various cognitive diagnosis models can be estimated, include the generalized deterministic inputs, noisy and gate (GDINA) model by de la Torre (2011) <DOI:10.1007/s11336-011-9207-7>, the sequential GDINA model by Ma and de la Torre (2016) <DOI:10.1111/bmsp.12070>, and many other models they subsume. Joint attribute distribution can be saturated, higher-order 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.

License GPL-3

LazyData TRUE

Depends R (>= 3.1.0)

Imports alabama, data.table, graphics, ggplot2, MASS, nloptr, numDeriv, Rcpp (>= 0.12.1), Rsolnp, stats, utils

Suggests CDM, testthat, shiny, shinydashboard

LinkingTo Rcpp, RcppArmadillo

URL https://github.com/Wenchao-Ma/GDINA

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

RoxygenNote 6.0.1

Collate 'CR.R' 'ExportedFuncs.R' 'GDINA.R' 'GDIL.R' 'GDINA-package.R'
'HO.R' 'HObsupplement.R' 'M.R' 'M.mono.R' 'Mstep.R'
'Msupplement.R' 'RcppExports.R' 'anova.GDINA.R' 'attr.struc.R'
'autoGDINA.R' 'designmatrix.R' 'dif.R' 'ecpe.R'
'extract.GDINA.R' 'extract.QVal.R' 'itemfit.R'
'extrac.itemfit.R' 'modelcomp.R' 'extract.modelcomp.R'
'simGDINA.R' 'extract.simGDINA.R' 'frac20.R'
'heatplot.itemfit.R' 'hoparm.R' 'initials.R' 'itemparm.GDINA.R'
'monocheck.R' 'personparm.GDINA.R' 'plotIRF.GDINA.R'
R topics documented:

GDINA-package .......................................................... 3
att.structure ............................................................ 5
attributepattern ....................................................... 6
autoGDINA ............................................................... 7
bdiagMatrix ............................................................. 10
cjoint ................................................................. 11
ClassRate ............................................................... 12
designmatrix ........................................................... 13
dif ...................................................................... 14
ecpe ................................................................. 16
extract ................................................................. 17
frac20 ................................................................. 19
GDINA ................................................................. 20
heatplot ............................................................... 36
hoparm ................................................................. 36
indlogLik .............................................................. 37
indlogPost ............................................................. 37
itemfit ................................................................. 38
itemparm .............................................................. 40
LC2LG ................................................................. 41
mesaplot ............................................................... 42
modelcomp ............................................................ 43
monocheck ........................................................... 45
npar .................................................................. 46
personparm .......................................................... 46
plotIRF .............................................................. 47
Qval ................................................................. 47
rowCount ............................................................ 49
rowMatch ............................................................ 50
score ................................................................. 51
sim10GDINA .......................................................... 51
sim20seqGDINA ..................................................... 52
sim21seqDINA ....................................................... 52

NeedsCompilation yes
Author Wenchao Ma [aut, cre],
   Jimmy de la Torre [aut]
Maintainer Wenchao Ma <wenchao.ma@rutgers.edu>
Repository CRAN
Date/Publication 2017-04-13 17:57:08 UTC
The Generalized DINA Model Framework

Description

For conducting CDM analysis within the G-DINA model framework

Details

This package provides a framework for a series of cognitive diagnostic analyses for dichotomous and polytomous responses.

Various cognitive diagnosis models (CDMs) can be calibrated using the `gdina` function, including the G-DINA model, 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). 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) is fitted and all aforementioned CDMs can be used as the processing function 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 de la Torre (2009), de la Torre (2011), Ma, Iaconangelo, & de la Torre (2016) and Ma, & de la Torre (2016). The higher-order attribute structure (de la Torre, & Douglas, 2004), polytomous attributes (Chen, & de la Torre, 2013) and attribute structures (e.g., linear, divergent) can also be accommodated. Monotonicity constraints for item/category success probabilities can also be specified.

Q-matrix validation (de la Torre, & Chiu, 2016; see `qval`), item and model fit statistics (Chen, de la Torre, & Zhang, 2013; see `itemfit`), model comparison at test and item level (de la Torre, & Lee, 2013; Ma, Iaconangelo, & de la Torre, 2016; see `modelcomp`), and differential item functioning (Hou, de la Torre, & Nandakumar, 2014; see `dif`) can also be conducted.

Author(s)

Wenchao Ma, Rutgers University, <wenchao.ma@rutgers.edu>
Jimmy de la Torre, The University of Hong Kong
References


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
att.structure  Generate hierarchical attribute structures

Description
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, 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
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, Rutgers University, <wenchao.ma@rutgers.edu>
Jimmy de la Torre, The University of Hong Kong

See Also
GDINA, autoGDINA

Examples
## 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))
att.structure(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))
att.structure(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))
att.structure(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))
att.structure(unstru,K)

## See Example 4 and 5 in GDINA function

## End(Not run)

---

**attributeattern** *Generate all possible attribute patterns*

**Description**

This function generates all possible attribute patterns. The Q-matrix is needed when any attributes are polytomous.

**Usage**

`attributeattern(K, poly = F, Q = NULL)`

**Arguments**

- `K` number of attributes
- `poly` logical; is Q matrix polytomous?
- `Q` Q-matrix; required when Q-matrix is polytomous
Value

attribute profiles for $2^K$ latent classes

Author(s)

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

Examples

```r
attributepattern(3)
q <- matrix(scan(text = "0 1 2 1 0 1 1 2 0"), ncol = 3)
q
attributepattern(ncol(q), poly=TRUE, q)
q <- matrix(scan(text = "0 1 1 0 1 1 1 0"), ncol = 3)
q
attributepattern(ncol(q), poly=TRUE, q)
```

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 `gdina` function; then, the Q-matrix is validated using the function `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 `modelcomp`. Lastly, the selected models are calibrated based on the suggested Q-matrix using the `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 Details for the rules of item-level model selection.

