Journal of Statistical Software

May 2020, Volume 93, Issue 14. doi: 10.18637/jss.v093.i14

GDINA: An R Package for Cognitive Diagnosis Modeling

Wenchao Ma
The University of Alabama

Jimmy de la Torre
The University of Hong Kong

Abstract

Cognitive diagnosis models (CDMs) have attracted increasing attention in educational measurement because of their potential to provide diagnostic feedback about students' strengths and weaknesses. This article introduces the feature-rich R package **GDINA** for conducting a variety of CDM analyses. Built upon a general model framework, a number of CDMs can be calibrated using the **GDINA** package. Functions are also available for evaluating model-data fit, detecting differential item functioning, validating the item and attribute association, and examining classification accuracy. A grapical user interface is also provided for researchers who are less familar with R. This paper contains both technical details about model estimation and illustrations about how to use the package for data analysis. The **GDINA** package is also used to replicate published results, showing that it could provide comparable model parameter estimation.

Keywords: cognitive diagnosis, psychometrics, item response theory, G-DINA model.

1. Introduction

Cognitively diagnostic assessments (CDA; de la Torre and Minchen 2014; Nichols, Chipman, and Brennan 1995) have gained increasing popularity in the past decades in the field of educational measurement. Traditional standardized educational assessments usually base on unidimensional item response theory (IRT; e.g., de Ayala 2013), which assumes that a single latent trait (or overall ability) is measured. Consequently, students are located on a continuum based on their performance in assessments using appropriate IRT models. In contrast, CDAs usually depend on cognitive diagnosis models (CDMs) with an intention to provide diagnostic information about students' strengths and weaknesses. To obtain such information, the assessments are typically designed to measure a set of finer-grained skills, which are usually referred to as attributes, and treated as binary latent variables with outcome

1 for mastery and 0 for nonmastery. The major goal of CDM analyses is to infer students' attribute profiles from their item responses.

A wide array of CDMs (e.g., Rupp, Templin, and Henson 2010) have been developed, and most of them, if not all, consist of two major components. One component is an item and attribute association matrix, or Q-matrix (Tatsuoka 1983), which specifies whether an attribute is needed to perform an item correctly. The Q-matrix is usually developed by domain experts, and assumed to be correct in the following CDM analyses. The second component is referred to as the condensation rule (Maris 1999), which defines how attributes are "condensed" to yield manifest item responses. Some CDMs are formulated to accommodate some specific condensation rule. For example, the deterministic inputs, noisy "and" gate (DINA; Haertel 1989) model employs a conjunctive rule, and assigns the highest probability of answering correctly to individuals that possess all of the required attributes. The deterministic inputs, noisy "or" gate (DINO; Templin and Henson 2006) model, however, adopts a disjunctive rule, and assigns the highest probability of answering correctly to individuals mastering at least one of the required attributes. In addition, the additive CDM (A-CDM; de la Torre 2011) assumes that each required attribute contributes to the success probability uniquely and independently. Apart from these specific CDMs, general or saturated CDMs subsuming many widely-used specific CDMs have also been developed, including the generalized DINA (G-DINA; de la Torre 2011) model, the general diagnostic model (GDM; von Davier 2008) and the log-linear CDM (LCDM; Henson, Templin, and Willse 2009).

The present work introduces the R (R Core Team 2020) package **GDINA** (Ma and de la Torre 2020b) for conducting a variety of CDM analyses based on the G-DINA model (de la Torre 2011) and its extensions. The package is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=GDINA`. Currently, there exist a few packages for the R programming environment which implement CDM analyses, including the **ACTCD** (Chiu and Ma 2018), **CDM** (George, Robitzsch, Kiefer, Gross, and Ünlü 2016), **dina** (Culpepper 2015), and **NPCD** (Zheng and Chiu 2019) packages. The **dina** package estimates the DINA model using the Gibbs sampler, and the **ACTCD** and **NPCD** packages are designed for nonparametric CDM analyses. The **CDM** package is capable of estimating the GDM, the structured CDMs (Formann 1992), the regularized latent class model (Chen, Li, Liu, and Ying 2017), as well as the G-DINA model. In addition to these R packages, some general purpose commercial software programs, such as **Mplus** (Muthén and Muthén 2017) and **Latent GOLD** (Vermunt and Magidson 2016), are also capable of fitting the G-DINA model by imposing some specific constraints. However, since they are not designed for CDM analyses, they usually lack some important functionalities (e.g., Q-matrix validation), and tend to be slow.

The **GDINA** package is primarily developed for conducting CDM analyses using the G-DINA model (de la Torre 2011) and its extensions (e.g., Chen and de la Torre 2013; Ma and de la Torre 2016; Ma, Terzi, Lee, and de la Torre 2017; Ma 2019b), hence the name. It has several distinguishing features compared with existing R packages for cognitive diagnosis at the time of writing. In particular, the **GDINA** package provides a set of unique tools for model diagnostics under the G-DINA model framework. For example, the **GDINA** package is the only program that allows for empirically validating the Q-matrix under a general model framework using a variety of approaches (de la Torre and Chiu 2016; de la Torre and Ma 2016; Ma and de la Torre 2020a; Najera, Sorrel, and Abad 2019) and selecting models at item level using the Wald test, likelihood ratio test or Lagrange multiplier test. The **GDINA**

package also allows users to assess global model-data fit using the M_2 statistic for dichotomous response (Hansen, Cai, Monroe, and Li 2016; Liu, Tian, and Xin 2016) and the M_{ord} statistic for ordinal response (Ma 2020). In addition, the **GDINA** package allows users to calibrate the generalized multiple-strategy models for dichotomous response (Ma and Guo 2019) and diagnostic tree model for polytomous response (Ma 2019b) when multiple strategies exist. Furthermore, the **GDINA** package provides a routine to validate the Q-matrix, choose the most appropriate CDMs for each item, and calibrate the selected CDMs at one fell swoop. Last, the **GDINA** package is the only R package that offers a graphical user interface for various CDM analyses.

The paper is organized as follows. In Section 2 we review the G-DINA model framework, including various parameterizations of item response functions and joint attribute distribution. In Section 3, we present the details of the EM algorithm for structural parameter estimation. We introduce the features of the **GDINA** package in Section 4 and analyze a real dataset for illustrative purpose in Section 5. In Section 6, we compare the **GDINA** package and commercial software programs in terms of parameter estimation and speed based on two sets of real data. We conclude with a discussion on possible future developments in Section 7.

2. The G-DINA model framework

Suppose a test with J items measures K binary attributes. The association between items and attributes is specified in a $J \times K$ Q-matrix (Tatsuoka 1983), with element $q_{jk} = 1$ indicating item j measures attribute k and $q_{jk} = 0$ indicating item j does not measure attribute k. Also, K binary attributes produce 2^K attribute profiles, each labeling a unique latent class. The attribute profile for latent class c is denoted as $\boldsymbol{\alpha}_c = [\alpha_{c1}, \ldots, \alpha_{cK}]^\top$, where $\alpha_{ck} = 1$ if attribute k is mastered and $\alpha_{ck} = 0$ if not. Superscript $\top$ is used to denote transposition. Let $\boldsymbol{a} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_i, \ldots, \boldsymbol{a}_N]^\top$ be a vector of attribute profiles of N individuals in a sample, where $\boldsymbol{a}_i \in \{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_c, \ldots, \boldsymbol{\alpha}_{2^K}\}$ is a K-dimensional random vector representing the attribute pattern of individual i. Let Y_{ij} be a response variable of individual i to item j, following a Bernoulli distribution with an observed realization y_{ij}. Also, denote $\boldsymbol{Y}_i = \{Y_{ij}\}$ as item response vector of individual i and $\boldsymbol{Y} = [\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_N]^\top$ the item response matrix from N individuals. To formulate a CDM, we need to define the item response function for the relation between item response and attribute profiles, and the model for joint attribute distribution, which gives the population proportion of each latent class.

