Package 'GDINA'

March 28, 2018

Type Package

Title The Generalized DINA Model Framework

Version 2.0.8

Date 2018-3-28

Description A set of psychometric tools for cognitive diagnosis modeling for both dichotomous and polytomous responses. Various cognitive diagnosis models can be estimated, include the generalized deterministic inputs, noisy and gate (G-DINA) model by de la Torre (2011) <DOI:10.1007/s11336-011-9207-7>, the sequential G-DINA model by Ma and de la Torre (2016) <DOI:10.1111/bmsp.12070>, and many other models they subsume. Joint attribute distribution can be independent, saturated, higher-order, loglinear smoothed or structured. Q-matrix validation, item and model fit statistics, model comparison at test and item level and differential item functioning can also be conducted. A graphical user interface is also provided.

License GPL-3

LazyData TRUE

Depends R (>= 3.1.0)

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

Suggests CDM, testthat

LinkingTo Rcpp, RcppArmadillo

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

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

RoxygenNote 6.0.1

Collate 'CA.R' 'CR.R' 'Est.R' 'ExportedFuncs.R' 'GDINA.R' 'GDI.R' 'GDINA-package.R' 'HO.R' 'M2.R' 'Mstep.R' 'RcppExports.R' 'anova.GDINA.R' 'att.struc.R' 'autoGDINA.R' 'bootSE.R' 'coef.R' 'designmatrix.R' 'dif.R' 'ecpe.R' 'simGDINA.R' 'itemfit.R' 'modelcomp.R' 'extract.R' 'frac20.R' 'heatplot.itemfit.R' 'initials.R' 'itemparm.GDINA.R' 'monocheck.R' 'personparm.GDINA.R' 'plotIRF.GDINA.R' 'plotPVAF.Qval.R' 's3GDINA.R' 'print.GDINA.R' 'score.R' 'sim10GDINA.R'

'sim20seqGDINA.R' 'sim21seqDINA.R' 'sim30DINA.R' 'sim30GDINA.R'
'sim30pGDINA.R' 'startGDINA.R' 'structuralparm.R'
'summary.GDINA.R' 'utils.R'

NeedsCompilation yes

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

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

Repository CRAN

Date/Publication 2018-03-28 08:18:50 UTC

R topics documented:

`GDINA-package` *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 cognitively diagnostic analyses for dichotomous and polytomous responses.

Various cognitive diagnosis models (CDMs) can be calibrated using the `GDINA` function, including the G-DINA model (de la Torre, 2011), the deterministic inputs, noisy and gate (DINA; de la Torre, 2009; Junker & Sijtsma, 2001) model, the deterministic inputs, noisy or gate (DINO; Templin & Henson, 2006) model, the reduced reparametrized unified model (R-RUM; Hartz, 2002), the additive CDM (A-CDM; de la Torre, 2011), and the linear logistic model (LLM; Maris, 1999), the multiple-strategy DINA model (de la Torre, & Douglas, 2008) and models defined by users under the G-DINA framework using different link functions and design matrices (de la Torre, 2011). Note that the LLM is also called compensatory RUM and the RRUM is equivalent to the generalized NIDA model.

For ordinal and nominal responses, the sequential G-DINA model (Ma, & de la Torre, 2016) can be fitted and most of the aforementioned CDMs can be used as the processing functions (Ma, & de la Torre, 2016) at the category level. Different CDMs can be assigned to different items within a single assessment. Item parameters are estimated using the MMLE/EM algorithm. Details about the estimation algorithm can be found in de la Torre (2009), de la Torre (2011), Ma, Iaconangelo, & de la Torre (2016) and Ma, & de la Torre (2016). The joint attribute distribution can be modelled using an independent model, a higher-order IRT model (de la Torre, & Douglas, 2004), a loglinear model (Xu & von Davier, 2008), a saturated model or a hierarchical structures (e.g., linear, divergent). Monotonicity constraints for item/category success probabilities can also be specified.

Q-matrix validation (de la Torre, & Chiu, 2016; see `Qval`), imodel fit statistics (see `modelfit` and `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, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

References

Chen, J., & de la Torre, J. (2013). A General Cognitive Diagnosis Model for Expert-Defined Polytomous Attributes. *Applied Psychological Measurement, 37*, 419-437.

Chen, J., de la Torre, J., & Zhang, Z. (2013). Relative and Absolute Fit Evaluation in Cognitive Diagnosis Modeling. *Journal of Educational Measurement, 50*, 123-140.

de la Torre, J. (2009). DINA Model and Parameter Estimation: A Didactic. *Journal of Educational and Behavioral Statistics, 34*, 115-130.

de la Torre, J. (2011). The generalized DINA model framework. *Psychometrika, 76*, 179-199.

de la Torre, J. & Chiu, C-Y. (2016). A General Method of Empirical Q-matrix Validation. *Psychometrika, 81*, 253-273.

de la Torre, J., & Douglas, J. A. (2004). Higher-order latent trait models for cognitive diagnosis. *Psychometrika, 69*, 333-353.

de La Torre, J., & Douglas, J. A. (2008). Model evaluation and multiple strategies in cognitive diagnosis: An analysis of fraction subtraction data. *Psychometrika, 73*, 595.

de la Torre, J., & Lee, Y. S. (2013). Evaluating the wald test for item-level comparison of saturated and reduced models in cognitive diagnosis. *Journal of Educational Measurement, 50*, 355-373.

Haertel, E. H. (1989). Using restricted latent class models to map the skill structure of achievement items. *Journal of Educational Measurement, 26*, 301-321.

Hartz, S. M. (2002). A bayesian framework for the unified model for assessing cognitive abilities: Blending theory with practicality (Unpublished doctoral dissertation). University of Illinois at Urbana-Champaign.

Hou, L., de la Torre, J., & Nandakumar, R. (2014). Differential item functioning assessment in cognitive diagnostic modeling: Application of the Wald test to investigate DIF in the DINA model. *Journal of Educational Measurement, 51*, 98-125.

Junker, B. W., & Sijtsma, K. (2001). Cognitive assessment models with few assumptions, and connections with nonparametric item response theory. *Applied Psychological Measurement, 25*, 258-272.

Ma, W., & de la Torre, J. (2016). A sequential cognitive diagnosis model for polytomous responses. *British Journal of Mathematical and Statistical Psychology. 69,* 253-275.

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

Maris, E. (1999). Estimating multiple classification latent class models. *Psychometrika, 64*, 187-212.

Xu, X., & von Davier, M. (2008). Fitting the structured general diagnostic model to NAEP data. ETS research report, RR-08-27.

