DeCarlo, L. T. (in press). Insights from reparameterized DINA and beyond. In M. von Davier, & Y.-S. Lee (Eds.), *Handbook of Diagnostic Classification Models* (pp.). New York: Springer.

Insights from Reparameterized DINA and Beyond

Lawrence T. DeCarlo

The purpose of cognitive diagnosis is to obtain information about the set of skills or attributes that examinees have or do not have. A cognitive diagnostic model (CDM) attempts to extract this information from the pattern of responses of examinees to test items. A number of general CDMs have been proposed, such as the general diagnostic model (GDM; von Davier, 2008), the generalized DINA model (GDINA; de la Torre, 2011), and the log-linear cognitive diagnostic model (LCDM; Henson, Templin, & Willse, 2009). These general models can be shown to include well-known models that are often used in cognitive diagnosis, such as the *deterministic inputs noisy and* gate model (DINA; Junker & Sijtsma, 2001), the *deterministic inputs noise or* gate model (DINO; Templin & Henson, 2006), the additive cognitive diagnosis model (ACDM; de la Torre, 2011), the linear logistic model (LLM; Maris, 1999), and the reduced reparameterized unified model (rRUM; Hartz, 2002).

This chapter starts with a simple reparameterized version of the DINA model and builds up to other models; all of the models are shown to be extensions or variations of the basic model. Working up to more general models from a simple form helps to illustrate basic aspects of the models and associated concepts, such as the meaning of model parameters, issues of estimation, monotonicity, duality, and the relation of the models to each other and more general forms. In

addition, reparameterizing CDMs as latent class models allows one to use standard software for latent class analysis (LCA), which offers a connection to latent class analysis and also allows one to take advantage of recent advances in LCA.

The Reparameterized DINA Model

A well-known CDM, the DINA model (Haertel, 1989; Junker & Sijtsma, 2001; Macready & Dayton, 1977), provides a useful starting point. Let Y_{ij} be a binary variable that indicates whether the response of the *i*th examinee to the *j*th item is correct or incorrect (1 or 0) and let $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_K)'$ denote the vector of K attributes that are needed to solve the items. The Q-matrix consists of elements q_{jk} that specify which of the K attributes are needed to solve the *j*th item. Thus, the Q-matrix elements consist of zeroes and ones, with a value of zero indicating that the *k*th attribute is *not* needed, and a value of one indicating that the attribute is needed. For the DINA model, the probability that an examinee gets an item correct is

$$p\left(Y_{ij} = 1 \middle| \boldsymbol{\alpha}\right) = \left(1 - s_j\right)^{\eta_{ij}} g_j^{1-\eta_{ij}},$$

with

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

Note that η_{ij} is simply a binary indicator with a value of one indicating that an examinee has all of the required attributes and a value of zero indicating that an examinee is lacking one or more of the required attributes. Thus, if an examinee has all of the required attributes, then $\eta_{ij} = 1$ and

$$p\left(Y_{ij} = 1 \middle| \boldsymbol{\alpha}\right) = \left(1 - s_j\right),$$

where the parameter s_j is the *slip* rate for examinee j. If an examinee is missing one or more of the required attributes, then $\eta_{ij} = 0$ and

$$p(Y_{ij} = 1|\boldsymbol{\alpha}) = g_{j,}$$

where the parameter g_j is the *guess* rate.

Although 'slipping' and 'guessing' were suggested as useful mnemonics by Junker and Sijtsma (2001), the relation of the concepts to basic ideas of *signal detection theory* (SDT) is also informative (Green & Swets, 1966; Macmillan & Creelman, 2005; Wickens, 2002). In SDT, $(1-s_j)$ is the *hit* rate – the examinee has the requisite attributes and gets the item correct, whereas s_j is the *miss* rate – the examinee has the requisite attributes, but gets the item incorrect. If an examinee does not have the requisite attributes yet gets the item correct, then that's a *false alarm*; note that guessing is an interpretation of false alarms.

It has previously been shown that a simple re-parameterization of the DINA model can be obtained by re-writing the false alarm rate, g_j, as

$$g_j = \frac{exp(f_j)}{1 + exp(f_j)},$$

where *exp* is the exponential function and the parameter f_j is the transformed false-alarm rate. Similarly, one minus the slip rate, the hit rate, can be re-written as

$$1 - s_j = \frac{exp(f_j + d_j)}{1 + exp(f_j + d_j)},$$

where d_j is a discrimination parameter. The DINA model can then be re-written as

$$\text{logit}\, p(Y_{ij} = 1|\boldsymbol{\alpha}) = f_j + d_j \prod_{k=1}^{K} \alpha_{ik}^{q_{jk}}, \quad (1)$$

which has been referred to as the reparameterized DINA model (rDINA; DeCarlo, 2011). The model is a special case of the general diagnostic model of von Davier (2008), with a change in notation to emphasize the signal detection aspects of the model. In particular, f_j is a transformed

false alarm rate whereas d_j is a difference of transformed hit and transformed false alarm rates that indicates the level of *discrimination* between having and not having the attributes (i.e., $\eta_{ij} = 1$ versus $\eta_{ij} = 0$). Note that the discrimination parameter in SDT is a function of both hits and false alarms, in contrast to looking at slips and guesses separately, in that it follows from the theory that one needs to examine the hit rate relative to the false alarm rate (to get the distance measure *d*); other variations of the discrimination parameter have also been considered (e.g., differences, ratios, etc.). From a statistical point of view, the intercept f_j is the *log odds* of a correct response given $\eta_{ij} = 0$, and the slope d_j is the *log odds ratio* of a correct response given $\eta_{ij} = 1$ versus $\eta_{ij} = 0$. The rDINA model is equivalent to the DINA model and so estimates of f_j and d_j can be transformed to get estimates of g_j and s_j.

Monotonicity

Note that, when fitting the model, a constraint must be used so that *monotonicity* is satisfied. In terms of the DINA model, monotonicity holds if

$$0 < g_j < 1 - s_j < 1.$$

Without the monotonicity constraint, examinees who have a required attribute could have a lower probability of getting an item correct than if they did not have the attribute (although this could be appropriate in certain situations). It is informative to interpret the monotonicity constraint in terms of signal detection theory, in that it simply implies that the hit rate ($1-s_j$) must be greater than the false alarm rate (g_j), in which case the receiver operating characteristic curve (ROC; see Macmillan & Creelman, 2005), which is a plot of hits versus false alarms, will lie above the diagonal (the diagonal represents zero discrimination) and the discrimination parameter *d* will be greater than zero. In terms of the rDINA model, monotonicity holds if

$$0 < \frac{exp(f_j)}{1 + exp(f_j)} < \frac{\exp(f_j + d_j)}{1 + \exp(f_j + d_j)} < 1,$$

and so monotonicity holds if the discrimination parameter d_j is greater than zero (for finite f_j and d_j). This constraint can be implemented in many software packages; the Appendix, for example, provides Latent Gold programs that show how to implement monotonicity by using the (+) command, which constrains the parameter to non-negative values.

