RESEARCH REPORT

March 2001
RR-01-06

User's Guide for SCORIGHT (version 1.2): A Computer Program for Scoring Tests Built of Testlets

Xiaohui Wang
Eric T. Bradlow
Howard Wainer

ETS®
Educational Testing Service

Statistics & Research Division
Princeton, NJ 08541

User's Guide for SCORIGHT (version 1.2): A computer program for scoring tests built of testlets

Xiaohui Wang, Eric T. Bradlow and Howard Wainer

Abstract

SCORIGHT is a very general computer program for scoring tests. It models tests that are made up of dichotomously or polytomously rated items, or any kind of combination of the two through the use of a generalized IRT formulation. The items can be presented independently, or grouped into clumps of allied items (testlets), or in any combination of the two. When there are testlets the program assesses the degree of local dependence and adjusts the estimates accordingly. The estimation is accomplished within a fully Bayesian framework using Markov chain Monte Carlo procedures. This methodology yields the complete posterior distributions of the parameters, which allows the easy calculation of many important characteristics of the scores that are not available with other methods.

Key words: Markov chain Monte Carlo (MCMC), Bayesian, testlets, dichotomous, polytomous, item response theory (IRT)

Xiaohui Wang is a doctoral student in statistics, University of North Carolina, Eric Bradlow is an Associate Professor of Marketing and Statistics, The Wharton School of the University of Pennsylvania, and Howard Wainer is a Principal Research Scientist, Statistics and Psychometrics Research Group, Educational Testing Service. This program was prepared with the support from the research allocation of the Educational Testing Service, and is part of a long term project on Testlet Response Theory that has been supported at various times by the research budgets of the Graduate Record Exam, the Test of English as a Foreign Language, and the Joint Staff Research & Development Committee. We are grateful for the opportunity to acknowledge their support. We would also like to express our gratitude to David Thissen who provided us with advice and the North Carolina Department of Public Instruction who provided us with the data from the North Carolina Test of Computer Skills. Last we would like to thank Catherine Hombo for her careful and thoughtful reading of an earlier version of this manual; responding to her suggestions have made both SCORIGHT and its description better.

1 Introduction

Since IRT's introduction as a primary scoring method for standardized tests more than 30 years ago, people have been questioning the fundamental assumptions that underlie its foundation. One of the most basic tenets in IRT is that, given an individual's ability θ_i, his or her correct/incorrect answers to the items are *conditionally independent.* While this assumption leads to more tractable answers and computation, and may be approximately true when the items are carefully written (although sequence effects put that in question), current trends in educational assessment tools make the conditional independence assumption increasingly impractical.

Specifically, as the need for richer and more highly diagnostic forms of assessment arise, test writers have moved towards tests composed wholly, or in part, of testlets (Wainer and Kiely, 1987). In short, a testlet is a subset of items (one or more) generated from a *single* stimulus, e.g. a reading comprehension passage. In a testlet, one could easily imagine that items' behavior is more highly correlated than pure unidimensional ability would predict due to misreading of the passage (yielding the effect of all items within the testlet being answered apropos to a lower ability than presumed under a unidimensional model), a common subarea expertise, etc... Much work, under the name of "appropriateness measurement" (see Levine and Drasgow, 1988), and nonparametric approaches to detecting violations of conditional independence (Stout, 1987 and Zhang and Stout, 1999) has been done in this area.

Our approach has been to step beyond detection and instead to use a parametric structure to actually model the violations of conditional independence due to testlets. We modify standard IRT models to include a random effect that is common to all item responses within a testlet, but differing across testlets. In this manner, the generalized IRT model allows fitted item responses given by an individual in a testlet to be more highly correlated than their corresponding responses across testlets. In addition, our parametric approach is Bayesian, as we specify prior distributions that will allow for sharing of information across persons, testlets, and items. This parametric approach, briefly reviewed in Section 2 was first described in Bradlow, Wainer and Wang (1999) and subsequently extended in Wainer, Bradlow and Du (2000), and Wang, Bradlow, and Wainer (2000). The program (SCORIGHT), described here, is based on Wang, Bradlow, and Wainer (2000).

Specifically, SCORIGHT is a computer program designed to facilitate analysis of item response data that may contain testlets. This program is completely general in that it can handle data composed of binary or polytomously scored items, which are independent or nested within testlets. In addition, the model considered for the binary data is the 3-PL model (Birnbaum, 1968) and that for the ordinal data is the Samejima (1969) ordinal response model. In this manner, our program can specialize to the standard Rasch, 2-PL and ordinal models that are often fit by commercial software (e.g. BILOG, MULTILOG).

The remainder of this manual is as follows. First, we explicitly lay out the model that is fit to the data. This is followed by a "how to use" section, giving instructions on how to utilize the software. This, in turn, is followed by a description of the model output files, and finally we give a worked-out example.

2 Models

In this section, we describe the base probability models used. As our model is Bayesian in nature and can be used for both binary and polytomous items, this requires us to specify the following probability models:

(a) the model for binary data,

(b) the polytomous data model, and

(c) the prior distributions for all parameters governing (a) and (b).

2.1 Model Specification

The models that are used in this program have two basic probability kernels since they need to encompass both dichotomous and polytomous items. For dichotomous items, we utilized the three-parameter logistic model:

$$\mathrm{P}(Y_{ij} = 1) = c_j + (1 - c_j)\mathrm{logit}^{-1}(t_{ij}), \text{ and}$$

for polytomous items, the ordinal probit response model introduced by Samejima (1969):

$$\mathrm{P}(Y_{ij} = r) = \Phi(d_r - t_{ij}) - \Phi(d_{r-1} - t_{ij})$$

where Y_{ij} is the response of examinee i at item j, c_j is the lower asymptote ("guessing" parameter) for dichotomous item j, d_r are the latent cutoffs for the polytomous items, $\mathrm{logit}(\mathbf{x}) = \log(x/(1-x))$, Φ is the normal cumulative density function, and t_{ij} is the latent linear predictor of score.

This program models the extra dependence due to testlets by extending the linear score predictor t_{ij} from its standard form:

$$t_{ij} = a_j(\theta_i - b_j)$$

where a_j, b_j, and θ_i have their standard interpretations as item slope, item difficulty and examinee proficiency to:

$$t_{ij} = a_j(\theta_i - b_j - \gamma_{id(j)})$$

with $\gamma_{id(j)}$ the testlet effect (interaction) of item j with person i which is nested within testlet $d(j)$. The extra dependence of items within the same testlet (for a given examinee) is modeled in this manner as both would share the effect $\gamma_{id(j)}$ in their score predictor. By definition, $\gamma_{id(j)} = 0$ for all independent items. Thus, in summing up the model extension here, it is the parameters $\gamma_{id(j)}$ that represent the difference between our model and standard approaches.