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, The University of Alabama, <wenchao.ma@ua.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)
```

`attributepattern` *Generate all possible attribute patterns*

Description

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

Usage

```
attributepattern(K, Q)
```

Arguments

K	number of attributes
Q	Q-matrix; required when Q-matrix is polytomous

Value

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

Author(s)

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

Examples

```
attributepattern(3)

q <- matrix(scan(text = "0 1 2 1 0 1 1 2 0"),ncol = 3)
q
attributepattern(Q=q)

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

---

autoGDINA *Q-matrix validation, model selection and calibration in one run*

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

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

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

Arguments

dat	A required $N \times J$ matrix or data.frame consisting of the responses of N individuals to J items. Missing values need to be coded as NA.
Q	A required matrix; The number of rows occupied by a single-strategy dichotomous item is 1, by a polytomous item is the number of nonzero categories, and by a mutiple-strategy dichotomous item is the number of strategies. The number of column is equal to the number of attributes if all items are single-strategy dichotomous items, but the number of attributes + 2 if any items are polytomous or have multiple strategies. For a polytomous item, the first column represents the item number and the second column indicates the nonzero category number. For a multiple-strategy dichotomous item, the first column represents the item number and the second column indicates the strategy number. For binary attributes, 1 denotes the attributes are measured by the items and 0 means the attributes are not measured. For polytomous attributes, non-zero elements indicate which level of attributes are needed (see Chen, & de la Torre, 2013). See Examples.
modelselection	logical; conducting model selection or not?
modelselectionrule	how to conducted model selection? Possible options include simpler, largestp and DS. See Details.
alpha.level	nominal level for the Wald test. The default is 0.05.
modelselection.args	arguments passed to modelcomp
Qvalid	logical; validate Q-matrix or not? TRUE is the default.
Qvalid.args	arguments passed to Qval
GDINA1.args	arguments passed to GDINA function for initial G-DINA calibration
GDINA2.args	arguments passed to GDINA function for the second G-DINA calibration
CDM.args	arguments passed to GDINA function for final calibration
object	GDINA object for various S3 methods
...	additional arguments

Details

After the Wald statistics for each reduced CDM were calculated for each item, the reduced models with p values less than the pre-specified alpha level were rejected. If all reduced models were

rejected for an item, the G-DINA model was used as the best model; if at least one reduced model was retained, three diferent rules can be implemented for selecting the best model:

When `modelselectionrule` is `simpler`:

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

When `modelselectionrule` is `largestp`:

The reduced model with the largest p-values is selected as the most appropriate model.

When `modelselectionrule` is `DS`:

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

Value

a list consisting of the following elements:

GDINA1.obj initial GDINA calibration of class `GDINA`

GDINA2.obj second GDINA calibration of class `GDINA`

Qval.obj Q validation object of class `Qval`

Wald.obj model comparison object of class `modelcomp`

CDM.obj Final CDM calibration of class `GDINA`

Methods (by generic)

- `summary`: print summary information

Note

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

Author(s)

Wenchao Ma, The University of Alabama, <wenchao.ma@ua.edu>
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 classification. *Applied Psychological Measurement, 40*, 200-217.

See Also

GDINA, modelcomp, Qval

Examples

```
## 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",
                   modelselection.args = list(models = c("DINO","DINA")))
out11
summary(out11)

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


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

## End(Not run)
```

---

bdiagMatrix *Create a block diagonal matrix*

Description

Create a block diagonal matrix

Usage

```
bdiagMatrix(mlist, fill = 0)
```

Arguments

mlist	a list of matrices
fill	value to fill the non-diagnoal elements

Value

a block diagonal matrix

See Also

bdiag in **Matrix**

Examples

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

---

bootSE *Calculating standard errors and variance-covariance matrix using bootstrap methods*

Description

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

Usage

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

Arguments

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

Value

itemparm.se standard errors for item probability of success in list format

delta.se standard errors for delta parameters in list format

lambda.se standard errors for structural parameters of joint attribute distribution

boot.est resample estimates

Author(s)

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

Examples

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

dat <- sim30GDINA$simdat
Q <- sim30GDINA$simQ
fit <- GDINA(dat = dat, Q = Q, model = "GDINA",att.dist = "higher.order")
boot.fit <- bootSE(fit,bootsample = 5,randomseed=123)
boot.fit$delta.se
boot.fit$lambda.se

## End(Not run)
```

CA *Calculate classification accuracy*

Description

This function calculate test-, pattern- and attribute-level classification accuracy indices based on GDINA estimates from the GDINA function using approaches in Iaconangelo (2017) and Wang, Song, Chen, Meng, and Ding (2015). It is only applicable for dichotomous attributes.

Usage

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

Arguments

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

Value

a list with elements

tau estimated test-level classification accuracy, see Iaconangelo (2017, Eq 2.2)

tau_l estimated pattern-level classification accuracy, see Iaconangelo (2017, p. 13)

tau_k estimated attribute-level classification accuracy, see Wang, et al (2015, p. 461 Eq 6)

CCM Conditional classification matrix, see Iaconangelo (2017, p. 13)

References

Iaconangelo, C.(2017). *Uses of Classification Error Probabilities in the Three-Step Approach to Estimating Cognitive Diagnosis Models.* (Unpublished doctoral dissertation). New Brunswick, NJ: Rutgers University.

Wang, W., Song, L., Chen, P., Meng, Y., & Ding, S. (2015). Attribute-Level and Pattern-Level Classification Consistency and Accuracy Indices for Cognitive Diagnostic Assessment. *Journal of Educational Measurement, 52* , 457-476.

Examples

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

## End(Not run)
```

`cjoint` *Combine R Objects by Columns*

Description

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

Usage

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

Arguments

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

Value

a data frame

See Also

`cbind`

Examples

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

ClassRate *Classification Rate Evaluation*

Description

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

Usage

```
ClassRate(att1, att2)
```

Arguments

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

Value

a list with the following components:

PCA the proportion of correctly classified attributes (i.e., attribute level classification rate)

PCV a vector giving the proportions of correctly classified attribute vectors (i.e., vector level classification rate). The fist 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

```
## Not run:
N <- 2000
# model does not matter if item parameter is probability of success
Q <- sim30GDINA$simQ
J <- nrow(Q)
gs <- matrix(0.1,J,2)

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

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

## End(Not run)
```

`designmatrix` *Generate design matrix*

Description

This function generates the design matrix for an item

Usage

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

Arguments

`Kj`	Required except for the MS-DINA model; The number of attributes required for item j
`model`	the model associated with the design matrix; It can be "GDINA","DINA","DINO", "ACDM" or "MSDINA". The default is "GDINA". Note that models "LLM" and "RRUM" have the same design matrix as the ACDM.
`Qj`	the Q-matrix for item j; This is required for "MSDINA" model; The number of rows is equal to the number of strategies and the number of columns is equal to the number of attributes.

Value

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

References

de la Torre, J. (2011). The generalized DINA model framework. *Psychometrika, 76*, 179-199.

Examples

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

## End(Not run)
```

dif *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) and the likelihood ratio test (Ma, Terzi, Lee, & de la Torre, 2017). It can only detect DIF for two groups currently.

Usage

```
dif(dat, Q, group, method = "wald", p.adjust.methods = "bonferroni",
  LR.type = "free.all", LR.approx = FALSE, difitem = "all", 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 matrix; The number of rows occupied by a single-strategy dichotomous item is 1, by a polytomous item is the number of nonzero categories, and by a mutiple-strategy dichotomous item is the number of strategies. The number of column is equal to the number of attributes if all items are single-strategy dichotomous items, but the number of attributes + 2 if any items are polytomous or have multiple strategies. For a polytomous item, the first column represents the item number and the second column indicates the nonzero category number. For a multiple-strategy dichotomous item, the first column represents the item number and the second column indicates the strategy number. For binary attributes, 1 denotes the attributes are measured by the items and 0 means the attributes are not measured. For polytomous attributes, non-zero elements indicate which level of attributes are needed (see Chen, & de la Torre, 2013). See Examples.
group	a numerical vector with integer 1, 2, ..., # of groups indicating the group each individual belongs to. It must start from 1 and its length must be equal to the number of individuals.
method	DIF detection method; It can be "wald" for Hou, de la Torre, and Nandakumar's (2014) Wald test method, and "LR" for likelihood ratio test (Ma, Terzi, Lee,& de la Torre, 2017).
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'.
LR.approx	Whether an approximated LR test is implemented? If TRUE, anchor item parameters will not be re-estimated but fixed.
difitem	Items for the DIF detection. By default, all items will be examined.
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	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, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

References

Hou, L., de la Torre, J., & Nandakumar, R. (2014). Differential item functioning assessment in cognitive diagnostic modeling: Application of the Wald test to investigate DIF in the DINA model. *Journal of Educational Measurement, 51*, 98-125.

