ACER ConQuest Manual

Ray Adams, Dan Cloney, Margaret Wu, Alla Berezner,
Alejandra Osses, Viktor Schwantner, Alvin Vista

2025-06-10

Contents

Chapter 1

ACER ConQuest: An Introduction

ACER ConQuest is currently at Version 5. To cite this version:

- Adams, R. J., Wu, M. L., Cloney, D., Berezner, A., & Wilson, M. (2020). *ACER ConQuest: Generalised Item Response Modelling Software* (Version 5.29) [Computer software]. Australian Council for Educational Research. https://www.acer.org/au/conquest

This section provides a brief survey of the models that ACER ConQuest can fit, and some applications to which these models can be applied.

1.1 What is ACER ConQuest?

ACER ConQuest is a computer program for fitting item response and latent regression models. It provides a comprehensive and flexible range of item response models to analysts, allowing them to examine the properties of performance assessments, traditional assessments and rating scales. ACER ConQuest also makes available, to the wider measurement and research community, the most up-to-date psychometric methods of multifaceted item response models, multidimensional item response models, latent regression models and drawing plausible values.

1.2 What are the models that ACER ConQuest can fit?

ACER ConQuest brings together in a single program a wide variety of item response models (including multidimensional models) and provides an integration of item response and regression analysis.

1.2.1 Rasch's Simple Logistic Model

Rasch's simple logistic model for dichotomies (Rasch, 1980) is the simplest of all commonly used item response models. This model is applicable for data that are scored into two categories, generally representing correct and incorrect answers. ACER ConQuest can fit this model to multiple choice and other dichotomously scored items.

1.2.2 Rating Scale Model

Andrich's extension of the simple logistic model (Andrich, 1978) allows the analysis of sets of rating items that have a common, multiple-category response format. The rating scale model is of particular value when examining the properties of the Likert-type items that are commonly used in attitude scales.

1.2.3 Partial Credit Model

Masters' extension of the simple logistic model (Masters, 1982) to the partial credit model allows the analysis of a collection of cognitive or attitudinal items that can have more than two levels of response. This model is now widely used with performance assessments that yield richer data than the dichotomous data that are typically generated by traditional assessment practices.

1.2.4 Ordered Partition Model

Wilson's extension of the partial credit model (Wilson, 1992) to the ordered partition model allows a many-to-one correspondence between item response categories and scores. Most item response models require a one-to-one correspondence between the categories of response to items and the level of performance that is attributed to those categories, for example, the dichotomous Rasch model, as it name implies, models two categories of performance on an item. These categories are usually identified with the digits 0 and 1, which serve both as category labels and score levels. Similarly, for the partial credit model, the digits 0, 1, 2 and so on serve both as category labels and score levels. In each case, there is a one-to-one correspondence between the category label and the score level. ACER ConQuest allows this correspondence between category labels and score levels to be broken by permitting items to have any number of categories assigned to the same score level, while the categories are still modelled separately. For example, an item that taps students' conceptual understanding of a science concept may elicit responses that reflect four different types of conceptual understanding. One of the responses may be considered very naive and scored as level zero, a second type of response may be regarded as indicative of a sophisticated understanding and be scored as level two, and the two remaining categories may both indicate partially correct, but qualitatively different, misconceptions that can each be reasonably scored as level one. ACER ConQuest can analyse this as a four-category item with three different score levels. It does this through the application of Wilson's ordered partition model (Wilson, 1992).

1.2.5 Linear Logistic Test Model

Fischer (1983) developed a form of Rasch's simple logistic model that allows the item difficulty parameters of items to be specified as linear combinations of more fundamental elements, such as the difficulties of cognitive subtasks that might be required by an item. ACER ConQuest is able to fit the linear logistic model to both dichotomous and polytomous response items.

1.2.6 Multifaceted Models

Linacre's multifaceted model (Linacre, 1994) is an extension of the linear logistic model to partial credit items. Standard item response models have assumed that the response data that are modelled result from the interaction between an object of measurement (a student, say) and an agent of measurement (an item, say). Linacre (1994) has labelled this two-faceted measurement, one facet being the object of measurement and the other the agent of measurement. In a range of circumstances, however, additional players, or facets, are involved in the production of the response. For example, in performance assessment, a judge or rater observes a student's performance on tasks and then allocates it to a response category. Here we have three-faceted measurement, where the response is determined by the characteristics of the student, the task and the rater. The general class of models that admit additional facets are now called multifaceted item response models.

1.2.7 Generalised Unidimensional Models

ACER ConQuest's flexibility, which enables it to fit all of the unidimensional models described above, derives from the fact that the underlying ACER ConQuest model is a repeated-measures, multinomial, logistic-regression model that allows the arbitrary specification of a linear design for the item parameters. ACER ConQuest can automatically generate the linear designs to fit models like those described above, or it can import user-specified designs that allow the fit of a myriad of other models to be explored (see section 2.10). Imported models can be used to fit mixtures of two-faceted and multifaceted responses, to impose equality constraints on the parameters of different items, and to mix rating scales with different formats, to name just a few possibilities.

1.2.8 Multidimensional Item Response Models

ACER ConQuest analyses are not restricted to models that involve a single latent dimension. ACER ConQuest can be used to analyse sets of items that are designed to produce measures on up to ten latent dimensions. Wang (1995) and Adams, Wilson, & Wang (1997) have described two types of multidimensional tests: multidimensional between-item tests and multidimensional within-item tests. Multidimensional between-item tests are made up of subsets of items that are mutually exclusive and measure different latent variables. That is, each item on the test serves as an indicator for a single latent dimension. In multidimensional within-item tests, each of the items can be an indicator of multiple latent dimensions. ACER ConQuest is able to fit all of the above-listed unidimensional models to undertake confirmatory analyses of either multidimensional within-item or multidimensional between-item tests.[1]

[1] If ACER ConQuest is being used to estimate a model that has within-item multidimensionality, then the `set` command argument `lconstraints=cases` must be provided. ACER ConQuest can be used to estimate a within-item multidimensional model without `lconstraints=cases`. This will, however, require the user to define and import an appropriate design matrix.

1.2.9 Latent Regression Models

The term latent regression refers to the direct estimation of regression models from item response data. To illustrate the use of latent regression, consider the following typical situation. We have two groups of students, group A and group B, and we are interested in estimating the difference in the mean achievement of the two groups. If we follow standard practice, we will administer a common test to the students and then use this test to produce achievement scores for all of the students. We would then follow a standard procedure, such as regression (which, in this simple case, becomes identical to a t-test), to examine the difference in the means. Depending upon the model that is used to construct the achievement scores, this approach can result in misleading inferences about the differences in the means. Using the latent regression methods described by Adams, Wilson, & Wang (1997), ACER ConQuest avoids such problems by directly estimating the difference in the achievement of the groups from the response data.

1.3 How does ACER ConQuest fit these models?

ACER ConQuest produces marginal maximum likelihood estimates for the parameters of the models summarised above. The estimation algorithms used are adaptations of the quadrature method described by Bock & Aitkin (1981), Gauss-Hermite quadrature, and the Monte Carlo method of Volodin & Adams (1995). The fit of the models is ascertained by generalisations of the Wright & Masters (1982) residual-based methods that were developed by Wu (1997). A summary of these procedures is provided in Estimation in Chapter 3, Technical matters.

1.4 Some applications of ACER ConQuest

1.4.1 Performing item analysis

With each of the models that it fits, ACER ConQuest provides parameter estimates, errors for those estimates, and diagnostic indices of fit. These are the basic components of an item analysis based on item response modelling. In addition to producing item response modelling-based information, ACER ConQuest produces an array of traditional item statistics, such as KR-20 and Cronbach's alpha coefficients of reliability, distractor analyses for multiple choice questions, and category analyses for multicategory items.

1.4.2 Examining Differential Item Functioning

ACER ConQuest provides powerful tools for examining differential item functioning. ACER ConQuest's facility for fitting multifaceted models and imposing linear constraints on item parameters allows convenient but rigorous testing of the equality of item parameter estimates in multiple groups.

1.4.3 Exploring Rater Effects

The exploration of rater effects is an important application of multifaceted models implemented in ACER ConQuest. Multifaceted models can be used to examine variation in the harshness or leniency of raters, they can be used to examine the propensity of raters to favour different response categories, and they can be used to examine the fit (or consistency) of individual raters with other raters.

1.4.4 Estimating Latent Correlations and Testing Dimensionality

By providing the opportunity to fit multidimensional item response models, ACER ConQuest allows the correlations between latent variables to be estimated. Estimating the correlations in this fashion avoids the problems associated with the influence of measurement error on obtaining accurate and unbiased estimates of correlations between constructed variables. Fitting ACER ConQuest with alternatively posited dimensionality structures and comparing the fit of these models also provides a powerful mechanism for formally checking dimensionality assumptions.

1.4.5 Drawing Plausible Values

The combination of item response modelling techniques and methods for dealing with missing-response data through multiple imputation has resulted in the so-called plausible values methodology (Mislevy, 1991) that is now widely used in sophisticated measurement contexts. Through the use of plausible values, secondary analysts are able to use standard software and techniques to analyse data that have been collected using complex matrix sampling designs. A particularly powerful feature of ACER ConQuest is its ability to draw plausible values for each of the models that it fits.