In order to combine all the information across examinees, items, and most importantly testlets in this setting, a hierarchical Bayesian framework is introduced into the model. The following prior distributions for parameters $\Lambda_1 = \{a_j, b_j, q_j, \theta_i, \gamma_{id(j)}\}$ are asserted:

$$\begin{aligned} a_j &\sim N(\mu_a, \sigma_a^2) \\ b_j &\sim N(\mu_b, \sigma_b^2) \\ q_j &\sim N(\mu_q, \sigma_q^2) \\ \theta_i &\sim N(0,1) \\ \gamma_{id(j)} &\sim N(0, \sigma_{d(j)}^2) \end{aligned}$$

where $q_j = \log(c_j/(1-c_j))$. That is, the parameters are assumed to be heterogeneously distributed across the population and drawn from common distributions. We note that the mean and variance of the distribution for θ and the mean of the distribution for testlet effects $\gamma_{id(j)}$ are fixed to identify the model.

To complete the model specification, a set of hyperpriors for parameters

$$\Lambda_2 = \{\mu_a, \mu_b, \mu_q, \sigma_a^2, \sigma_b^2, \sigma_q^2, \sigma_{d(j)}^2\}$$

is added to reflect the uncertainty in their values. The distributions for these parameters were chosen out of convenience as conjugate priors to Λ_1. For the distribution means, we chose $\mu_a \sim N(0, V_a)$, $\mu_b \sim N(0, V_b)$, and $\mu_q \sim N(0, V_q)$, where $V_a^{-1} = V_b^{-1} = V_q^{-1}$ are set to 0. These uninformative priors were chosen to indicate our lack of knowledge regarding these parameters. Slightly informative hyperpriors (to ensure proper posteriors) were used for all prior variances given by $\sigma_z^2 \sim \chi_{g_z}^{-2}$, an inverse chi-square random variable with g_z degrees of freedom where $g_z = 0.5$ for all distributions.

We note that in a future version of SCORIGHT, we have a few modifications planned. First, it is likely that for a given item the item parameters a_j, b_j, c_j are correlated. In our model, we impose *independent* priors which should be generalized. This could easily be done by assuming that the item parameters come from a multivariate normal distribution with some unknown covariance matrix. A second and potentially more interesting extension would be to assume that $\log(\sigma_{d(j)}^2) \sim N(Z'_{d(j)}\beta, \sigma^2)$. That is, we will model the common testlet effect distribution as a function of testlet covariates $Z_{d(j)}$. For example, testlet covariates may include the number of words in the testlet stimuli, the type of stimuli, etc.... In this manner, we will be able to explore the factors that lead to larger interdependence (correlation) among testlet item parameters. This will have great practical importance for test design. We are also considering extensions that would allow the insertion of covariates for some of the other parameters as well (especially the important location parameters θ and b).

2.2 Computation

To draw inferences under this Bayesian testlet model, samples from the posterior distribution of the model's parameters are obtained using Markov chain Monte Carlo techniques. Details of the techniques are presented in Wang, Bradlow, and Wainer (2000). The relevant aspects of computation (from the user's perspective) for implementing the model are contained in this manual, and described in Section 3.

The model to be utilized in fitting the data is specified by the user. The choice for the dichtomous data is to fit a Rasch (1960) model, 2-PL or 3-PL model. For polytomous items, Samejima's model for graded responses is utilized.

3 How to use it

The SCORIGHT program is run under a DOS environment. Basically, the user at the keyboard starts this program, then answers the questions put forward by typing an appropriate response. These responses are used to determine the location of the input data files and the details of how SCORIGHT will be run.

On the following pages, the output from SCORIGHT is printed in `this font type`, and everything typed by the user from the keyboard is printed in **boldface**. We have written the manual in terms of a specific set of step-by-step instructions.

Step-by-Step User Instructions for SCORIGHT

Step 1: Start SCORIGHT

At the DOS window, in the subdirectory where the SCORIGHT program is installed, type:

scoright.exe

to start SCORIGHT. Then on the screen, the following will be printed:

```
This program estimates the ability and item parameters for both
dichotomous and polytomous items which could be independent
or nested within testlets using the Gibbs sampler.  To run this program,
you need to provide the following information.


Please enter the number of examinees and the number of items
in your dataset separated by at least one space:
```

Step 2: Enter the number of examinees and items

Now the user has to answer the question by typing two numbers separated by a single space, multiple spaces, a single tab, multiple tabs, or even a return key. SCORIGHT interprets all of these in the same way. The first number to be input is the total number of examinees, and the second is the total number of items. If one examinee is given an item, his or her response should be how he or she answers this item. If some items are not assigned to an examinee, the response of this examinee to these items should be coded as "N". "N" stands for not assigned. No other sorts of missing data are allowed in the current version of SCORIGHT. This is an area for future extension. At the current time, if you have other kinds of missing data you will need to preprocess them in one way or another to accomodate this limitation (e.g., impute values from an appropriate kind of missing data model).

For example, following the question, you can type:

```
Please enter the number of examinees and the number of items
in your dataset separated by at lease one space:  100 30
```

indicating that there are a total of 100 examinees each of whom has responded to a total of (up to) 30 test items (Again we note that under many test designs, e.g. CAT, examinees receive only a fraction of the total item test pool). If the user enters anything other than two numbers separated by spaces, SCORIGHT will print out an error message and ask the user to restart the program. Any wrong input, throughout the entire program for the numbers input to SCORIGHT will result in the same error message and a need to restart. If SCORIGHT asks for the name of an input file and the user enters a name of a file which the program could not find, the program will ask the user to reenter the name of the input file, rather than require a restart.