Usage

```r
autoGDINA(dat, Q, modelselection = TRUE, Qvalid = TRUE,
          reducedCDM = c("DINA", "DINO", "ACDM", "LLM", "RRUM"),
          alpha.level = 0.05,
          modelselectionrule = "simpler", eps = 0.95, GDINA1.option = list(),
          GDINA2.option = list(), CDM.option = 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 \( J \times K \) item or category and attribute association matrix, where \( J \) represents the number of items or nonzero categories and \( K \) represents the number of attributes. For binary attributes, entry 1 indicates that the attribute is measured by the item, and 0 otherwise. For polytomous attributes, non-zero elements indicate the level of attributes that are needed for an individual to answer the item correctly (see Chen, & de la Torre, 2013). Note that for polytomous items, the sequential G-DINA model is used and either restricted or unrestricted category-level Q-matrix is needed. In the category-level Q-matrix, 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. See Examples.

`modelselection` logical; conducting model selection or not?

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

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

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

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

`eps` cut-off value for PVAF if `Qvalid=TRUE`. The default is 0.95.

`GDINA1.option` options for initial G-DINA calibration

`GDINA2.option` options for second G-DINA calibration

`CDM.option` options for final calibration

`object` estimated 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 model selection rule 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 dis-similar the reduced model is from the G-DINA model. See Ma, Iaconangelo, and
de la Torre (2016).

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, Rutgers University, <wenchao.ma@rutgers.edu>
Jimmy de la Torre, The University of Hong Kong

References

Ma, W., Iaconangelo, C., & de la Torre, J. (2016). Model similarity, model selection and attribute

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")
out1
summary(out1)
AIC(out1$CDM.obj)

# using the other selection rule
out11 <- autoGDINA(dat,misQ,modelselectionrule="simpler",reducedCDM = c("DINO","DINA"))
out11
summary(out11)

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

# -- Only consider some reduced CDMs
out2 <- autoGDINA(dat=misQ,reducedCDM = c("RRUM","LLM"))

# Disable Q-matrix validation
out3 <- autoGDINA(dat = dat, Q = misQ, Qvalid = FALSE, alpha.level=0.01)
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
cjoint

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)

cjoint v1 v2 v3 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)
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.

Examples

```r
## Not run:
N <- 2000
# ---- Simulated G-DINA model ---- #

# using probability of success in matrix format ---- #
# model does not matter if item parameter is probability of success
Q <- sim30GDINA$simQ
J <- nrow(Q)
gs <- matrix(c(0,1,1,2)

set.seed(12345)
sim <- simGDINA(N,Q,gs.parm = gs)
GDINA.est <- GDINA(sim$dat,Q)

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

## End(Not run)
```
designmatrix

Design matrix for parameter transformation

Description

This function calculates the design matrix $M_j$ in de la Torre (2011), which can be useful for parameter transformation between probability of success and delta.

Usage

```r
designmatrix(Kj, model = "GDINA")
```

Arguments

- `Kj`: the number of attributes for item j
- `model`: the model fitted to item j; it can be "GDINA", "DINA", "DINO", "ACDM", "LLM", or "RRUM". The default is "GDINA".

Value

A design matrix ($M_j$) which plays a critical role of transforming probability of success with delta parameters. See de la Torre (2011) for details.

References


Examples

```r
# transform probability of success to delta
# based on saturated GDINA model
# assuming an item with 2 attributes has the following
# probability of success for 00, 10, 01 and 11
pj <- c(0.2, 0.4, 0.6, 0.8)
Mj <- designmatrix(2)
# delta parameters can be calculated in this way:
deltaj <- solve(Mj) %*% pj
# for reduced CDMs, OLS or WLS may be used
```
Differential item functioning for cognitive diagnosis models

Description

This function is used to detect differential item functioning based on the models estimated in the GDINA function using the Wald test (Hou, de la Torre, & Nandakumar, 2014)

Usage

dif(dat, Q, group, method = "wald", p.adjust.methods = "bonferroni",
    LR.type = "free.all", difitem = "all", parm = "delta", digits = 4,
    SE.type = 2, ...)

# S3 method for class 'dif'
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 \(J \times K\) item or category and attribute association matrix, where \(J\) represents the number of items or nonzero categories and \(K\) represents the number of attributes. For binary attributes, entry 1 indicates that the attribute is measured by the item, and 0 otherwise. For polytomous attributes, non-zero elements indicate the level of attributes that are needed for an individual to answer the item correctly (see Chen, & de la Torre, 2013). Note that for polytomous items, the sequential G-DINA model is used and either restricted or unrestricted category-level Q-matrix is needed. In the category-level Q-matrix, 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. See Examples.

group a scalar indicating which column in dat is group indicator or a numerical vector indicating the group each individual belongs to. If it is a vector, its length must be equal to the number of individuals. Only at most two groups can be handled currently.

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.

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. "bonferroni" is the default.

LR.type Type of likelihood ratio test for DIF detection. It can be 'free.all' or 'free.one'.

difitem Items for the DIF detection. By default, all items will be examined.
parm The type of parameters associated with the Wald test for the DIF detection. It can be either "itemprob" or "delta" for item probabilities and delta parameters, respectively.
digits How many decimal places in each number? The default is 4.
SE.type Type of standard error estimation methods for the Wald test.
... Other arguments passed to GDINA function for model calibration
object estimated GDINA object for various S3 methods

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

Methods (by generic)
• summary: print summary information

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

References

See Also
GDINA

Examples
## Not run:
set.seed(123456)
N <- 1000
Q <- simGHDINA$simQ
gs <- matrix(c(0.1,0.2,
  0.1,0.2,
  0.1,0.2,
  0.1,0.2,
  0.1,0.2,
  0.1,0.2,
  0.1,0.2,
  0.1,0.2),ncol = 2, byrow = TRUE)
# By default, individuals are simulated from uniform distribution
# and deltas are simulated randomly
sim1 <- simGDINA(N,Q,gs.parm = gs,model="DINA")
**ecpe**

*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 (2013), and Templin and Bradshaw (2014), among others.

**Usage**

ecpe

**Format**

A list of responses and Q-matrix with components:

- dat: Responses of 2922 examinees to 28 items.
- Q: The $28 \times 3$ 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.

**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 (2013)
itemparm(mod2,"rrum")

## End(Not run)
```

---

**extract**

**extract elements from objects of various classes**

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**
- **att.prior** attribute prior weights for calculating marginalized likelihood in the last iteration
- **att.str** argument att.str
- **BIC**
- **call** function call
- **catprob.parm** category success probability for each latent group; the same as itemprob.parm for dichotomous response items.
- **catprob.se** SE associated with the category success probability for each latent group.
- **catprob.cov** variance-covariance matrix of item endorsement probabilities for all items
- **conv.crit** argument conv.crit
- **dat** item responses analyzed
- **delta.parm** delta parameters for each category
- **delta.cov** Convariance matrix associated with the delta parameters.
- **delta.se** SE associated with the delta parameters for each latent group.
- **deviance** deviance: -2 times observed log-likelihood value
- **dif.LL** absolute change in deviance in the last EM iteration
- **dif.p** max absolute change in success probabilities in the last EM iteration
digits  argument digits
discrim  GDINA discrimination index
empirical  argument empirical
d.end.time  end time
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
higher.order.method  argument higher.order$method
higher.order.model  argument higher.order$model
HO.parm.history  HO.parm.history in diagnosis mode
initial.catprob  initial item category probability parameters
iter.history  iter.history in diagnosis mode
item.names  argument item.names
itemprob.history  itemprob.history in diagnosis mode
itemprob.parm  item success probability for each latent group
itemprob.se  SE associated with the item success probability for each latent group.
LCprob.parm  category success probability for each latent class
logLik  observed log-likelihood value
loglikelihood.i  log-likelihood for each examinee
likepost.history  likepost.history in diagnosis mode
logposterior.i  log-posteriori for each examinee
maxitr  argument maxitr
models  fitted CDMs for each item/category
mono.constraint  argument mono.constraint
natt  number of attributes
ncat  number of categories excluding category zero
ngroup  number of groups
nitem  number of items
nitr  number of iterations
nobs  number of individuals
npar  number of parameters
npar.item  number of item parameters
npar.att  number of attribute parameters
nstarts  argument nstarts
prevalence  prevalence of each attribute
posterior.prob  posterior weights for each latent class
Q  Q-matrix
Usage