Ma, W., Terzi, R., Lee, S., & de la Torre, J. (2017, April). Multiple group cognitive diagnosis models and their applications in detecting differential item functioning. Paper presented at the Annual Meeting ofthe American Educational Research Association, San Antonio, TX.

See Also

GDINA

Examples

```
## Not run:
set.seed(123456)
N <- 3000
Q <- sim10GDINA$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,
                      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")
sim2 <- simGDINA(N,Q,gs.parm = gs,model=c(rep("DINA",9),"DINO"))
dat <- rbind(extract(sim1,"dat"),extract(sim2,"dat"))
gr <- c(rep(1,N),rep(2,N))
dif.out <- dif(dat,Q,group=gr)
dif.out2 <- dif(dat,Q,group=gr,method="LR")

## End(Not run)
```

---

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 (2014), 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 28×3 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

Feng, Y., Habing, B. T., & Huebner, A. (2014). Parameter estimation of the reduced RUM using the EM algorithm. *Applied Psychological Measurement, 38*, 137-150.

Henson, R. A., & Templin, J. (2007, April). Large-scale language assessment using cognitive diagnosis models. Paper presented at the annual meeting of the National Council for Measurement in Education in Chicago, Illinois.

Templin, J., & Bradshaw, L. (2014). Hierarchical diagnostic classification models: A family of models for estimating and testing attribute hierarchies. *Psychometrika, 79*, 317-339.

Templin, J., & Hoffman, L. (2013). Obtaining diagnostic classification model estimates using Mplus. *Educational Measurement: Issues and Practice, 32*, 37-50.

Examples

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

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

## 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:

att.prior attribute prior weights for calculating marginalized likelihood in the last EM iteration

discrim GDINA discrimination index

designmatrix A list of design matrices for each item/category

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

linkfunc link functions for each item

initial.catprob initial item category probability parameters

prevalence prevalence of each attribute

posterior.prob posterior weights for each latent class

Usage

```
extract(object, what, ...)
```

Arguments

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

Examples

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

## End(Not run)
```

`frac20` *Tatsuoka's fraction subtraction data*

Description

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

Usage

```
frac20
```

Format

A list of responses and Q-matrix with components:

`dat` responses of 536 examinees to 20 items

`Q` The 20×8 Q-matrix

References

Tatsuoka, C. (2002). Data analytic methods for latent partially ordered classification models. *Journal of the Royal Statistical Society, Series C, Applied Statistics, 51*, 337-350.

Examples

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

## End(Not run)
```

GDINA *CDM calibration under the G-DINA model framework*

Description

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

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

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

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

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

Usage

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

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

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

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

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

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

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

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

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

Arguments

dat	A required $N \times J$ matrix or data.frame consisting of the responses of N individuals to J items. Missing values need to be coded as NA.
Q	A required matrix; The number of rows occupied by a single-strategy dichotomous item is 1, by a polytomous item is the number of nonzero categories, and by a mutiple-strategy dichotomous item is the number of strategies. The number of column is equal to the number of attributes if all items are single-strategy dichotomous items, but the number of attributes + 2 if any items are polytomous or have multiple strategies. For a polytomous item, the first column represents the item number and the second column indicates the nonzero category number. For a multiple-strategy dichotomous item, the first column represents the item number and the second column indicates the strategy number. For binary attributes, 1 denotes the attributes are measured by the items and 0 means the attributes are not measured. For polytomous attributes, non-zero elements indicate which level of attributes are needed (see Chen, & de la Torre, 2013). See

Examples.

model — A vector for each item or nonzero category, or a scalar which will be used for all items or nonzero categories to specify the CDMs fitted. The possible options include "GDINA","DINA","DINO","ACDM","LLM", "RRUM", "MSDINA" and "UDF". When "UDF", indicating user defined function, is specified for any item, arguments design.matrix and linkfunc need to be defined.

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

att.dist — How is the joint attribute distribution estimated? It can be (1) saturated, which is the default, indicating that the proportion parameter for each permissible latent class is estimated separately; (2) higher.order, indicating that a higher-order joint attribute distribution is assumed (higher-order model can be specified in higher.order argument); (3) fixed, indicating that the weights 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; (4) independent, indicating that all attributes are assumed to be independent; and (5) loglinear, indicating a loglinear model is employed. 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. However, if a higher-order model is used for any group, it must be used for all groups.

mono.constraint — logical; TRUE indicates that $P(\boldsymbol{\alpha}_1) <= P(\boldsymbol{\alpha}_2)$ if for all k, $\alpha_{1k} < \alpha_{2k}$. 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 numerical vector with integer 1, 2, ..., # of groups indicating the group each individual belongs to. It must start from 1 and its length must be equal to the number of individuals.

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

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

latent.var — A string indicating the nature of the latent variables. It is "att" (by default) if the latent variables are attributes, and "bugs" if the latent variables are misconceptions. When "bugs" is specified, only the DINA, DINO or G-DINA model can be specified in model argument (Kuo, Chen, Yang & Mok, 2016).

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

att.str — logical; are attributes structured? If yes, att.prior must be specified where impossible latent classes have prior weights 0. If attributes are structured, only the DINA, DINO or G-DINA model can be specified in model argument.

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

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

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

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

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

control — A list of control parameters with elements:

- maxitr A vector for each item or nonzero category, or a scalar which will be used for all items or nonzero categories to specify the maximum number of EM cycles allowed. Default = 2000.
- conv.crit The convergence criterion for max absolute change in item parameters or deviance. Default = 0.0001.
- conv.type How is the convergence criterion evaluated? A vector with possible elements: "ip", indicating the maximum absolute change in item success probabilities, "mp", representing the maximum absolute change in mixing proportion parameters, "delta", indicating the maximum absolute change in delta parameters or neg2LL indicating the absolute change in negative two times loglikeihood. Multiple criteria can be specified. If so, all criteria need to be met. Default = c("ip", "mp").
- nstarts how many sets of starting values? Default = 1.
- lower.p A vector for each item or nonzero category, or a scalar which will be used for all items or nonzero categories to specify the lower bound for success probabilities. Default = .0001.

- `upper.p` A vector for each item or nonzero category, or a scalar which will be used for all items or nonzero categories to specify the upper bound for success probabilities. Default = .9999.
- `lower.prior` The lower bound for mixing proportion parameters (latent class sizes). Default = .Machine$double.eps.
- `randomseed` Random seed for generating initial item parameters. Default = 123456.
- `smallNcorrection` A numeric vector with two elements specifying the corrections applied when the expected number of individuals in some latent groups are too small. If the expected no. of examinees is less than the second element, the first element and two times the first element will be added to the numerator and denominator of the closed-form solution of probabilities of success. Only applicable for the G-DINA, DINA and DINO model estimation without monotonic constraints.
- `MstepMessage` Integer; Larger number prints more information from Mstep optimizer. Default = 1.

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

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

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

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

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

`...` additional arguments

`object` GDINA object for various S3 methods

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

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

For calculating incidental parameters using `personparm`, `what` can be

- `EAP` - EAP estimates of attribute pattern.
- `MAP` - MAP estimates of attribute pattern.
- `MLE` - MLE estimates of attribute pattern.
- `mp` - marginal mastery probabilities.