1.5 Where to find more information

ACER ConQuest is able to fit a large range of statistically sophisticated models, and it is not possible for either the measurement or statistical theory underpinning those models to be adequately discussed in this manual. Nor is it possible for the manual to do anything but touch on the range of applications to which the models can be applied. For those interested in further information on the ACER ConQuest models and their application, we refer you to the following papers: Adams & Wilson (1996); Adams, Wilson, & Wang (1997); Adams, Wilson, & Wu (1997); Mislevy et al. (1992); Wright & Masters (1982); and Wright & Stone (1979).

1.6 Installing ACER ConQuest

ACER ConQuest requires installation on both Windows (64 Bit) and Mac OS (Both arm64 and x86). On Windows, double click the installer file on Windows (`ACER ConQuest.msi`) to be guided through installation.

On Mac OS, open the installer disk image (`ConQuest_X_YY_Z.dmg` - where X, Y, and Z is the version number) and drag the folder `ConQuest` to the Applications folder (or any other location you would like to install to).

To open ACER ConQuest, double click the icon in the install location, or type the location of the executable in a console window. The default install locations are:

- Windows
  - `%ProgramFiles%/ACER ConQuest/ConQuestConsole.exe`
  - `%ProgramFiles%/ACER ConQuest/ConQuestGUI.exe`
  - NOTE: There is a deprecated 32 Bit version of ConQuest that will be installed to `%ProgramFiles(x86)%` instead.
- Mac OS (arm64 and x86)
  - `~/Applications/ConQuest/ConQuest`
  - NOTE: The Mac version is console-only.

1.6.1 Licence key instructions

Once you have installed ACER ConQuest, you will need to activate you licence. The licence key activates both console and GUI versions (GUI is Windows only).

1.6.1.1 Console Mode

Start the program, type the following ACER ConQuest set command (where you use the key provided to you instead of 'std-999-0000') and press enter.

```
set softkey=std-999-0000;
```

Then close and restart the program.

1.6.1.2 GUI Mode

Start the program and select File, then New. In the Input Window, type the following ACER ConQuest set (where you use the key provided to you instead of 'std-999-0000').

```
set softkey=std-999-0000;
```

Now select Run, then Run All. Then close and restart the program.

Chapter 2

An ACER ConQuest Tutorial

This section of the manual contains 13 sample ACER ConQuest analyses. They range from the traditional analysis of a multiple choice test through to such advanced applications of ACER ConQuest as the estimation of multidimensional Rasch models and latent regression models. Our purpose here is to describe how to use ACER ConQuest to address particular problems; it is not a tutorial on the underlying methodology. For those interested in developing a greater familiarity with the mathematical and statistical methods that ACER ConQuest employs, the sample analyses in the tutorials should be supplemented by reading the material that is cited in the discussions.

In each sample analysis, the command statements used by ACER ConQuest are explained. For a comprehensive description of each command, see Chapter 4, ACER ConQuest Command Reference.

The files used in the sample analyses can be found on the ACER Notes and Tutorials website.

Before beginning the tutorials, this section starts with a description of the basic elements of the ACER ConQuest user interfaces.

2.1 The ACER ConQuest User Interfaces

ACER ConQuest is available with both a graphical user interface (GUI) and a simple command line or console interface (CMD). The ACER ConQuest command statement syntax (described in Chapter 4, ACER ConQuest Command Reference) used by the GUI and the console versions is identical. The tutorials are presented assuming use of the GUI version of ACER ConQuest.

Both the console version of the program and the GUI version are compatible with Microsoft Windows. The console version of the program is available for Mac OSX. There is no GUI version for Mac OSX.

The console version runs faster than the GUI version and may be preferred for larger and more complex analyses. The GUI version is more user friendly and provides plotting functions that are not available with the console version.

The two interfaces are described below.

2.1.1 GUI Version

Figure 2.1 shows the screen when the GUI version of ACER ConQuest is launched (double-click on the file `ConQuestGUI.exe`). You can now proceed in one of three ways.

1. Open an existing command file (`File`→`Open`).[1]
2. Open a previously saved ACER ConQuest system file (`File`→`Get System File`)
3. Create a new command file (`File`→`New`).

If you choose to open an existing command file, a standard Windows File/Open dialog box will appear (see Figure 2.2). Locate the file you want to open. Note that, by default, the list of files will be restricted to those with the extension `.cqc`, which is the default extension for ACER ConQuest command files. To list other files, change the *file type* to *All Files*.

If you choose to read a previously created system file, a standard Windows File/Open dialog box will appear. Locate the file you want to open. Note that, by default, the list of files will be restricted to those with the extension `.CQS`, which is the default extension for ACER ConQuest system files.

If you choose to create a new command file, or after you have selected an existing command file or system file from the File/Open dialog box, two windows will be created: an input window and an output window. These windows are illustrated in Figure 2.3.

A status bar reporting on the current activity of the program is located at the bottom of the ACER ConQuest window.

[1] We use the notation `File`→`Open` to indicate that the menu item `Open` should be chosen from the `File` menu.