Estimation

A benefit of the rDINA model of Equation 1 is that it is simple to fit with standard software for latent class analysis, such as Latent Gold (LG; Vermunt & Magidson, 2016) or the freely available LEM (Vermunt, 1997), given that Equation 1 is simply a logistic latent class model with latent dichotomous interaction terms. The program provided in the Appendix shows that it is straightforward to fit the rDINA model in LG by specifying interaction terms for the latent dichotomous attributes; the Q-matrix being used is also made transparent in the program.

Note that the rDINA model discussed so far is the 'examinee-level' part of the model, whereas the complete model also includes a higher-level model for the attributes, that is, an 'attribute-level' model. This is a model for the probabilities of the various skill combinations, that is, $p(\alpha_1, \alpha_2, \ldots, \alpha_K)$. Latent Gold and LEM programs to fit the rDINA model with either an independence structure or a higher-order structure (with a continuous latent variable) for the attribute-level model have previously been provided (DeCarlo, 2011). Here it is shown how to specify an *unstructured* attribute-level model (not previously shown) in Latent Gold or other latent class software. Note that when CDM researchers refer to fitting 'the DINA model', they usually mean the DINA model with an unstructured attribute-level model.

To implement the unstructured attribute model one merely needs to use a *saturated model* as the higher-level model. A simple way to do this in Latent Gold is to specify a saturated *association model* (Agresti, 2002) for the attribute-level model, with one parameter set to zero, so that the model includes $2^k - 1$ parameters for the 2^k attribute patterns (and so it is saturated). An example of this approach is given by the first rDINA LG program provided in the Appendix. In this case, the first cell is restricted to be zero, using the command r[1,1] = 0, for identification. Estimates of the class sizes for each attribute pattern, and their standard errors, are given in the section of LG output that is labeled as "Profile", along with estimates of the marginal class sizes, that is, estimates of $p(\alpha_k)$.

Another option is to specify a sequence of path models, which gives the same results as when an association model is used, given that both models are saturated. The LG program in the Appendix also illustrates this approach (in the comments, which are specified in LG by the symbol '//'); running the program shows that the results are the same as those obtained using the saturated association model. Depending on the software that one uses, one or the other of the approaches for the attribute-level model might be simpler to implement.

Philipp, Strobl, de la Torre, and Zeilis (2018) recently noted that there is a problem with respect to estimation of the standard errors in CDMs. In particular, "it is common to compute the standard errors only for the parameters that are used to specify the item response function while ignoring the parameters used to specify the joint distribution of the attributes." (p. 2). They note that this common approach can lead to underestimation of the standard errors in both parts of the model (also see von Davier, 2014). Note that, with the LCA approach, the standard errors are estimated for both the examinee-level parameters (i.e., the item response function) and the

attribute-level parameters. Latent Gold also offers a robust (sandwich) estimator of the SEs, as well as others, details of which are given in the technical manual (Vermunt & Magidson, 2016).

Using software for LCA also makes available a wide array of tools and output for CDMs. For example, with Latent Gold, one obtains estimates of the parameters for both the examinee-level and attribute-level models, along with their standard errors, absolute and relative fit statistics (e.g., Chi-square goodness of fit, information criteria such BIC and AIC, etc.), bivariate residuals, various classification statistics and tables, different types of plots, output files with posterior classifications for each examinee, as well as details about the iterations and any convergence or identification problems. In addition, different algorithms are available, such as versions of the Newton-Raphson and Expectation-Maximization algorithms (LG starts with the EM and moves to NR when in the vicinity of the solution), as well as the option to use posterior mode estimation with different Bayes constants, which controls the degree of smoothing, along with many other options.

Boundary Problems

A well-known problem that often arises in latent class analysis is known as a *boundary problem* (Clogg & Eliason, 1987; DeCarlo, 2011; Maris, 1999). Boundary problems occur when parameter estimates and SEs are large or indeterminate, or probability estimates are close to zero or one, which is also related to identification problems (such as weak identification). This problem has been somewhat neglected in CDMs (some exceptions are noted below), partly because the SEs are sometimes not reported, and partly because, in the original probability version of the model, finding slipping or guessing parameters close to zero, for example, is not in and of itself cause for alarm (not having slipping or guessing can be viewed as a good thing),

whereas it could actually be reflecting an overlooked boundary problem. The reparameterized model is useful in this regards because the model transforms the zero-one probability scale (for g_j and s_j) to a minus infinity to positive infinity scale for f_j and greater than zero to infinity scale for d_j (because of the monotonicity constraint), and so boundary problems or weak identifiability will tend to be more obvious, in that they will appear as overly large or infinite parameter estimates and/or estimated standard errors. For example, Table 4 of de la Torre (2009) shows parameter estimates for a fit of the DINA model to a subset of 15 items (out of 20) of the well-known fraction subtraction data. The estimate of g_1 for the first item is shown as 0.00 with a standard error of 0.05 (which is the largest standard error in the table) and the estimate of s_1 is 0.28 (the highest in the table) with a standard error of 0.013. A fit of the rDINA model with LG to this data (with unstructured attributes) gives an estimate of f_1 of about −24 (whereas the lowest f_j for all the other items is around −4.5) with an SE of 0.12, and an estimate of d_1 of 25 with an SE of 1000 (i.e., infinite), and so there are clearly identification problems for this item.

In addition to boundary problems appearing in the examinee-level part of the model, they can also appear in the attribute-level part of the model, particularly when the unstructured attribute model is used. For the fraction subtraction example with 15 items just discussed, LG shows that there are 22 boundary problems in the unstructured attribute model (with 22 SEs appearing as 1000). It is interesting to note that if one fits the original 20 item version of the fraction subtraction data with an unstructured attribute model, as has been widely used in many studies, then Latent Gold reports that there are 198 non-identified parameters (note that the unstructured model has $2^8 - 1 = 255$ parameters; problems also appear for a higher-order model). Estimates of the standard errors for the attribute-level model parameters are also all excessively

large (>20), again reflecting identification problems. Thus, even though the unstructured attributes model has been widely used, identification problems with the attribute-level model, and the possible effects of this on estimation for the examinee-level model, have generally not been considered.

Boundary problems for fits of the DINA model, additive cognitive diagnosis model (ACDM), and GDINA model to real-world data given in the R package *pks* (Heller & Wickelmaier, 2013) were recently noted by Philipp, Strobl, de la Torre, & Zeileis (2018); they noted boundary problems in both the examinee-level and attribute-level parts of the model (p. 20). von Davier (2014) discussed identification problems for the well-known Examination for the Certificate of Proficiency in English (ECPE) data; he noted that, even with constraints, weak identifiability still appeared for the LCDM; also see Templin and Bradshaw (2014).