- HO - EAP estimates of higher-order ability if a higher-order is fitted.

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

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

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

Value

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

Methods (by generic)

- anova: Model comparison using likelihood ratio test
- coef: extract structural parameter estimates
- extract: extract various elements of GDINA estimates
- personparm: calculate person attribute patterns and higher-order ability
- 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 collaspes all latent classes into two latent groups for each item, if item j requires K_j^* attributes, the G-DINA model collapses 2^K latent classes into $2^{K_j^*}$ latent groups with unique success probabilities on item j, where $K_j^* = \sum_{k=1}^{K} q_{jk}$.

Let $\boldsymbol{\alpha}_{lj}^*$ 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, $\boldsymbol{\alpha}_{lj}^* = (\alpha_{l1}, \alpha_{lK})$. For notational convenience, the first K_j^* attributes can be assumed to be the required attributes for item j as in de la Torre (2011). The probability of success $P(X_j = 1|\boldsymbol{\alpha}_{lj}^*)$ is denoted by $P(\boldsymbol{\alpha}_{lj}^*)$. To model this probability of success, different link functions as in the generalized linear

models are used in the G-DINA model. The item response function of the G-DINA model using the identity link can be written as

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

or in matrix form,

$$f[\boldsymbol{P}_j] = \boldsymbol{M}_j\boldsymbol{\delta}_j,$$

where δ_{j0} is the intercept for item j, δ_{jk} is the main effect due to α_{lk}, $\delta_{jkk'}$ is the interaction effect due to α_{lk} and $\alpha_{lk'}$, $\delta_{j12\ldots K_j^*}$ 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 breifly. 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 η for person i and item j is defined as

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

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 δ_0 and $\delta_{12\ldots K_j^*}$ need to be fixed to zero, that is,

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

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

`DINO model` The DINO model can be given by

$$P(\boldsymbol{\alpha}_{lj}^*) = \delta_{j0} + \delta_{j1} I(\boldsymbol{\alpha}_{lj}^* \neq \mathbf{0})$$

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

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

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

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

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

The item response function for LLM can be given by

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

and lastly, the RRUM, can be written as

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

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).

Joint Attribute Distribution

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

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

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

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

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

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

where θ_i is the ability of examinee i. λ_{0k} and λ_{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 λ_1 is estimated. The probability of joint attributes can be written as

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

.

Model Estimation

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

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

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 $2^{K_j^*}$ item parameters. Apart from item parameters, the parameters involved in the estimation of joint attribute distribution need to be estimated as well. When using the saturated attribute structure, there are $2^K - 1$ parameters for joint attribute distribution estimation; when using a higher-order attribute structure, there are K, $K + 1$, and $2 \times K$ parameters for the Rasch model, 1PL model and 2PL model, respectively. For polytomous response data using the sequential G-DINA model, the number of item parameters are counted at category level.

Note

anova function does NOT check whether models compared are nested or not.

Author(s)

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

References

Bock, R. D., & Aitkin, M. (1981). Marginal maximum likelihood estimation of item parameters: Application of an EM algorithm. *Psychometrika, 46*, 443-459.

Bock, R. D., & Lieberman, M. (1970). Fitting a response model forn dichotomously scored items. *Psychometrika, 35*, 179-197.

Bor-Chen Kuo, Chun-Hua Chen, Chih-Wei Yang, & Magdalena Mo Ching Mok. (2016). Cognitive diagnostic models for tests with multiple-choice and constructed-response items. *Educational Psychology, 36*, 1115-1133.

Carlin, B. P., & Louis, T. A. (2000). Bayes and empirical bayes methods for data analysis. New York, NY: Chapman & Hall

de la Torre, J., & Douglas, J. A. (2008). Model evaluation and multiple strategies in cognitive diagnosis: An analysis of fraction subtraction data. *Psychometrika, 73*, 595-624.

de la Torre, J. (2009). DINA Model and Parameter Estimation: A Didactic. *Journal of Educational and Behavioral Statistics, 34*, 115-130.

de la Torre, J. (2011). The generalized DINA model framework. *Psychometrika, 76*, 179-199.

de la Torre, J., & Douglas, J. A. (2004). Higher-order latent trait models for cognitive diagnosis. *Psychometrika, 69*, 333-353.

de la Torre, J., & Lee, Y. S. (2013). Evaluating the wald test for item-level comparison of saturated and reduced models in cognitive diagnosis. *Journal of Educational Measurement, 50*, 355-373.

Haertel, E. H. (1989). Using restricted latent class models to map the skill structure of achievement items. *Journal of Educational Measurement, 26*, 301-321.

Hartz, S. M. (2002). A bayesian framework for the unified model for assessing cognitive abilities: Blending theory with practicality (Unpublished doctoral dissertation). University of Illinois at Urbana-Champaign.

Huo, Y., & de la Torre, J. (2014). Estimating a Cognitive Diagnostic Model for Multiple Strategies via the EM Algorithm. *Applied Psychological Measurement, 38*, 464-485.

Junker, B. W., & Sijtsma, K. (2001). Cognitive assessment models with few assumptions, and connections with nonparametric item response theory. *Applied Psychological Measurement, 25*, 258-272.

Ma, W., & de la Torre, J. (2016). A sequential cognitive diagnosis model for polytomous responses. *British Journal of Mathematical and Statistical Psychology. 69,* 253-275.

Ma, W. (in press). A Diagnostic Tree Model for Polytomous Responses with Multiple Strategies. *British Journal of Mathematical and Statistical Psychology.*

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

Ma, W. (2017). *A Sequential Cognitive Diagnosis Model for Graded Response: Model Development, Q-Matrix Validation,and Model Comparison. Unpublished doctoral dissertation.* New Brunswick, NJ: Rutgers University.

Maris, E. (1999). Estimating multiple classification latent class models. *Psychometrika, 64*, 187-212.

Tatsuoka, K. K. (1983). Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement, 20*, 345-354.

Templin, J. L., & Henson, R. A. (2006). Measurement of psychological disorders using cognitive diagnosis models. *Psychological Methods, 11*, 287-305.

Tutz, G. (1997). Sequential models for ordered responses. In W.J. van der Linden & R. K. Hambleton (Eds.), Handbook of modern item response theory p. 139-152). New York, NY: Springer.

Xu, X., & von Davier, M. (2008). Fitting the structured general diagnostic model to NAEP data. ETS research report, RR-08-27.

See Also

See `autoGDINA` for Q-matrix validation, item-level model comparison and model calibration in one run; See `modelfit` and `itemfit` for model and item fit analysis, `Qval` for Q-matrix validation, `modelcomp` for item level model comparison and `simGDINA` for data simulation. Also see `gdina` in **CDM** package for the G-DINA model estimation.