2.1. Item response function

When not all attributes are required for item j, the G-DINA model (de la Torre 2011) collapses 2^K latent classes into $2^{K_j^*}$ latent groups, where $K_j^* = \sum_{k=1}^{K} q_{jk}$. The collapsed latent classes have the same success probability to item j. To simplify the notation, the first K_j^* attributes are assumed to be required for item j, and $\boldsymbol{\alpha}_{lj}^*$ is the reduced attribute vector consisting of the columns of the required attributes, where $l = 1, \ldots, 2^{K_j^*}$. The probability of individual i with attribute profile $\boldsymbol{\alpha}_c$ answering item j correctly is denoted by $\mathsf{P}(Y_{ij} = 1|\boldsymbol{\alpha}_c) = \mathsf{P}_j(\boldsymbol{\alpha}_c)$. Note that if latent class c is collapsed into latent group l, we have $\mathsf{P}_j(\boldsymbol{\alpha}_c) = \mathsf{P}(\boldsymbol{\alpha}_{lj}^*)$, where for the latter, the subscript j is dropped to avoid the redundancy. The item response function

(IRF) of the G-DINA model is given by

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

where $g[\cdot]$ stands for identity, log or logit link functions. Additionally, δ_{j0} is the intercept for item j, δ_{jk} is the main effect due to α_k, $\delta_{jkk'}$ is the interaction effect due to α_k and $\alpha_{k'}$, $\delta_{j12\cdots K_j^*}$ is the interaction effect due to $\alpha_1,\ldots,\alpha_{K_j^*}$. By setting appropriate constraints, as shown in de la Torre (2011), several widely used CDMs can be obtained. Specifically, to obtain the DINA model, all terms in the identity link G-DINA model, except δ_0 and $\delta_{12\cdots K_j^*}$, are constrained to zero, that is,

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

For the DINO model, the IRF is given by

$$\mathsf{P}(\boldsymbol{\alpha}_{lj}^*) = \delta_{j0} + \delta_{jk}\alpha_{lk},$$

where $\delta_{jk} = -\delta_{jk'k''} = \cdots = (-1)^{K_j^*+1}\delta_{j12\cdots K_j^*}$, for $k = 1,\ldots,K_j^*, k' = 1,\ldots,K_j^*-1$, and $k'' > k',\ldots,K_j^*$. This results in two parameters for the DINO model as well as the DINA model for each item, regardless of the number of attributes required. In addition, by setting all interactions at zero in the G-DINA model, the A-CDM, linear logistic model (LLM; Maris 1999), and reduced reparameterized unified model (R-RUM; Hartz 2002) can be obtained. Specifically, the A-CDM is the constrained identity-link G-DINA model without the interaction terms. It can be written by

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

The LLM is the logit link G-DINA model without any interaction terms, and can be formulated as

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

The R-RUM is the log-link G-DINA model without any interaction terms, and can be given by

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

These three models assume that mastering attribute α_{lk} raises the probability of success on item j, but the increases may be different due to the different scales they employ. As additive models, mastery of each attribute does not affect other attributes. Additionally, they all have the same number of parameters for item j (i.e., K_j^*+1).

2.2. Joint attribute distribution

Maris (1999) discussed a variety of approaches to parameterizing the joint attribute distribution. In this paper, the probability mass function of attribute profile $\boldsymbol{\alpha}_c$ is denoted by

$\pi_c = h(\boldsymbol{\lambda})$, where $\boldsymbol{\lambda}$ is a vector of unknown structural parameters. One of the simplest approaches is the independent model, which assumes that all attributes are independent statistically and can be written by

$$\pi_c = \prod_{k=1}^{K} \mathsf{P}(\alpha_k = 1)^{\alpha_{ck}} [1 - \mathsf{P}(\alpha_k = 1)]^{1-\alpha_{ck}}.$$

The independent model involves K parameters, $\boldsymbol{\lambda} = [\mathsf{P}(\alpha_1 = 1), \ldots, \mathsf{P}(\alpha_K = 1)]^\top$, where $\mathsf{P}(\alpha_k = 1)$ is the probability of mastering attribute k. Another model with simple parameterization is the saturated model, which involves 2^K parameters, $\boldsymbol{\lambda} = [\pi_1, \ldots, \pi_{2^K}]^\top$, with the constraint of $\sum_{c=1}^{2^K} \pi_c = 1$. Note that π_c is sometimes referred to as mixing proportion parameter, indicating the proportion of individuals in latent class c. The saturated model is considered the most general parameterization of the joint attribute distribution, but tends to involve too many parameters when K is large. In addition, Xu and von Davier (2008) proposed to use a loglinear model to smooth the joint attribute distribution, which can be written as

$$\log[n(\boldsymbol{\alpha}_c)] = \lambda_0 + \sum_{k=1}^{K} \lambda_k \alpha_k + \sum_{k=1}^{K-1} \sum_{k'=k+1}^{K} \lambda_{kk'} \alpha_k \alpha_{k'},$$

where $n(\boldsymbol{\alpha}_c) = N\pi_c$ is the number of individuals with attribute profile $\boldsymbol{\alpha}_c$. The above loglinear structural model considers main effects and first-order interactions. It has $1+K(K+1)/2$ parameters, namely, $\boldsymbol{\lambda} = [\lambda_0, \lambda_1, \ldots, \lambda_{K-1,K}]^\top$. It is very flexible in that it can be simplified by removing all interactions between attributes, or extended by including higher-order attribute interactions, but its parameters do not have straightforward interpretation. Another method for parameterizing the joint attribute distribution is the higher-order model (de la Torre and Douglas 2004), which assumes that the mastery of each attribute is influenced by a higher-order latent trait and that their relation is defined using IRT models, such as the two parameter logistic (2PL; see de Ayala 2013) model:

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

where θ represents the unidimensional general ability and $\boldsymbol{\lambda} = [\lambda_{01}, \ldots, \lambda_{0K}, \lambda_{11}, \ldots, \lambda_{1K}]^\top$ is a vector of higher-order structural parameters. Note that setting $\lambda_{1k} = \lambda_1 \ \forall k$ yields the one parameter logistic (1PL; see de Ayala 2013) model, and setting $\lambda_{1k} = 1 \ \forall k$ produces the Rasch model (Rasch 1960). Under the assumption of local independence,

$$\mathsf{P}(\boldsymbol{\alpha}_c|\theta) = \prod_{k=1}^{K} \mathsf{P}_k(\theta)^{\alpha_{ck}} \left[1 - \mathsf{P}_k(\theta)\right]^{1-\alpha_{ck}}$$

and the probability mass function can be obtained by integrating out θ, which can be approximated by Gauss-Hermite quadrature:

$$\pi_c = \int \mathsf{P}(\boldsymbol{\alpha}_c|\theta) f(\theta) d\theta \approx \sum_{s=1}^{S} \mathsf{P}(\boldsymbol{\alpha}_c|\tilde{\theta}_s) W(\tilde{\theta}_s),$$

where $\tilde{\theta}_s$ and $W(\tilde{\theta}_s)$ are the quadrature nodes and weights, respectively. The accuracy of the approximation can be improved by increasing the number of quadrature nodes S.

Note that the higher-order model defined in de la Torre and Douglas (2004) is more general by allowing the higher-order latent trait to be multidimensional, but only the unidimensional θ is considered in this paper, as well as in the **GDINA** package. There is no existing software program that can accommodate higher-order CDMs with multidimensional higher-order latent traits so far. This is one of the features that we consider incorporating into the **GDINA** package later.

3. Model estimation

Let $\boldsymbol{\gamma} = [\boldsymbol{\delta}^\top, \boldsymbol{\lambda}^\top]^\top$ denote a vector of structural parameters involved in a CDM, including item parameters $\boldsymbol{\delta}$ and joint attribute distribution parameters $\boldsymbol{\lambda}$. In addition, for individual i, attribute profile $\boldsymbol{\alpha}_i$, as well as higher-order ability θ_i when a higher-order model is employed for joint attribute distribution, need to be estimated, which are typically referred to as incidental parameters. The G-DINA model was estimated by maximizing the marginalized likelihood (de la Torre 2011), which, as shown in Bock and Aitkin (1981), can be implemented using the expectation-maximization algorithm (EM; Dempster, Laird, and Rubin 1977). The EM algorithm is a general procedure for finding MLEs when missing data exist. In the current context, for individual i, item responses $\boldsymbol{y}_i$ can be regarded as "incomplete" data, person parameter $\boldsymbol{\alpha}_i$ as missing data, and $\boldsymbol{x}_i = [\boldsymbol{y}_i^\top, \boldsymbol{\alpha}_i^\top]^\top$ as the "complete" data. When a higher-order model is used, the complete-data for individual i is $\boldsymbol{x}_i = [\boldsymbol{y}_i^\top, \boldsymbol{\alpha}_i^\top, \theta_i]^\top$ since both $\boldsymbol{\alpha}_i$ and θ_i are missing data.

The goal of the EM algorithm is to find the maxima of the incomplete-data likelihood indirectly by maximizing the complete-data log likelihood iteratively. More specifically, the EM algorithm consists of two steps: the expectation (E) step and the maximization (M) step. In the E-step, we need to calculate the so-called Q function (Dempster *et al.* 1977), which is the expected log likelihood of the complete-data conditional on the observed data and current parameter estimates, and takes the following form:

$$Q(\boldsymbol{\gamma}; \boldsymbol{\gamma}') = E_{\boldsymbol{X}|\boldsymbol{y},\boldsymbol{\gamma}'}\big[\log L(\boldsymbol{\gamma}; \boldsymbol{X})\big],$$

where $\boldsymbol{\gamma}'$ denotes the parameter estimates from the previous step. In the M-step, the Q function is maximized. The E- and M-steps repeat until certain convergence criteria have been met.