Figure 2.1: The ACER ConQuest Screen at Startup

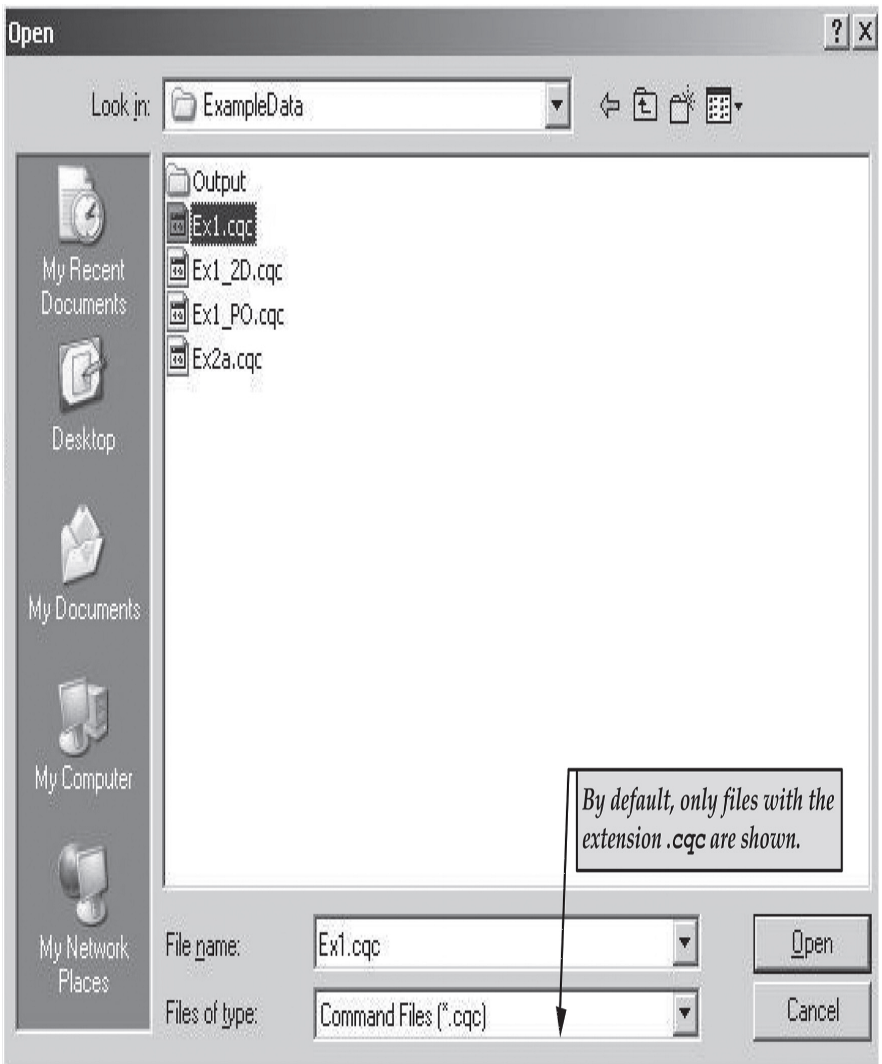


Figure 2.2: File/Open Dialog Box

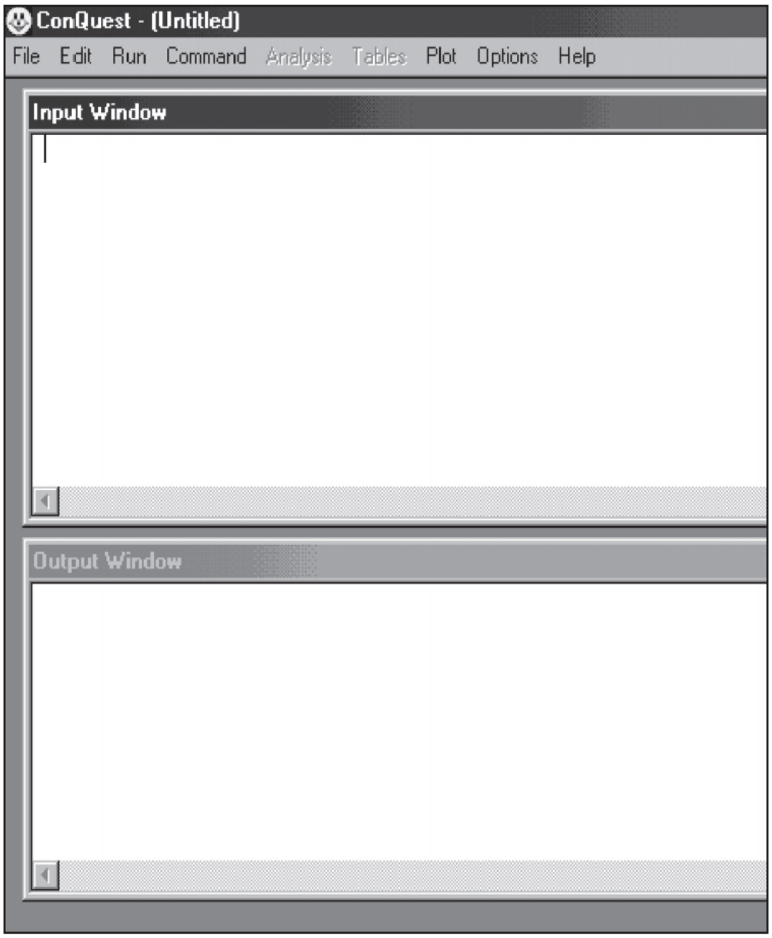


Figure 2.3: The ACER ConQuest Input and Output Windows

2.1.1.1 The Input Window

The input window is an editing window. If you have opened an existing ACER ConQuest command file, it will contain the file. If you have opened a system file or selected new, the input window will be blank.

Type or edit the ACER ConQuest command statements in the input window. To start execution of the command statements, choose `Run`→`Run All`, if you wish to run all of the commands in the input window. To execute a subset of the commands then highlight the desired commands, choose `Run`→`Run Selection`. ACER ConQuest will execute the command statements that are selected. This is illustrated in Figure 2.4. If nothing is highlighted, then ACER ConQuest will not execute any commands.

Figure 2.4: Running a Selection

2.1.1.2 The Output Window

The output window displays the results and the progress of the execution of the command statements. As statements are executed by ACER ConQuest, they are echoed in the output window. When ACER ConQuest is estimating item response models, progress information is displayed in the output window. Certain ACER ConQuest statements produce displays of the results of analyses. Unless these results are redirected to a file, they will be shown in the output window.

The output window has a limited amount of buffer space. When the buffer is full, material from the top of the buffer will be deleted. The contents of the buffer can be saved or edited at any time that ACER ConQuest is not busy undertaking computations. The output is cleared whenever `Run`→`Run All` is chosen to execute all statements in the input window, whenever ACER ConQuest executes a `reset` statement, and whenever `Command`→`Clear Output` is selected.

2.1.1.3 Using the Menus and Commands

The menus listed in the menu bar have several characteristics familiar to Windows users. The menus are drop-down menus. Menus are closed by either selecting a command or pressing the Esc key. Menus can be activated by pressing the Alt key plus the underlined character in the menu name. A command can then be selected by pressing the underlined character in the command name. Keyboard shortcuts are indicated next to the commands.

Only one file can be open at any time.

2.1.1.4 Description of the Input Window Menu Items

When the input window is the active window, the menu bar contains the following items:

- `File`, `Edit`, `Run`, `Command`, `Analysis`, `Tables`, `Plot`, `Options`, and `Help`
  The majority of the items under these menus, which are for controlling ACER ConQuest's data specification, estimation and result display, are described in detail in other sections. *Note that this section describes the commands that are not described elsewhere in the manual.*

- `File`→`New`
  Creates a new input and output window. If you already have a file displayed in the

input window, you will be prompted to save any changes, if necessary, before ACER ConQuest closes that file and creates a new input and output window.

- `File→Open`
  Opens an existing ACER ConQuest command file and places it in the input window. If you already have a file displayed in the input window, you will be prompted to save any changes, if necessary, before ACER ConQuest closes that file and displays the File/Open dialog box.

- `File→Save`
  Saves the contents of the input window under the current file name or prompts for a new file name if you have not yet named the file.

- `File→Save As`
  Prompts for a new file name and saves the contents of the input window under the new file name.

- `File→Print`
  Sends the contents of the input window to the printer.

- `File→Exit`
  Terminates the ACER ConQuest program. If you have a file displayed in the input window, you will be prompted to save any changes, if necessary, before ACER ConQuest terminates.

- `Edit→Undo`, `Edit→Cut`, `Edit→Copy`, `Edit→Paste`, `Edit→Delete`, `Edit→Select All`, `Edit→Font`
  These are standard Windows editing commands. The `Edit→Undo` command undoes the most recent edit only.

- `Run→Run All`
  Starts execution of all of the command statements that are contained in the input window.

- `Run→Run Selection`
  Starts execution of the command statements that are completely highlighted. If nothing is highlighted, ACER ConQuest will execute all the command statements in the input window. Note that complete and legal commands, or sets of commands, must be highlighted, if only part of a statement is highlighted, ACER ConQuest will display an error message.

- `Run`→`Stop`
  Interrupts a current analysis

- `Plot`→`Launch PlotQuest`
  Starts the ACER ConQuest plotting program

- `Options`→`Display Progress`
  Toggles display of a dialog box that reports on estimation progress.

- `Help`→`About this program`
  Shows the version number of the ACER ConQuest program you are using.

2.1.1.5 Description of the Output Window Menu Items

When the output window is the active window, the menu bar displays the following commands:

- `File`→`Save`
  Saves the contents of the output window under the current file name or prompts for a new file name if you have not yet named the file.

- `File`→`Save As`
  Prompts for a new file name and saves the contents of the output window under the new file name.

- `File`→`Print`
  Sends the contents of the output window to the printer.

2.1.2 Console Version

The console version of ACER ConQuest provides a command line interface that does not draw upon the GUI features of the host operating system. This version of ACER ConQuest is substantially faster than the GUI version but is more limited in its functionality.

Figure 2.5 shows the screen when the console version of ACER ConQuest is started (double-click on the file ConQuestConsole.exe). The less than character (`<`) is the ACER ConQuest prompt. When the ACER ConQuest prompt is displayed, any appropriate ACER ConQuest statement can be entered. As with any command line interface, ACER ConQuest attempts to execute the command statement when you press the Enter key. If you have not yet entered a semi-colon (`;`) to indicate the end of the statement, the ACER ConQuest prompt changes to a plus sign (`+`) to indicate that the statement is continuing on a new line.

Figure 2.5: The Console ACER ConQuest Screen at Startup

The syntax of ACER ConQuest commands is described in section 4.1, and the remaining sections in this section illustrate various sets of command statements.

To exit from the ACER ConQuest program, enter the statement `quit;` at the ACER ConQuest prompt.

On many occasions, a file containing a set of ACER ConQuest statements (an ACER ConQuest command file) will be prepared with a text editor, and you will want ACER ConQuest to run the set of statements that are in the file. For example if the file is called `myfile.cqc`, then the statements in the file can be executed in two ways.

- In the first method, start ACER ConQuest (see the Installation Instructions if you don't know how to start ACER ConQuest) and then type the command

  ```
  submit myfile.cqc;
  ```

- A second method, which will work on operating systems that allow ACER ConQuest to be launched from a command line interface, is to provide the command file as a command line argument. That is, launch ACER ConQuest using

  ```
  ConQuestCMD myfile.cqc;
  ```

With either method, after you press the Enter key, ACER ConQuest will proceed to execute each statement in the file. As statements are executed, they will be echoed on the screen. If you have requested displays of the analysis results and have not redirected them to a file, they will be displayed on the screen.

ACER ConQuest system files can be exchanged between the console and GUI versions. For large analyses it may be advantageous to fit the model with the console version, save a system file and then read that system file with the GUI version, for the purpose of preparing output plots and other displays.

2.1.3 Temporary Files

While ACER ConQuest is running, a number of temporary files will be created. These files have prefix "laji" (e.g., laji000.1, laji002.1, etc.). ACER ConQuest removes these files before closing the program. If these temporary files remain when ACER ConQuest is not running, you should remove them, as these files are typically large in size.

2.2 A Dichotomously Scored Multiple Choice Test

Multiple choice items are perhaps the most widely applied tool in testing. This is particularly true in the case of the testing of the cognitive abilities or achievements of a group of students.[2] The analysis of the basic properties of dichotomous items and of tests containing a set of dichotomous items is the simplest application of ACER ConQuest. This first

[2]The term 'student' or 'students' is used to indicate the object of the measurement process, that is, the entity that is being measured. This term has been chosen because most of the sample analyses are set in an educational context where the object of measurement is typically a student. The methods, however, are applicable well beyond the measurement of students.

sample analysis, shows how ACER ConQuest can be used to fit Rasch's simple logistic model to data gathered with a multiple choice test. ACER ConQuest can also generate a range of traditional test item statistics.[3]

2.2.1 Required files

The files used in this sample analysis are:

filename	content
ex1.cqc	The command statements.
ex1_dat.txt	The data.
ex1_lab.txt	The variable labels for the items on the multiple choice test.
ex1_shw.txt	The results of the Rasch analysis.
ex1_itn.txt	The results of the traditional item analyses.