Posterior Mode Estimation and Bayesian Estimation

A number of authors have discussed the use of *posterior mode estimation* (PME) to deal with boundary problems (e.g., DeCarlo, 2011; DeCarlo, Kim, & Johnson, 2011; Maris, 1999; Vermunt & Magidson, 2016). PME is less computationally intense than a full Bayesian analysis in that it does not require that the full posterior distribution be generated, but rather only the mode needs to be found. As a result, PME has a computational speed advantage over a full Bayesian analysis, which is useful when performing computer simulations. In addition, standard algorithms that implement maximum likelihood estimation can often easily be modified to implement PME. The approach basically smooths infinite or large parameter estimates and/or estimates of the standard errors. The use of PME in CDMs is a topic for future research; this option is currently available in Latent Gold. The use of PME for a simple latent class signal

detection model (DeCarlo, 2002; 2005), which is the same as a CDM with a single latent dichotomous attribute (the latent signal), has been examined in simulations presented in DeCarlo (2008; 2010); the use of PME for a hierarchical rater signal detection model with ordinal latent classes has been examined in simulations presented in Kim (2009), and the use of PME with real-world data was examined in DeCarlo, Kim, and Johnson (2011).

Another option is to use a full Bayesian analysis to fit CDMs (e.g., Culpepper, 2015; de la Torre & Douglas, 2004; DeCarlo, 2012; Henson, Templin, & Willse, 2009). For example, an OpenBugs program (Spiegelhalter, Thomas, Best, & Lunn, 2014) to fit the rDINA model with Bayesian estimation was given in DeCarlo (2012; with a monotonicity constraint implemented by restricting d_j to be greater than zero). The approach also generalized the model by allowing for uncertainty about some elements of the Q-matrix, and simulations suggested adequate recovery of those elements using posterior distributions. The Bayesian approach allows for interesting extensions; for example one can extend the model with a few uncertain Q-matrix elements to allow all of the Q-matrix elements to be uncertain; this approach was examined by DeCarlo and Kinghorn (2016; with monotonicity restrictions) and by Culpepper (2015; with completeness restrictions).

Classification

Classification in latent class analysis is typically done using the modal posterior probabilities (e.g., Clogg, 1995; Dayton, 1998). For example, one approach, *maximum a priori* (MAP) classification, is to simply classify each examinee into the attribute set with the largest posterior probability. Another option is to use marginal probabilities to classify examinees for each skill separately, as in *expected a posteriori* (EAP) classification. In maximum likelihood

estimation (MLE), classification is accomplished by finding attribute patterns that maximize the posterior. The various approaches were compared in the context of CDMs by Huebner and Wang (2011).

Identifiability

Identifiability is concerned with whether one can obtain unique estimates of the model parameters. Xu and Zhang (2016) gave necessary and sufficient conditions for identifiability of the model parameters for the DINA model (also see Chen, Liu, Xu, & Ying, 2015). They also noted that their results could be extended to the DINO model, because of the duality of the models (see below). The issue of boundary problems discussed above is also related to the issue of identifiability, with large standard errors often indicating 'weak identification', in which case the data (or model) are not informative about the parameters.

The effect of identifiability on classification has also been discussed. Chiu, Douglas, and Li (2009) noted that *completeness* of a Q-matrix is generally needed for identification of all possible attribute patterns. For the DINA and DINO models, for example, completeness is satisfied if, for each attribute, there is an item that measures that attribute alone. Köhn and Chiu (2017) noted that the conditions for completeness depend on the model and examined completeness for several CDMs.

For the fraction subtraction data, DeCarlo (2011) noted that, because of incompleteness of the Q-matrix, some of the posterior classifications from the DINA model depend solely on the priors, and so the data offers no additional information over the priors. Zhang, DeCarlo, and Ying (2013) noted that, although certain attribute patterns are in the same equivalence class for the fraction-subtraction data and so are not identifiable, individual attributes within an

equivalence class may still be identifiable. They proposed a measure of the *marginal identifiability rate*, which is the proportion of the population for which each attribute is marginally identifiable, and suggested that it can be viewed as a measure of test (and model) quality. Zhang et al. also proposed classification algorithms that took into account the effects of marginal identifiability.

Reparameterized DINO

Here it is shown that a reparameterized version of the DINO model (Templin & Henson, 2006) clarifies issues about the relations between the DINO and DINA models (duality) and their parameters. The DINO model is similar to the DINA model with the exception that, instead of requiring that all of the skills be present in order to solve an item, only one or more of the skills need to be present. The DINO condensation rule is usually referred to as being *disjunctive*, whereas the DINA condensation rule is *conjunctive* (Rupp, Templin, & Henson, 2010). The model is

$$p\left(Y_{ij}=1\middle|\boldsymbol{\alpha}\right)=\left(1-s_j'\right)^{\omega_{ij}}g_j'^{1-\omega_{ij}},$$

where $\omega_{ij}=1-\prod_{k=1}^{K}(1-\alpha_{ik})^{q_{jk}}$ equals one if any required skill is present, and zero only if all of the required skills are absent.

It is important to note that slips and guesses, s_j' and g_j', are defined differently in the DINO model as compared to the DINA model. In the DINO model, the hit rate $1-s_j'$ is the probability of a correct response given that an examinee has *at least one* of the attributes, whereas in the DINA model, the hit rate $1-s_j$ is the probability of a correct response given that an examinee has *all* of the attributes. Similarly, the false alarm rate g_j' in the DINO model is the

probability of a correct response given that an examinee has *none* of the attributes, whereas the false alarm rate g_j in the DINA model is the probability of a correct response given that an examinee is *missing at least one* attribute.

The DINO model can be reparameterized using the same approach used above for the DINA model, which gives,

$$\text{logit}\, p(Y_{ij} = 1|\boldsymbol{\alpha}) = f_j' + d_j'[1 - \prod_{k=1}^{K}(1 - \alpha_{ik})^{q_{jk}}], \tag{2}$$

which will be referred to as the *rDINO model*; the model was also recently derived in terms of the GDM by Köhn and Chiu (2016). Once again, monotonicity is satisfied if d_j' is greater than zero.

As for the rDINA model, the model in this form is straightforward to fit using software for latent class analysis. Suppose, for example, that Item 1 requires the first three skills. The model for the first item is then

$$\text{logit}\, p(Y_{i1} = 1|\boldsymbol{\alpha}) = f_1' + d_1'(\alpha_{i1} + \alpha_{i2} + \alpha_{i3} - \alpha_{i1}\alpha_{i2} - \alpha_{i1}\alpha_{i3} - \alpha_{i2}\alpha_{i3} + \alpha_{i1}\alpha_{i2}\alpha_{i3}),$$

which is a logistic model with all main effects and higher order interaction terms. Further, the coefficients (d_j') are restricted to be equal across all terms and have alternating signs across the two and three way interactions, as was also noted by de la Torre (2011) for the G-DINA model. A sample rDINO program that shows how to implement the parameter constraints of Equation 2 in Latent Gold is given in the Appendix.