It can be shown that, when individuals are independent and the joint attribute distribution is not modeled using the higher-order model,

$$Q(\boldsymbol{\gamma}; \boldsymbol{\gamma}') = \sum_{c=1}^{2^K} n_c \log\left[\pi_c\right] + \sum_{j=1}^{J}\sum_{c=1}^{2^K}\Big[r_{jc}\log\left[\mathsf{P}_j(\boldsymbol{\alpha}_c)\right] + (n_c - r_{jc})\log\left[1-\mathsf{P}_j(\boldsymbol{\alpha}_c)\right]\Big],$$

where n_c is the expected number of individuals in latent class c and r_{jc} is the expected number of individuals in latent class c who answer item j correctly. They can be calculated by

$$n_c = \sum_{i=1}^{N} \mathsf{P}(\boldsymbol{\alpha}_c|\boldsymbol{y}_i, \boldsymbol{\gamma}') \text{ and}$$

$$r_{jc} = \sum_{i=1}^{N} y_{ij}\mathsf{P}(\boldsymbol{\alpha}_c|\boldsymbol{y}_i, \boldsymbol{\gamma}'),$$

where $\mathsf{P}(\boldsymbol{\alpha}_c|\boldsymbol{y}_i, \boldsymbol{\gamma}')$ is the posterior probability of individual i being assigned to latent class c, and can be calculated using the Bayes rule:

$$\mathsf{P}(\boldsymbol{\alpha}_c|\boldsymbol{y}_i, \boldsymbol{\gamma}') = \frac{\mathsf{P}(\boldsymbol{y}_i|\boldsymbol{\alpha}_c, \boldsymbol{\gamma}')\pi_c}{\sum_c \mathsf{P}(\boldsymbol{y}_i|\boldsymbol{\alpha}_c, \boldsymbol{\gamma}')\pi_c}.$$

Here,

$$\mathsf{P}(\boldsymbol{y}_i|\boldsymbol{\alpha}_c, \boldsymbol{\gamma}') = \prod_{j=1}^{J} \left[\mathsf{P}(Y_{ij} = 1|\boldsymbol{\alpha}_c, \boldsymbol{\gamma}')\right]^{y_{ij}} \left[1 - \mathsf{P}(Y_{ij} = 1|\boldsymbol{\alpha}_c, \boldsymbol{\gamma}')\right]^{1-y_{ij}}.$$

When individuals are independent and the joint attribute distribution is parameterized using a higher-order model,

$$\begin{aligned} Q(\boldsymbol{\gamma}; \boldsymbol{\gamma}') = &\sum_{j=1}^{J}\sum_{c=1}^{2^K} \Big[r_{jc} \log\left[\mathsf{P}_j(\boldsymbol{\alpha}_c)\right] + (n_c - r_{jc}) \log\left[1 - \mathsf{P}_j(\boldsymbol{\alpha}_c)\right]\Big] + \\ &\sum_{k=1}^{K}\sum_{s=1}^{S} \Big[r_{ks} \log\left[\mathsf{P}_k(\tilde{\theta}_s)\right] + (n_s - r_{ks}) \log\left[1 - \mathsf{P}_k(\tilde{\theta}_s)\right]\Big] + \sum_{s=1}^{S} n_s \log\left[W(\tilde{\theta}_s)\right], \end{aligned}$$

where n_s is the expected number of individuals having ability $\tilde{\theta}_s$, which can be calculated by

$$n_s = \sum_{i=1}^{N}\sum_{c=1}^{2^K} \mathsf{P}(\boldsymbol{\alpha}_c, \tilde{\theta}_s|\boldsymbol{y}_i) = \sum_{c=1}^{2^K} n_c \frac{\mathsf{P}(\boldsymbol{\alpha}_c|\tilde{\theta}_s)W(\tilde{\theta}_s)}{\sum_s \mathsf{P}(\boldsymbol{\alpha}_c|\tilde{\theta}_s)W(\tilde{\theta}_s)},$$

and r_{ks} is the expected number of individuals with ability $\tilde{\theta}_s$ mastering attribute k, and can be calculated by

$$r_{ks} = \sum_{i=1}^{N}\sum_{c=1}^{2^K} \alpha_{ck}\mathsf{P}(\boldsymbol{\alpha}_c, \tilde{\theta}_s|\boldsymbol{y}_i) = \sum_{c=1}^{2^K} n_c \frac{\alpha_{ck}\mathsf{P}(\boldsymbol{\alpha}_c|\tilde{\theta}_s)W(\tilde{\theta}_s)}{\sum_s \mathsf{P}(\boldsymbol{\alpha}_c|\tilde{\theta}_s)W(\tilde{\theta}_s)}.$$

In the E-step, n_c and r_{jc}, as well as n_s and r_{ks} when a higher-order model is employed, are calculated based on $\boldsymbol{\gamma}'$, and then, in the M-step, $Q(\boldsymbol{\gamma}; \boldsymbol{\gamma}')$ is maximized with respect to model parameters $\boldsymbol{\gamma}$.

4. Implementation in the GDINA package

The main function of the R package **GDINA** is `GDINA()`, which allows the calibration of a variety of CDMs within the G-DINA model framework. In the **GDINA** package, the G-DINA model is specified similar to the generalized linear model using design matrix and link function (de la Torre 2011). In particular, let $\boldsymbol{\delta}_j = [\delta_{j0}, \ldots, \delta_{j12\ldots K_j^*}]^\top$ be a vector of parameters of item j and $\mathbf{P}_j = \{\mathsf{P}(\boldsymbol{\alpha}_{lj}^*)\}$ be a vector of item success probabilities. The G-DINA model is written as

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

where $\boldsymbol{M}_j$ is the design matrix (de la Torre 2011) and $g[\cdot]$ the link function. Let us assume item j requires 2 attributes, for the G-DINA model,

$$\boldsymbol{M}_j^{\text{G-DINA}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

Note that the aforementioned DINA, DINO, A-CDM, LLM and R-RUM models can also be written in this form by specifying appropriate design matrices. For example, the $\boldsymbol{M}_j$ for the DINA and DINO models can be written by

$$\boldsymbol{M}_j^{\text{DINA}} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \text{ and } \boldsymbol{M}_j^{\text{DINO}} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix},$$

respectively. Furthermore, it is possible to define the design matrix and link function to obtain new models. For example, as pointed out by de la Torre (2011), the multiple-strategy DINA model (de la Torre and Douglas 2008) can be obtained by defining the design matrix appropriately. The Bug-DINO model (Kuo, Chen, Yang, and Mok 2016) treats attributes as misconceptions and assumes that individuals are not expected to answer an item correctly if they possess any of the misconceptions measured by the item. A similar model, referred to as Bug-DINA model, can be defined by assuming that individuals are expected to answer an item correctly if they do not possess all of the misconceptions measured by the item. Both can be viewed as special cases of the G-DINA model with design matrices:

$$\boldsymbol{M}_j^{\text{Bug-DINA}} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix} \text{ and } \boldsymbol{M}_j^{\text{Bug-DINO}} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}.$$

From the design matrices, it is straightforward to note that the Bug-DINA and Bug-DINO models are just reparameterized DINA and DINO models. In the **GDINA** package, one can define their own models by specifying the design matrix and link function for each item.

Model parameters are estimated using the EM algorithm. The E-step and many other computationally intensive functions are written in C++ through the **Rcpp** (Eddelbuettel and François 2011) and **RcppArmadillo** (Eddelbuettel and Sanderson 2014) R packages to speed up the execution. In the M-step, for a few models, closed-form solutions exist; whereas for other models, some optimization routines are needed to maximize the Q function. Possible solvers include `optim` from the **stats** package (R Core Team 2020), `slsqp` fromthe **nloptr** package (Johnson 2010), `auglag` from the **alabama** package (Varadhan 2015), or `solnp` from the **Rsolnp** package (Ghalanos and Theussl 2015). By default, the solvers are automatically chosen according to the model to be calibrated, and if one optimization routine fails, another one may be employed. Users can specify the solver in the `GDINA()` function if they have any preference.

Several methods are available to further analyze the object returned from the `GDINA()` function. For example, `print()` and `summary()` can be used to display some summary information of model estimation, and `AIC()` and `BIC()` can be used to calculate the Akaike (1974) information criterion and the Schwarz (1978) information criterion, respectively. In addition, `coef()` can be used to extract structural parameters while `personparm()` can be used to estimate person parameters.