Step 3: Enter the number of testlets

Then the next question SCORIGHT asks is:

```
Enter the total number of testlets in the test:
```

If there are no testlets, i.e., all the items in the test are independent, enter **0**. Otherwise, enter the number of testlets. For example, if there are 4 testlets within the dataset, type **4** following the question. Note, that the number of testlets to be typed is the number of testlets made up only of dichotomous items plus the number of testlets made up only of polytomous items plus the number of testlets that are a mixture of dichotomous and polytomous items.

```
Enter the total number of testlets in the test:  4
```

Step 4: Enter the name/path of the data file

The next question asks the user to input the name of the file that contains the response data matrix. The whole name of the file including the drive name and all subdirectory names should be entered (i.e. the entire path). For example:

```
Enter the name of the file that contains the testdata:  c:\subdirectory\test.dat
```

The name of the file is case sensitive, since SCORIGHT is designed to be used under either a PC or Unix environment. The input data set is required to have a specific format.

(i) Each examinee's data record is to be contained in one row with item responses recorded sequentially.

(ii) Each item response occupies only one column in the file.

(iii) Do NOT put spaces between the item responses.

(iv) It is not necessary that the response to the first item start in the first column of the record, only that all respondent's answers start in the same column.

(v) If there are testlets, the item responses nested within each testlet should be ordered sequentially (clustered) in the data set.

For example:

```
EXAM0001 NNNN01110111111113111111121112
EXAM0002 011000101111111121112212311231
EXAM0003 010001101001111111111111211111
EXAM0004 111111101011111111433555322111
EXAM0005 010001001110111123112112121132
```

(vi) The responses for dichotomous items should be 1 for correct responses, 0 for incorrect.

(vii) For polytomous items, responses start from 1 to the highest category. For example, if a polytomous item has a total of 4 different responses, the responses on the data file should be 1, 2, 3, or 4. The model does not have any restrictions in the total number of categories for polytomous items; however, the current version of this program can only handle items with total categories equal to or less than 9. This was done to keep the format of input files consistent.

(viii) For items which are not assigned to the examinee (i.e., the nonresponse observed is ignorable), the responses should be coded "N".

Step 5: Enter the beginning and ending column of the test data

Then according to the format of the input file, answer the following query:

```
Enter the starting and ending columns of the test scores
for the data file:  10 39
```

For example, if the data are as above,

```
EXAM0001 NNNN01110111111113111111121112
EXAM0002 011000101111111121112212311231
EXAM0003 010001101001111111111111211111
EXAM0004 111111101011111111433555322111
EXAM0005 010001001110111123112112121132
```

then the beginning column would be 10, and ending column would be 39 (indicating a 30 item test). The two numbers entered should be separated by spaces. Here SCORIGHT will check the user's input, if the number of the ending column minus the number of the beginning column is not equal to the total number of items input, SCORIGHT will print out an error message and ask the user to restart the program.

Step 6: Enter the beginning and ending columns for the testlet items

If the user has answered **0** for the question about how many testlets there are in the test, the following questions won't be shown. Otherwise, the user has to provide information about the testlets' starting and ending columns.

For example if there were two testlets consisting of six items each, starting at the 13th and 19th column of the data set, the user would type:

```
Enter the starting and ending columns of Testlet #1:  13 18
Enter the starting and ending columns of Testlet #2:  19 24
...
```

The user has to fill all the information out about each testlet until all testlets are done. The number of questions about testlets will correspond to that number inputted in Step 3.

Step 7: Enter the beginning and ending rows of the data set

SCORIGHT then asks the question:

```
Enter the starting and ending rows of the test scores:  1 100
```

This would indicate that the data starts at the top of the file (as you can see this is not required) and goes to row 100, indicating 100 examinees. If the number of the ending row minus the number of the starting row plus one is not equal to the total number of examinees that was originally input, SCORIGHT will print out a suitable error message and ask the user to restart the program.

Step 8: Polytomous items: Yes or No

The next question is:

```
Do you have any polytomous items in the test?
Enter y for YES, n for NO: y
```

The input is not case sensitive (e.g., for example, the user could answer **y** or **Y**, and both will be interpreted in the same way). Users who answer **y** have to provide the information about the polytomous items through another file (Step 9).

Step 9: Create an information file about the items

This file indicates which items are dichotomous and which are polytomous by using one character: "D" for dichotomous items and "P" for polytomous items. The user has to put a "D" or "P" in the first column of each row of the file followed by a number which indicates the total number of categories for this item. The two entries should be separated by spaces (e.g., one or more spaces or tabs). If an item is dichotomous, the number of categories should be 2. Each item occupies one row of the file with the first item in the starting row, until all items are described. The following is an example of part of an item input file with the file name "index" under "c:\subdirectory\".

```
D 2
D 2
D 2
P 5
D 2
P 4
...
```

This means the first three items on the test are dichotomous items, the fourth one is a polytomous item with 5 categories, the fifth item is also a dichotomous item, and the sixth item is polytomous with 4 categories, etc.

Step 10: Enter the name/path of the item information file

The user is then asked for the location of the item information file (created in Step 9):

```
Enter the name of the file that contains the polytomous item information:
c:\subdirectory\index
```

It is the same requirement as in Step 4 for the item response data file.

Step 11: Enter the name/path where the output files should go

Since SCORIGHT generates many output files, the user is allowed to put all the output files within a user specified subdirectory.

```
Please enter the name of the subdirectory (include
the last backslash) where you want to put the analysis
results:  c:\result\
```

Step 12: Enter the number of iterations for the Gibbs sampler.

This computer program uses Gibbs sampling methods for inference. For the inferences to be valid, the Gibbs sampler must have "converged". The convergence rate depends on the data and initial values of the item parameters. In this step, the user must specify the number of iterations to run. Typically this would be at least 2000 iterations, and potentially a larger number. One way to diagnose "convergence" of the sampler, which we strongly recommend, is to take your data set and run SCORIGHT a number of times with different starting values for the parameters. Convergence would be indicated by the "same output" each time. Note that in Step 14 the user will be asked to designate a number k such that every k- th draw after convergence is kept for estimation. Therefore, the total number of iterations run should be set accordingly. See Step 14 for more details.

```
Enter the number of needed iterations of sampling:  2000
```

Step 13: Enter the number of initial draws to be discarded

As mentioned in Step 12, the sampler must have converged before valid inferences under the model can be obtained. Therefore, iterations (and their draws) obtained prior to convergence should be discarded for estimating quantities of interest. In this step of SCORIGHT, the user specifies the number of initial iterations of draws to be discarded for inference purposes. For example:

```
Enter the number of draws to be discarded:  1000
```

i.e. the draws after the initial 1000 will be considered for output (see Step 14) and further estimation or computation will be based on these.