extract(object, what, ...)

Arguments

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

frac20        Tatsuoka’s fraction subtraction data

Description

Fraction Subtraction data consists of responses of 536 examinees to 20 items measuring 8 attributes. The data set is obtained from http://onlinelibrary.wiley.com/journal/10.1111/(ISSN)1467-9876/homepage/series_c_datasets.htm.

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

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)
```

GDINA

Calibrate dichotomous and polytomous responses

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) 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), and the linear logistic model (LLM; Maris, 1999) 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). Different models can be fitted to different items in a single test. The attributes can be either dichotomous or polytomous (Chen & de la Torre, 2013). Joint attribute distribution can be saturated, structured or higher-order (de la Torre & Douglas, 2004) when attributes are binary. Marginal maximum likelihood method with Expectation-Maximization (MMLE/EM) algorithm is used for item parameter estimation.

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

To extract higher-order parameters, use method `hoparm`.

To extract lower-order structural (item) parameters, use method `itemparm`.

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

Usage

GDINA(dat, Q, model = "GDINA", sequential = FALSE, att.dist = "saturated",
att.prior = NULL, att.str = FALSE, mono.constraint = FALSE,
group = NULL, verbose = 1, catprob.parm = NULL, lower.p = 1e-04,
upper.p = 0.9999, item.names = NULL, nstarts = 1, conv.crit = 0.001,
lower.prior = -1, conv.type = "max.p.change", maxitr = 1000,
digits = 4, diagnosis = 0, Mstep.warning = FALSE, optimizer = "all",
```
randomseed = 123456, smallncorrection = c(5e-04, 0.001),
higher.order = list(model = "Rasch", method = "MMLE", nquad = 19, type =
"testwise", slope.range = c(0.1, 5), intercept.range = c(-3, 3), slope.prior =
c(0, 0.25), intercept.prior = c(0, 1)), optim.control = list())

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

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

## S3 method for class 'GDINA'
hoparm(object, withSE = FALSE, theta.est = FALSE,
digits = 4, ...)

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

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

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

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

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

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

## S3 method for class 'GDINA'
mpar(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 $J \times K$ item or category and attribute association matrix, where $J$ represents the number of items or nonzero categories and $K$ represents the number of attributes. For binary attributes, entry 1 indicates that the attribute is measured by the item, and 0 otherwise. For polytomous attributes, non-zero elements indicate the level of attributes that are needed for an individual to answer the item correctly (see Chen, & de la Torre, 2013). Note that for polytomous items, the sequential G-DINA model is used and either restricted or unrestricted category-level Q-matrix is needed. In the category-level Q-matrix, 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. 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", and "RRUM". It is also possible to specify CDMs using numbers. Particularly, 0,1,2,3,4 and 5 represents "GDINA", "DINA", "DINO", "ACDM", "LLM", and "RRUM", respectively.

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

att.dist  How is the joint attribute distribution estimated? It can be saturated, indicating that the proportion parameter for each permissible latent class is estimated separately; higher order, indicating that a higher-order joint attribute distribution is assumed (higher-order model can be specified in higher.order argument); or fixed, indicating that the weights specified in att.prior argument are fixed in the estimation process. If att.prior is not specified, a uniform joint attribute distribution is employed initially. If different groups have different joint attribute distributions, specify att.dist as a character vector with the same number of elements as the number of groups.

att.prior  A vector of length $2^K$ for single group estimation, or a matrix of dimension $2^K \times$ no. of groups to specify attribute prior distribution for $2^K$ latent classes for all groups. Only applicable for dichotomous attributes. The sum of all elements does not have to be equal to 1; however, it will be transformed so that the sum is equal to 1 before model calibration. The label for each latent class can be obtained by calling attributepattern(K). See examples for more info.

att.str  logical; are attributes structured?

mono.constraint  logical; TRUE indicates that $P(\alpha_1) \leq P(\alpha_2)$ if for all $k$, $\alpha_{1k} \leq \alpha_{2k}$. Can be a vector for each item or nonzero category or a scalar which will be used for all items to specify whether monotonicity constraint should be added.

group  a scalar indicating which column in dat is group indicator or a numerical vector indicating the group each individual belongs to. If it is a vector, its length must be equal to the number of individuals. Only at most two groups can be handled currently.

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.
| **catprob.parm** | A list of initial success probability parameters for each nonzero category. |
| **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. The default is $1e^{-4}$ for all items. |
| **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. The default is 0.9999 for all items. |
| **item.names** | A vector giving the item names. By default, items are named as "Item 1", "Item 2", etc. |
| **nstarts** | how many sets of starting values? The default is 1. |
| **conv.crit** | The convergence criterion for max absolute change in item parameters or deviance. |
| **lower.prior** | The lower bound for prior weights. Only applicable for nonstructured attributes. The default value is -1, which means the lower bound is $1/2^K/100$. |
| **conv.type** | How is the convergence criterion evaluated? Can be "max.p.change", indicating the maximum absolute change in success probabilities, or "dev.change", representing the absolute change in deviance. |
| **maxitr** | The maximum number of EM cycles allowed. |
| **digits** | How many decimal places in each number? The default is 4. |
| **diagnosis** | Run in diagnostic mode? If it is 1 or 2, some intermediate results obtained in each iteration can be extracted. |
| **Mstep.warning** | Logical; Should the warning message in Mstep, if any, be output immediately. |
| **optimizer** | A string indicating which optimizer should be used in M-step. |
| **randomseed** | Random seed for generating initial item parameters. The default random seed is 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. |
| **higher.order** | A list specifying the higher-order joint attribute distribution with the following components: (1) 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 (see Details). "Rasch" is the default model when att.dist = "higher.order". (2) method - a character indicating the algorithm for the higher-order structural parameter estimation; Can be either "BL", "MLE", which is the default, or "BMLE", which allows parameter priors to be imposed. (3) nquad - a scalar specifying the number of integral nodes. (4) type - a character specifying whether all higher-order structural parameters are estimated at the same time (i.e., type="testwise") or estimated attribute by attribute (i.e., type="attwise", |
only applicable when method="MML" or method="BML"). (5) slope.range - a vector of length two specifying the range of slope parameters. (6) intercept.range - a vector of length two specifying the range of intercept parameters. (7) slope.prior - a vector of length two specifying the mean and variance of log(slope) parameters, which are assumed normally distributed. (8) intercept.prior - a vector of length two specifying the mean and variance of intercept parameters, which are assumed normally distributed.