(The last two files are created when the command file is executed.)

The data used in this tutorial comes from a 12-item multiple-choice test that was administered to 1000 students. The data have been entered into the file `ex1_dat.txt`, using one line per student. A unique student identification code has been entered in columns 1 through 5, and the students' responses to each of the items have been recorded in columns 12 through 23. The response to each item has been allocated one column; and the codes a, b, c and d have been used to indicate which alternative the student chose for each item. If a student failed to respond to an item, an M has been entered into the data file. An extract from the data file is shown in Figure 2.6.

2.2.2 Syntax

In this sample analysis, the Rasch (1980) simple logistic model will be fitted to the data, and traditional item analysis statistics are generated. `ex1.cqc` is the command file used in this tutorial, and is shown in the code box below. A list explaining each line of syntax follows.

The syntax for ACER ConQuest commands is presented in section 4.1.

`ex1.cqc`:

[3] The analysis of dichotomous tests with traditional methods is usually referred to as classical test theory.

```
         1         2
12345678901234567890123 (column numbers)

40016     acdabaeadacd
655       acdcccecbaca
31140     eccdbcebbacb
 .              .
 .              .
50321     dabcMcebdaca
30782     acddbcebbacc
 .              .
 .              .
```

Figure 2.6: Extract from the Data File `ex1\\_dat.txt`. Each column of the data file is labelled so that it can be easily referred to in the text. The actual ACER ConQuest data file does not have any column labels.

```
1 datafile ex1_dat.txt;
2 format id 1-5 responses 12-23;
3 labels << ex1_lab.txt;
4 key acddbcebbacc ! 1;
5 model item;
6 estimate;
7 show >> results/ex1_shw.txt;
8 itanal >> results/ex1_itn.txt;
9 /* rout option is for use in R using conquestr: */
10 plot icc ! rout=results/icc/ex1_;
11 plot mcc! legend=yes;
```

- **Line 1**
  The `datafile` statement indicates the name and location of the data file. Any file name that is valid for the operating system you are using can be used here.

- **Line 2**
  The `format` statement describes the layout of the data in the file `ex1_dat.txt`. This `format` statement indicates that a field that will be called `id` is located in columns 1 through 5 and that the responses to the items are in columns 12 through 23 of the data file. Every `format` statement must give the location of the responses. In fact, the explicit variable `responses` must appear in the `format` statement or ACER ConQuest will not run. In this particular sample analysis, the responses are those made by the students to the multiple choice items; and, by default, `item` will be the implicit variable name that is used to indicate these responses. The levels of the `item` variable (that is, item 1, item 2 and so on) are implicitly identified through their location within the set of responses (called the response block) in the `format` statement; thus, in this sample analysis, the data for item 1 is located in column 12, the data for item 2 is in column 13, and so on.

  > **EXTENSION:** The item numbers are determined by the order in which the column locations are set out in the response block. If you use the following:
  > `format id 1-5 responses 12-23;`
  > item 1 will be read from column 12. If you use:
  > `format id 1-5 responses 23,12-22;`
  > item 1 will be read from column 23

  > **TIP:** In some testing contexts, it may be more informative to refer to the response variable as something other than `item`. Using the variable name

`task` or `question` may lead to output that is better documented. Altering the name of the response variable is easy. If you want to use the name `tasks` rather than `item`, simply add an option to the `format` statement as follows:

```
format id 1-5 responses 12-23 ! tasks(12);
```

The variable name `tasks` must then be used to indicate the response variable in other ACER ConQuest commands. For example in the `model` statement in Line 5.

- **Line 3**
  The `labels` statement indicates that a set of labels for the variables (in this case, the items) is to be read from the file `ex1_lab.txt`. An extract of `ex1_lab.txt` is shown in Figure 2.7. (This file must be text only; if you create or edit the file with a word processor, make sure that you save it using the text only option.) The first line of the file contains the special symbol `===>` (a string of three equals signs and a greater than sign) followed by one or more spaces and then the name of the variable to which the labels are to apply (in this case, `item`). The subsequent lines contain two pieces of information separated by one or more spaces. The first value on each line is the level of the variable (in this case, `item`) to which a label is to be attached, and the second value is the label. If a label includes spaces, then it must be enclosed in double quotation marks (`" "`). In this sample analysis, the label for item 1 is `BSMMA01`, the label for item 2 is `BSMMA02`, and so on.

  TIP: Labels are not required by ACER ConQuest, but they improve the readability of any ACER ConQuest printout, so their use is strongly recommended.

- **Line 4**
  The `key` statement identifies the correct response for each of the multiple choice test items. In this case, the correct answer for item 1 is `a`, the correct answer for item 2 is `c`, the correct answer for item 3 is `d`, and so on. The length of the argument in the `key` statement is 12 characters, which is the length of the response block given in the `format` statement.

  If a `key` statement is provided, ACER ConQuest will recode the data so that any response a to item 1 will be recoded to the value given in the key statement option (in this case, `1`). All other responses to item 1 will be recoded to the value of the `key_default` (in this case, 0). Similarly, any response `c` to item 2 will be recoded to `1`, while all other responses to item 2 will be recoded to 0; and so on.

```
===> item
1    BSMMA01
2    BSMMA02
3    BSMMA03
4    BSMMA04
5    BSMMA05
6    BSMMA06
7    BSMSA07
8    BSMSA08
9    BSMSA09
10   BSMSA10
11   BSMSA11
12   BSMSA12
```

Figure 2.7: Contents of the Label File `ex1_lab.txt`.

- **Line 5**
The `model` statement must be provided before any traditional or item response analyses can be undertaken. When undertaking simple analyses of multiplechoice tests, as in this example, the argument for the `model` statement is the name of the variable that identifies the response data that are to be analysed (in this case, `item`).

- **Line 6**
The `estimate` statement initiates the estimation of the item response model.

> **NOTE:** The order in which commands can be entered into ACER ConQuest is not fixed. There are, however, logical constraints on the ordering. For example, `show` statements cannot precede the `estimate` statement, which in turn cannot precede the `model`, `format` or `datafile` statements.

- **Line 7**
The `show` statement produces a sequence of tables that summarise the results of fitting the item response model. In this case, the redirection symbol (`>>`) is used so that the results will be written to the file `ex1_shw.txt` in your current directory. If redirection is omitted, the results will be displayed on the console (or in the output window for the GUI version).

- **Line 8**
The `itanal` statement produces a display of the results of a traditional item analysis. As with the `show` statement, the results are redirected to a file (in this case, `ex1_itn.txt`).

- **Line 10**
The `Plot icc` statement will produce 12 item characteristic curve plots, one for each item. The plots will compare the modelled item characteristic curves with the empirical item characteristic curves. Note that this command is not available in the console version of ACER ConQuest.

- **Line 11**
The `Plot mcc` statement will produce 12 category characteristic curve plots, one for each item. The plots will compare the modelled item characteristic curves with the empirical item characteristic curves (for correct answers) and will also show the behavior of the distractors. Note that this command is not available in the console version of ACER ConQuest.

2.2.3 Running the Multiple Choice Sample Analysis

To run this sample analysis, start the GUI version. Open the file `ex1.cqc` and choose `Run`→`Run All`.

Alternatively, you can launch the console version of ACER ConQuest, by typing the command[4] (on Windows) `ConQuestConsole.exe ex1.cqc`.

ACER ConQuest will begin executing the statements that are in the file `ex1.cqc` and as they are executed, they will be echoed on the screen (or output window). When ACER ConQuest reaches the `estimate` statement, it will begin fitting Rasch's simple logistic model to the data, and as it does so it will report on the progress of the estimation. Figure 2.8 is an extract of the information that is provided during the estimation (in this case, the changes in the estimates after four iterations).

After the estimation is completed, the two statements that produce text output (`show` and `itanal`) will be processed and then, in the case of the GUI version two sets of 12 plots will be produced. In this case, the `show` statement will produce all six of its tables. All of these tables will be in the file `ex1_shw.txt`. The contents of the first table are shown in Figure 2.9.

This table is provided for cross-referencing and record-keeping purposes. It indicates the data set that was analysed, the format that was used to read the data, the model that was requested and the sample size. It also provides the number of parameters that were estimated, the number of iterations that the estimation took, and the reason for the termination of the estimation. The deviance is a statistic that indicates how well the item response model has fit the data; it will be discussed further in future sample analyses.

As Figure 2.9 shows, in this analysis 13 parameters were estimated. They are: (a) the mean and variance of the latent achievement that is being measured by these items; and (b) 11 item difficulty parameters. Following the usual convention of Rasch modelling, the mean of the item difficulty parameters has been made zero, so that a total of 11 parameters is required to describe the difficulties of the 12 items.