Step 14: Enter the the size of the gap between posterior draws

Since the posterior draws are highly autocorrelated, it is often useful to reduce the level of autocorrelation by keeping only every k-th draw. When making the posterior draws, the user can specify how large k should be. For example:

```
Enter the the size of the gap between posterior draws:  10
```

In this case every tenth observation after convergence will be kept. Of course the user must consider this when specifying how many iterations to run (Step 12) as the output saved will only consist of every k-th (e.g. 10-th) obtained after convergence. As an example, if 2,000 iterations are run (Step 12) and 1,000 are used for burn-in (Step 13) and k is set equal to 10, then only (2,000-1,000)/10 = 100 will remain for estimation.

Step 15: Enter initial values for the parameters

The convergence of SCORIGHT may depend, in part, on initial starting values (guesses) for the parameters values. SCORIGHT will automatically select starting values for the user. However, sometimes the user may have some information (perhaps from the output of a different program) that suggests a reasonable set of starting values for either abilities θ_i or item parameters a_j, b_j, c_j. This part of SCORIGHT allows the user to use those values. In addition, SCORIGHT allows the user (if desired) to fix the values of a set of parameters, if those parameters are to be treated as fixed and known (although this is counter to the Bayesian nature of SCORIGHT, it is what allows SCORIGHT to nest the Rasch and 2-PL models).

```
Do you want to input the initial values
for parameter a?  If yes, enter 1,
otherwise, enter 0:

Please enter the name of the file
which contains the initial values
of the a parameters:

Do you want to fix the parameter values
of a throughout the whole analysis, if yes,
please enter 1, otherwise, enter 0:
```

These three questions are about the initial values of item parameter a. If the user answers **0** to the first question, the next two questions won't be shown. If the user answers **1** to the first question, then the following two must be answered. After asking the information of parameter a, SCORIGHT will ask similar questions about the parameters b and c, as well as the values of proficiency parameter θ.

The format of the files that contain the initial values of all three item parameters a, b, and c are the same and must take a specific form. For example, the file containing the initial values of item parameter a should have as many rows as items, and each row must contain the starting value of that item's value. SCORIGHT requires each initial value for each item to occupy one row. It is not necessary that the starting values begin at the first column or that each one start at the same column. For example, the file which contains:

0.65
0.74
1.43
...

would indicate $a_1 = 0.65, a_2 = 0.74, a_3 = 1.43$, etc... If a starting value file is used for the examinee abilities θ_i then there must be as many rows as examinees (matching Step 2).

This completes the user input for SCORIGHT.

4 Model Output on the Screen

After answering all the questions about the initial values, SCORIGHT prints out the input information for the user to check before it starts estimating the parameters, i.e. running the Gibbs sampler. If an item is an independent item, SCORIGHT prints -2. For all the items nested within the first testlet, the program prints 0 for each of them. For all the items nested within the second testlet, SCORIGHT prints 1 for each of them, and so on, until the last testlet. The user can therefore check the input by comparing the printout to the desired data structure.

```
Please check the input:
-2 indicates independent items,
0 indicates items in the first testlet,
1 indicates items in the second testlet,
··· and so on:

-2-2-2-2-2-2-2-2-2-2-2-2000000111111222222

If the input is correct, enter 1, otherwise, enter 0:
```

If the input is correct, the user should enter **1** as a response to the above question, and SCORIGHT will start running. Otherwise, the user should enter **0** to restart the program. SCORIGHT will then print some summary information about the analysis on the screen: the starting time and the time of completion of each iteration, like the following:

```
Starting time:  Thu Jun 15 21:11:31 2000

Time after 1 cycle:  Thu Jun 15 21:11:34 2000

Time after 2 cycles:  Thu Jun 15 21:11:36 2000

Time after 3 cycles:  Thu Jun 15 21:11:39 2000

Time after 4 cycles:  Thu Jun 15 21:11:42 2000

...
```

For example, in this output we see that the sampler is taking roughly 3 seconds per iteration. This indicates that running 2,000 iterations would take 6,000 seconds or 100 minutes. Of course, faster processors will yield shorter run lengths. More experience with SCORIGHT is necessary to provide efficient rules of thumb for the number of iterations required. Until such experience is amassed caution suggests going in the direction of too many iterations rather than too few. We commonly go 5,000 iterations, although so far this has always turned out to be a substantial overkill.

After finishing all the iterations, SCORIGHT prints the final message of the analysis. Note that the values indicated here for the number of iterations, number of initial iterations to be discarded, etc... will correspond to those values input in Steps 1 - 15 above. For example:

```
The Gibbs sampling of 2000 iterations is completed.
The point estimates are computed from the last 1000 iterations.
The theta estimates and their standard errors are in file -- thet.est.
The a, b, c estimates and their standard errors are in file -- itmp.est.
The jump and the acceptance ratio for a is 0.229635 and 0.333333
The jump and the acceptance ratio for b is 0.1782 and 0.466667
The jump and the acceptance ratio for q is 0.35 and 0.216667
The jump and the acceptance ratio for theta is 0.73205 and 0.45025
The jump and acceptance ratio for gamma in Testlet 1 is 0.06561 and 0.17175
The jump and acceptance ratio for gamma in Testlet 2 is 0.06561 and 0.1795
The jump and acceptance ratio for gamma in Testlet 3 is 0.06561 and 0.16275
```

The first two lines indicate the completion of the iterations. The third and fourth lines give the names of the output files of the item parameters estimates and the estimates of the examinees' proficiency θ values . From the fifth line to the second to the last, SCORIGHT prints the information of the jumps and the acceptance ratios for all the estimates. These correspond to a technical issue of how the sampler draws the parameter values. The details of their meaning are presented in Wang, Bradlow, and Wainer (2000).

5 Output Files and Format

There are several output files that the user will find under the subdirectory previously specified in Step 11. They are:

itmP.est
thet.est
a_DrawsC
b_DrawsC
c_DrawsC
t_DrawsC
d_DrawsC (optional)
gamm.est (optional)
gamV.est (optional)

The file "d_DrawsC" will be generated if the test has polytomous items. The file "gamm.est" and the file "gamV.est" will only be generated if the test has at least one testlet.