In addition to model estimation, several statistical procedures are available in the **GDINA** package. First, the Q-matrix can be empirically validated using the `Qval()` function, which implements de la Torre and Chiu's (2016) ς^2 approach, the ς^2 approach based on empirical

cutoffs (Najera *et al.* 2019), and the stepwise method based on the Wald test (Ma and de la Torre 2020a). A mesa plot (de la Torre and Ma 2016; Ma 2019a) can be created using the `plot()` function, which is a line graph based on the proportion of variance accounted for by each candidate q-vector (de la Torre and Chiu 2016) to provide a way of visually pinpointing the best q-vector(s) for each item. Second, the `modelcomp()` function can be used to evaluate whether, for an item which requires at least two attributes, the G-DINA model can be replaced by a reduced model without a significant loss of model-data fit using the Wald test (de la Torre 2011; de la Torre and Lee 2013; Ma, Iaconangelo, and de la Torre 2016; Ma and de la Torre 2018), score test (Sorrel, Abad, Olea, de la Torre, and Barrada 2017a), likelihood ratio (LR) test or two-step approximated LR test (Ma and de la Torre 2018; Sorrel, de la Torre, Abad, and Olea 2017b). Further, the `modelfit()` function evaluates model-data fit by calculating the M_2 statistic, M_{ord} statistic, RMSEA_2 and SRMSR (Maydeu-Olivares and Joe 2014; Hansen *et al.* 2016; Liu *et al.* 2016; Ma 2020), and the `itemfit()` function calculates the log odds ratio and transformed correlations proposed by Chen, de la Torre, and Zhang (2013), which provide more details about the absolute fit for item pairs, and may be used to identify the sources of misfit. To compare nested models at the test level, an LR test can be conducted using `anova()`. Additionally, `dif()` detects differential item functioning using the Wald test (Hou, de la Torre, and Nandakumar 2014) or LR test (Ma *et al.* 2017).

5. Illustrations

The data for this illustration were collected from a learning experiment at the University of Tuebingen in Germany in 2010, and previously used by Philipp, Strobl, de la Torre, and Zeileis (2018). Twelve items in elementary probability theory were presented to participants and four attributes were involved: (α_1) calculate the probability of the complement of an event, (α_2) calculate the probability of two independent events, (α_3) calculate the classic probability of an event, and (α_4) calculate the probability of the union of two disjoint events. Responses of 504 participants from the first part of the experiment were used. The data and Q-matrix are available from the R package **pks** (Heller and Wickelmaier 2013). Specifically, after installing the **pks** package, the data can be loaded by

```
R> data("probability", package = "pks")
R> pb <- probability[, sprintf("b1%.2i", 1:12)]
```

Note that the data contain some missing values, which are coded as `NA`. The Q-matrix is

```
R> Q <- read.table(header = TRUE, text = "
+    cp id pb un
+    0  0  1  0
+    1  0  0  0
+    0  0  0  1
+    0  1  0  0
+    1  0  1  0
+    1  0  1  0
+    0  0  1  1
+    0  0  1  1
+    0  1  1  0
```

```
+    1  1  0  0
+    1  1  1  0
+    0  1  1  1")
```

We first fit the G-DINA model to the data by specifying data and Q-matrix as below. A challenge for the EM algorithm is that the solutions could be local maxima. We set `nstarts = 200` in argument `control` to request the function to evaluate the observed log likelihood based on 200 sets of randomly generated initial values. The best set of initial values is used for the following model calibration.

```
R> library("GDINA")
R> GDINA_est <- GDINA(dat = pb, Q = Q, control = list(nstarts = 200))
```

As shown below, some information about the data, model and estimation can be printed. The estimation converges quickly.

```
R> GDINA_est

Call:
GDINA(dat = pb, Q = Q, control = list(nstarts = 200))

  GDINA version 2.8.0 (2020-05-23)
================================================
Data
------------------------------------------------
# of individuals    groups    items
             504         1       12
================================================
Model
------------------------------------------------
Fitted model(s)       = GDINA
Attribute structure   = saturated
Attribute level       = Dichotomous
================================================
Estimation
------------------------------------------------
Number of iterations  = 161

For the final iteration:
Max abs change in item success prob. = 0.0001
Max abs change in mixing proportions = 0.0000
Change in -2 log-likelihood          = 0.0000
Converged?                           = TRUE

Time used             = 0.3979 secs
```

`summary()` can be used to print additional summary information about relative model-data fit, number of parameters and attribute prevalence. For attribute prevalence, columns labeled

"Level0" and "Level1" give the proportions of individuals who do not master and who master each attribute.

```
R> summary(GDINA_est)

Test Fit Statistics

Loglik = -2424.84

AIC    = 4975.69  | penalty [2 * p]  = 126.00
BIC    = 5241.71  | penalty [log(n) * p]  = 392.02
CAIC   = 5304.71  | penalty [(log(n) + 1) * p]  = 455.02
SABIC  = 5041.74  | penalty [log((n + 2)/24) * p]  = 192.05

No. of parameters (p)  = 63
No. of estimated item parameters =  48
No. of fixed item parameters =  0
No. of distribution parameters =  15

Attribute Prevalence

   Level0 Level1
A1 0.1191 0.8809
A2 0.2894 0.7106
A3 0.1732 0.8268
A4 0.1199 0.8801
```

`coef()` can be used to extract various types of item parameters. The first argument of `coef()` is the object returned from `GDINA()`. By default, `coef()` returns a list consisting of $\mathsf{P}(\boldsymbol{\alpha}_{lj}^*)$ for each item.

```
R> coef(GDINA_est)

$`Item 1`
  P(0)   P(1)
0.2245 0.9344

$`Item 2`
  P(0)   P(1)
0.2751 0.9741

$`Item 3`
  P(0)   P(1)
0.0971 0.9607

$`Item 4`
  P(0)   P(1)
```

```
0.1253 0.9625

$`Item 5`
 P(00)  P(10)  P(01)  P(11)
0.0993 0.6218 0.9999 0.8457

$`Item 6`
 P(00)  P(10)  P(01)  P(11)
0.1374 0.5730 0.8986 0.9571

$`Item 7`
 P(00)  P(10)  P(01)  P(11)
0.2632 0.6390 0.6701 0.9406

$`Item 8`
 P(00)  P(10)  P(01)  P(11)
0.3490 0.8735 0.9999 0.9476

$`Item 9`
 P(00)  P(10)  P(01)  P(11)
0.0847 0.5165 0.6607 0.7728

$`Item 10`
 P(00)  P(10)  P(01)  P(11)
0.0710 0.0351 0.9999 0.8068

$`Item 11`
P(000) P(100) P(010) P(001) P(110) P(101) P(011) P(111)
0.0766 0.0001 0.9999 0.0636 0.5102 0.0165 0.0007 0.7044

$`Item 12`
P(000) P(100) P(010) P(001) P(110) P(101) P(011) P(111)
0.0418 0.0001 0.0001 0.0001 0.2928 0.0001 0.0521 0.8206
```

`coef()` takes an argument `what`, where users can specify the type of item parameters. For example, by specifying `what = "delta"`, δ parameters in Equation 1 can be extracted (outputs are omitted).

```
R> coef(GDINA_est, what = "delta")
```

By specifying `what = "lambda"`, parameters involved in the joint attribute distribution can be extracted. By default, the saturated model is used and therefore, the mixing proportion parameters are printed:

```
R> coef(GDINA_est, what = "lambda")

p(0000) p(1000) p(0100) p(0010) p(0001) p(1100) p(1010) p(1001)
 0.0818  0.0129  0.0040  0.0176  0.0097  0.0000  0.0000  0.0494
```

```
p(0110) p(0101) p(0011) p(1110) p(1101) p(1011) p(0111) p(1111)
 0.0000  0.0012  0.0000  0.0036  0.0143  0.1180  0.0049  0.6827
```

Individuals' attribute patterns can be estimated using maximum likelihood estimation (MLE), expected a posteriori (EAP) or maximum a posteriori (MAP) as methods. To obtain attribute estimates, use the function `personparm()` and specify the object returned from `GDINA()` as the input. Similar to `coef()`, `personparm()` also has an argument called `what`, where users can specify the method for person attribute estimation. By default, the EAP method is employed. When using MLE or MAP, `personparm()` will print an additional column consisting of `TRUE` or `FALSE` indicating whether the likelihood or posterior have multiple modes or not. If there are multiple modes, the estimate returned is randomly selected. The following code shows the MAP estimates of the first 6 individuals and it can be found that the posterior distributions for all of these 6 individuals are unimodal.

```
R> map <- personparm(GDINA_est, what = "MAP")
R> map[1:6, ]

  A1 A2 A3 A4 multimodes
1  1  1  1  1      FALSE
2  1  1  1  1      FALSE
3  1  0  1  1      FALSE
4  1  1  1  1      FALSE
5  1  0  1  1      FALSE
6  1  1  1  1      FALSE
```

To fit other CDMs, we can modify argument `model` of the `GDINA()` function. It can be a character vector for each item or a scalar which will be used for all items. For example, to fit the DINA model, we set `model = "DINA"`. We can also modify `att.dist` to specify the model for the joint attribute distribution. The code below estimates a higher-order DINA model using the Rasch model. Note that the argument `higher.order` allows further specifications of the higher-order model.

```
R> HoDINA_est <- GDINA(dat = pb, Q = Q, model = "DINA",
+    control = list(nstarts = 200), att.dist = "higher.order",
+    higher.order = list(model = "Rasch"))
```

We can still use `summary()` to print summary information, `coef()` to print model parameters involved in item response function and joint attribute distribution, and `personparm()` to estimate person parmeters (including higher-order ability). For example, to print the structural parameters of the joint attribute distribution,

```
R> coef(HoDINA_est, what = "lambda")

   slope intercept
A1     1    3.1993
A2     1    1.1419
A3     1    2.5979
A4     1    2.6201
```