DINO/DINA Duality

The rDINO model of Equation 2 can be re-written as

$$\text{logit}\, p\left(Y_{ij}=1\middle|\boldsymbol{\alpha}\right)=\left(f_j'+\ d_j'\right)-d_j'\prod_{k=1}^{K}(1-\alpha_{ik})^{q_{jk}}. \tag{3}$$

Note that, if one replaces $1-\alpha_{ik}$ in the above with reverse coded $\alpha_{ik}^{*}=1-\alpha_{ik}$, then Equation 3 has the same form as the rDINA model of Equation 1, with a redefined intercept and a negative slope. This was also shown by Köhn and Chiu (2016, see Section 3.3) by replacing $1-\alpha_{ik}$ with α_{ik}^{*} and by reverse coding the data, so that $Y_{ij}^{*}=1-Y_{ij}$, which merely reverses the signs of Equation 3, given that logit p = −logit (1−p), and so

$$\text{logit}\, p\left(Y_{ij}^{*}=1\middle|\boldsymbol{\alpha}^{*}\right)=\left(-f_j'-d_j'\right)+d_j'\prod_{k=1}^{K}(\alpha_{ik}^{*})^{q_{jk}}. \tag{3a}$$

Equation 3a is clearly related in form to the rDINA model of Equation 1, with a redefined intercept (and different parameters). In this respect, there is a *duality* between the rDINA and rDINO models (and so between the DINA and DINO models as well; Chen, Liu, Xu, & Ying, 2015; Köhn & Chiu, 2016; Liu, Xu, & Ying, 2011).

An important consequence of the duality between the DINA and DINO models (and rDINA and rDINO) is that theoretical results developed for one model can be applied to the other model (Liu, Xu, & Ying, 2011). For example, Köhn and Chiu (2016) used duality to determine the conditions necessary for completeness of the Q-matrix for both the DINA and DINO models.

Köhn and Chiu (2016) noted another interesting consequence of duality, which is that it implies that the DINO model can be fit by using a DINA program. The simple reparameterized versions of the models presented here are helpful in that they suggest more than one way that this can be done. To start, note that the rDINO model can be fit directly as given in Equation 2, as shown by the rDINO program given in the Appendix. In this case, the program is a little more

involved than the rDINA program because of the parameter restrictions implied by the rDINO model (i.e., equal d_j' and alternating signs).

Equation 3a suggests another option, also suggested by Köhn and Chiu (2016), which is to use a DINA program to fit the DINO model. This can be done if one can fit the DINA model with reverse coded α_{ik}^* in lieu of α_{ik}, which is the key to the difference between the models. Köhn and Chiu (2016) accomplished this by reverse coding the data and maintaining the monotonicity constraint. Note that, for symmetric links such as the logit, reverse coding the data simply reverses the parameter signs. However, because the monotonicity constraint is also maintained, the model cannot account for the reversed *Y* with a negative sign for the discrimination parameter, but rather with a reversed α (i.e., α^*). Thus, reverse coding the data and maintaining monotonicity is simply a way to induce the use α^* in the model in lieu of α.

A practical advantage of the above approach is that one can then fit the rDINO model using the simpler rDINA program given in the Appendix by reverse coding the data and keeping the monotonicity constraint (i.e., positive values of d_j'). Note that if the monotonicity constraint is removed, then fitting the reverse coded data will simply give results for the rDINA model with reversed parameter signs, and not the rDINO model, as the reader can verify.

Another interesting option is suggested by Equation 3 – fit the rDINO model with an rDINA program, but impose a *negative monotonicity* constraint, that is, restrict d_j' in Equation 3 to be less than zero. An interesting aspect of this approach is that it again allows one to fit the rDINO model with an rDINA program, but there is *no need to reverse code the data*. That is, one can fit the original data, again using an rDINA program, by simply replacing (+) in the LG program given in the Appendix with (−), to give a negative monotonicity constraint. Specifying a

negative monotonicity constraint will tend to lead to α_{ik}^{*} being used in the model in lieu of α_{ik}, in which case the rDINO model of Equation 3 is fit (and not the rDINA model).

It is apparent that the simplest approach in Latent Gold is the third one – simply use the original data and impose a negative monotonicity constraint in an rDINA program to get the rDINO model of Equation 3. It would be interesting in future research to see if there are any differences across the three approaches to fitting the rDINO model, in terms of estimation advantages or disadvantages.

It should be noted that using a negative monotonicity constraint or reverse coding the data and using a positive monotonicity constraint may not be sufficient to lead to α_{ik}^{*} being used in the model in lieu of α_{ik} (this is also related to 'label switching' issues discussed in latent class analysis, although it is not simply label switching in this case in that the likelihood differs, but this is beyond the scope of the current chapter), this needs to be considered more closely in future research. For example, if Equation 3a is used, then one must check that all of the d_j' estimates have positive signs, so that monotonicity holds. Another useful check is to compare the latent class size estimates, that is, the estimates of $p(\alpha_k)$ for the rDINO model, to those obtained for a fit of the rDINA model – the class size estimates will usually differ (beyond a simple reversal in categories). If they are the same, apart from a category reversal, then it is likely that the rDINA model was fit, not the rDINO model.

Equations 2, 3, and 3a are useful in that they also show exactly what estimates are obtained with each of the three approaches. That is, if one fits the rDINO model as specified in Equation 2, then the intercept and slope of the logistic model will give direct estimates of f_j' and d_j' respectively; the estimates of $p(\alpha_k)$, the latent class sizes, will be given in LG as Class Size 2

(i.e., the class size for having the attribute). If the data are reverse coded and Köhn and Chiu's (2016) approach is used, then the intercept gives an estimate of $-f_j' - d_j'$ and the slope gives an estimate of d_j' (see Equation 3a). Thus, one has to add the estimate of d_j' to the intercept and then reverse the sign to get an estimate of f_j'. The latent class sizes will also be reversed, so that Class Size 1 in Latent Gold will give the class size for having the attribute. Finally, if one fits rDINO with Equation 3, then the intercept will give an estimate of $f_j' + d_j'$ and the slope will give an estimate of $-d_j'$. Thus, one simply adds the slope estimate to the intercept estimate to get an estimate of f_j' and reverses the sign of the slope to get d_j'. The latent class sizes will again be reversed, and so Class Size 1 again gives the desired estimate. It is instructive to use the three approaches and compare the results; simulated DINO data is available at the author's website, along with LG programs, so that the three approaches can be implemented and compared.