Examples

```
## 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

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

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

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

# -------- DINO model -------#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod3 <- GDINA(dat = dat, Q = Q, model = "DINO")
#slip and guessing
coef(mod3, what = "gs") # guess and slip parameters
coef(mod3, what = "gs",withSE = TRUE) # guess and slip parameters + 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")
mod4
# --------- LLM model -------#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod4b <- GDINA(dat = dat, Q = Q, model = "LLM")
mod4b
# --------- RRUM model -------#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod4c <- GDINA(dat = dat, Q = Q, model = "RRUM")
mod4c

# --- Different CDMs for different items --- #

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


####################################
#        Example 2.                #
#        Model estimations         #
# With monotonocity constraints    #
####################################
```

```
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
# for item 10 only
mod11 <- GDINA(dat = dat, Q = Q, model = "GDINA",mono.constraint = c(rep(FALSE,9),TRUE))
mod11
mod11a <- GDINA(dat = dat, Q = Q, model = "DINA",mono.constraint = TRUE)
mod11a
mod11b <- GDINA(dat = dat, Q = Q, model = "ACDM",mono.constraint = TRUE)
mod11b
mod11c <- GDINA(dat = dat, Q = Q, model = "LLM",mono.constraint = TRUE)
mod11c
mod11d <- GDINA(dat = dat, Q = Q, model = "RRUM",mono.constraint = TRUE)
mod11d
coef(mod11d,"delta")
coef(mod11d,"rrum")

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

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

####################################
#           Example 3b.            #
#        Model estimations         #
#   With log-linear att structure  #
####################################

# --- DINA model with loglinear smoothed attribute space ---#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod23 <- GDINA(dat = dat, Q = Q, model = "DINA",att.dist="loglinear",loglinear=1)
coef(mod23,"lambda") # intercept and three main effects

####################################
#           Example 3c.            #
#        Model estimations         #
#  With independent att structure  #
####################################
```

```
# --- GDINA model with independent attribute space ---#
dat <- sim10GDINA$simdat
Q <- sim10GDINA$simQ
mod33 <- GDINA(dat = dat, Q = Q, att.dist="independent")
coef(mod33,"lambda") # mastery probability for each attribute

####################################
#          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,
                1,0,0,
                1,1,0,
                1,1,0,
                1,1,1,
                1,1,1,
                1,0,0,
                1,0,0,
                1,1,0,
                1,1,0,
                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
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
true.att <- attributepattern(K)[true.lc,]
 gs <- matrix(rep(0.1,2*nrow(Q)),ncol=2)
 # data simulation
 simD <- simGDINA(N,Q,gs.parm = gs, model = "DINA",attribute = true.att)
 dat <- extract(simD,"dat")

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


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

```
#           Example 6.             #
#        Model estimations         #
# With user-specified initial pars #
####################################

 # 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
 # estimation- only 20 iterations for illustration purposes
 mod.initial <- GDINA(dat,Q,control = list(maxitr=20))
 par <- coef(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),control = list(maxitr = 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, control = list(maxitr=0)) # re-estimation
 anova(mod.initial,mod.fix2) # identical results

####################################
#           Example 8.             #
#        polytomous attribute      #
#          model estimation        #
#    see Chen, de la Torre 2013    #
####################################


# --- polytomous attribute G-DINA model --- #
dat <- sim30pGDINA$simdat
Q <- sim30pGDINA$simQ
#polytomous G-DINA model
pout <- GDINA(dat,Q)
```

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

####################################
#           Example 9.             #
#        Sequential G-DINA model   #
#    see Ma, & de la Torre 2016    #
####################################

# --- polytomous attribute G-DINA model --- #
dat <- sim20seqGDINA$simdat
Q <- sim20seqGDINA$simQ
Q
#    Item Cat A1 A2 A3 A4 A5
#       1   1  1  0  0  0  0
#       1   2  0  1  0  0  0
#       2   1  0  0  1  0  0
#       2   2  0  0  0  1  0
#       3   1  0  0  0  0  1
#       3   2  1  0  0  0  0
#       4   1  0  0  0  0  1
#       ...

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

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

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

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

```
                lambda = data.frame(a=rep(1,K),b=seq(-2,2,length.out=K))))

# combine data
# see ?bdiagMatrix
dat <- bdiagMatrix(list(extract(sim1,"dat"),extract(sim2,"dat")),fill=NA)
Q <- rbind(Q,Q)
gr <- rep(c(1,2),c(3000,3000))
# Fit G-DINA model
mg.est <- GDINA(dat = dat,Q = Q,group = gr,att.dist="higher.order",
higher.order=list(model = "1PL",Prior=TRUE))
summary(mg.est)
extract(mg.est,"posterior.prob")



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

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

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

# combine data
# see ?bdiagMatrix
dat <- bdiagMatrix(list(extract(sim1,"dat"),extract(sim2,"dat")),fill=NA)
Q <- rbind(Q,Q)
gr <- rep(c(1,2),c(3000,3000))
# Fit G-DINA model
mg.est <- GDINA(dat = dat,Q = Q,group = gr,att.dist="higher.order",
higher.order=list(model = "Rasch",Prior=FALSE,Type = "SameLambda"))
summary(mg.est)
coef(mg.est,"lambda")


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

```
set.seed(123)
Q <- sim10GDINA$simQ # 1 represents misconceptions/bugs
ip <- list(
c(0.8,0.2),
c(0.7,0.1),
c(0.9,0.2),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1),
c(0.9,0.1,0.1,0.1,0.1,0.1,0.1,0.1))
sim <- simGDINA(N=1000,Q=Q,catprob.parm = ip)
dat <- extract(sim,"dat")
# use latent.var to specify a bug model
est <- GDINA(dat=dat,Q=Q,latent.var="bugs",model="DINO")
coef(est)

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

set.seed(123)
Q <- sim10GDINA$simQ # 1 represents misconceptions/bugs
ip <- list(
c(0.8,0.2),
c(0.7,0.1),
c(0.9,0.2),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.1),
c(0.9,0.9,0.9,0.9,0.9,0.9,0.9,0.1))
sim <- simGDINA(N=1000,Q=Q,catprob.parm = ip)
dat <- extract(sim,"dat")
# use latent.var to specify a bug model
est <- GDINA(dat=dat,Q=Q,latent.var="bugs",model="DINA")
coef(est)

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

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

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

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

# compare two models => identical
anova(lcdm,iGDINA)

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

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

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

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

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

####################################
#           Example 14.            #
#     Multiple-strategy DINA model #
####################################

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

```
  item3=c(0.2,0.6),
  item4=c(0.2,0.7),
  item5=c(0.1,0.8))

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

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

## End(Not run)
```

`heatplot` *Item fit plots*

Description

Create plots of bivariate heatmap for item fit

Usage

```
heatplot(object, ...)
```

Arguments

`object`	model object of class `itemfit`
`...`	additional arguments

See Also

`GDINA`, `itemfit`

Examples

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

fit <- GDINA(dat = dat, Q = Q, model = "GDINA")
ift <- itemfit(fit)
heatplot(ift)

## End(Not run)
```

indlogLik *Extract log-likelihood for each individual*

Description

Extract individual log-likelihood.

Usage

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

Arguments

object	GDINA object
...	additional arguments

Examples

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

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

## End(Not run)
```

indlogPost *Extract log posterior for each individual*

Description

Extract individual log posterior.

Usage

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

Arguments

object	GDINA object
...	additional arguments

Examples

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

## End(Not run)
```

`internal_GDINA` *Functions for internal use*

Description

Functions for internal use

Usage

```
internal_Lik(...)

internal_Lik2(...)

internal_aggregateCol(...)

internal_eta(...)

internal_matchMatrix(...)

internal_RowNormalize(...)

internal_ColNormalize(...)

internal_RowProd(...)

internal_l2m(...)

internal_m2l(...)

internal_uP(...)
```

Arguments

...	arguments passed to internal functions

`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:

p the proportion correct statistics, adjusted and unadjusted p values for each item

r the transformed correlations, adjusted and unadjusted p values for each item pair

logOR the log odds ratios, adjusted and unadjusted p values for each item pair

maxitemfit the maximum proportion correct, transformed correlation, and log-odds ratio for each item with associated item-level adjusted p-values

Methods (by generic)

- `extract`: extract various elements from `itemfit` objects
- `heatplot`: plot bivariate heatmap for misfit detection
- `summary`: print summary information

Author(s)

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

References

Chen, J., de la Torre, J., & Zhang, Z. (2013). Relative and Absolute Fit Evaluation in Cognitive Diagnosis Modeling. *Journal of Educational Measurement, 50*, 123-140.