We can use `anova()` to compare two models:

```
R> anova(HoDINA_est, GDINA_est)

Information Criteria and Likelihood Ratio Test

           #par   logLik Deviance     AIC     BIC    CAIC   SABIC chisq df
HoDINA_est   28 -2579.39  5158.79 5214.79 5333.02 5361.02 5244.15 309.1 35
GDINA_est    63 -2424.84  4849.69 4975.69 5241.71 5304.71 5041.74
           p-value
HoDINA_est  <0.001
GDINA_est
```

In addition, we can explore whether the G-DINA model can be constrained to some reduced models for each item. To achieve this goal, `modelcomp()` can be used with the object returned from `GDINA()` as the input:

```
R> mc <- modelcomp(GDINA.obj = GDINA_est)
```

By default, `modelcomp()` performs the Wald test for each item that requires two or more attributes. The following code prints the primary results of model comparsions. The column labeled `"models"` gives the suggested CDM for each item based on the "simpler model + largest p value rule" (Ma *et al.* 2016). Specifically, if 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 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 the nominal level, the DINA and DINO models were preferred over the A-CDM, LLM, and R-RUM models because of their simplicity. The suggested models for the first four items are the G-DINA model because they are all single-attribute items.

```
R> mc

Item-level model selection:

test statistic: Wald
Decision rule: simpler model + largest p value rule at 0.05 alpha level.
Adjusted p values were based on holm correction.

        models pvalues adj.pvalues
Item 1   GDINA
Item 2   GDINA
Item 3   GDINA
Item 4   GDINA
Item 5    DINO   0.148           1
Item 6     LLM  0.4296           1
Item 7     LLM  0.6948           1
```

```
Item 8    DINO  0.7135          1
Item 9    DINO  0.2646          1
Item 10   DINA    0.68          1
Item 11   DINA  0.6206          1
Item 12   DINA  0.9999          1
```

Users can specify `method = "LM"` or `method = "LR"` in `modelcomp()` for score and LR tests. Note that the null hypothesis is that the reduced model can fit the data as well as the G-DINA model, and hence, a non-significant result implies that the reduced model may be used instead of the G-DINA model. Note that `modelcomp()` is only suitable for nested models: the saturated G-DINA model and the models it subsumes. To compare non-nested models, users can use `AIC()` and `BIC()` functions.

Another important function in the **GDINA** package is `Qval()` for Q-matrix validation. Based on the object returned from `GDINA()`, the following code examines whether any element in the Q-matrix is potentially misspecified using the stepwise Wald test (Ma and de la Torre 2020a).

```
R> Qv <- Qval(GDINA.obj = GDINA_est, method = "wald")
R> Qv

Q-matrix validation based on Stepwise Wald test

Suggested Q-matrix:

   A1 A2 A3 A4
1   0  0  1  0
2   1  0  0  0
3   0  0  0  1
4   0  1  0  0
5   1  0  1  0
6   1  0  1  0
7   0  0  1  1
8   0  0  1  1
9   0  0* 1  0
10 0* 1  0  0
11 0* 1  0* 0
12 0  1  0* 0*
Note: * denotes a modified element.
```

It can be observed that based on the data, a few modifications are suggested to the q-vectors of items 9, 10, 11 and 12. To better understand the results, we can draw mesa plots for these items. For example, the code below draws a mesa plot for item 12 (see Figure 1):

```
R> plot(Qv, item = 12)
```

The mesa plot in Figure 1, by default, has the best q-vectors given the number of attributes required on the x-axis. **0** is not a valid q-vector but presented for reference. de la Torre and

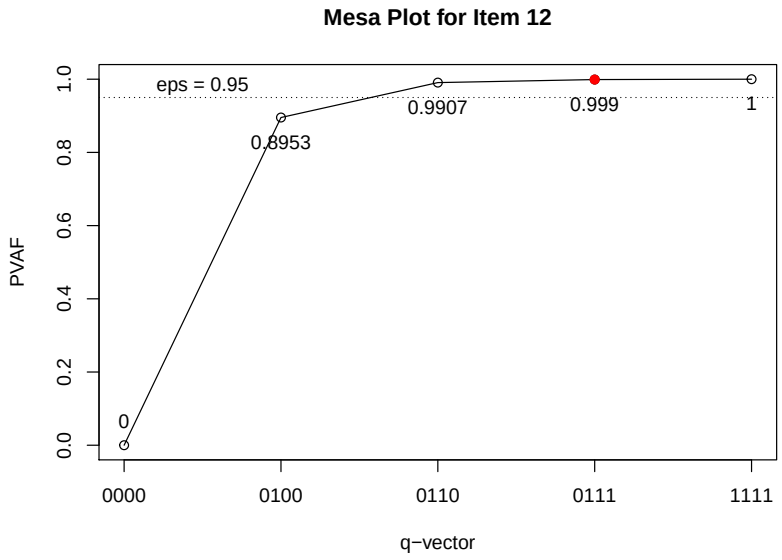

Figure 1: Mesa plot for item 12.

Ma (2016) noted that the q-vector on the edge of the mesa is potentially the best. In the figure, the one with red dot is the original q-vector. The plot shows that attribute 2 itself (i.e., $\boldsymbol{q}_j = [0100]^\top$) can explain most variation in the success probabilities, and that attributes 3 and 4 do not contribute much. This implies that attributes 3 and 4 may not be necessary to correctly answer this item. Note that the Q-matrix validation procedures in the **GDINA** package are based on the data at hand, and whether the suggested modifications should be incorporated should be subject to the judgement of domain experts.

6. Program comparisons

In this section, we use the R package **GDINA** to replicate published results based on two sets of real data analyzed using commercial software programs. The first replication is based on 536 students' responses to 20 fraction subtraction items, which has been previously analyzed by many researchers (e.g., de la Torre and Douglas 2004). **flexMIRT** (Cai 2017) has been used to fit the 1PL higher-order DINA model to the data, and the corresponding syntax and outputs can be found in its manual (Houts and Cai 2016, p. 156–159). The data and Q-matrix can be loaded by:

```
R> data("frac20", package = "GDINA")
R> frac_data <- frac20$dat
R> frac_Q <- frac20$Q
```

Using the **GDINA** R package, the 1PL higher-order DINA model can be estimated using the code below:

```
R> est <- GDINA(dat = frac_data,  Q = frac_Q,  model = "DINA",
+    att.dist = "higher.order", higher.order = list(model = "1PL",
+    InterceptRange = c(-5, 5), nquad = 49),
+    control = list(conv.crit = 1e-6))
```

Item	**flexMIRT**		**GDINA**	
	$\mathsf{P}(\boldsymbol{\alpha}_{lj}^* = \mathbf{0})$	$\mathsf{P}(\boldsymbol{\alpha}_{lj}^* = \mathbf{1})$	$\mathsf{P}(\boldsymbol{\alpha}_{lj}^* = \mathbf{0})$	$\mathsf{P}(\boldsymbol{\alpha}_{lj}^* = \mathbf{1})$
1	0.04	0.90	0.04	0.90
2	0.03	0.96	0.03	0.96
3	0.00	0.88	0.00	0.88
4	0.22	0.89	0.22	0.89
5	0.30	0.82	0.30	0.82
6	0.01	0.96	0.01	0.96
7	0.03	0.80	0.03	0.80
8	0.45	0.81	0.45	0.81
9	0.18	0.75	0.18	0.75
10	0.03	0.79	0.03	0.79
11	0.07	0.93	0.06	0.93
12	0.13	0.96	0.13	0.96
13	0.02	0.67	0.02	0.67
14	0.05	0.94	0.05	0.94
15	0.03	0.90	0.03	0.90
16	0.10	0.88	0.10	0.88
17	0.04	0.86	0.04	0.86
18	0.12	0.85	0.12	0.85
19	0.02	0.76	0.02	0.76
20	0.01	0.84	0.01	0.84

Table 1: Item parameter estimates from **flexMIRT** and **GDINA**.

Attribute	**flexMIRT**		**GDINA**	
	Slope	Intercept	Slope	Intercept
A1	3.82	−0.08	3.82	−0.08
A2	3.82	3.75	3.82	3.75
A3	3.82	2.34	3.82	2.34
A4	3.82	1.08	3.82	1.08
A5	3.82	−0.11	3.82	−0.11
A6	3.82	4.27	3.82	4.27
A7	3.82	3.99	3.82	3.99
A8	3.82	3.08	3.82	3.08

Table 2: Higher-order structural parameter estimates from **flexMIRT** and **GDINA**.