The equations also help to clarify similarities and differences between the DINA and DINO models. First, the relation between Equations 1 and 3a does not imply that the rDINA model and the rDINO model are *equivalent*; they can and will give different log likelihoods when fit to the same data (and different log posteriors, in a Bayesian approach), given that they impose a different structure on the data (for items that involve two or more attributes). Note that Köhn and Chiu (2016; Section 3.3) showed that the expected item response function for DINO with Y^* and α^* is equivalent to that for DINA with Y and α, as also shown by Equation 3a. This does not mean however that the DINO model is equivalent to the DINA model. Rather, it should be recognized that if one uses a DINA program to fit the model with α_{ik}^*, regardless of whether or not Y is reverse coded, then the DINO model is being fit, as shown by Equations 3 and 3a, and

not the DINA model. That is, using notation suggested by a reviewer, Equations 3 and 3a, respectively, show that

$$\text{rDINO}(Y,\alpha) = \text{rDINA}(Y^*,\alpha^*) = \text{rDINA}(Y,\alpha^*),$$

with the parameters related as shown above. This is what allows one to use a DINA program to fit the DINO model – use α^* in the DINA model in lieu of α, irrespective of whether Y or Y^* is used, and the DINO model is being fit. However, this does not mean that the DINO model is equivalent to the DINA model, that is,

$$\text{rDINO}(Y,\alpha) \neq \text{rDINA}(Y,\alpha)$$

as can be seen by comparing Equations 1 and 2 (for items that load on two or more attributes). Thus, rDINO and rDINA (and DINO and DINA) are structurally different models, the above just shows that using α^* in place of α in a DINA program, and maintaining monotonicity, results in the DINO model being fit. A useful exercise is to use the rDINA LG program given in the Appendix with reverse coded data, but remove the monotonicity constraint; the result is that the rDINA model is fit, not the rDINO model, and the parameter signs are simply reversed from those obtained for a fit of the rDINA model to the original data. On the other hand, if the monotonicity constraint is enforced for the reverse coded data, then the rDINO model will be fit, not the rDINA model, and the parameter estimates and likelihood will differ.

Another point of clarification is that the models also differ with respect to the parameter estimates, that is, the DINO parameter estimates are not transformations of the DINA parameter estimates (contrary to some claims). The models are structurally different and involve different parameters. Table 1 shows a simple example with two attributes. The third column shows the condensation rule η_j for the DINA model and the fourth column shows the condensation rule ω_j

for the DINO model. The fifth column shows the DINA parameters for a correct response. The column shows that, for the DINA model, false alarms, g_j, occur for the first three rows, whereas the last row represents hits, $1 - s_j$. The sixth column shows the DINO parameters; in this case, only the first row represents false alarms, g'_j, whereas the other rows all correspond to hits, $1 - s'_j$.

A comparison of the second and third rows of Table 1, where only one of the skills is present, helps to highlight differences between the models. For DINA, the second and third row parameters are the same as the first row, that is, they are all false alarms g_j. In contrast, for DINO, the second and third row parameters are the same as the fourth row, the hit rate $1 - s'_j$. Thus, different parameter estimates will generally be obtained for fits of the two models (for items that involve two or more attributes) and one set of parameters are not simply transformations of the other set, that is, $g'_j \neq g_j$ and $1 - s'_j \neq 1 - s_j$.

To summarize, if one uses a DINA program and induces the use of α^* in place of α, either by reverse coding the data or enforcing negative monotonicity, then one is fitting the DINO model and not the DINA model.

A General Reparameterized Model

The rDINA and rDINO models show a clear and simple pattern. Consider the rDINA model for an item that requires three skills,

$$\text{logit}\, p(Y_{ij} = 1|\boldsymbol{\alpha}) = f_j \;\; + \;\; d_j \;\; \alpha_{i1}\alpha_{i2}\alpha_{i3},$$

whereas the corresponding rDINO model is

$$\text{logit}\, p(Y_{ij} = 1|\boldsymbol{\alpha}) = f'_j + \; d'_j(\alpha_{i1} + \alpha_{i2} + \alpha_{i3} - \alpha_{i1}\alpha_{i2} - \alpha_{i1}\alpha_{i3} \; - \alpha_{i2}\alpha_{i3} + \alpha_{i1}\alpha_{i2}\alpha_{i3}),$$

and similarly for the other items. It is clear that the rDINA model only includes the highest-order interaction term whereas the rDINO model also includes main effects and lower order interaction terms. Further, rDINO restricts the coefficients of all the terms to be equal and the signs to be alternating.

It is immediately obvious that both the rDINA and rDINO models, as well as others, are simply special cases of a more general model that includes all main effects and higher order interactions. For example, for the above item with three attributes, a general reparameterized model is

$$g[\, p(Y_{ij} = 1|\boldsymbol{\alpha})] = f_j \;\; + \; d_{j1}\alpha_{i1} + d_{j2}\alpha_{i2} + d_{j3}\alpha_{i3} + d_{j,12}\alpha_{i1}\alpha_{i2} + d_{j,13}\alpha_{i1}\alpha_{i3}$$
$$+ d_{j,23}\alpha_{i2}\alpha_{i3} + \; d_{j,123}\alpha_{i1}\alpha_{i2}\alpha_{i3},$$

where *g* is a link function, such as the logit, probit, or complementary log-log link (all available in LG, along with others). Note that the discrimination parameters are now attribute-specific, that is, d_j in the rDINA and rDINO models is replaced with the attribute-specific d_{jk}, for the first-order terms. For the interaction terms, the discrimination parameter subscripts indicate which attributes are involved; for example, for a three-way interaction term the discrimination parameter is $d_{j,kk'k''}$, where the *j* indicates the item, as before, and *kk'k''* indicates the three attributes involved in the interaction (giving $d_{j,123}$ in the example above). Applying the model to every item, according to the Q-matrix structure, gives a *general reparameterized model* (GRM), which is simple to fit with software such as Latent Gold. The notation makes clear that the added parameters are discrimination parameters that show how (transformed) hits increase compared to (transformed) false alarms.

The GRM with logit link gives a saturated version of the GDM of von Davier (2008) and the LCDM of Henson et al. (2009); with an identity link it is a saturated version of the GDINA model of de la Torre (2011); also see von Davier (2013; 2014). With appropriate parameter restrictions, the GRM includes the rDINA and the rDINO models discussed above. Another simplification is to only include main effects, which gives the linear logistic model (LLM) of Maris (1999); using an identity link gives the additive cognitive diagnosis model (ACDM) of de la Torre (2011). With constraints placed on the coefficients of the higher-order interaction terms, one can obtain the reduced reparameterized unified model (rRUM; Hartz, 2002), given that Chiu and Köhn (2016) recently showed that rRUM is a (non-saturated) logistic model with parameter constraints. The parameter constraints for rRUM, however, are somewhat complex (and apparently cannot be implemented in LG at this time).

Although an unrestricted saturated model is quite simple to fit in software such as Latent Gold, one has to pay close attention to parameter restrictions that might need to be imposed. For example, if the monotonicity constraint is to be satisfied, then the coefficients d_{jk} of single attribute terms should be restricted to be greater than zero. Note that if one fits the model (as GDINA) using 'rule=GDINA2' in the CDM package in R (George et al., 2016; also see Chapter 26 in this volume), for example, then monotonicity is not enforced, as can be verified using the ECPE data – the first item gives a negative d_{jk} for the first attribute, and so monotonicity does not hold for the first item. The GDINA package in R (Ma & de la Torre, 2017; also see Chapter 29 in this volume) allows one to place monotonicity constraints on the parameters. As before, in Latent Gold, non-negativity for d_{jk} is implemented by using the monotonicity constraint (+),

whereas negative monotonicity is implemented by using (−), as shown by the programs given in the Appendix.