Item parameter output file: itmP.est

The file named "itmP.est" has as many rows as the total number of items. It contains the information of the estimated parameters of each item and is formatted as follows:

I4, 1X, 1C, 6F11.4, 2X, I1, mF11.5

which is a typical FORTRAN format. *I4* means the first 4 columns of each record (row) is an integer number which indicates the item number; *1X* means an empty space; *1C* means one character for item type, "D" for dichotomous item, "P" for polytomous item; *6F11.4* are six floating point values each occupying 11 columns, they are the estimated value of parameter a and its estimated standard error, the estimated value of parameter b and its estimated standard error, and the estimated value of parameter c and its estimated standard error. If the item is a polytomous item, the corresponding spaces for the estimated values of parameter c and its corresponding standard error contains "NA"s. After them, the output includes an integer that indicates the total number of categories, $m/2+1$ (if m is even) or $(m+1)/2$ (if m is odd), for this polytomous item. And *mF11.5* means m floating point values which are the $m/2$ estimated values of the cutoffs and their corresponding standard error for each category of a polytomous item. If any parameter is fixed with initial values, the corresponding output will be printed as "NA"s as well (e.g., a model without guessing parameters would have all c's indicated as NA).

Ability parameter output file: thet.est

The file named "thet.est" contains the estimated value of each examinee's proficiency θ value and its corresponding standard error. This file has the same number of records (rows) as examinees. Its format is:

I6, 2F11.4

I6 means the first 6 columns is an integer indicating the examinee number; *2F11.4* means that there are two floating values each occupying 11 columns, which are the estimated proficiency (θ) value and its estimated standard error. If the initial values of θ are supplied and fixed, there is no estimated standard error and hence the corresponding spaces of the output will be printed as "NA"s.

Output files containing parameter draws: *DrawsC

All the file names ending with "DrawsC" contain the random draws from the posterior distribution of the corresponding parameters. This information can be used to calculate ANY interesting statistic or to get some further statistical inference from them. File "a_DrawsC" contains the random draws for item parameters $a_1, ..., a_J$ with format

JF11.6

where J is the total number of items. There are J floating point values which are the draws. The file contains the same number of rows as the number of iterations specified (Step 12) minus the number of initial iterations that were discarded (Step 13) divided by the step size (Step 14) [g=(Step 12 - Step 13)/Step 14 =(number of iterations - number iterations in burn in)/k. For our example in this manual, the total records (rows) is equal to $g = (2000 - 1000)/10 = 100$. So, in summary the files "a_DrawsC", "b_DrawsC", "c_DrawsC" contain g rows and J columns corresponding to each of the J test items. For example, given the inputs typed as an example in this document, these files would have $g = 100$ rows and 30 columns (a 30 item test).

The files "b_DrawsC" and "c_DrawsC" are similar to the file "a_DrawsC". It should be noted that in the output for the item parameter c, if the item is a polytomous item, the estimated value of parameter c as well as its corresponding standard error are both NA, because there is no guessing parameter for polytomous items, only for dichotomous ones.

The format of file "t_DrawsC" is

nF11.6

where n is the total number of examinees. There are n floating point values that are the draws of the examinees' proficiency θ values from the sampler. The total number of records (rows) of this file is as same as the one in the file "a_DrawsC" (equal to g).

The format of the file "d_DrawsC" is

KF10.6

where K is the total number of polytomous cutoffs that need to be estimated for the model. Suppose there are L polytomous items in the test, and for each polytomous item, $d_i, i = 1, \ldots, L$, is the number of categories. Since the first cutoff of each polytomous item is set to 0, there are $d_i - 2$ estimated cutoffs needed for each polytomous item and hence

$$K = \sum_{i}^{L}(d_i - 2)$$

are needed in total. Thus, each record of the file contains the random draws of cutoffs for each polytomous item in sequence (that is the first $d_1 - 2$ are for the first polytomous item, and so on). As before, the number of records (rows) is the same as "a_DrawsC".

File "gamm.est" contains the estimated value of γ for each examinee and each testlet. It has as many records (rows) as the number of examinees. For each record, it has the following format:

I6, DF9.4

I6 means that the first 6 columns is an integer containing the examinee number. D is the total number of testlets. So there are D estimated γ values for each examinee.

File "gamV.est" contains all the estimated values of the variance of γ for each testlet for all the iterations. This is unlike all the other files which only contain every k-th draw after convergence. For example, as input before, this file has 2000 records. And each record has the following format:

I5, DF11.6

I5 means that the first 5 columns is an integer, the iteration number. D is total number of testlets. Since this file contains all the draw values from the first iteration, it is feasible to assess how fast the program has converged. We strongly recommend using the output in "gamV.est to help diagnose convergence.

6 Example: The North Carolina Test of Computer Skills

We use data from the North Carolina Test of Computer Skills (NCTCS) to demonstrate the approach. These data have a total 266 examinees and 26 items (Step 2), which comprise 4 testlets (Step 3) with the following testlet structure:

Testlet 1:	item 1-3,	all polytomous items
Testlet 2:	item 4-13,	all dichotomous items
Testlet 3:	item 14-20,	3 dichotomous items, 2 polytomous items, 1 dichotomous item, 1 polytomous item
Testlet 4:	item 21-26,	5 dichotomous items, 1 polytomous item

The part of the data containing just the item responses is shown below with the file name "nctcs.data" (Step 4):

```
42300101110011103101000002
22301000000011101101000011
23310111110001112101100012
33300111110111102201000103
22310111110001012101000001
...
```

which indicates a 26 item test. Since there are polytomous items, a file indicating which items are polytomous and which are dichotomous is needed. We named this file "index" (Step 10), which follows (Step 9):

```
P 4
P 4
P 4
D 2
D 2
D 2
D 2
D 2
D 2
D 2
D 2
D 2
D 2
D 2
D 2
D 2
P 3
P 3
D 2
P 3
D 2
D 2
D 2
D 2
D 2
P 3
```

In the North Carolina analysis, none of the dichotomous items are multiple choice. Thus the the likelihood of an examinee correctly guessing the answer is very low. This suggests that the dichotomous data can be well modeled with the two-parameter logistic model. To do this we set the initial values of the parameter c to 0. We do the same thing for all the polytomous items, although this is not strictly necessary, as SCORIGHT will turn the guessing parameters to 0s by default. Thus when the question (Step 14) is asked:

```
Do you want to fix the parameter values
of c through the whole analysis, if yes,
please enter 1, otherwise, enter 0:
```

enter **1**. The value of each item parameter c will be fixed through the entire analysis at their initial starting values; in this case at 0. Thus, as a result this turns the three-parameter logistic model into a two-parameter logistic model. A file named "c.raw" contains the initial values of parameter c (Step 14), part of which is:

```
0.0
0.0
0.0
0.0
...
```