optim.control Control options for optimizers in the M-step. Only available when optimizer is one specific optimization method, including BFGS from optim, slsqp, solnp and auglag. For the auglag method, optim.control specifies control.outer.

object estimated GDINA object for various S3 methods

... additional arguments

what argument for various S3 methods

SE.type type of standard errors.

withSE argument for method itemparm; show standard errors or not?

theta.est logical; Estimating higher-order person ability or not? The default is FALSE.

Value

GDINA returns an object of class GDINA. Methods for GDINA objects include extract for extracting various components, itemparm for extracting item parameters, personparm for calculating 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
- extract: extract various elements of GDINA estimates
- hoparm: extract higher-order parameters
- itemparm: extract various item parameters
- personparm: calculate person attribute patterns and higher-order ability
- AIC: calculate AIC
- BIC: calculate BIC
- logLik: calculate log-likelihood
- deviance: calculate deviance
- 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 \( 2^K_j \) latent classes into \( 2^{K_j^*} \) latent groups with unique success probabilities on item \( j \), where \( K_j^* = \sum_{k=1}^{K_j} q_{jk} \).

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, \( \alpha_{lj}^* = (\alpha_{l1}, \alpha_{lK_j^*}) \). 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 \( P(X_{ij} = 1|\alpha_{lj}^*) \) 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

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

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 \( \alpha_{lk'} \), \( \delta_{j12...K_j^*} \) 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

\[
\eta_{ij} = \prod_{k=1}^{K_j} \alpha_{ik}^{q_{jk}}
\]

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 \( \delta_{j12...K_j^*} \) need to be fixed to zero, that is,

\[
P(\alpha_{lj}^*) = \delta_{j0} + \delta_{j12...K_j^*} \prod_{k=1}^{K_j^*} \alpha_{lk}
\]

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

DINO model The DINO model can be given by

\[
P(\alpha_{lj}^*) = \delta_{j0} + \delta_{j1} I(\alpha_{lj}^* \neq 0)
\]
where $I(\cdot)$ is an indicator variable. The DINO model is also a constrained identity link GDINA model. As shown by de la Torre (2011), the appropriate constraint is

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

for $k = 1, \ldots, K_j^*, k' = 1, \ldots, K_j^* - 1$, and $k'' > k', \ldots, K_j^*$. 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 GDINA model, respectively. Specifically, the A-CDM can be formulated as

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

The item response function for LLM can be given by

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

and lastly, the RRUM, can be written as

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

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).

Model Estimation

The MMLE/EM algorithm is implemented in this package. For G-DINA, DINA and DINO models, closed-form solutions can be found. Specifically, for the G-DINA model,

$$P(\alpha_{lj}^*) = R_{jl}/N_{jl}$$

where $R_{jl}$ is the expected number of examinees with attribute pattern $\alpha_{lj}^*$ answering item $j$ correctly and $N_{jl}$ is the expected number of examinees with attribute pattern $\alpha_{lj}^*$. For DINA or DINO model, $R_{jl}$ and $N_{jl}$ are collapsed for latent classes having the same probability of success. 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. See Ma, Iaconangelo and de la Torre (2016) for details. The selection of optimization techniques mainly depends on whether some specific constraints need to be added. It should be noted that adding monotone constraints to the G-DINA model may dramatically increase running time especially when the number of required attributes are large.

The sequential G-DINA model can be estimated as in Ma & de la Torre (2016a) using optimization techniques. However, Ma & de la Torre (2016b) found that the sequential G-DINA, DINA and DINO models can be estimated using close-form solutions, which can be implemented in a straightforward manner using the observation-coding (Tutz, 1997).
For estimating the joint attribute distribution, by default, an empirical Bayes method (saturated; Carlin & Louis, 2000) is adopted, which is referred to as the saturated attribute structure. Specifically, the prior distribution of joint attributes is uniform at the beginning, and then updated after each EM iteration based on the posterior distribution.

The joint attribute distribution can also be modeled using some higher-order IRT models, which is referred to as higher-order attribute structure. The higher-order attribute structure was originally proposed by de la Torre and Douglas (2004) for the DINA model. It has been extended in this package for the G-DINA model, DINA, DINO, A-CDM, LLM and RRUM. 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

\[
P(\alpha|\theta_i, \lambda) = \prod_k P(\alpha_k|\theta_i, \lambda)
\]

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 \( K_0 + 1 \), and \( 2 \times K \) parameters for the Rasch model, 1PL model and 2PL model, respectively. For polynomials 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, Rutgers University, <wenchao.ma@rutgers.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 `itemfit` for item fit analysis, `Qval` for Q-matrix validation, `modelcomp` for item level model comparison and `simGDINA` for data simulation. 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  #
#   #
###################################################