Another consideration has to do with whether or not restrictions should be placed on the coefficients of the interaction terms. For example, if they are left unrestricted, then it is possible that the probability of a correct response can be lower when an examinee has two required attributes as compared to only one of the attributes. If this is viewed as being theoretically undesirable (although in some cases one could possibly argue for an interference effect) then restrictions should be placed on the interaction parameters. For example, for an item that requires two attributes, the restriction $d_{j,kk'} > -1*\min(d_{jk}, d_{jk'})$ will ensure that the probability of a correct response when an examinee has both attributes will not be lower than when they only have one of the attributes (namely the one that gives the highest probability of a correct response). The restriction can also be written as $d_{j,kk'} > -d_{jk}$ and $d_{j,kk'} > -d_{jk'}$, which is the form used by Templin and Hoffman (2013) for an implementation of the model in Mplus.

With respect to fitting the model in Latent Gold, although there is a way to implement order restrictions (as shown below), the multiple restrictions required above cannot currently be implemented simultaneously (to my knowledge). For example, for items that require two attributes, there are four required restrictions: $d_{j1} > 0$, $d_{j2} > 0$, $d_{j,12} > -d_{j1}$, and $d_{j,12} > -d_{j2}$; three of the four restrictions can be implemented in LG, but not all four. A simple work-around is to use a two-step approach: in the first step, fit the GRM with monotonicity constraints on the first order terms but with no restrictions on the interaction terms and examine the parameter estimates; in the second step constrain interaction parameters where necessary (i.e., they violate the above order condition) using information gained in the first step. That is, if one finds that the estimate

of d_{j2} is greater than the estimate of d_{j1}, then only the restriction $d_{j,12} > -d_{j2}$ is needed, in addition to the two monotonicity restrictions.

To illustrate the suggested approach, consider the well-known ECPE data, where the saturated LCDM with appropriate restrictions has previously been fit (using Mplus; Templin & Bradshaw, 2014). The first step is to fit a saturated model in LG with a monotonicity constraint on the first-order terms, but unrestricted interaction terms; the program in the Appendix shows that this is very simple to do in LG. The results then allow one to see 1) if and where the above restriction on the interaction term is violated and 2) if it is, which discrimination parameter is smallest, which gives one information about $\min(d_{jk}, d_{jk'})$, and so only three of the four restrictions noted above are needed. For example, for the ECPE data, it was apparent from a fit of the GRM that there were problems with Item 7; the coefficient for the second attribute for this item was also clearly smaller than for the first attribute. Thus, for the second step, the saturated model was fit adding the constraint $d_{j,12} > -d_{j2}$ to Item 7, and the results reproduce those shown in Table 1 and Figure 1 of Templin and Bradshaw (2014) to two decimal places; the log-likelihood was also identical to that obtained with Mplus. This is not to say that the two-step approach will work in general, but the point here is to show possible ways to implement more complex restrictions in current software.

The program given in the Appendix shows how to implement the order constraint for Item 7 in LG; a ‘trick’ is used of adding a positive constant to the coefficient that must be greater than zero by introducing an additional interaction term that is restricted to be greater than zero, which implements the order constraint.

It should be noted that there is also a computational speed advantage of Latent Gold for CDMs, which is useful when conducting simulations. For example, on a machine with 8GB RAM, 2.30 GHz Intel Core processor, and 64-bit OS, an Mplus program to fit the saturated LCDM to the ECPE data (retrieved from https://jonathantemplin.com/dcm-workshop-spring-2012-ncme/) took 47 minutes to converge, whereas the model fit with LG (using the program provided in the Appendix) took less than 3 seconds (packages in R also have shorter run times).

Several researchers have suggested starting with a saturated model and attempting to determine which sub-model might be more appropriate (e.g., Rupp, Templin, & Henson, 2010). Recent studies have examined this approach using information criteria (Chen, de la Torre & Zhang, 2013) and the Wald test (de la Torre & Lee, 2013) for the DINA, DINO, and ACDM models. Given the ability to fit the saturated model and the various sub-models in Latent Gold, it is straightforward to implement these types of model comparisons.

Discussion

Reparameterized models are useful both for illustrating and understanding basic aspects of CDMs, as well as providing a bridge to latent class models and accompanying software. The importance of recognizing the signal detection nature of the parameters is emphasized. Monotonicity, for example, is seen to be a simple restriction on the discrimination parameter (i.e., that it is greater than zero) which ensures that the corresponding ROC curves lie above the diagonal line (the diagonal represents zero discrimination). The models also help make concepts such as duality more transparent, and are useful for showing how different models are related. For example, the models show that duality leads to a simple way to fit the rDINO model with a program for the rDINA model by using a negative monotonicity constraint. It also clarifies that

DINA and DINO are structurally different models with different parameters. All the options for estimation, classification, and other output and tools available in latent class software become immediately available for CDMs. One can also go beyond the GRM, in that one can consider models with nominal or ordinal indicators with more than two categories, models with continuous indicators, models with nominal, ordinal, or continuous latent variables, and models with other link functions besides the logit, all in a very straightforward manner, and all available in current software.

References

Agresti, A. A. (2002). *Categorical data analysis* (2nd ed.). NJ: John Wiley & Sons.

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.

Chen, Y., Liu, J., Xu, G., & Ying, Z. (2015). Statistical analysis of Q-matrix based diagnostic classification models. *Journal of the American Statistical Association*, *110*, 850-866.

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

Chiu, C-Y., & Köhn, H-F. (2016). The reduced rum as a logit model: parameterization and constraints. *Psychometrika*, *2*, 350-370.

Clogg, C. C. (1995). Latent class models. In G. Arminger, C. C. Clogg, & M. E. Sobel (Eds.), *Handbook of statistical modeling for the social and behavioral sciences* (pp. 211-359). New York: Plenum.

Clogg, C. C., & Eliason, S. R. (1987). Some common problems in log-linear analysis. *Sociological Methods and Research, 16*, 8-44.

Culpepper, S. A. (2015). Bayesian estimation of the DINA model with Gibbs sampling. *Journal of Educational and Behavioral Statistics, 5*, 454-476.

Dayton, C. M. (1998). *Latent class scaling analysis*. Thousand Oaks, CA: Sage.

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.

DeCarlo, L. T. (2002). A latent class extension of signal detection theory, with applications. *Multivariate Behavioral Research, 37*, 423-451.

DeCarlo, L. T. (2005). A model of rater behavior in essay grading based on signal detection theory. *Journal of Educational Measurement, 42*, 53-76.