Examples

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

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

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

itmfit

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

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

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

## End(Not run)
```

`itemparm` *extract item parameters (deprecated)*

Description

This function has been deprecated; use `coef` instead.

Usage

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

## S3 method for class 'GDINA'
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.
`withSE`	show standard errors or not?
`SE.type`	Type of standard errors.
`digits`	how many decimal places for the ouput?
`...`	additional arguments

References

Philipp, M., Strobl, C., de la Torre, J., & Zeileis, A.(2017). On the estimation of standard errors in cognitive diagnosis models. *Journal of Educational and Behavioral Statistics, 43*, 88-115.

Examples

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

## End(Not run)
```

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

```
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 Kj attributes, 2^K latent classes can be combined into 2^{Kj} latent groups. This matrix gives which latent group each of 2^K latent classes belongs to for each item.

Author(s)

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

Examples

```
attributepattern(3)

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

`mesaplot` *Mesa plot for Q-matrix validation*

Description

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

Usage

```
mesaplot(Qval.obj, item, type = "best", no.qvector = 10,
  data.label = TRUE, eps = "auto", 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

References

de la Torre, J., & Ma, W. (2016, August). Cognitive diagnosis modeling: A general framework approach and its implementation in R. A Short Course at the Fourth Conference on Statistical Methods in Psychometrics, Columbia University, New York.

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)

## End(Not run)
```

---

modelcomp *Item-level model comparison using Wald, LR or LM tests*

Description

This function evaluates whether the saturated G-DINA model can be replaced by reduced CDMs without significant loss in model data fit for each item using the Wald test, likelihood ratio (LR) test or Lagrange multiplier (LM) test. For Wald test, see de la Torre and Lee (2013), and Ma, Iaconangelo and de la Torre (2016) for details. For LR test and a two-step LR approximation procedure, see Sorrel, de la Torre, Abad, and Olea (2017) and Ma (2017). For LM test, which is only applicable for DINA, DINO and ACDM, see Sorrel, Abad, Olea, de la Torre, and Barrada (2017). This function also calculates the dissimilarity between the reduced models and the G-DINA model, which can be viewed as a measure of effect size (Ma, Iaconangelo & de la Torre, 2016).

Usage

```
modelcomp(GDINA.obj = NULL, method = "Wald", items = "all",
  p.adjust.methods = "bonferroni", models = c("DINA", "DINO", "ACDM", "LLM",
  "RRUM"), DS = FALSE, Wald.args = list(SE.type = 2, varcov = NULL),
  LR.args = list(LR.approx = FALSE), LM.args = list(reducedMDINA = NULL,
  reducedMDINO = NULL, reducedMACDM = NULL, SE.type = 2))

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

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

Arguments

GDINA.obj	An estimated model object of class GDINA
method	method for item level model comparison; can be wald, LR or LM.
items	a vector of items to specify the items for model comparsion
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.
models	a vector specifying which reduced CDMs are possible reduced CDMs for each item. The default is "DINA","DINO","ACDM","LLM",and "RRUM".
DS	whether dissimilarity index should be calculated? FALSE is the default.
Wald.args	a list of options for Wald test including (1) SE.type giving the type of covariance matrix for the Wald test; by default, it uses outer product of gradient based on incomplete information matrix; (2) varcov for user specified variance-covariance matrix. If supplied, it must be a list, giving the variance covariance matrix of success probability for each item or category. The default is NULL, in which case, the estimated variance-covariance matrix from the GDINA function is used.
LR.args	a list of options for LR test including for now only LR.approx, which is either TRUE or FALSE, indicating whether a two-step LR approximation is implemented or not.
LM.args	a list of options for LM test including reducedMDINA, reducedMDINO, and reducedMACDM for DINA, DINO and ACDM estimates from the GDINA function; SE.type specifies the type of covariance matrix.
object	object of class modelcomp for various S3 methods
what	argument for S3 method extract indicating what to extract; It can be "wald" for wald statistics, "wald.p" for associated p-values, "df" for degrees of freedom, and "DS" for dissimilarity between G-DINA and other CDMs.
digits	How many decimal places in each number? The default is 4.
...	additional arguments

Value

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

stats Wald or LR statistics

pvalues p-values associated with the test statistics

adj.pvalues adjusted p-values

df degrees of freedom

DS dissimilarity between G-DINA and other CDMs

Methods (by generic)

- extract: extract various elements from modelcomp objects
- summary: print summary information

Author(s)

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

References

de la Torre, J., & Lee, Y. S. (2013). Evaluating the wald test for item-level comparison of saturated and reduced models in cognitive diagnosis. *Journal of Educational Measurement, 50*, 355-373.

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

Ma, W. (2017). *A Sequential Cognitive Diagnosis Model for Graded Response: Model Development, Q-Matrix Validation,and Model Comparison. Unpublished doctoral dissertation.* New Brunswick, NJ: Rutgers University.

Sorrel, M. A., Abad, F. J., Olea, J., de la Torre, J., & Barrada, J. R. (2017). Inferential Item-Fit Evaluation in Cognitive Diagnosis Modeling. *Applied Psychological Measurement, 41,* 614-631.

Sorrel, M. A., de la Torre, J., Abad, F. J., & Olea, J. (2017). Two-Step Likelihood Ratio Test for Item-Level Model Comparison in Cognitive Diagnosis Models. *Methodology, 13*, 39-47.

See Also

GDINA, autoGDINA

Examples

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

###################
#
# Wald test
#
###################

w <- modelcomp(fit)
w
# wald statistics
extract(w,"stats")
#p values
extract(w,"pvalues")

#########################
#
# LR and Two-step LR test
#
```

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

lr <- modelcomp(fit,models = c("DINA","DINO"),method = "LR")
lr
TwostepLR <- modelcomp(fit,items =c(6:10),method = "LR",LR.args = list(LR.approx = TRUE))
TwostepLR

##########################
#
# LM test
#
##########################

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



## End(Not run)
```

`modelfit` *Model fit statistics*

Description

Calculate various model-data fit statistics

Usage

```
modelfit(GDINA.obj, CI = 0.9, ...)
```

Arguments

`GDINA.obj`	An estimated model object of class `GDINA`
`CI`	numeric value from 0 to 1 indicating the range of the confidence interval for RMSEA. Default returns the 90% interval.
`...`	arguments passed to the function

Details

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

Author(s)

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

References

Ma, W. (under review). Evaluating model data fit using limited information statistics for the sequential G-DINA model.

Maydeu-Olivares, A. (2013). Goodness-of-Fit Assessment of Item Response Theory Models. *Measurement, 11*, 71-101.

Liu, Y., Tian, W., & Xin, T. (2016). An Application of M2 Statistic to Evaluate the Fit of Cognitive Diagnostic Models. *Journal of Educational and Behavioral Statistics, 41*, 3-26.

Examples

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

## End(Not run)
```

---

monocheck *This function checks if monotonicity is violated*

Description

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

Usage

```
monocheck(object, strict = FALSE)
```

Arguments

object	object of class GDINA
strict	whether a strict monotonicity is checked?