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

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

# 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
```
GDINA

#plot item response functions for item 10
plotIRF(mod1, item = 10)
plotIRF(mod1, item = 10, errorbar = TRUE) # with error bars
plotIRF(mod1, item = c(6,10))

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

# Model comparison at 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")
# slip and guessing
itemparm(mod3, what = "gs") # guess and slip parameters
itemparm(mod3, what = "gs", withSE = TRUE) # guess and slip parameters and standard errors

# Model comparison at 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")

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

# ------- 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)
GDINA

###################################################################
# Example 2. #
# Model estimations #
# With monotonocity constraints #
###################################################################
dat <- sim0GDINA$simdat
Q <- sim0GDINA$simQ
# for item 10 only
mod11 <- GDINA(dat = dat, Q = Q, model = "GDINA",mono.constraint = c(rep(FALSE,9),TRUE))
mod11a <- GDINA(dat = dat, Q = Q, model = "DINA",mono.constraint = TRUE)
mod11b <- GDINA(dat = dat, Q = Q, model = "ACDM",mono.constraint = TRUE)
mod11c <- GDINA(dat = dat, Q = Q, model = "LLM",mono.constraint = TRUE)
mod11d <- GDINA(dat = dat, Q = Q, model = "RRUM",mono.constraint = TRUE)
itemparm(mod11d,"delta")
itemparm(mod11d,"rrum")

###################################################################
# Example 3. #
# Model estimations #
# With Higher-order att structure #
###################################################################
dat <- sim0GDINA$simdat
Q <- sim0GDINA$simQ
# --- Higher order G-DINA model ---#
mod12 <- GDINA(dat = dat, Q = Q, model = "DINA",
att.dist="higher.order",higher.order=list(method="MMLE",nquad=31))
hoest=hoparm(mod12) # extract higher-order parameters
hoest$theta # ability
hoest$lambda # structural parameters
# --- Higher order DINA model ---#
mod22 <- GDINA(dat = dat, Q = Q, model = "DINA",
att.dist="higher.order",higher.order=list(method="BMLE"))
# --- Higher order DINO model ---#
mod23 <- GDINA(dat = dat, Q = Q, model = "DINO",att.dist="higher.order")
# --- Higher order ACDM model ---#
mod24 <- GDINA(dat = dat, Q = Q, model = "ACDM",
att.dist="higher.order",higher.order=list(model="1PL"))
# --- Higher order LLM model ---#
mod25 <- GDINA(dat = dat, Q = Q, model = "LLM",att.dist="higher.order")
# --- Higher order RRUM model ---#
mod26 <- GDINA(dat = dat, Q = Q, model = "RRUM",att.dist="higher.order")

###################################################################
# Example 4. #
# Model estimations #
# With user-specified att structure#

# --- User-specified attribute priors ----#
# prior distribution is fixed during calibration
# Assume each of 000, 100, 010 and 001 has probability of 0.1
# and each of 110, 101, 011 and 111 has probability of 0.15
# Note that the sum is equal to 1
#
# 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.
# dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
modp1 <- GDINA(dat = dat, Q = Q, att.prior = prior, att.dist = "fixed")
# See the posterior weights
extract(modp1,what = "posterior.prob")
extract(modp1,what = "att.prior")

# ----Linear structure of attributes ------#
# Assuming A1 -> A2 -> A3
Q <- matrix(c(1, 0, 0,
              1, 1, 0,
              1, 1, 0,
              1, 1, 1,
              1, 1, 1,
              1, 0, 0,
              1, 1, 0,
              1, 1, 0,
              1, 1, 1,
              1, 1, 1,
              1, 1, 1), ncol = 3, byrow = TRUE)
# item parameters for DINA model (guessing and slip)
gs <- matrix(rep(0.1, 24), ncol = 2)
N <- 5000
# attribute simulation
att <- rbind(matrix(0, nrow = 500, ncol = 3),
              matrix(rep(c(1, 0, 0), 1000), ncol = 3, byrow = TRUE),
              matrix(rep(c(1, 1, 0), 1000), ncol = 3, byrow = TRUE),
              matrix(rep(c(1, 1, 1), 2500), ncol = 3, byrow = TRUE))
# data simulation
simD <- simGDINA(N, Q, gs.parm = gs, model = "DINA", attribute = att)
dat <- simD$dat
# setting structure: A1 -> A2 -> A3
# note: latent classes with prior 0 are assumed impossible
prior <- c(0.1, 0.2, 0.0, 0.2, 0, 0, 0.5)
out <- GDINA(dat, Q, att.prior = prior, att.str = TRUE, att.dist = "fixed", model = "DINA")
# check posterior dist.
extract(out, what = "posterior.prob")
extract(out, what = "att.prior")

out2 <- GDINA(dat, Q, att.prior = prior, att.str = TRUE, att.dist = "saturated", model = "DINA")
# check posterior dist.
extract(out2, what = "posterior.prob")
extract(out2,what = "att.prior")

# Example 5.      #
# Model estimations  #
# With 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
true.lc <- sample(c(1:2*K),N,replace=TRUE,prob=struc$att.prob)
table(true.lc) #check the sample
ture.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 = "DINA",attribute = true.att)
dat <- extract(simD,"dat")

modp1 <- GDINA(dat = dat, Q = Q, att.prior = struc$att.prob, att.str = TRUE, att.dist = "saturated")
modp1
# Note that fixed priors were used for all iterations
extract(modp1,what = "att.prior")
# Posterior weights were slightly different
extract(modp1,what = "posterior.prob")
modp2 <- GDINA(dat = dat, Q = Q, att.prior = struc$att.prob, att.str = TRUE, att.dist = "fixed")
modp2
extract(modp2,what = "att.prior")
extract(modp2,what = "posterior.prob")

# check initials to see the format for initial item parameters
initials <- sim10GDINA$simItempar
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod.ini <- GDINA(dat,Q,catprob.parm = initials)
extract(mod.ini,"initial.catprob")
# Example 7.

## Model estimation

## Without M-step

# -------------Fix User-specified item parameters
# Item parameters are not estimated
# Only person attributes are estimated
# attribute prior distribution matters if interested in the marginalized likelihood
dat <- frac20$dat
Q <- frac20$Q
mod.initial <- GDINA(dat,Q,maxit=20) # estimation- only 10 iterations for illustration purposes
par <- itemparm(mod.initial,digits=8)
weights <- extract(mod.initial,"posterior.prob",digits=8) #posterior weights
# use the weights as the priors
mod.fix <- GDINA(dat,Q,catprob.parm = par,att.prior=c(weights),maxit=0) # re-estimation
anova(mod.initial,mod.fix) # very similar - good approximation most of time
# prior used for the likelihood calculation for the last step
priors <- extract(mod.initial,"att.prior")
# use the priors as the priors
mod.fix2 <- GDINA(dat,Q,catprob.parm = par,att.prior=priors,maxit=0) # re-estimation
anova(mod.initial,mod.fix2) # identical results

# Example 8.

## polytomous attribute

## model estimation

## see Chen, de la Torre 2013

# polytomous 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
# Item Cat A1 A2 A3 A4 A5
GDINA

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

Q <- sim0GDINA$simQ

# parameter simulation
# Group 1 - female
N1 <- 2000
gs1 <- matrix(rep(0.1,2*nrow(Q)),ncol=2)
# Group 2 - male
N2 <- 2000
gs2 <- matrix(rep(0.2,2*nrow(Q)),ncol=2)

# data simulation for each group
sim1 <- simGDINA(N1,Q,gs.parm = gs1,model = "DINA")
sim2 <- simGDINA(N2,Q,gs.parm = gs2,model = "DINO")

# combine data
# see ?bdiagMatrix
dat <- bdiagMatrix(list(extract(sim1,"dat"),extract(sim2,"dat")),fill=NA)
Q <- rbind(Q,Q)
gr <- rep(c("female","male"),each=2000)
# Fit G-DINA model
mg.est <- GDINA(dat = dat,Q = Q,group = gr)
summary(mg.est)
extract(mg.est,"posterior.prob")

# Fit G-DINA model with different joint attribute dist.
mg.est2 <- GDINA(dat = dat,Q = Q,group = gr,
att.dist = c("saturated","fixed"))
summary(mg.est2)

## End(Not run)
```
heatplot  

Item fit plots

Description
Create plots of bivariate heatmap for item fit

Usage
heatplot(object, ...)