DeCarlo, L. T. (2008). *Studies of a latent-class signal-detection model for constructed response scoring*. ETS Research Report No. RR-08-63. Princeton NJ: Educational Testing Service.

DeCarlo, L. T. (2010). *Studies of a latent class signal detection model for constructed response scoring II: Incomplete and hierarchical designs*. ETS Research Report No. RR-10-08. Princeton NJ: Educational Testing Service.

DeCarlo, L. T. (2011). On the analysis of fraction subtraction data: The DINA model, classification, latent class sizes, and the Q-matrix. *Applied Psychological Measurement, 35*, 8-26.

DeCarlo, L. T. (2012). Recognizing uncertainty in the Q-matrix via a Bayesian extension of the DINA model. *Applied Psychological Measurement, 36*, 447-468.

DeCarlo, L. T., Kim, Y. K., & Johnson, M. S. (2011). A hierarchical rater model for constructed responses with a signal detection rater model. *Journal of Educational Measurement, 48*, 333-356.

DeCarlo, L. T., & Kinghorn, B. R. C. (2016, April). *An exploratory approach to the Q-matrix via Bayesian estimation*. Paper presented at the 2016 meeting of the National Council on Measurement in Education, Washington, DC.

George, A. C., Robitzsch, A., Kiefer, T., Gross, J., & Uenlue, A. (2016). The R package CDM for cognitive diagnosis models. *Journal of Statistical Software, 74*, 1-24.

Green, D. M., & Swets, J. A. (1966). *Signal detection theory and psychophysics*. NY: John Wiley.

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*. Unpublished doctoral dissertation.

Heller, J., & Wickelmaier, F. (2013). Minimum discrepancy estimation in probabilistic knowledge structures. *Electronic Notes in Discrete Mathematics, 42*, 49-56.

Henson, R. A., Templin, J. L., & Willse, J. T. (2009). Defining a family of cognitive diagnosis models using log-linear models with latent variables. *Psychometrika, 74*, 191-210.

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

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.

Kim, Y. K. (2009). *Combining constructed response items and multiple choice items using a hierarchical rater model*. Doctoral dissertation, Teachers College, Columbia University.

Köhn, H-F., & Chiu, C-Y. (2016). A proof of the duality of the DINA model and the DINO model. *Journal of Classification, 33*, 171-184.

Köhn, H-F., & Chiu, C-Y. (2017). A procedure for assessing the completeness of the Q-matrices of cognitively diagnostic tests. *Psychometrika, 82*, 112-132.

Liu, J., Xu, G., & Ying, Z. (2011). "Learning Item-Attribute Relationship in Q-matrix Based Diagnostic Classification Models", Report: arXiv:1106.0721v1, New York City, NY: Columbia University, Department of Statistics. Retrieved from the website https://arxiv.org/abs/1106.0721v1.

Ma, W., & de la Torre, J. (2017). GDINA: The generalized DINA model framework. R package version 1.4.2. Retrieved from https://cran.r-project.org/web/packages/GDINA/index.html

Macmillan, N. A., & Creelman, C. D. (2005). *Detection theory: A user's guide* (2nd ed.). New York: Cambridge University Press.

Macready, G. B., & Dayton, C. M. (1977). The use of probabilistic models in the assessment of mastery. *Journal of Educational Statistics*, *2*, 99-120.

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

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*, 88-115.

Rupp, A. A., Templin, J., & Henson, R. A. (2010). *Diagnostic measurement: Theory, methods, and applications*. NY: Guilford Press.

Spiegelhalter, D., Thomas, A., Best, N., & Lunn, D. (2014). *OpenBUGS Version 3.2.3 User Manual*. Helsinki, Finland. Retrieved from http://www.openbugs.net/w/Manuals.

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.L., & Henson, R. A. (2006), "Measurement of Psychological Disorders Using Cognitive Diagnosis Models," *Psychological Methods, 11*, 287–305.

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

Vermunt, J. K. (1997). LEM: A general program for the analysis of categorical data. Tilburg University. Tilburg, The Netherlands.

Vermunt, J. K., & Magidson, J. (2016). *Technical Guide for Latent GOLD 5.1: Basic, Advanced, and Syntax*. Belmont, MA: Statistical Innovations Inc.

von Davier, M. (2008). A general diagnostic model applied to language testing data. *British Journal of Mathematical and Statistical Psychology*, *61*, 287-307.

von Davier, M. (2013). The DINA model as a constrained general diagnostic model – two variants of a model equivalency. *British Journal of Mathematical and Statistical Psychology*, *67*, 49-71.

von Davier, M. (2014). *The log-linear cognitive diagnostic model (LCDM) as a special case of the general diagnostic model (GDM)*. ETS Research Report No. RR-14-40. Princeton NJ: Educational Testing Service.

Wickens, T. D. (2002). *Elementary signal detection theory*. New York: Oxford University Press.

Xu, G., & Zhang, S. (2016). Identifiability of diagnostic classification models. *Psychometrika*, *81*, 625-649.

Zhang, S. S., DeCarlo, L. T., & Ying, Z. (2013). Non-identifiability, equivalence classes, and attribute-specific classification in Q-matrix based cognitive diagnosis models. Available at: http://arxiv.org/abs/1303.0426v1

Table 1: Relation between terms and parameters for the DINA and DINO models

α_1	α_2	η_j	ω_j	DINA	DINO
0	0	0	0	g_j	g_j'
1	0	0	1	g_j	$1 - s_j'$
0	1	0	1	g_j	$1 - s_j'$
1	1	1	1	$1-s_j$	$1 - s_j'$

Table notes: for DINA, $\eta_j = \prod_{k=1}^{K} \alpha_k^{q_{jk}}$; for DINO, $\omega_j = 1 - \prod_{k=1}^{K}(1 - \alpha_k)^{q_{jk}}$; the DINA and DINO columns show the parameters that correspond to a correct response.