Note that we specify the 1PL model as the higher-order model by setting `model = "1PL"` in argument `higher.order`. We also set the range of intercept parameters at $[-5, 5]$ and the number of quadrature nodes at 49. In addition, we use a more stringent convergence criterion, that is, `1e-6`, instead of the default `1e-4`. The following code can be used to extract item and higher-order structural parameters:

```
R> gs <- coef(est, what = "gs")
R> ho <- coef(est, what = "lambda")
```

The results are given in Tables 1 and 2. The item parameter estimates of **flexMIRT** in Table 1

Item	Templin and Hoffman (2013)				**GDINA**			
	δ_{j0}	δ_{j1}			δ_{j0}	δ_{j1}		
	δ_{j0}	δ_{j1}	δ_{j2}	δ_{j12}	δ_{j0}	δ_{j1}	δ_{j2}	δ_{j12}
1	0.835	0.000	0.600	1.222	0.835	0.000	0.600	1.222
2	1.037	1.247			1.037	1.247		
3	−0.340	0.748	0.346	0.535	−0.340	0.748	0.346	0.535
4	−0.139	1.691			−0.139	1.691		
5	1.082	2.015			1.082	2.015		
6	0.865	1.692			0.865	1.692		
7	−0.106	2.855	0.952	−0.952	−0.106	2.855	0.952	−0.952
8	1.482	1.922			1.482	1.922		
9	0.119	1.195			0.119	1.195		
10	0.055	2.050			0.055	2.050		
11	−0.039	0.818	0.961	0.777	−0.039	0.818	0.961	0.777
12	−1.769	0.000	1.290	1.515	−1.768	0.000	1.290	1.515
13	0.660	1.630			0.660	1.630		
14	0.176	1.368			0.176	1.368		
15	0.996	2.114			0.996	2.114		
16	−0.104	2.341	0.892	−0.864	−0.104	2.344	0.892	−0.867
17	1.354	0.767	0.596	0.076	1.354	0.767	0.596	0.075
18	0.926	1.389			0.926	1.389		
19	−0.195	1.848			−0.195	1.848		
20	−1.389	0.243	0.908	1.410	−1.389	0.243	0.908	1.410
21	0.164	1.053	1.130	0.042	0.164	1.053	1.130	0.042
22	−0.872	2.245			−0.872	2.245		
23	0.664	2.071			0.664	2.071		
24	−0.673	1.522			−0.673	1.522		
25	0.092	1.136			0.092	1.136		
26	0.164	1.119			0.164	1.119		
27	−0.887	1.713			−0.886	1.713		
28	0.568	1.745			0.568	1.745		

Table 3: Item parameter estimates from **Mplus** and **GDINA**.

were obtained from its manual (Houts and Cai 2016, p. 159), where numbers were reported to only two decimal places. All item parameter estimates in Table 1 from **flexMIRT** and **GDINA** are virtually identical. The higher-order structural parameters of **flexMIRT** in Table 2 were obtained by re-running the **flexMIRT** code in its manual (see Example 7-7; Houts and Cai 2016, p. 156). The outputs are provided as supplementary material. It can be observed that the **GDINA** package and **flexMIRT** also yielded virtually identical estimates of higher-order structural parameter. It is worth noting that the **GDINA** package completed its calibration in 4.11 seconds, whereas on the same laptop, the **flexMIRT** calculation took 23.27 seconds.

The second replication is based on the Examination for the Certificate of Proficiency in English (ECPE) data set, which consists of 2922 students' responses to 28 items. The data and Q-matrix can be loaded by:

```
R> data("ecpe", package = "GDINA")
```

```
R> ecpe_data <- ecpe$dat
R> ecpe_Q <- ecpe$Q
```

Templin and Hoffman (2013) fit the loglinear CDM (LCDM; Henson *et al.* 2009) to the data using **Mplus** (Muthén and Muthén 2017). Templin and Hoffman (2013) also imposed monotonic constraints to the success probabilities so that mastering of additional attributes will not lead to a lower probability of success. More formally, the monotonic constraint implies that $\mathsf{P}(\boldsymbol{\alpha}_{lj}^*) \leq \mathsf{P}(\boldsymbol{\alpha}_{l'j}^*)$ if $\alpha_{lj}^* \prec \alpha_{l'j}^*$. Since the LCDM is equivalent to the logit link G-DINA model, it can be estimated using the code below by specifying `model = "logitGDINA"`:

```
R> ecpe_est <- GDINA(dat = ecpe_data, Q = ecpe_Q, model = "logitGDINA",
+    mono.constraint = TRUE, control = list(conv.crit = 1e-6))
```

In the code above, `mono.constraint = TRUE` ensures that the monotonic constraints are met during the calibration. Table 3 gives the δ parameter estimates from the **GDINA** package and Templin and Hoffman (2013) using **Mplus**. It can be found that the estimates are virtually identical with only a few differences at the third decimal places. We also provided **Mplus** code as supplemental material, which yielded identical parameter estimates as those in Table 3 from Templin and Hoffman (2013), with only a few exceptions at the third decimal places for items 16 and 23. Note that the **GDINA** package completed its calibration in 21.42 seconds, whereas **Mplus** took 32 minutes and 4 seconds.

7. Summary and discussion

This paper introduces the R package **GDINA** for conducting a variety of CDM analyses under the G-DINA model framework. There are several features that are potentially useful but were not demonstrated in this paper. First, the G-DINA model has been extended to accommodate polytomous attributes (Chen and de la Torre 2013), polytomous responses (Ma and de la Torre 2016) and multiple-group analysis (Ma *et al.* 2017). These models can also be estimated using the `GDINA()` function. Second, the package can also calibrate various models for multiple strategies, including the multiple strategy DINA model (Huo and de la Torre 2014), generalized multiple strategy CDMs (Ma and Guo 2019), and diagnostic tree models (Ma 2019b). Other models and approaches for cognitive diagnosis that can be handled by the package include the multiple-choice DINA model (de la Torre 2009) and iterative latent class analysis (Jiang 2019). Besides, `autoGDINA()` is a wrapper function to simplify the CDM analyses by conducting the Q-matrix validation, item-level model selection and model calibration sequentially in a one run. In addition, to simulate item responses based on various CDMs under the G-DINA model framework, `simGDINA()` can be employed. Last but not least, to make the package more accessible to users who are less familar with R, a graphical user interface (GUI), which is developed using **shiny** (Chang, Cheng, Allaire, Xie, and McPherson 2020) and **shinydashboard** (Chang and Borges Ribeiro 2018), can be started for various CDM analyses by calling `startGDINA()`. The GUI session will be launched in the user's default browser. Figure 2 shows the start-up page of the interface, where users could browse and import the item response matrix and Q-matrix in various formats. Figure 3 shows the second tab, where users could choose a CDM from a dropdown list for model calibration or specify different CDMs for different items. Users can also choose the options for validating Q-matrix and performing item-level model selection using the Wald test. After clicking the "CLICK

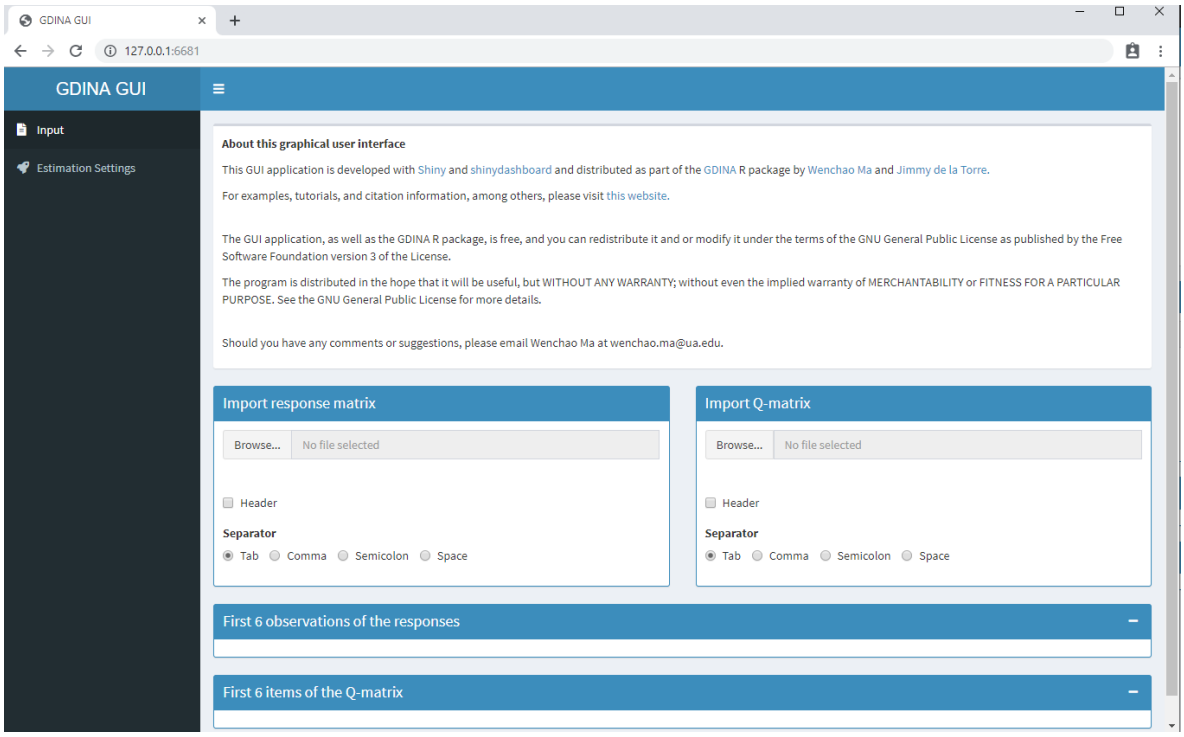

Figure 2: Start-up page of the **GDINA** GUI.