Arguments
object  model object of class itemfit
...

See Also
GDINA, itemfit

hoparm  

extract higher-order parameters

Description
Function to extract higher-order parameters when a higher-order model is fitted.

Usage
hoparm(object, withSE = FALSE, theta.est = FALSE, digits = 4, ...)

Arguments
object  estimated GDINA object returned from GDINA
withSE  estimate standard errors for lambda parameters or not?
theta.est  logical: Estimating higher-order person ability or not? The default is FALSE.
digits  how many decimal places for the output?
...

Value
a list with element theta for higher-order incidental (ability) parameters and lambda for higher-order structural parameters.
indlogLik

Author(s)

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

indlogLik

Extract log-likelihood for each individual

Description

Extract individual log-likelihood See GDINA for examples.

Usage

indlogLik(object, ...)

Arguments

object GDINA object
... additional arguments

indlogPost

Extract log posterior for each individual

Description

Extract individual log posterior See GDINA for examples.

Usage

indlogPost(object, ...)

Arguments

object GDINA object
... additional arguments
itemfit  

**Item fit statistics**

**Description**

Calculate item fit statistics (Chen, de la Torre, & Zhang, 2013)

**Usage**

itemfit(GDINA.obj, person.sim = "post", p.adjust.methods = "bonferroni",
        digits = 4, N.resampling = NULL, randomseed = 123456)

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

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

## 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. "bonferroni" is the default.
- **digits** How many decimal places in each number? The default is 4.
- **N.resampling** the sample size of resampling. By default, it is maximum of 1e+5 or 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:

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

Methods (by generic)

- \texttt{extract}: extract various elements from itemfit objects
- \texttt{heatplot}: plot bivariate heatmap for misfit detection
- \texttt{summary}: print summary information

Author(s)

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

References


Examples

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

eval2 <- GDINA(data, model="DINA")
itemfit2 <- itemfit(eval2)
# misfit heatmap
heatplot(itemfit2)
itemfit2

## End(Not run)

---

**itemparm**

*extract lower-order structural (item) parameters*

**Description**

Function to extract various item parameters, including "itemprob" for item success probabilities of each reduced attribute pattern, "catprob" for category success probabilities of each reduced attribute pattern, "LCprob" for item success probabilities of each attribute pattern, "gs" for guessing and slip parameters, "delta" for delta parameters, "rrum" for RRUM parameters when items are estimated using RRUM. Standard errors can be estimated if withSE = TRUE. See GDINA for examples.

**Usage**

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

**Arguments**

- **object**: estimated GDINA object returned from GDINA.
- **what**: what to show; It can be "itemprob" for item success probabilities of each reduced attribute pattern, "catprob" for category success probabilities of each reduced attribute pattern, "LCprob" for item success probabilities of each attribute pattern, "gs" for guessing and slip parameters, "delta" for delta parameters, "rrum" for RRUM parameters when items are estimated using RRUM. The default is "catprob".
- **withSE**: show standard errors or not?
- **SE.type**: Type of standard errors. Can be 1, 2 or 3, indicating outer product of gradient (OPG) estimates based on itemwise, incomplete or complete information matrix. See Philipp, Strobl, de la Torre, & Zeileis (2016). Currently, the OPG method based on the complete information matrix assumes that all latent classes are identifiable.
- **digits**: how many decimal places for the output?
- **...**: additional arguments
**Author(s)**

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

**References**


---

**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)
```

**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?

**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, Rutgers University, <wenchao.ma@rutgers.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)
```
mesaplot

Mesa plot for Q-matrix validation

Description

Mesa plot for Q-matrix validation

Usage

mesaplot(Qval.obj, item, type = "best", no.qvector = 10,
  data.label = TRUE, eps = 0.95, original.q.label = FALSE,
  auto ylim = TRUE, ...)

Arguments

Qval.obj 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

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)
mesaplot(out, item=item, data.label=FALSE, type="all")
mesaplot(out, item=10, type="best", eps=0.95)
mesaplot(out, item=10, type="all", no.qvector=5)
modelcomp	Item-level model comparison using Wald test

Description
This function evaluates whether the saturated G-DINA model can be replaced by reduced CDMs without significant loss in model data fit for each item. See de la Torre and Lee (2013), and Ma, Iaconangelo and de la Torre (2016) for details. 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
modelcomp(GDINA.obj, item = "all", DS = FALSE, SE.type = 2,
models = c("DINA", "DINO", "ACDM", "LLM", "RRUM"), varcov = NULL)

## S3 method for class 'modelcomp'
extract(object, what = c("wald", "wald.p", "df", "DS",
"models"), digits = 4, ...)

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

Arguments
GDINA.obj	An estimated model object of class GDINA
item	a vector of items to specify which items the Wald test is applied to
DS	whether dissimilarity index should be calculated? FALSE is the default.
SE.type	the type of standard error estimates.
models	a vector specifying which reduced CDMs are possible reduced CDMs for each item. The default is "DINA","DINO","ACDM","LLM", and "RRUM".
varcov	Optional; user specified variance-covariance matrix. If supplied, it must be a list of length J, giving the variance covariance matrix of item success probability for each item. The default is NULL, in which case, the estimated variance-covariance matrix from the GDINA function is used.
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
Value

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

    wald  wald statistics
    wald.p p-values associated with the wald statistics
    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, Rutgers University, <wenchao.ma@rutgers.edu>
    Jimmy de la Torre, The University of Hong Kong
    Wenchao Ma & Jimmy de la Torre

References


See Also

    GDINA, autoGDINA

Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
# --- GDINA model ---#
mod1 <- GDINA(dat = dat, Q = Q, model = "GDINA")
mod1
wmod1 <- modelcomp(mod1)
wmod1
# wald statistics
extract(wmod1,"wald")
# p values
extract(wmod1,"wald.p")
wmod1r <- modelcomp(mod1,DS=TRUE)
# dissimilarity index
extract(wmod1r,"DS")
wmod2 <- modelcomp(mod1,models = c("DINA","DINO"))
wmod2
wmod3 <- modelcomp(mod1,item=c(8,9,10),DS=FALSE)
```
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

```r
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 = "GDINA")
check <- monocheck(mod1)
check
mod2 <- GDINA(dat = dat, Q = Q, model = "GDINA", 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 of population parameters. See *GDINA* for examples.