The following contains the details of how to enter the information into SCORIGHT and the results of the analysis of North Carolina Computer Skill Test data.

```
This program estimates the ability and item parameters for both
dichotomous and polytomous items which could be independent items
or nested with some testlets via a Gibbs sampler.  To run this program,
you need to provide the following information.

Please enter the number of examinees and the number of items
to be estimated separated by spaces:  266 26

Enter the number of testlets in the whole test:  4

Enter the name of the file that contains the testdata:  c:\nccs\nctcs.data
Enter the starting and ending columns of the test scores:  1 26
Enter the starting and ending columns of Testlet #1:  1 3
Enter the starting and ending columns of Testlet #2:  4 13
Enter the starting and ending columns of Testlet #3:  14 20
Enter the starting and ending columns of Testlet #4:  21 26
Enter the starting and ending rows of the test scores:  1 266

Do you have polytomous items in the test?
Enter y for YES, n for NO: y

Enter the name of the file that contains the polytomous items information:
c:\nctcs\index

Please enter the name of the subdirectory (include
the last backslash) where you want to put the analysis
results:  c:\nctcs\result\

Enter the number of needed iterations of sampling:  6000

Enter the number of initial draws to be discarded:  5000

Enter the size of the gaps between the posterior draws:  1

Do you want to input the initial values
for parameter a?  If yes, enter 1,
otherwise, enter 0:  0

Do you want to input the initial values
for parameter b?  If yes, enter 1,
otherwise, enter 0:  0
```

```
Do you want to input the initial values
for parameter c?  If yes, enter 1,
otherwise, enter 0:  1
Please enter the name of the file
which contains the initial values
of parameter c's:  c:\nctcs\c.raw
Do you want to fix the parameter values
of c through the whole analysis, if yes,
please enter 1, otherwise, enter 0:  1
Do you want to input the initial values
for parameter theta?  If yes, enter 1,
otherwise, enter 0:  0
Please check the input:
-2 indicates independent items,
0 indicates items in the first testlet,
1 indicates items in the second testlet,
··· and so on:
000111111111122222222333333

If it is correct, enter 1, otherwise, enter 0:  1

Starting time:  Fri Jun 16 14:41:10 2000
Time after 1 cycle:  Fri Jun 16 14:41:10 2000
Time after 2 cycles:  Fri Jun 16 14:41:11 2000
...
Time after 6000 cycles:  Fri Jun 16 15:21:11 2000
The Gibbs sampling of 6000 iterations is completed.
The point estimates are computed from the last 1000 iterations.
The theta estimates and their standard errors are in file -- thet.est.
The a, b, c estimates and their standard errors are in file -- itmp.est.
The jump and the acceptance ratio for a is 0.25515 and 0.166667
The jump and the acceptance ratio for b is 0.162 and 0.423077
The jump and the acceptance ratio for q is 0.25515 and 0
The jump and the acceptance ratio for theta is 0.6655 and 0.510025
The jump and acceptance ratio for gamma in Testlet 1 is 0.0729 and 0.270677
The jump and acceptance ratio for gamma in Testlet 2 is 0.0729 and 0.265664
The jump and acceptance ratio for gamma in Testlet 3 is 0.0729 and 0.260652
The jump and acceptance ratio for gamma in Testlet 4 is 0.0729 and 0.255639
```

All the input files are under c:\nctsc\ and the output files are put under the subdirectory c:\nctsc\result\.

Let us first look in subdirectory `c:\nctsc\result\`. There are the following files:

itmP.est
thet.est
gamm.est
a_DrawsC
b_DrawsC
c_DrawsC
t_DrawsC
d_DrawsC
gamV.est

The following shows the first four lines of the file “itmP.est”:

```
 1   P   0.5190   0.0029   -5.5841   0.4111       NA   NA   4   0.00000   NA   1.95773   0.02334   3.50403   0.12563
 2   P   0.4424   0.0019   -5.6483   0.3766       NA   NA   4   0.00000   NA   2.02058   0.17329   3.29577   0.03789
 3   P   0.7424   0.0054   -4.5611   0.2019       NA   NA   4   0.00000   NA   1.28050   0.05673   3.74034   0.21672
 4   D   0.7769   0.0192   -1.0180   0.0640   0.0000   NA
...
```

If we look at the first line of the file *itmP.est*, it says that the first item is a polytomous item, the estimated value of parameter a is 0.5190, and the corresponding estimated standard error is 0.0029. The estimated value of parameter b for item 1 is -5.5841, and its corresponding estimated standard error is 0.4111. Since it is a polytomous item, there is no estimated value of parameter c. Hence, the next two columns have been coded NA and NA reflecting that they are not in the model. On the ninth column, 4 means this polytomous item has 4 total categories. Therefore, it has 3 cutoffs, in which the first one is set to 0.00, the estimated value of second one is 1.95773, and the estimated value of the third one is 3.50403. If we look at the fourth line of the file, it says the fourth item is a dichotomous item. Since we chose the 2-PL model, there is no estimated value for parameter c and its corresponding estimated standard error. There is no further information about the cutoffs since it is not a polytomous item.

The following are the first few lines from the file “thet.est”, which contains all the estimated values of the 266 examinees' proficiency, θ (which is calculated as the *mean* of each examinee's posterior density).

```
     1   -0.8026    0.4440
     2   -1.6045    0.5155
...
   265    -0.0791    0.3855
   266     0.4122    0.3240
```

The first line shows the estimated proficiency value of examinee # 1 and its corresponding estimated standard error. The second line corresponds to the second examinee, and so on. The last line of this file (line 266) contains the estimated proficiency value and its corresponding estimated standard error for the last examinee.

The following lines are from the file “gamm.est”, which contains the estimated γ values for each examinee across the all testlets.

```
     1  -0.1543   0.5946  -0.2277   1.2369
     2   0.2694   1.5695   0.2926   0.2399
 ...
   266  -0.1483  -1.0774   0.4527  -0.5897
```

Since there are 4 testlets in this analysis, the first line of this file says: examinee #1 has the estimated values of γ for testlet 1, 2, 3, and 4 as -0.1543, 0.5946, -0.2277, and 1.2369, respectively. Each subsequent line represents one examinee; the 266th line is the entry for the last examinee.