Figure 2.10 shows the second table from the file `ex1_shw.txt`. This table gives the parameter estimates for each of the test items along with their standard errors and some diagnostics tests of fit. The estimation algorithm and the methods used for computing standard errors and fit statistics are discussed in Chapter 3. In brief, the item parameter estimates are marginal maximum likelihood estimates obtained using an EM algorithm,

[4] If you wish to launch ACER ConQuest in this fashion on command-based systems, `ConQuestConsole.exe` must be in the directory you are working in or a path must have been set up; otherwise, you must type the entire path name.

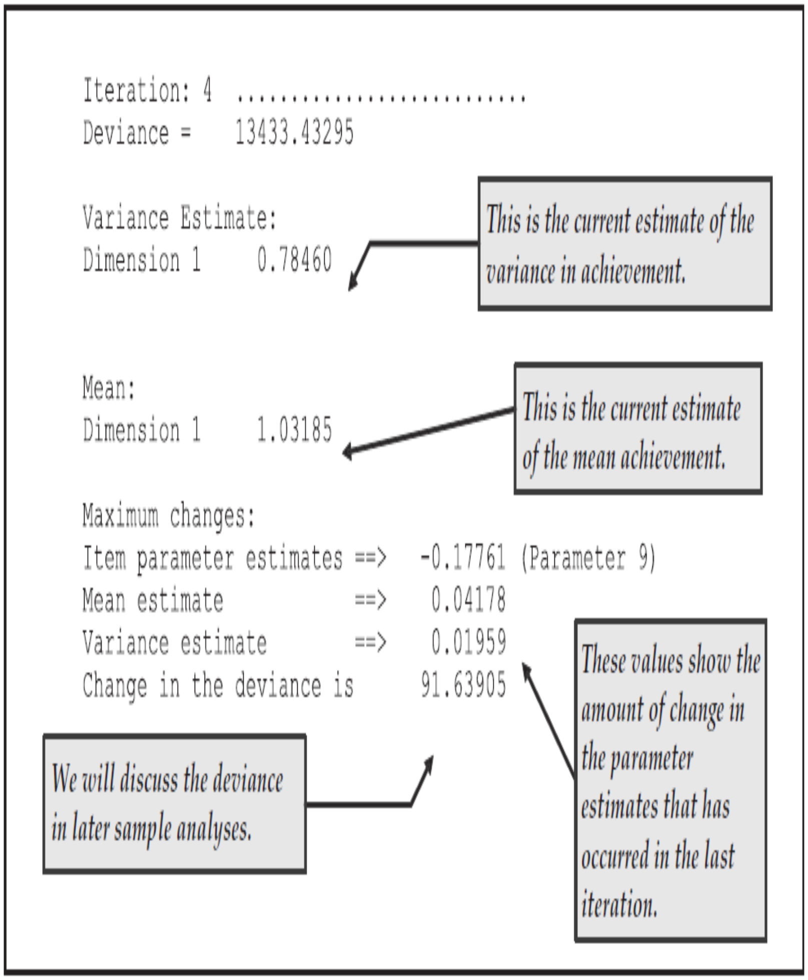


Figure 2.8: Reported Information on the Progress of Estimation

```
================================================================================
ConQuest: Generalised Item Response Modelling Software  Tue Oct 03 10:19 2006
SUMMARY OF THE ESTIMATION
================================================================================
Estimation method was: Gauss-Hermite Quadrature with 15 nodes
Assumed population distribution was: Gaussian
Constraint was: DEFAULT
The Data File: ex1.dat
The format:  id 1-5 responses 12-23
The regression model:
Grouping Variables:
The item model: item
Sample size: 1000
Final Deviance:    13274.87603
Total number of estimated parameters: 13
The number of iterations: 46
Termination criteria:  Max iterations=1000, Parameter Change= 0.00010
                       Deviance Change= 0.00010
Iterations terminated because the deviance convergence criteria was reached
Random number generation seed:    1.00000
Number of nodes used when drawing PVs: 2000
Number of nodes used when computing fit: 200
Number of plausible values to draw: 5
Maximum number of iterations without a deviance improvement: 100
Maximum number of Newton steps in M-step: 10
Value for obtaining finite MLEs for zero/perfects:    0.30000
```

Figure 2.9: Summary Table

the standard errors are asymptotic estimates given by the inverse of the hessian, and the fit statistics are residual-based indices that are similar in conception and purpose to the weighted and unweighted fit statistics that were developed by Wright & Stone (1979) and Wright & Masters (1982) for Rasch's simple logistic model and the partial credit model respectively.

For the MNSQ fit statistics we provide a ninety-five percent confidence interval for the expected value of the MNSQ (which under the null hypothesis is 1.0). If the MNSQ fit statistic lies outside that interval then we reject the null hypothesis that the data conforms to the model. If the MNSQ fit statistic lies outside the interval then the corresponding T statistics will have an absolute value that exceeds 2.0.

At the bottom of the table an item separation reliability and chi-squared test of parameter equality are reported. The separation reliability is as described in Wright & Stone (1979). This indicates how well the item parameters are separated; it has a maximum of one and a minimum of zero. This value is typically high and increases with increasing sample sizes. The null hypothesis for the chi-square test is equality of the set of parameters. In this case equality of all of the parameters is rejected because the chi-square is significant. This test is not useful here, but will be of use in other contexts, where parameter equivalence (e.g., rater severity) is of concern.

The third table in the `show` statement's output (not shown for the sake of brevity) gives the estimates of the population parameters. In this case, these are simply estimates of the mean of the latent ability distribution and of the variance of that distribution. In this case, the mean is estimated as 1.070, and the variance is estimated as 0.866.

Extension: In Rasch modelling, it is usual to identify the model by setting the mean of the item difficulty parameters to zero. This is also the default behaviour for ACER ConQuest, which automatically sets the value of the 'last' item parameter to ensure an average of zero. In ACER ConQuest, however, you can, as an alternative, choose to set the mean of the latent ability distribution to zero. To do this, use the set command as follows:

```
set lconstraints=cases;
```

If you want to use a different item as the constraining item, then you can read the items in a different order. For example:

```
format id 1-5 responses 12-15, 17-23, 16;
```

would result in the constraint being applied to the item in column 16. But be aware, it will now be called item 12, not item 5, as it is the twelfth item in the response block.

```
==============================================================================
ConQuest: Generalised Item Response Modelling Software Tue Oct 03 10:19 2006
TABLES OF RESPONSE MODEL
==============================================================================
TERM 1: item
--------------------------------------------------------------------------------
   VARIABLES                      UNWEIGHTED FIT            WEIGHTED FIT
-------------                 -----------------------  -----------------------
     item     ESTIMATE ERROR^ MNSQ       CI        T   MNSQ       CI        T
--------------------------------------------------------------------------------
 1   BSMMA01    0.364  0.050  0.87 ( 0.91, 1.09) -3.1  0.90 ( 0.94, 1.06) -3.4
 2   BSMMA02   -0.177  0.052  0.99 ( 0.91, 1.09) -0.3  0.98 ( 0.92, 1.08) -0.4
 3   BSMMA03   -0.024  0.051  0.89 ( 0.91, 1.09) -2.5  0.93 ( 0.93, 1.07) -2.1
 4   BSMMA04    0.837  0.049  0.97 ( 0.91, 1.09) -0.6  0.97 ( 0.95, 1.05) -1.2
 5   BSMMA05    1.179  0.049  1.10 ( 0.91, 1.09)  2.2  1.07 ( 0.95, 1.05)  2.9
 6   BSMMA06   -0.313  0.052  1.01 ( 0.91, 1.09)  0.3  1.01 ( 0.92, 1.08)  0.3
 7   BSMSA07   -0.391  0.053  1.09 ( 0.91, 1.09)  2.0  1.04 ( 0.92, 1.08)  0.8
 8   BSMSA08   -0.326  0.053  1.15 ( 0.91, 1.09)  3.3  1.09 ( 0.92, 1.08)  2.2
 9   BSMSA09   -0.966  0.056  0.91 ( 0.91, 1.09) -2.2  0.97 ( 0.88, 1.12) -0.5
 10  BSMSA10   -0.391  0.053  1.11 ( 0.91, 1.09)  2.4  1.06 ( 0.92, 1.08)  1.4
 11  BSMSA11   -0.499  0.053  0.91 ( 0.91, 1.09) -2.0  0.97 ( 0.91, 1.09) -0.7
 12  BSMSA12    0.707* 0.172  1.02 ( 0.91, 1.09)  0.4  1.02 ( 0.95, 1.05)  0.6
--------------------------------------------------------------------------------
```

This table provides parameter estimates for every term in the **model** *statement. In this sample analysis, we have just one term:* **item**.

The asterisk indicates that this parameter has been constrained.

Figure 2.10: The Item Parameter Estimates

This table also provides a set of reliability indices.

The fourth table in the output, Figure 2.11, provides a map of the item difficulty parameters.

The file `ex1_shw.txt` contains one additional table, labelled `Map of Latent Distributions and Thresholds`. In the case of dichotomously scored items and a model statement with a single term[5], these maps provide the same information as that shown in Figure 2.11, so they are not discussed further.