**Usage**

`npar(object, ...)`

**Arguments**

- **object**: GDINA object
- **...**: additional arguments

---

personparm

*calculate lower-order incidental (person) parameters*

**Description**

Function to calculate various person attribute parameters, including "EAP", "MAP", and "MLE", for EAP, MAP and MLE estimates of attribute patterns, "mp" for marginal mastery probabilities. See *GDINA* for examples. To estimate higher-order person parameters, see *hoparm*.

**Usage**

`personparm(object, what = c("EAP", "MAP", "MLE", "mp"), 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.
- **digits**: number of decimal places.
- **...**: additional arguments

**Author(s)**

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

**References**

plotIRF

Plot item success probability

Description

Create plots of item/category success probability for each latent group

Usage

plotIRF(object, item, errorbar = FALSE, ...)

Arguments

- **object**: model object of class `GDINA` or `dif`
- **item**: a vector specifying which item(s) the plots are drawn for
- **errorbar**: add error bar to the plot?
- **...**: additional arguments

See Also

`GDINA`, `autoGDINA`, `dif`

Examples

```r
## Not run:
dat <- sim1@GDINA$simdat
Q <- sim1@GDINA@simQ
mod1 <- GDINA(dat = dat, Q = Q, model = "GDINA")
# plot item response functions for item 10
plotIRF(mod1, 10)
pplotIRF(mod1, 9, errorbar = TRUE)
## End(Not run)
```

Qval

Q-matrix validation

Description

Q-matrix validation for the G-DINA model based on de la Torre and Chiu (2016).
Usage

```r
Qval(GDINA.obj, method = "PVAF", eps = 0.95, digits = 4)
```

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

```r
# 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?
- **eps**: cutoff value for PVAF. 0.95 is the default.
- **digits**: How many decimal places in each number? The default is 4.
- **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, Rutgers University, <wenchao.ma@rutgers.edu>
Jimmy de la Torre, The University of Hong Kong

References

See Also

GDINA, mesaplot

Examples

```r
## Not run:
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
Q[10,] <- c(0,1,0)
mod1 <- GDINA(dat = dat, Q = Q, model = "GDINA")
out <- Qval(mod1,eps = 0.95)
out
#If many entries are modified, you may want to check
#the PVAF plot using the function plotPVAF or
#to change eps. eps = 0.9 or 0.8 seems another two
#reasonable choices.
extract(out,what = "PVAF")
#See also:
extract(out,what = "varsigma")
extract(out,what = "sug.Q")

# Draw a mesa plot
mesaplot(out,item=10,type="all",no.qvector=5)

## End(Not run)
```

---

**rowCount**

*Count and label unique rows in a data frame*

**Description**

Count and label unique rows in a data frame

**Usage**

`rowCount(df)`

**Arguments**

- `df` a data frame or matrix

**Value**

- `freq` the number of rows
- `group` the data frame with a column named `row.no` giving unique labels for all unique rows
Examples

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

---

**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(1L,12))
rowMatch(df,c(2,"B",1))
```
### Score

**Score function**

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

**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

### Simulated data, Q-matrix and item parameters (10 items, G-DINA model)

**Description**

Artificial Q-matrix and item parameters for a 10-item test measuring 3 attributes are given. Based on the Q-matrix and item parameters, responses of 1000 examinees were simulated using the random seed of 12345.

**Usage**

```r
sim10GDINA
```

**Format**

A list of Q-matrix and item parameters for a 10-item test with components:

- `simdat` simulated responses of 1000 examinees
- `simQ` artificial Q-matrix
- `simItempar` artificial item parameters (probability of success for each latent group)
sim20seqDINA  
Simulated data, Q-matrix and item parameters (20 items, sequential DINA model)

**Description**

Artificial Qc-matrix and item parameters for a 20-item test measuring 5 attributes are given. Based on the Qc-matrix and item parameters, responses of 2000 examinees were simulated using the random seed of 12345.

**Usage**

```r
sim20seqDINA
```

**Format**

A list of Q-matrix and item parameters for a 20-item test 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)

sim21seqDINA  
Simulated data and Qc-matrix based on the sequential DINA model

**Description**

Artificial Qc-matrix for a 21-item test measuring 5 attributes are given. Based on the Q-matrix, responses of 2000 examinees were simulated.

**Usage**

```r
sim21seqDINA
```

**Format**

A list of Qc-matrix and responses for a 30-item test with components:

- `simdat`: simulated responses of 2000 examinees
- `simQ`: artificial Qc-matrix
**sim30DINA**

*Simulated data Q-matrix and item parameters (30 items, DINA model)*

**Description**

Artificial Q-matrix and item parameters for a 30-item test measuring 5 attributes are given. Based on the Q-matrix and item parameters, responses of 1000 examinees were simulated using the random seed of 12345.

**Usage**

`sim30DINA`

**Format**

A list of Q-matrix and item parameters for a 30-item test with components:

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

---

**sim30GDINA**

*Simulated data, Q-matrix and item parameters (30 items, G-DINA model)*

**Description**

Artificial Q-matrix and item parameters for a 30-item test measuring 5 attributes are given. Based on the Q-matrix and item parameters, responses of 1000 examinees were simulated using the random seed of 12345.

**Usage**

`sim30GDINA`

**Format**

A list of responses, Q-matrix and item parameters for a 30-item test 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)
sim30pGDINA

Simulated data, Q-matrix and item parameters (30 items, polytomous G-DINA model)

Description

Artificial Q-matrix and item parameters for a 30-item test measuring 5 attributes are given. Based on the Q-matrix and item parameters, responses of 3000 examinees were simulated using the random seed of 12345.

Usage

sim30pGDINA

Format

A list of Q-matrix and item parameters for a 30-item test with components:

- simdat: simulated responses of 3000 examinees
- simQ: artificial Q-matrix
- simItempar: artificial item parameters (probability of success for each latent group)

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

simgdina(N, Q, gs.parm = NULL, model = "GDINA", sequential = FALSE, type = "random", catprob.parm = NULL, delta.parm = NULL, mono.constraint = TRUE, 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), 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 $J \times K$ item/category and attribute association matrix, where $J$ represents the number of items/categories and $K$ represents the number of attributes. For binary attributes, 1 denotes attributes are measured by items and 0 means attributes are not necessary. For polytomous attributes, non-zero elements indicate which level of attributes are needed. Note that for polytomous items, the sequential G-DINA model is used and either restricted or unrestricted category-level Q-matrix is needed. The first column represents the item number and the second column indicates the category number. See Examples.