The other files contain the random draws from the posterior distributions. For example, the file "a_DrawC" has the last 1,000 random draws. The number (g) is specified by user as the difference between the number of iterations (Step 12) and the number of initial draws to discard as burn-in (Step 13) divided by the space between draws that are kept (Step 14), for the 26 "a" parameters. The NCTCS data file contains a 1000×26 matrix. The i-th row contains the draws of the 26 a's from the i-th iteration of the sampler, and the j-th column contains 1,000 draws from the posterior distribution of the j-th test item.

To demonstrate some of the full power of the Bayesian approach, let us examine in a little more detail some interesting inferences that could be obtained from the file "a_DrawsC" that would not be obtainable via standard software (BILOG, etc...). The power in the Bayesian approach, implemented here, is that we have obtained *draws* from the posterior distributions of interest, and hence have greater flexibility to answer a richer set of questions. Each of these inferences can be carried out similarly for any of the other item parameters, examinee abilities, or testlet effects. For ease of explanation and brevity, we only provide a demonstration for the "a" parameters.

(i) The posterior distribution for any given item's discrimination. One inference available is the entire posterior distribution for a given item's parameter a. This is simply done by constructing a frequency histogram of the draws from a given column of the file "a_DrawsC". For example, in Figure 1 below, we have constructed the posterior distribution for item 1 from the NCTCS analysis. We note that from this figure we can infer the shape of the posterior distribution, and an idea of the uncertainty/variance of the posterior distribution. As we observe, the posterior here appears to be somewhat multimodal. This suggests even more that the Bayesian approach is most appropriate in that we do not base our inferences on a normal approximation to this distribution.

(ii) The posterior probability that the true parameter value is greater than some number w. Imagine you were interested in assessing whether a given item parameter had a value of a greater than some number, say $w = 1.2$. Then an *exact* way to assess this is to simply count the number of draws in a *given* column that exceed the value w. This provides a *probability* that the discrimination of a given item is above a certain level. So note here the improved inference in that we now can give an assessment of the uncertainty with which we make a statement. This can be done in a similar manner, for example, for ability assessments in pass/fail exams.

(iii) The posterior probability that one item has discrimination greater than a second item. This

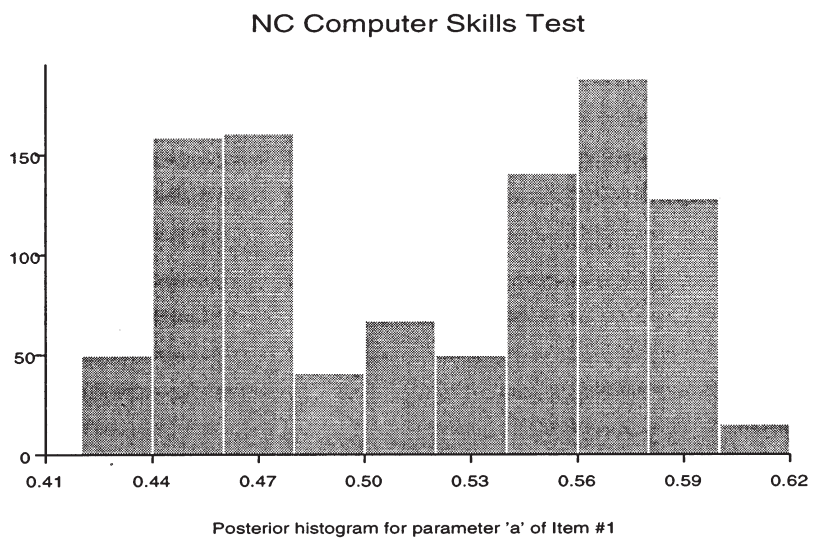

Figure 1: Histogram of the draws for the first item's discrimination parameter.

can be calculated directly from two columns of the file "a_drawsc". We simply count the fraction of times that the value in one column is greater than the corresponding value in the second column. This may be of interest if, say, one was interested in choosing from among a set of items to administer.

(iv) Constructing confidence intervals for parameters. In BILOG and/or other standard software programs, confidence intervals for parameters are constructed by using point estimates +/- a certain number of asymptotic standard errors. These asymptotic standard errors are computed numerically and assume large sample normal theory. As item (i) above showed, many parameter distributions are not normally distributed and hence these confidence intervals are of limited validity. Exact posterior intervals can be constructed from the Gibbs draws as follows. For any given column, compute the 2.5% and 97.5% sample quantile of the draws. This would provide you an exact 95% posterior interval. For example, in Figure 2 we have a plot of the 95% posterior intervals for each of the item parameters. From this we can assess the relative uncertainty with regards to each parameter.

(v) Graphical display of draws. In some instances, it may be of interest to look a sequential plot of the draws for a given parameter. This may be to assess whether the sampler has converged, to assess whether there are any extreme values, and/or the degree of autocorrelation among the draws. This is done by creating a plot where on the x-axis you utilize iteration number of the sampler, and on the y-axis you have the drawn value of a given parameter (i.e. a column of the file "a_DrawsC"). Note however, that the file "a_DrawsC" only contains every k-th draw "after convergence" so in this case this would be of limited use. An example of such a

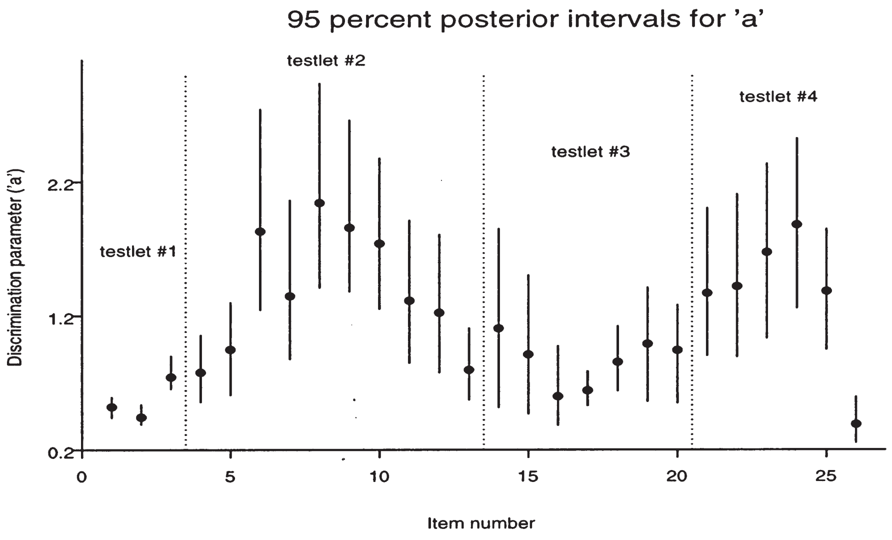

Figure 2: 95% posterior intervals for each of the 26 item discriminations. X-axis is the item number. The plotted points are the posterior means.

plot is given below in Figure 3.