Appendix

Complete Latent Gold program to fit the rDINA model, 15 items, 4 unstructured attributes.

```
model
options
  maxthreads=all;
  algorithm
    tolerance=1e-008 emtolerance=0.01 emiterations=250 nriterations=50 ;
  startvalues
    seed=0 sets=16 tolerance=1e-005 iterations=50;
  bayes
    categorical=0 variances=0 latent=0 poisson=0;
  montecarlo
    seed=0 sets=0 replicates=500 tolerance=1e-008;
  quadrature  nodes=10;
  missing  excludeall;
  output
    parameters=effect  betaopts=wl standarderrors profile probmeans=posterior
    bivariateresiduals estimatedvalues=model;
variables
  dependent y1 cumlogit, y2 cumlogit, y3 cumlogit, y4 cumlogit, y5 cumlogit,
  y6 cumlogit, y7 cumlogit, y8 cumlogit, y9 cumlogit, y10 cumlogit, y11 cumlogit,
  y12 cumlogit, y13 cumlogit, y14 cumlogit, y15 cumlogit;
  latent
   a1 ordinal 2 score=(0 1), a2 ordinal 2 score=(0 1),
   a3 ordinal 2 score=(0 1), a4 ordinal 2 score=(0 1);
equations
//next line uses a saturated association model for the attribute model//
 (r~full) a1 <-> a2 <-> a3 <-> a4;
//for sequential path approach, replace above with://
//a1 <- 1; a2 <- 1 + a1; a3 <- 1 + a1 + a2 + a1 a2//
//a4 <- 1 + a1 + a2 + a3 + a1 a2 + a1 a3 + a2 a3 + a1 a2 a3//
  y1 <- 1 + (+)a1;
  y2 <- 1 + (+)a2;
  y3 <- 1 + (+)a3;
  y4 <- 1 + (+)a4;
  y5 <- 1 + (+)a1 a2;
  y6 <- 1 + (+)a1 a3;
  y7 <- 1 + (+)a1 a4;
  y8 <- 1 + (+)a2 a3;
  y9 <- 1 + (+)a2 a4;
  y10 <- 1 + (+)a3 a4;
  y11 <- 1 + (+)a1 a2 a3;
```

```
   y12 <- 1 + (+)a1 a2 a4;
   y13 <- 1 + (+)a1 a3 a4;
   y14 <- 1 + (+)a2 a3 a4;
   y15 <- 1 + (+)a1 a2 a3 a4;
//remove next line for the sequential path approach//
r[1,1]=0;
end model
```

Latent Gold program to fit the rDINO model of Equation 2 (starting from variables statement)

```
variables
   dependent y1 cumlogit, y2 cumlogit, y3 cumlogit, y4 cumlogit, y5 cumlogit,
 y6 cumlogit, y7 cumlogit, y8 cumlogit, y9 cumlogit, y10 cumlogit, y11 cumlogit,
 y12 cumlogit, y13 cumlogit, y14 cumlogit, y15 cumlogit;
   latent
     a1 ordinal 2 score=(0 1), a2 ordinal 2 score=(0 1),
     a3 ordinal 2 score=(0 1), a4 ordinal 2 score=(0 1);
equations
 (r~full) a1 <-> a2 <-> a3 <-> a4;
  y1 <- 1 + (+)a1;
  y2 <- 1 + (+)a2;
  y3 <- 1 + (+)a3;
  y4 <- 1 + (+)a4;
  y5 <- 1 + (+a)a1 + (+a)a2 + (-a)a1 a2;
  y6 <- 1 + (+b)a1 + (+b)a3 + (-b)a1 a3;
  y7 <- 1 + (+c)a1 + (+c)a4 + (-c)a1 a4;
  y8 <- 1 + (+d)a2 + (+d)a3 + (-d)a2 a3;
  y9 <- 1 + (+e)a2 + (+e)a4 + (-e)a2 a4;
  y10 <- 1 + (+f)a3 + (+f)a4 + (-f)a3 a4;
  y11 <- 1 + (+g)a1 + (+g)a2 + (+g)a3 + (-g)a1 a2 + (-g)a1 a3
   + (-g)a2 a3 + (+g)a1 a2 a3;
  y12 <- 1 + (+h)a1 + (+h)a2 + (+h)a4 + (-h)a1 a2 + (-h)a1 a4
   + (-h)a2 a4 + (+h)a1 a2 a4;
  y13 <- 1 + (+i)a1 + (+i)a3 + (+i)a4 + (-i)a1 a3 + (-i)a1 a4
   + (-i)a3 a4 + (+i)a1 a3 a4;
  y14 <- 1 + (+j)a2 + (+j)a3 + (+j)a4 + (-j)a2 a3 + (-j)a2 a4
   + (-j)a3 a4 + (+j)a2 a3 a4;
  y15 <- 1 + (+k)a1 + (+k)a2 + (+k)a3 + (+k)a4 + (-k)a1 a2
   + (-k)a1 a3 + (-k)a1 a4 + (-k)a2 a3 + (-k)a2 a4 + (-k)a3 a4
   + (+k)a1 a2 a3 +(+k)a1 a2 a4 + (+k)a1 a3 a4 + (+k)a2 a3 a4
   + (-k)a1 a2 a3 a4;
r[1,1]=0;
end model
```

Latent Gold program to fit the restricted (and unrestricted) GRM to the ECPE data

```
variables
   dependent i1 cumlogit, i2 cumlogit, i3 cumlogit, i4 cumlogit, i5 cumlogit, i6 cumlogit,
   i7 cumlogit, i8 cumlogit, i9 cumlogit, i10 cumlogit, i11 cumlogit, i12 cumlogit, i13 cumlogit,
   i14 cumlogit, i15 cumlogit, i16 cumlogit, i17 cumlogit, i18 cumlogit, i19 cumlogit,
   i20 cumlogit, i21 cumlogit, i22 cumlogit, i23 cumlogit, i24 cumlogit, i25 cumlogit,
   i26 cumlogit, i27 cumlogit, i28 cumlogit;
   latent
      a1 ordinal 2 score=(0 1), a2 ordinal 2 score=(0 1), a3 ordinal 2 score=(0 1);
equations
   (r~full) a1 <-> a2 <-> a3;
   i1 <- 1 + (+)a1 + (+)a2 + (+)a1 a2;
   i2 <- 1 + (+)a2;
   i3 <- 1 + (+)a1 + (+)a3 + (+)a1 a3;
   i4 <- 1 + (+)a3;
   i5 <- 1 + (+)a3;
   i6 <- 1 + (+)a3;
 //i7 <- 1 + (+)a1 + (+)a3 + (+)a1 a3;//
 //order restriction on Item 7 can be done as follows//
   i7 <- 1 + (+)a1 + (+a)a3 + (-a)a1 a3 + (+)a1 a3;
   i8 <- 1 + (+)a2;
   i9 <- 1 + (+)a3;
   i10 <- 1 + (+)a1;
   i11 <- 1 + (+)a1 + (+)a3 + (+)a1 a3;
   i12 <- 1 + (+)a1 + (+)a3 + (+)a1 a3;
   i13 <- 1 + (+)a1;
   i14 <- 1 + (+)a1;
   i15 <- 1 + (+)a3;
   i16 <- 1 + (+)a1 + (+)a3 + (+)a1 a3;
   i17 <- 1 + (+)a2 + (+)a3 + (+)a2 a3;
   i18 <- 1 + (+)a3;
   i19 <- 1 + (+)a3;
   i20 <- 1 + (+)a1 + (+)a3 + (+)a1 a3;
   i21 <- 1 + (+)a1 + (+)a3 + (+)a1 a3;
   i22 <- 1 + (+)a3;
   i23 <- 1 + (+)a2;
   i24 <- 1 + (+)a2;
   i25 <- 1 + (+)a1;
   i26 <- 1 + (+)a3;
   i27 <- 1 + (+)a1;
   i28 <- 1 + (+)a3;
  r[1,1]=0;
 end model
```