The traditional item analysis is invoked by the `itanal` statement, and its results have been written to the file `ex1_itn.txt`. The `itanal` output includes a table showing classical difficulty, discrimination, and point-biserial statistics for each item. Figure 2.12 shows the results for item 2.

Summary results, including coefficient alpha for the test as a whole, are printed at the end of the file `ex1_itn.txt` as shown in Figure 2.13. Discussion of the usage of the statistics can be found in any standard text book, such as Crocker & Algina (1986).

Figure 2.14 shows one of the 12 plots that were produced by the `plot icc` command. The ICC plot shows a comparison of the empirical item characteristic curve (the broken line, which is based directly upon the observed data) with the modelled item characteristic curve (the smooth line).

Figure 2.15 shows a matching plot produced by the `plot mcc` command. In addition to showing the modelled curve and the matching empirical curve, this plot shows the characteristics of the incorrect responses—the distractors. In particular it shows the proportion of students in each of a sequence of ten ability groupings[6] that responded with each of the possible responses.

> **TIP:** Whenever a `key` statement is used, the `itanal` statement will display results for all valid data codes. If a `key` statement is not used, the `itanal` statement will display the results of an analysis done after recoding has been applied.

2.2.4 Summary

This section shows how ACER ConQuest can be used to analyse a multiple-choice test. Some key points covered in this section are:

[5] In this case the single term was 'item'.

[6] Ten ability groupings is a default setting that can be altered.

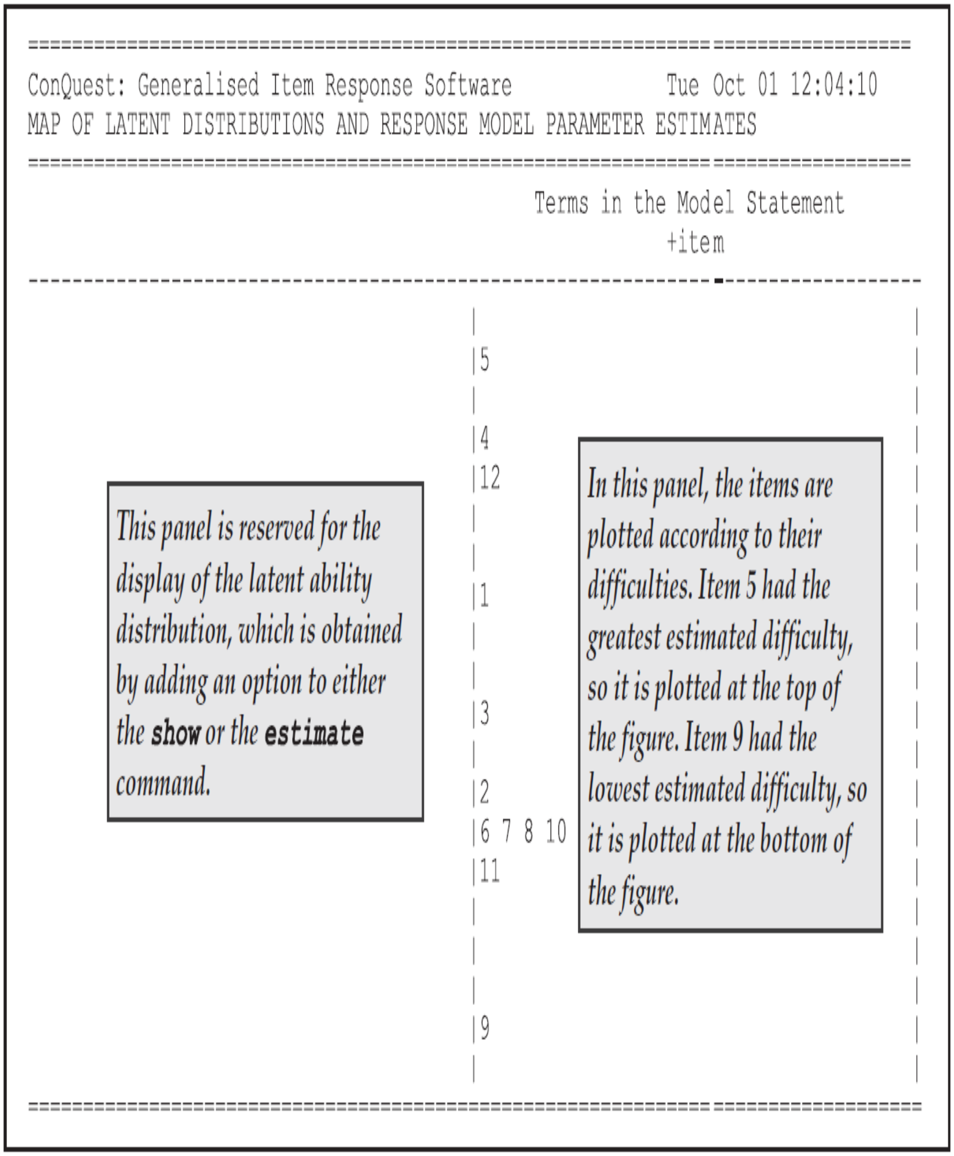


Figure 2.11: The Item and Latent Distribution Map for the Simple Logistic Model

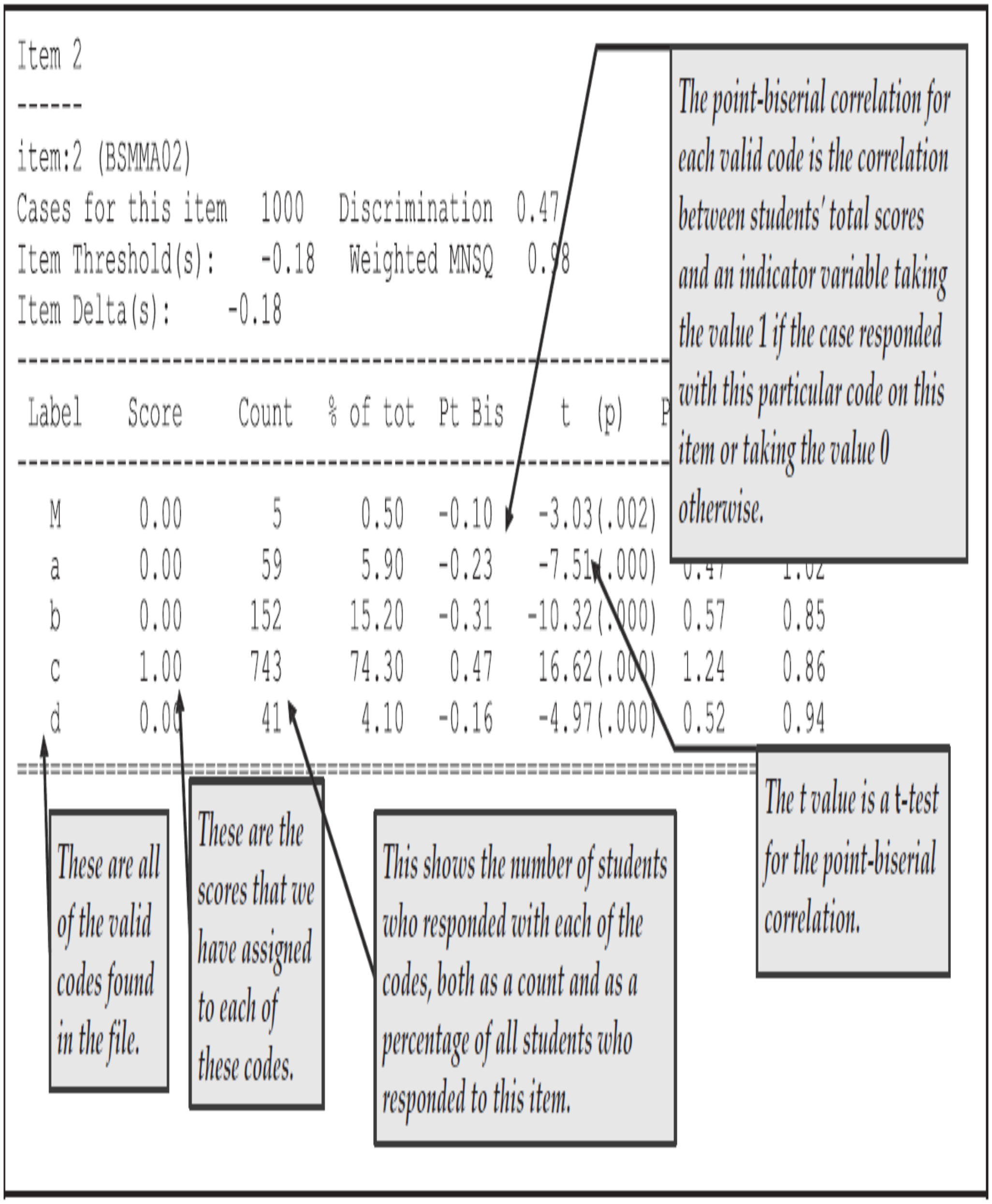


Figure 2.12: Example of Traditional Item Analysis Results

```
--------------------------------------------------------------------
The following traditional statistics are only meaningful for complete
designs and when the amount of missing data is minimal.
In this analysis  0.00%  of the data are missing.

The following results are scaled to assume that a single response
was provided for each item.

N                               1000
Mean                            8.44
Standard Deviation              2.40
Variance                        5.78
Skewness                       -0.60
Kurtosis                       -0.19
Standard error of mean          0.08
Standard error of measurement   1.43
Coefficient Alpha               0.64
====================================================================
```

Some traditional statistics are not computed if more than 10% of the response data is missing.