- **gs.parm**
  - A matrix or data frame for guessing and slip parameters. It must be of dimension $J \times 2$, where the first column represents the guessing parameters (or $P(0)$), and the second column represents slip parameters (or $1 - P(1)$). This needs to be used in conjunction with the argument type if generating models include ACDM, LLM, or RRUM, and model.

- **model**
  - A vector for each item/category or a scalar which will be used for all items/categories to specify which model is fitted to each item/category. The possible options include "G-DINA", "DINA", "DINO", "ACDM", "LLM", and "RRUM". If model is a scalar, the specified model is fitted to all items. Different models can be assigned to different items or categories.

- **sequential**
  - logical; TRUE if the sequential model is used for polytomous responses simulation, and FALSE if there is no polytomously scored items.

- **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.

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

- **delta.parm**
  - A list of delta parameters for each latent group for each item or category.

- **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.

- **attribute**
  - optional user-specified person attributes. It is a $N \times K$ 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" or "mvnorm" for uniform, higher-order and multivariate normal distribution, 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.
**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` × `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`.

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

**object**  
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., \( P(\alpha^*_{lj}) \) or \( S(\alpha^*_{ljh}) \)) 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, Rutgers University, <wenchao.ma@rutgers.edu>
Jimmy de la Torre, The University of Hong Kong

References

Examples

```r
## Not run:
# Examples
# Data simulation (DINA)
N <- 500
Q <- sim30GDINA$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.param = 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.param = gs,model = "RRUM")

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

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

# Example 3
# Data simulation (LLM)
```
N <- 500
t <- simGDINA$n
Q <- simGDINA$n
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 equally
sim <- simGDINA(N,Q,gs parm = gs,model = "LLM",type="equal")
#check below for what the equal contribution means
extract(sim,what = "delta.parm")
# simulated data
eXtract(sim,what = "dat")
# simulated attributes
extract(sim,what = "attribute")

# Example 4
# Data simulation (all CDMs)
set.seed(12345)
N <- 500
Q <- simGDINA$n
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,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)
item parm.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 <- sim0GDINA$simQ
# When simulating data using catprob.parm argument,
# it is not necessary to specify model and type
sim <- simGDINA(N,Q,catprob.parm = item parm.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(0.1,0.1,0.2,0.3,0.0,0.0,0.1,0.1))

model <- c("GDINA","GDINA","GDINA","DINA","DINO","GDINA","ACDM","LLM","RRUM","GDINA")
N <- 500
Q <- sim0GDINA$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 <- sim0GDINA$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,lamba = lambda))

# Example 8 
# Data simulation (higher-order CDMs)

Q <- sim3DGDINA$simQ
gs <- matrix(0.1,nrow(Q),2)
models <- c(rep("GDINA",5),
            rep("DINO",5),
            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,lamba = 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
true.lc <- sample(c(1:2^K),N,replace=TRUE,prob=struct$att.prob)
table(true.lc) #check the sample
ture.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 = "DINA",attribute = true.att)

# (GDINA with monotonicity constraints)
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",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")

# Example 12
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",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 <- sim@seqGDINA$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 = "DINA")

# 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")

## End(Not run)

startGDINA Graphical user interface of the GDINA function

Description

An experimental interactive Shiny application for running GDINA function

Usage

startGDINA()

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
vec <- c(4, 2, 3, 5, 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 a restricted Qc matrix

Description
Generate unrestricted Qc matrix from a restricted Qc matrix

Usage
unrestrQ(Qc)
Arguments

\( Q_c \)  
an restricted \( Q_c \) matrix

Value

an unrestricted \( Q_c \) matrix

Examples

\[
\begin{align*}
Q_c & \leftarrow \text{sim2seqDINA$simQc} \\
Q_c & \quad \text{unrestrQ}(Q_c)
\end{align*}
\]
Index

+Topic **datasets**
  ecpe, 16  
  frac20, 19  
  sim10GDINA, 51  
  sim20seqGDINA, 52  
  sim21seqGDINA, 52  
  sim30DINA, 53  
  sim30GDINA, 53  
  sim30pGDINA, 54  

+Topic **package**
  GDINA-package, 3  

AIC, GDINA (GDINA), 20  
anova, 20  
anova.GDINA (GDINA), 20  
att.structure, 5  
attribute_pattern, 6  
auglag, 24  
autoGDINA, 5, 7, 29, 42, 44, 47  

bdiagMatrix, 10  
BIC, GDINA (GDINA), 20  

cbind, 11  
cjoint, 11  
ClassRate, 12  

designmatrix, 13  
deviance.GDINA (GDINA), 20  
dif, 3, 14, 47  

ecee, 16  
eextract, 9, 17, 20, 24  
eextract.GDINA (GDINA), 20  
eextract.itemfit (itemfit), 38  
eextract.modelcomp (modelcomp), 43  
eextract.Qval (Qval), 47  
eextract.simGDINA (simGDINA), 54  

frac20, 19  

GDINA, 3, 5, 7, 9, 14, 15, 20, 36, 37, 40, 44–47, 49, 54, 56  
GDINA-package, 3  

heatplot, 36  
heatplot.itemfit (itemfit), 38  
hoparm, 20, 36, 46  
hoparm.GDINA (GDINA), 20  

indlogLik, 37  
indlogLik.GDINA (GDINA), 20  
indlogPost, 37  
indlogPost.GDINA (GDINA), 20  
itemfit, 3, 17, 29, 36, 38  
itemparm, 20, 24, 40  
itemparm.GDINA (GDINA), 20  

LC2LG, 41  
logLik.GDINA (GDINA), 20  

mesaplot, 42, 49  
modelcomp, 3, 7, 9, 17, 29, 43  
monocheck, 20, 45  

npar, 46  
npar.GDINA (GDINA), 20  

optim, 24  

personparm, 20, 24, 46  
personparm.GDINA (GDINA), 20  
plotIRF, 20, 47  
Qval, 3, 7, 9, 17, 29, 42, 47  

rowCount, 49  
rowMatch, 50  

score, 51  
sim10GDINA, 51  
sim20seqGDINA, 52
INDEX

sim21seqDINA, 52
sim30DINA, 53
sim30GDINA, 53
sim30pGDINA, 54
simGDINA, 29, 54
slsqp, 24
solnp, 24
startGDINA, 20, 63
summary.autoGDINA (autoGDINA), 7
summary.dif (dif), 14
summary.GDINA (GDINA), 20
summary.itemfit (itemfit), 38
summary.modelcomp (modelcomp), 43
summary.Qval (Qval), 47
unique, 64
unique_only, 64
unrestrQ, 64