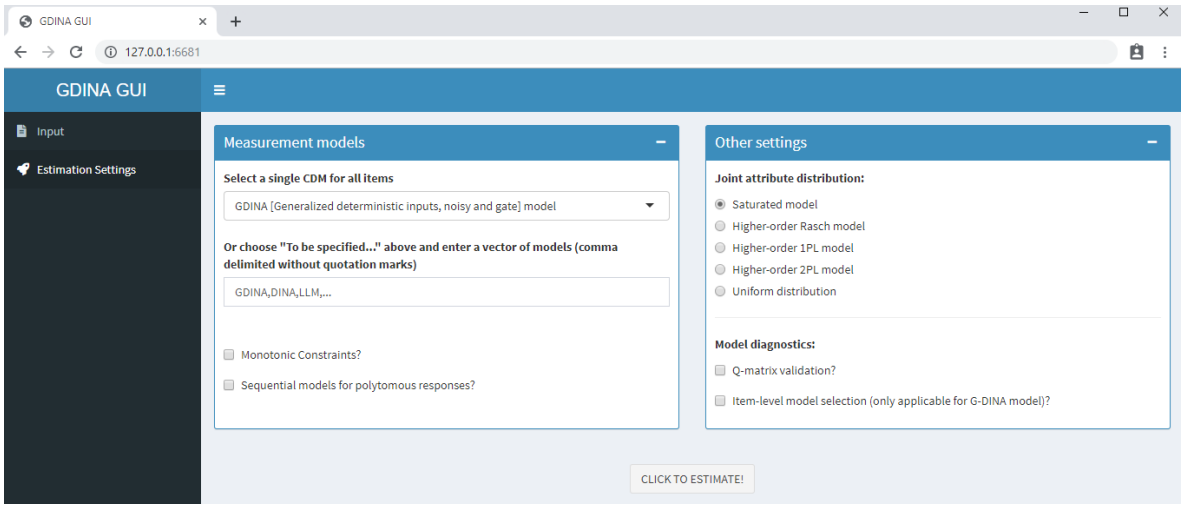

Figure 3: Model specification page of the **GDINA** GUI.

TO ESTIMATE!" button, a few other tabs will become visible to give various outputs. More details on the use of the GUI for CDM analyses can be found in Ma and de la Torre (2019) and de la Torre and Akbay (2019).

Because of the various functionality, the **GDINA** package could provide a set of useful tools for researchers and practitioners who are interested in CDMs. The package is being developed actively, and the features that may be incorporated into the package in the future include (1)

accommodating complex sampling design, (2) providing other methods for estimating standard errors in the EM algorithm, (3) providing procedures for detecting model identifiability and (4) accommodating a large dataset and Q-matrix.

Computational details

The results in this paper were obtained using R 4.0.0 with the **GDINA** 2.8.0 package. R itself and all packages used are available from CRAN at https://CRAN.R-project.org/. All analyses were conducted on a laptop with Intel i7-7600U CPU, 8GB RAM and Windows 10 OS.

References

Akaike H (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control*, **19**(6), 716–723. doi:10.1109/TAC.1974.1100705.

Bock RD, Aitkin M (1981). "Marginal Maximum Likelihood Estimation of Item Parameters: Application of an EM Algorithm." *Psychometrika*, **46**(4), 443–459. doi:10.1007/bf02293801.

Cai L (2017). "**flexMIRT** 3.51: Flexible Multilevel Multidimensional Item Analysis and Test Scoring." Vector Psychometric Group.

Chang W, Borges Ribeiro B (2018). ***shinydashboard**: Create Dashboards with **shiny***. R package version 0.7.1, URL https://CRAN.R-project.org/package=shinydashboard.

Chang W, Cheng J, Allaire JJ, Xie Y, McPherson J (2020). ***shiny**: Web Application Framework for R*. R package version 1.4.0.2, URL https://CRAN.R-project.org/package=shiny.

Chen J, de la Torre J (2013). "A General Cognitive Diagnosis Model for Expert-Defined Polytomous Attributes." *Applied Psychological Measurement*, **37**(6), 419–437. doi:10.1177/0146621613479818.

Chen J, de la Torre J, Zhang Z (2013). "Relative and Absolute Fit Evaluation in Cognitive Diagnosis Modeling." *Journal of Educational Measurement*, **50**(2), 123–140. doi:10.1111/j.1745-3984.2012.00185.x.

Chen Y, Li X, Liu J, Ying Z (2017). "Regularized Latent Class Analysis with Application in Cognitive Diagnosis." *Psychometrika*, **82**(3), 660–692. doi:10.1007/s11336-016-9545-6.

Chiu CY, Ma W (2018). "**ACTCD**: Asymptotic Classification Theory for Cognitive Diagnosis." R package version 1.2.0, URL https://CRAN.R-project.org/package=ACTCD.

Culpepper SA (2015). "Bayesian Estimation of the DINA Model with Gibbs Sampling." *Journal of Educational and Behavioral Statistics*, **40**(5), 454–476. doi:10.3102/1076998615595403.

de Ayala RJ (2013). *The Theory and Practice of Item Response Theory*. Guilford Publications.

de la Torre J (2009). "A Cognitive Diagnosis Model for Cognitively Based Multiple-Choice Options." *Applied Psychological Measurement*, **33**(3), 163–183. doi:10.1177/0146621608320523.

de la Torre J (2011). "The Generalized DINA Model Framework." *Psychometrika*, **76**(2), 179–199. doi:10.1007/s11336-011-9207-7.

de la Torre J, Akbay L (2019). "Implementation of Cognitive Diagnosis Modeling Using the **GDINA** R Package." *Eurasian Journal of Educational Research*, **80**, 171–192. doi:10.14689/ejer.2019.80.9.

de la Torre J, Chiu CY (2016). "A General Method of Empirical Q-Matrix Validation." *Psychometrika*, **81**(2), 253–273. doi:10.1007/s11336-015-9467-8.

de la Torre J, Douglas J (2008). "Model Evaluation and Multiple Strategies in Cognitive Diagnosis: An Analysis of Fraction Subtraction Data." *Psychometrika*, **73**(4), 595–624. doi:10.1007/s11336-008-9063-2.

de la Torre J, Douglas JA (2004). "Higher-Order Latent Trait Models for Cognitive Diagnosis." *Psychometrika*, **69**(3), 333–353. doi:10.1007/bf02295640.

de la Torre J, Lee YS (2013). "Evaluating the Wald Test for Item-Level Comparison of Saturated and Reduced Models in Cognitive Diagnosis." *Journal of Educational Measurement*, **50**(4), 355–373. doi:10.1111/jedm.12022.

de la Torre J, Ma W (2016). "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.

de la Torre J, Minchen N (2014). "Cognitively Diagnostic Assessments and the Cognitive Diagnosis Model Framework." *Psicología Educativa*, **20**(2), 89–97. doi:10.1016/j.pse.2014.11.001.

Dempster AP, Laird NM, Rubin DB (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society B*, **39**(1), 1–38.

Eddelbuettel D, François R (2011). "**Rcpp**: Seamless R and C++ Integration." *Journal of Statistical Software*, **40**(8), 1–18. doi:10.18637/jss.v040.i08.

Eddelbuettel D, Sanderson C (2014). "**RcppArmadillo**: Accelerating R with High-Performance C++ Linear Algebra." *Computational Statistics & Data Analysis*, **71**, 1054–1063. doi:10.1016/j.csda.2013.02.005.

Formann AK (1992). "Linear Logistic Latent Class Analysis for Polytomous Data." *Journal of the American Statistical Association*, **87**(418), 476–486. doi:10.1080/01621459.1992.10475229.

George AC, Robitzsch A, Kiefer T, Gross J, Ünlü A (2016). "The R Package **CDM** for Cognitive Diagnosis Models." *Journal of Statistical Software*, **74**(2), 1–24. doi:10.18637/jss.v074.i02.

Ghalanos A, Theussl S (2015). ***Rsolnp**: General Non-Linear Optimization Using Augmented Lagrange Multiplier Method*. R package version 1.16, URL https://CRAN.R-project.org/package=Rsolnp.

Haertel EH (1989). "Using Restricted Latent Class Models to Map the Skill Structure of Achievement Items." *Journal of Educational Measurement*, **26**(4), 301–321. doi:10.1111/j.1745-3984.1989.tb00336.x.

Hansen M, Cai L, Monroe S, Li Z (2016). "Limited-Information Goodness-of-Fit Testing of Diagnostic Classification Item Response Models." *British Journal of Mathematical and Statistical Psychology*, **69**(3), 225–252. doi:10.1111/bmsp.12074.