In the case of dichotomous data, the Coefficient Alpha is equal to KR-20.

Figure 2.13: Summary Statistics from Traditional Item Analysis Results

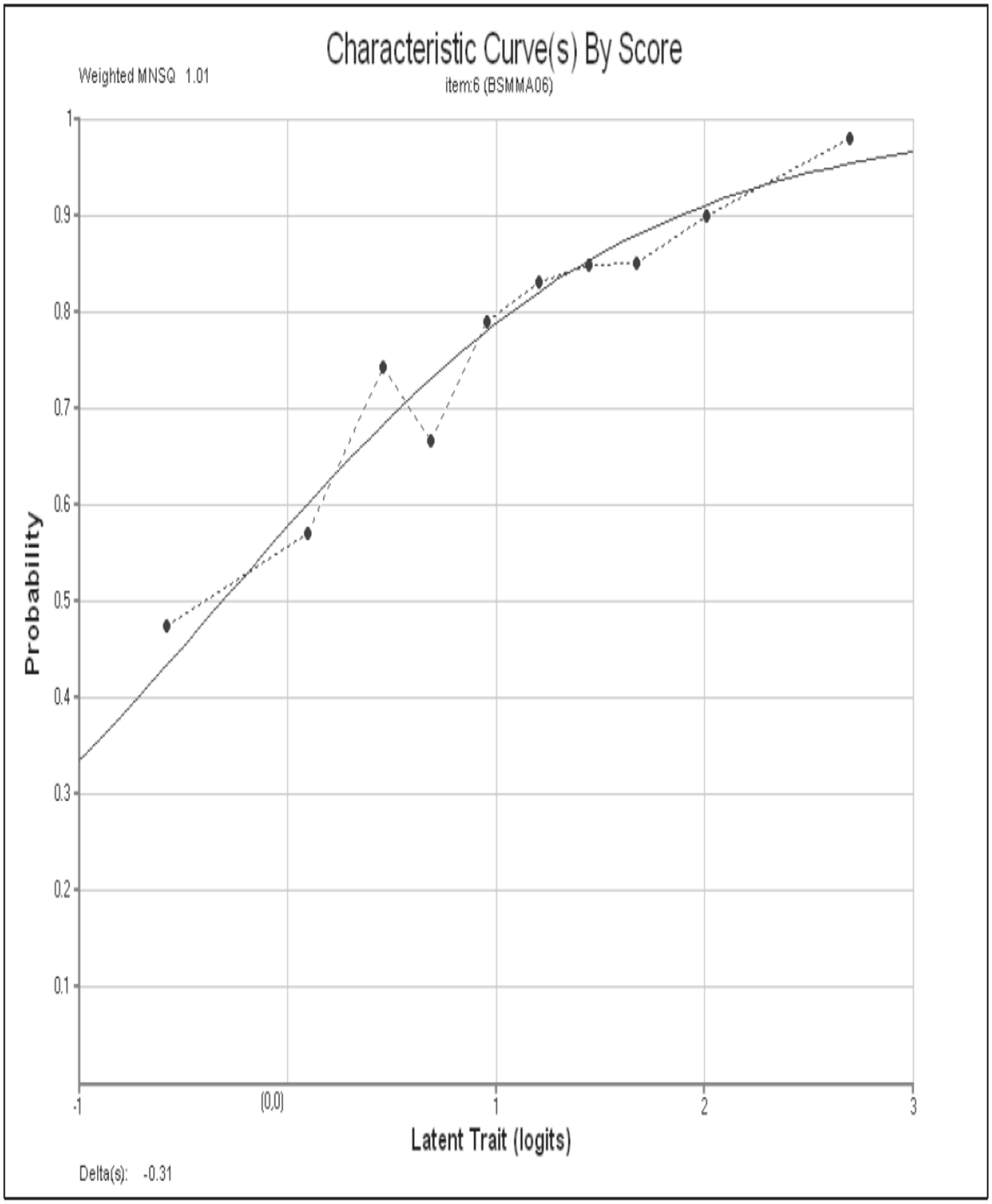


Figure 2.14: Modelled and Empirical Item Characteristic Curves for Item 6

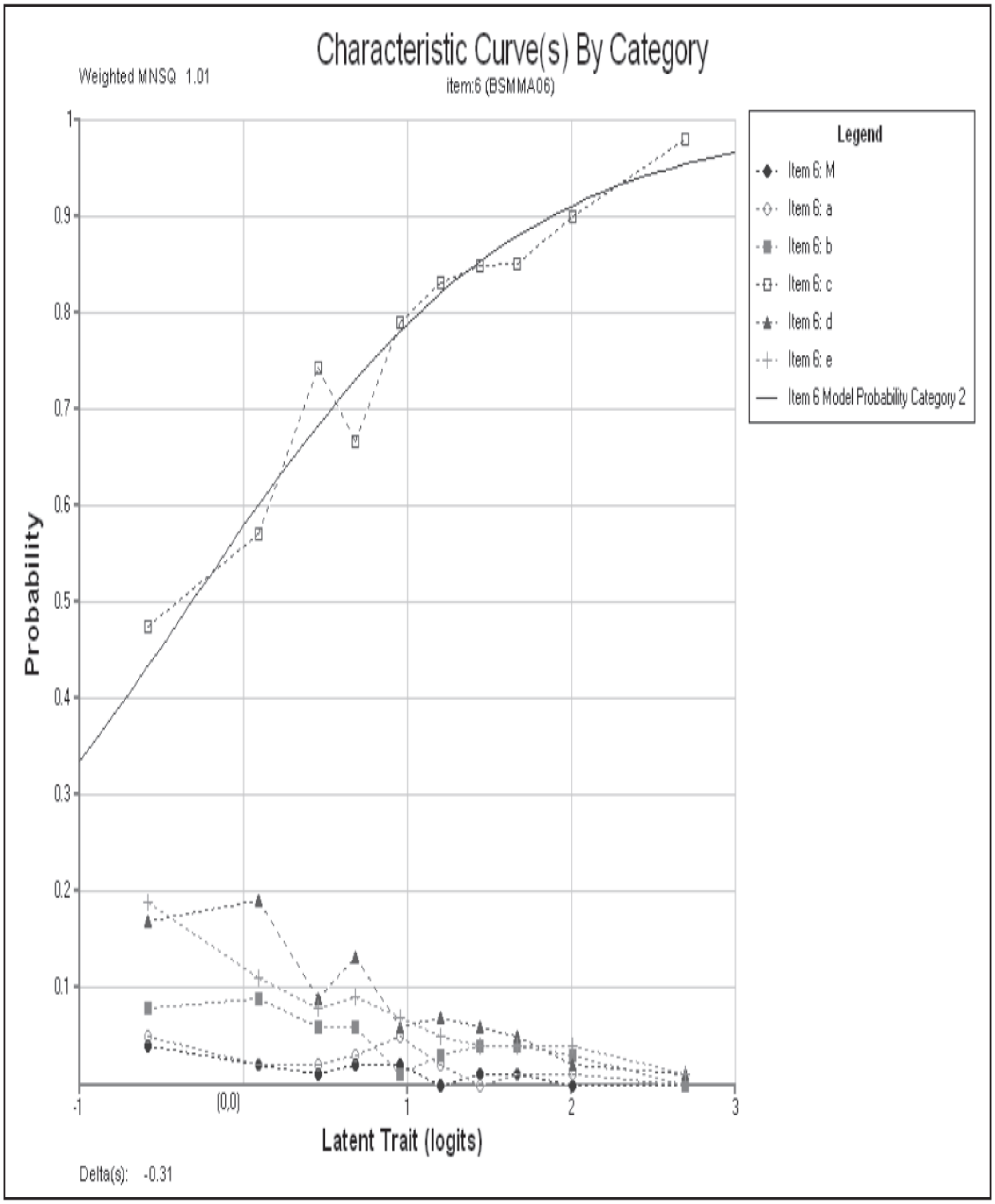


Figure 2.15: Modelled and Empirical Category Characteristics Curves for Item 6

- the `datafile`, `format` and `model` statements are prerequisites for data set analysis.
- the `key` statement provides an efficient method for scoring multiple choice tests.
- the `estimate` statement is used to fit an item response model to the data.
- the `itanal` statement generates traditional item statistics.
- the `plot` statement displays graphs which illustrate the relationship between the empirical data and the model's expectation.

EXTENSION: ACER ConQuest can fit other models to multiple choice tests, including models such as the ordered partition model.