Value

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

Examples

```
## 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 parameters of joint attribute distribution.

Usage

```
npar(object, ...)
```

Arguments

object	GDINA object
...	additional arguments

Examples

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

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

## End(Not run)
```

personparm *calculate person (incidental) parameters*

Description

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

Usage

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

Arguments

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

Author(s)

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

References

Huebner, A., & Wang, C. (2011). A note on comparing examinee classification methods for cognitive diagnosis models. *Educational and Psychological Measurement, 71*, 407-419.

Examples

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

## End(Not run)
```

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 (estimate - SE, estimate + SE) to the plot?
...	additional arguments

See Also

GDINA, autoGDINA, dif

Examples

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

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

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

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

Arguments

GDINA.obj	An estimated model object of class GDINA
method	which Q-matrix validation method is used?
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, The University of Alabama, <wenchao.ma@ua.edu>
Jimmy de la Torre, The University of Hong Kong

References

de la Torre, J. & Chiu, C-Y. (2016). A General Method of Empirical Q-matrix Validation. *Psychometrika, 81*, 253-273.

See Also

GDINA, mesaplot

Examples

```
## 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)
```

---

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

Description

Count the frequency of a row vector in a data frame

Usage

```
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

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

---

score *Score function*

Description

Calculate score function for each dichotomous item or each nonzero category for polytomous items Only applicable to saturated model ofr joint attribute distribution

Usage

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

Arguments

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

Value

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

Examples

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

## End(Not run)
```

sim10GDINA *Simulated data (10 items, G-DINA model)*

Description

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

Usage

```
sim10GDINA
```

Format

A list with components:

`simdat` simulated responses of 1000 examinees

`simQ` artificial Q-matrix

`simItempar` artificial item parameters (probability of success for each latent group)

sim20seqGDINA *Simulated data (20 items, sequential G-DINA model)*

Description

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

Usage

```
sim20seqGDINA
```

Format

A list with components:

`simdat` simulated polytomous responses of 2000 examinees

`simQ` artificial Qc-matrix

`simItempar` artificial item parameters (category level probability of success for each latent group)

sim21seqDINA *Simulated data (21 items, sequential DINA model)*

Description

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

Usage

```
sim21seqDINA
```

Format

A list with components:

`simdat` simulated responses of 2000 examinees

`simQ` artificial Qc-matrix

sim30DINA *Simulated data (30 items, DINA model)*

Description

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

Usage

```
sim30DINA
```

Format

A list with components:

`simdat` simulated responses of 1000 examinees

`simQ` artificial Q-matrix

`simItempar` artificial item parameters (probability of success for each latent group)

sim30GDINA *Simulated data (30 items, G-DINA model)*

Description

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

Usage

```
sim30GDINA
```

Format

A list with components:

`simdat` simulated responses of 1000 examinees

`simQ` artificial 30×5 Q-matrix

`simItempar` artificial item parameters(probability of success for each latent group)

sim30pGDINA *Simulated data (30 items, polytomous G-DINA model)*

Description

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

Usage

```
sim30pGDINA
```

Format

A list 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, delta.parm = NULL, catprob.parm = NULL,
  model = "GDINA", sequential = FALSE, gs.args = list(type = "random",
  mono.constraint = TRUE), delta.args = list(design.matrix = NULL, linkfunc =
  NULL), attribute = NULL, att.dist = "uniform", item.names = NULL,
  higher.order.parm = list(theta = NULL, lambda = NULL),
  mvnorm.parm = list(mean = NULL, sigma = NULL, cutoffs = NULL),
  att.prior = NULL, digits = 4)

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

Arguments

`N`	Sample size.
`Q`	A required matrix; The number of rows occupied by a single-strategy dichotomous item is 1, by a polytomous item is the number of nonzero categories, and by a mutiple-strategy dichotomous item is the number of strategies. The number of column is equal to the number of attributes if all items are single-strategy dichotomous items, but the number of attributes + 2 if any items are polytomous or have multiple strategies. For a polytomous item, the first column represents the item number and the second column indicates the nonzero category number. For a multiple-strategy dichotomous item, the first column represents the item number and the second column indicates the strategy number. For binary attributes, 1 denotes the attributes are measured by the items and 0 means the attributes are not measured. For polytomous attributes, non-zero elements indicate which level of attributes are needed. See `Examples`.
`gs.parm`	A matrix or data frame for guessing and slip parameters. The number of rows occupied by a dichotomous item is 1, and by a polytomous item is the number of nonzero categories. The number of columns must be 2, where the first column represents the guessing parameters (or $P(0)$), and the second column represents slip parameters (or $1 - P(1)$). This may need to be used in conjunction with the argument `gs.args`.

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

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

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

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

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

- `type` How are the delta parameters for ACDM, LLM, RRUM generated? It can be either `"random"` or `"equal"`. `"random"` means the delta parameters are simulated randomly, while `"equal"` means that each required attribute contributes equally to the probability of success (P), logit(P) or log(P) for ACDM, LLM and RRUM, respectively. See `Details` for more information.
- `mono.constraint` A vector for each item/category or a scalar which will be used for all items/categories to specify whether monotonicity constraints should be satisfied if the generating model is the G-DINA model. Note that this is applicable only for the G-DINA model when `gs.parm` is used. For ACDM, LLM and RRUM, monotonicity constraints are always satisfied and therefore this argument is ignored.

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

- `linkfunc` a vector of link functions for each item/category; It can be `"identity"`,`"log"` or `"logit"`. Only necessary when, for some items, `model="UDF"`.
- `design.matrix` a list of design matrices; Its length must be equal to the number of items (or nonzero categories for sequential models). If CDM for item j is specified as "UDF" in argument `model`, the corresponding design matrix must be provided; otherwise, the design matrix can be `NULL`, which will be generated automatically.

`attribute` optional user-specified person attributes. It is a $N \times K$ 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"`, `"mvnorm"` or `"multinomial"` for uniform, higher-order, multivariate normal and multinomial distributions, respectively. The default is the uniform distribution. To specify structural parameters for the higher-order and multivariate normal distributions, see `higher.order.parm` and `mvnorm.parm`, respectively. To specify the probabilities for the multinomial distribution, use `att.prior` argument.

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

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

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

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

`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 $P(\boldsymbol{\alpha}_{lj}^* = 0)$ and $1 - P(\boldsymbol{\alpha}_{lj}^* = 1)$ for all items for dichotomous items and $S(\boldsymbol{\alpha}_{ljh}^* = 0)$ and $1 - S(\boldsymbol{\alpha}_{ljh}^* = 1)$ for all nonzero categories for polytomous items. Note that $1 - P(\boldsymbol{\alpha}_{lj}^* = 0) - P(\boldsymbol{\alpha}_{lj}^* = 1)$ or $1 - S(\boldsymbol{\alpha}_{lj}^* = 0) - S(\boldsymbol{\alpha}_{lj}^* = 1)$ 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., $P(\boldsymbol{\alpha}_{lj}^*)$ or $S(\boldsymbol{\alpha}_{ljh}^*)$) 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 inviduals' attribute patterns

catprob.parm a list of non-zero category success probabilities for each latent group

delta.parm a list of delta parameters

higher.order.parm Higher-order parameters

mvnorm.parm multivariate normal distribution parameters

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

Author(s)

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

References

Chiu, C.-Y., Douglas, J. A., & Li, X. (2009). Cluster analysis for cognitive diagnosis: Theory and applications. *Psychometrika, 74*, 633-665.

de la Torre, J. (2011). The generalized DINA model framework. *Psychometrika, 76*, 179-199.

de la Torre, J., & Douglas, J. A. (2004). Higher-order latent trait models for cognitive diagnosis. *Psychometrika, 69*, 333-353.

Haertel, E. H. (1989). Using restricted latent class models to map the skill structure of achievement items. *Journal of Educational Measurement, 26*, 301-321.

Hartz, S. M. (2002). A bayesian framework for the unified model for assessing cognitive abilities: Blending theory with practicality (Unpublished doctoral dissertation). University of Illinois at Urbana-Champaign.

Junker, B. W., & Sijtsma, K. (2001). Cognitive assessment models with few assumptions, and connections with nonparametric item response theory. *Applied Psychological Measurement, 25*, 258-272.

Ma, W., & de la Torre, J. (2016). A sequential cognitive diagnosis model for polytomous responses. *British Journal of Mathematical and Statistical Psychology. 69,* 253-275.

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

Maris, E. (1999). Estimating multiple classification latent class models. *Psychometrika, 64*, 187-212.

Templin, J. L., & Henson, R. A. (2006). Measurement of psychological disorders using cognitive diagnosis models. *Psychological Methods, 11*, 287-305.

Examples