Hartz SM (2002). *A Bayesian Framework for the Unified Model for Assessing Cognitive Abilities: Blending Theory With Practicality*. Ph.D. thesis, University of Illinois at Urbana-Champaign.

Heller J, Wickelmaier F (2013). "Minimum Discrepancy Estimation in Probabilistic Knowledge Structures." *Electronic Notes in Discrete Mathematics*, **42**, 49–56. doi:10.1016/j.endm.2013.05.145.

Henson RA, Templin JL, Willse JT (2009). "Defining a Family of Cognitive Diagnosis Models Using Log-Linear Models with Latent Variables." *Psychometrika*, **74**(2), 191–210. doi:10.1007/s11336-008-9089-5.

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**(1), 98–125. doi:10.1111/jedm.12036.

Houts CR, Cai L (2016). "**flexMIRT** User's Manual Version 3.51: Flexible Multilevel Multidimensional Item Analysis and Test Scoring." Vector Psychometric Group.

Huo Y, de la Torre J (2014). "Estimating a Cognitive Diagnostic Model for Multiple Strategies via the EM Algorithm." *Applied Psychological Measurement*, **38**(6), 464–485. doi:10.1177/0146621614533986.

Jiang Z (2019). "Using the Iterative Latent-Class Analysis Approach to Improve Attribute Accuracy in Diagnostic Classification Models." *Behavior Research Methods*, **51**(3), 1075–1084. doi:10.3758/s13428-018-01191-0.

Johnson SG (2010). "The **NLopt** Nonlinear-Optimization Package." URL http://ab-initio.mit.edu/nlopt.

Kuo BC, Chen CH, Yang CW, Mok MMC (2016). "Cognitive Diagnostic Models for Tests with Multiple-Choice and Constructed-Response Items." *Educational Psychology*, **36**(6), 1115–1133. doi:10.1080/01443410.2016.1166176.

Liu Y, Tian W, Xin T (2016). "An Application of M_2 Statistic to Evaluate the Fit of Cognitive Diagnostic Models." *Journal of Educational and Behavioral Statistics*, **41**(1), 3–26. doi:10.3102/1076998615621293.

Ma W (2019a). "Cognitive Diagnosis Modeling Using the **GDINA** R Package." In M von Davier, YS Lee (eds.), *Handbook of Diagnostic Classification Models*, pp. 593–601. Springer-Verlag. doi:10.1007/978-3-030-05584-4_29.

Ma W (2019b). "A Diagnostic Tree Model for Polytomous Responses with Multiple Strategies." *British Journal of Mathematical and Statistical Psychology*, **72**(1), 61–82. doi:10.1111/bmsp.12137.

Ma W (2020). "Evaluating Model Data Fit Using Limited Information Statistics for the Sequential G-DINA Model." *Applied Psychological Measurement*, **44**(3), 167–181. doi:10.1177/0146621619843829.

Ma W, de la Torre J (2016). "A Sequential Cognitive Diagnosis Model for Polytomous Responses." *British Journal of Mathematical and Statistical Psychology*, **69**(3), 253–275. doi:10.1111/bmsp.12070.

Ma W, de la Torre J (2018). "Category-Level Model Selection for the Sequential G-DINA Model." *Journal of Educational and Behavioral Statistics*, **44**(1), 45–77. doi:10.3102/1076998618792484.

Ma W, de la Torre J (2019). "Digital Module 05: Diagnostic Measurement – The G-DINA Framework." *Educational Measurement: Issues and Practice*, **38**(2), 114–115. doi:10.1111/emip.12262.

Ma W, de la Torre J (2020a). "An Empirical Q-Matrix Validation Method for the Sequential G-DINA Model." *British Journal of Mathematical and Statistical Psychology*, **73**(1), 142–163. doi:10.1111/bmsp.12156.

Ma W, de la Torre J (2020b). "**GDINA**: The Generalized DINA Model Framework." R package version 2.8.0, URL https://CRAN.R-project.org/package=GDINA.

Ma W, Guo W (2019). "Cognitive Diagnosis Models for Multiple Strategies." *British Journal of Mathematical and Statistical Psychology*, **72**(2), 370–392. doi:10.1111/bmsp.12155.

Ma W, Iaconangelo C, de la Torre J (2016). "Model Similarity, Model Selection, and Attribute Classification." *Applied Psychological Measurement*, **40**(3), 200–217. doi:10.1177/0146621615621717.

Ma W, Terzi R, Lee S, de la Torre J (2017). "Multiple Group Cognitive Diagnosis Models and their Applications in Detecting Differential Item Functioning." Paper presented at the Annual Meeting of the American Educational Research Association, San Antonio.

Maris E (1999). "Estimating Multiple Classification Latent Class Models." *Psychometrika*, **64**(2), 187–212. doi:10.1007/bf02294535.

Maydeu-Olivares A, Joe H (2014). "Assessing Approximate Fit in Categorical Data Analysis." *Multivariate Behavioral Research*, **49**(4), 305–328. doi:10.1080/00273171.2014.911075.

Muthén LK, Muthén BO (2017). ***Mplus*** *User's Guide*. Muthén & Muthén, Los Angeles, 8th edition.

Najera P, Sorrel M, Abad P (2019). "Reconsidering Cutoff Points in the General Method of Empirical Q-Matrix Validation." *Educational and Psychological Measurement*, **79**(4), 727–753. doi:10.1177/0013164418822700.

Nichols PD, Chipman SF, Brennan RL (1995). *Cognitively Diagnostic Assessment*. Lawrence Erlbaum Associates, Inc.

Philipp M, Strobl C, de la Torre J, Zeileis A (2018). "On the Estimation of Standard Errors in Cognitive Diagnosis Models." *Journal of Educational and Behavioral Statistics*, **43**(1), 88–115. doi:10.3102/1076998617719728.

Rasch G (1960). "Probabilistic Models for Some Intelligence and Achievement Tests." *Technical report*, Danish Institute for Educational Research, Copenhagen.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Rupp AA, Templin JL, Henson RA (2010). *Diagnostic Measurement: Theory, Methods, and Applications*. Guilford Press, New York.

Schwarz G (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**, 461–464. doi:10.1214/aos/1176344136.

Sorrel MA, Abad FJ, Olea J, de la Torre J, Barrada JR (2017a). "Inferential Item-Fit Evaluation in Cognitive Diagnosis Modeling." *Applied Psychological Measurement*, **41**(8), 614–631. doi:10.1177/0146621617707510.

Sorrel MA, de la Torre J, Abad FJ, Olea J (2017b). "Two-Step Likelihood Ratio Test for Item-Level Model Comparison in Cognitive Diagnosis Models." *Methodology: European Journal of Research Methods for the Behavioral and Social Sciences*, **13**(Suppl 1), 39–47. doi:10.1027/1614-2241/a000131.

Tatsuoka KK (1983). "Rule Space: An Approach for Dealing with Misconceptions Based on Item Response Theory." *Journal of Educational Measurement*, **20**(4), 345–354. doi:10.1111/j.1745-3984.1983.tb00212.x.

Templin JL, Henson RA (2006). "Measurement of Psychological Disorders Using Cognitive Diagnosis Models." *Psychological Methods*, **11**(3), 287–305. doi:10.1037/1082-989x.11.3.287.

Templin JL, Hoffman L (2013). "Obtaining Diagnostic Classification Model Estimates Using **Mplus**." *Educational Measurement: Issues and Practice*, **32**(2), 37–50. doi:10.1111/emip.12010.

Varadhan R (2015). ***alabama**: Constrained Nonlinear Optimization*. R package version 2015.3-1, URL https://CRAN.R-project.org/package=alabama.

Vermunt JK, Magidson J (2016). ***Latent GOLD** 5.1 User's Guide*. Statistical Innovations Inc, Belmont.

von Davier M (2008). "A General Diagnostic Model Applied to Language Testing Data." *British Journal of Mathematical and Statistical Psychology*, **61**(2), 287–307. doi:10.1348/000711007x193957.

Xu X, von Davier M (2008). "Fitting the Structured General Diagnostic Model to NAEP Data." *ETS Research Report Series*, **2008**(1), i–18. doi:10.1002/j.2333-8504.2008.tb02113.x.

Zheng Y, Chiu CY (2019). "**NPCD**: Nonparametric Methods for Cognitive Diagnosis." R package version 1.0-11, URL https://CRAN.R-project.org/package=NPCD.

Affiliation:

Wenchao Ma
Department of Educational Studies in Psychology, Research Methodology, and Counseling
College of Education
The University of Alabama
520 Colonial Dr. Room 307B
Tuscaloosa, AL 35487, United States of America
E-mail: wenchao.ma@ua.edu

Jimmy de la Torre
Faculty of Education
The University of Hong Kong
Pokfulam Road, Hong Kong
E-mail: j.delatorre@hku.hk

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
May 2020, Volume 93, Issue 14 *Submitted:* 2018-03-21
doi:10.18637/jss.v093.i14 *Accepted:* 2019-05-12