The other files are similar to "a_DrawsC", except "gamV.est". The file, "gamV.est", contains all the draws (i.e. not just those beyond convergence as in the other parameters) of the drawn values of the variance of the γ's. In this case, for the NCTCS data, it is a matrix with 6000 rows (the number of iterations of the sampler) and four columns (the number of testlets). In Figure 4 is a trace plot of iteration number versus σ_γ^2 for the first testlet. From the figure, it is obvious that the variance of the γ's (for this testlet) is considerably greater than 0 (indicating a strong testlet effect). As above, a frequency histogram, posterior probabilities, etc... can also be computed from this file.

In summary, the draws that are available from the Gibbs sampler provide the opportunity to do exact inferences, inferences for interesting events whose probabilities can not be computed in closed-form, and to make statements with an appropriate degree of certainty. The bottom line is that the Gibbs sampler provides a true sample rather than simply referring to a theoretical distribution whose asymptotic properties are of uncertain relevance to a specific finite situation. Basic statistical practice tells us how to estimate things with samples. That is the real power of MCMC methods.

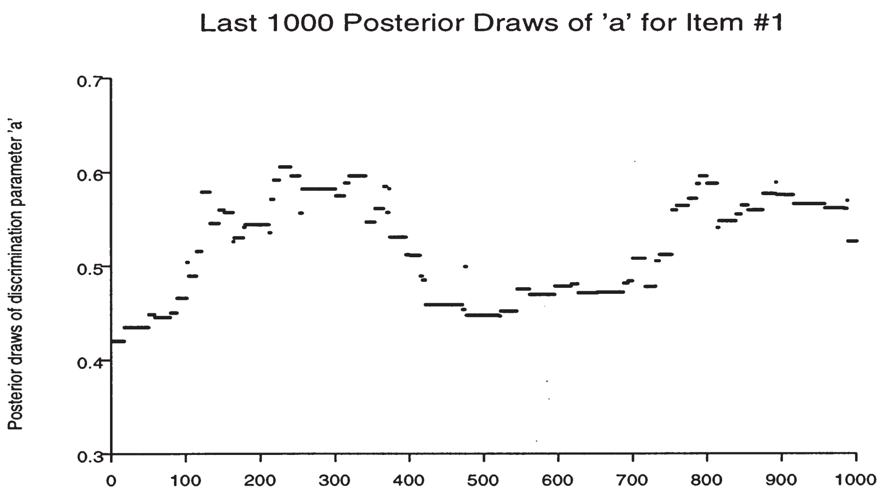

Figure 3: For the first item of the NC Computer Skill test: X-axis represents the iteration number of the sampler, Y-axis represents the drawn value of the a parameter.

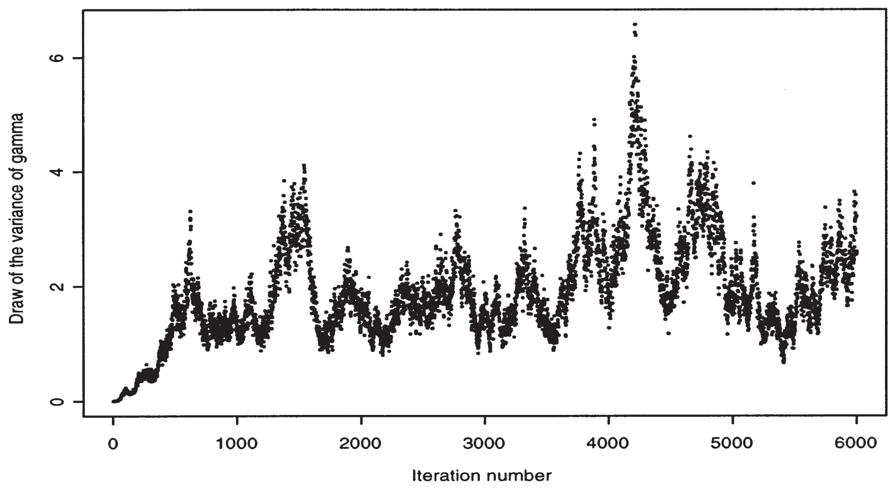

Figure 4: For the first testlet of NC Computer Skill test: X-axis represents the all 6000 iteration times, Y-axis represents the draws of the variance of the γ's.

References

Birnbaum, A. (1968). Some latent trait models. Chapter 17 in F.M. Lord & M. R. Novick's *Statistical Theories of Mental Test Scores.* Reading, Mass.: Addison Wesley.

Bradlow, E. T., Wainer, H. & Wang, X. (1999). A Bayesian random effects model for testlets. *Psychometrika*, 64, 153-168.

Levine, M.V. & Drasgow, F. (1988), Optimal Appropriateness Measurement. *Psychometrika*, 53, 161-176.

Rasch, G. (1980). *Probabilistic models for some intelligence and attainment tests.* Chicago, Ill.: University of Chicago Press (original work published 1960).

Samejima, F. (1969). Estimation of latent ability using a response pattern of graded scores. *Psychometrika Monographs* (Whole No. 17).

Stout, W. F. (1987). A nonparametric approach for assessing latent trait dimensionality. *Psychometrika*, 52, 589-617.

Wainer, H. , & Kiely, G. (1987). Item clusters and computerized adaptive testing: A case for testlets. *Journal of Educational Measurement*, 24, 185-202.

Wainer, H. , Bradlow, E. T. , & Du, Z. (2000) Testlet response theory: An analog for the 3-PL useful in adaptive testing. Chapter 13, in W. J. van der Linden & C. A. W. Glas's (Eds.) *Computerized adaptive testing:Theory and practice.* Boston, MA: Kluwer-Nijhoff.

Wang, X. Bradlow, E. T., & Wainer, H. (in press). A general Bayesian model for testlets: Theory and applications.*Applied Psychological Measurement.*

Zhang, J., & Stout, W. F. (1999). The theoretical DETECT index of dimensionality and its application to approximate simple structure. *Psychometrika*, 64, 213-249.