```
## Not run:
#####################################################
#                     Example 1                     #
#             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.parm = gs,model = "DINA")

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

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

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

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

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

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

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

#####################################################
#                     Example 3                     #
#             Data simulation (LLM)                 #
#####################################################
N <- 500
```

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

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

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

#####################################################
#                   Example 4                       #
#          Data simulation (all CDMs)               #
#####################################################

set.seed(12345)

N <- 500
Q <- sim10GDINA$simQ
J <- nrow(Q)
gs <- data.frame(guess=rep(0.1,J),slip=rep(0.1,J))
# Simulated different CDMs for different items
models <- c("GDINA","DINO","DINA","ACDM","LLM","RRUM","GDINA","LLM","RRUM","DINA")
sim <- simGDINA(N,Q,gs.parm = gs,model = models,gs.args = list(type="random"))

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

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

#####################################################
#                   Example 5                       #
#          Data simulation (all CDMs)               #
#  using probability of success in list format      #
#####################################################

# success probabilities for each item need to be provided in list format as follows:
# if item j requires Kj attributes, 2^Kj success probabilities
# need to be specified
# e.g., item 1 only requires 1 attribute
# therefore P(0) and P(1) should be specified;
# similarly, item 10 requires 3 attributes,
# P(000),P(100),P(010)...,P(111) should be specified;
# the latent class represented by each element can be obtained
# by calling attributepattern(Kj)
itemparm.list <- list(item1=c(0.2,0.9),
```

```
                    item2=c(0.1,0.8),
                    item3=c(0.1,0.9),
                    item4=c(0.1,0.3,0.5,0.9),
                    item5=c(0.1,0.1,0.1,0.8),
                    item6=c(0.2,0.9,0.9,0.9),
                    item7=c(0.1,0.45,0.45,0.8),
                    item8=c(0.1,0.28,0.28,0.8),
                    item9=c(0.1,0.4,0.4,0.8),
                    item10=c(0.1,0.2,0.3,0.4,0.4,0.5,0.7,0.9))
set.seed(12345)
N <- 500
Q <- sim10GDINA$simQ
# When simulating data using catprob.parm argument,
# it is not necessary to specify model and type
sim <- simGDINA(N,Q,catprob.parm = itemparm.list)


####################################################
#                   Example 6                      #
#            Data simulation (all CDMs)            #
#      using delta parameters in list format       #
####################################################

delta.list <- list(c(0.2,0.7),
                    c(0.1,0.7),
                    c(0.1,0.8),
                    c(0.1,0.7),
                    c(0.1,0.8),
                    c(0.2,0.3,0.2,0.1),
                    c(0.1,0.35,0.35),
                    c(-1.386294,0.9808293,1.791759),
                    c(-1.609438,0.6931472,0.6),
                    c(0.1,0.1,0.2,0.3,0.0,0.0,0.1,0.1))

model <- c("GDINA","GDINA","GDINA","DINA","DINO","GDINA","ACDM","LLM","RRUM","GDINA")
N <- 500
Q <- sim10GDINA$simQ
# When simulating using delta.parm argument, model needs to be
# specified
sim <- simGDINA(N,Q,delta.parm = delta.list, model = model)


####################################################
#                   Example 7                      #
#      Data simulation (higher order DINA model)   #
####################################################

Q <- sim30GDINA$simQ
gs <- matrix(0.1,nrow(Q),2)
N <- 500
set.seed(12345)
theta <- rnorm(N)
K <- ncol(Q)
```

```
lambda <- data.frame(a=rep(1,K),b=seq(-2,2,length.out=K))
sim <- simGDINA(N,Q,gs.parm = gs, model="DINA", att.dist = "higher.order",
                higher.order.parm = list(theta = theta,lambda = lambda))

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

Q <- sim30GDINA$simQ
gs <- matrix(0.1,nrow(Q),2)
models <- c(rep("GDINA",5),
            rep("DINO",5),
            rep("DINA",5),
            rep("ACDM",5),
            rep("LLM",5),
            rep("RRUM",5))
N <- 500
set.seed(12345)
theta <- rnorm(N)
K <- ncol(Q)
lambda <- data.frame(a=runif(K,0.7,1.3),b=seq(-2,2,length.out=K))
sim <- simGDINA(N,Q,gs.parm = gs, model=models, att.dist = "higher.order",
                higher.order.parm = list(theta = theta,lambda = lambda))


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


# See Chiu et al., (2009)

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

####################################
#          Example 10              #
#        Simulation using          #
#      user-specified att structure#
####################################

# --- User-specified attribute structure ----#
```

```
Q <- sim30GDINA$simQ
K <- ncol(Q)
# divergent structure A1->A2->A3;A1->A4->A5;A1->A4->A6
diverg <- list(c(1,2),
               c(2,3),
               c(1,4),
               c(4,5))
struc <- att.structure(diverg,K)

# data simulation
N <- 1000
# data simulation
gs <- matrix(0.1,nrow(Q),2)
simD <- simGDINA(N,Q,gs.parm = gs,
                  model = "DINA",att.dist = "multinomial",att.prior = struc$att.prob)


####################################################
#                   Example 11                     #
#                Data simulation                   #
#  (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",gs.args=list(mono.constraint=TRUE))

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

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

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

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

####################################################
#                   Example 12                     #
#                Data simulation                   #
# (Sequential G-DINA model - polytomous responses) #
####################################################

set.seed(12345)

N <- 2000
```

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

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

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

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

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

## End(Not run)

#####################################################
#                   Example 13
#         DINA model Attribute generated using
#             multinomial distribution
#####################################################

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

#####################################################
#                   Example 14
#                  MS-DINA model
#####################################################


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

```
  item3=c(0.2,0.6),
  item4=c(0.2,0.7),
  item5=c(0.1,0.8))

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

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

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

`startGDINA` *Graphical user interface of the GDINA function*

Description

An experimental interactive Shiny application for running GDINA function

Usage

```
startGDINA()
```

Author(s)

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

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

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

unrestrQ *Generate unrestricted Qc matrix from an restricted Qc matrix*

Description

Generate unrestricted Qc matrix from an restricted Qc matrix

Usage

```
unrestrQ(Qc)
```

Arguments

Qc	an restricted Qc matrix

Value

an unrestricted Qc matrix

Examples

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

Index