2.3 Modelling Polytomously Scored Items with the Rating Scale and Partial Credit Models

The rating scale model (Andrich, 1978; Wright & Masters, 1982) and the partial credit model (Masters, 1982; Wright & Masters, 1982) are extensions to Rasch's simple logistic model and are suitable for use when items are scored polytomously. The rating scale model was initially developed by Andrich for use with Likert-style items, while Masters' extension of the rating scale model to the partial credit model was undertaken to facilitate the analysis of cognitive items that are scored into more than two ordered categories. In this section, the use of ACER ConQuest to fit the partial credit and rating scale models is illustrated through two sets of sample analyses. In the first, the partial credit model is fit to some cognitive items; and in the second, the fit of the rating scale and partial credit models to a set of items that forms an attitudinal scale is compared.

2.3.1 a) Fitting the Partial Credit Model

The data for the first sample analysis are the responses of 515 students to a test of science concepts related to the Earth and space. Previous analyses of some of these data are reported in Adams et al. (1991).

2.3.1.1 Required files

The files used in this sample analysis are:

filename	content
ex2a.cqc	The command statements.
ex2a_dat.txt	The data.
ex2a_lab.txt	The variable labels for the items on the partial credit test.
ex2a_shw.txt	The results of the partial credit analysis.
ex2a_itn.txt	The results of the traditional item analyses.

(The last two files are created when the command file is executed.)

The data have been entered into the file `ex2a_dat.txt`, using one line per student. A unique identification code has been entered in columns 2 through 7, and the students' response to each of the items has been recorded in columns 10 through 17. In this data, the upper-case alphabetic characters A, B, C, D, E, F, W, and X have been used to indicate the different kinds of responses that students gave to these items. The code Z has been used to indicate data that cannot be analysed. For each item, these codes are scored (or, more correctly, mapped onto performance levels) to indicate the level of quality of the response. For example, in the case of the first item (the item in column 10), the response coded A is regarded as the best kind of response and is assigned to level 2, responses B and C are assigned to level 1, and responses W and X are assigned to level 0. An extract of the file `ex2a_dat.txt` is shown in Figure 2.16.

> **NOTE:** In most Rasch-type models, a one-to-one match exists between the label that is assigned to each response category to an item (the category label) and the response level (or score) that is assigned to that response category. This need not be the case with ACER ConQuest.
>
> In ACER ConQuest, the distinction between a response category and a response level is an important one. When ACER ConQuest fits item response models, it actually models the probabilities of each of the response categories for each item. The scores for each of these categories need not be unique. For example, a four-alternative multiple choice item can be modelled as a four-response category item with three categories assigned zero scores and one category assigned a score of one, or it can be modelled in the usual fashion as a two-category item where the scores identify the categories.

2.3.1.2 Syntax

The command file used in this analysis of a Partial Credit Test is `ex2a.cqc`, which is shown in the code box below. Each line of the command file is described in the list underneath

```
          1         2
12345678901234567890123 (column numbers)

2110104ZHWBDCBBCBEABBBB
2110106ZEACDBXBCXXXXXXX
2110109ZHBWBBBWCAXAXXXX
2110113ZIWBWBXWCXXXABBB
2110115ZHWBFBBWCWXAWAXX
2110121ZHWWEBWBBCAABABA
2110123YIBWWBEWBWXABABB
2110305ZHCBABABAABACCCA
2110313YBBCFBDBCXXXXXXX

 .                .
 .                .
```

Figure 2.16: Extract from the Data File `ex2a_dat.txt`

the code box.

`ex2a.cqc`:

```
1  Title Partial Credit Model: What happened last night;
2  data ex2a_dat.txt;
3  format name 2-7 responses 10-17;
4  labels << ex2a_lab.txt;
5  codes 3,2,1,0;
6  recode (A,B,C,W,X) (2,1,1,0,0)           !items(1);
7  recode (A,B,C,W,X) (3,2,1,0,0)           !items(2);
8  recode (A,B,C,D,E,F,W,X) (3,2,2,1,1,0,0,0)!items(3);
9  recode (A,B,C,W,X) (2,1,0,0,0)           !items(4);
10 recode (A,B,C,D,E,W,X) (3,2,1,1,1,0,0)   !items(5);
11 recode (A,B,W,X) (2,1,0,0)               !items(6);
12 recode (A,B,C,W,X) (3,2,1,0,0)           !items(7);
13 recode (A,B,C,D,W,X) (3,2,1,1,0,0)       !items(8);
14 model item + item*step;
15 estimate;
16 show !estimates=latent >> results/ex2a_shw.txt;
17 itanal >> results/ex2a_itn.txt;
18 plot expected! gins=2;
19 plot icc! gins=2;
20 plot ccc! gins=2;
```

- **Line 1**
  Gives a title for this analysis. The text supplied after the command `title` will appear on the top of any printed ACER ConQuest output. If a title is not provided, the default, `ConQuest: Generalised Item Response Modelling Software`, will be used.

- **Line 2**
  Indicates the name and location of the data file. Any name that is valid for the operating system you are using can be used here.

- **Line 3**
  The `format` statement describes the layout of the data in the file `ex2a_dat.txt`. This format indicates that a field called `name` is located in columns 2 through 7 and that the responses to the items are in columns 10 through 17 (the response block) of the data file.

- **Line 4**
A set of labels for the items are to be read from the file `ex2a_lab.txt`. If you take a look at these labels, you will notice that they are quite long. ACER ConQuest labels can be of any length, but most ACER ConQuest printouts are limited to displaying many fewer characters than this. For example, the tables of parameter estimates produced by the `show` statement will display only the first 11 characters of the labels.

- **Line 5**
The `codes` statement is used to restrict the list of codes that ACER ConQuest will consider valid. In the sample analysis in section 2.2, a `codes` statement was not used. This meant that any character in the response block defined by the `format` statement — except a blank or a period (`.`) character (the default missing-response codes) — was considered valid data. In this sample analysis, the valid codes have been limited to the digits 0, 1, 2 and 3; any other codes for the items will be treated as missing-response data. It is important to note that the `codes` statement refers to the codes *after* the application of any recodes.

- **Lines 6-13**
The eight `recode` statements are used to collapse the alphabetic response categories into a smaller set of categories that are labelled with the digits `0`, `1`, `2` and `3`. Each of these `recode` statements consists of three components:

  - The first component is a list of codes contained within parentheses. These are codes that will be found in the data file `ex2a_dat.txt`, and these are called the *from* codes.
  - The second component is also a list of codes contained within parentheses, these codes are called the *to* codes. The length of the *to* codes list must match the length of the *from* codes list. When ACER ConQuest finds a response that matches a *from* code, it will change (or recode) it to the corresponding *to* code.
  - The third component (the option of the `recode` command) gives the levels of the variables for which the `recode` is to be applied. Line 11, for example, says that, for item 6, A is to be recoded to 2, B is to be recoded to 1, and W and X are both to be recoded to 0.

  Any codes in the response block of the data file that do not match a code in the *from* list will be left untouched. In these data, the Z codes are left untouched; and since Z is not listed as a valid code, all such data will be treated as missing-response data.

When ACER ConQuest models these data, the number of response categories that will be assumed for each item will be determined from the number of distinct codes for that item. Item 1 has three distinct codes (2, 1 and 0), so three categories will be modelled; item 2 has four distinct codes (3, 2, 1 and 0), so four categories will be modelled.

- **Line 14**
  The `model` statement for these data contains two terms (`item` and `item*step`) and will result in the estimation of two sets of parameters. The term `item` results in the estimation of a set of item difficulty parameters, and the term `item*step` results in a set of item step-parameters that are allowed to vary across the items. This is the partial credit model.

  In the section `[The Structure of ACER ConQuest Design Matrices]` in chapter 3, there is a description of how the terms in the `model` statement specify different versions of the item response model.

- **Line 15**
  The `estimate` statement is used to initiate the estimation of the item response model.

- **Line 16**
  The `show` statement produces a display of the item response model parameter estimates and saves them to the file `ex2a_shw.txt`. The option `estimates=latent` requests that the displays include an illustration of the latent ability distribution.

- **Line 17**
  The `itanal` statement produces a display of the results of a traditional item analysis. As with the `show` statement, the results have been redirected to a file (in this case, `ex2a_itn.txt`).

- **Lines 18-20**
  The `plot` statements produce a sequence of three displays for item 2 only. The first requested plot is a comparison of the observed and the modelled expected score curve. The second plot is a comparison of the observed and modelled item characteristics curves, and the third plot shows comparisons of the observed and expected cumulative item characteristic curves.

2.3.1.3 Running the Partial Credit Sample Analysis

To run this sample analysis, start the GUI version. Open the file `ex2a.cqc` and choose `Run`→`Run All`.

ACER ConQuest will begin executing the statements that are in the file `ex2a.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting the partial credit model to the data, and as it does so it will report on the progress of the estimation.

After the estimation is complete, the two statements that produce output (`show` and `itanal`) will be processed. As in the previous sample analysis, the `show` statement will produce six separate tables. All of these tables will be in the file `ex2a_shw.txt`. The contents of the first table were discussed in section 2.2. The first half of the second table, which contains information related to the parameter estimates for the first term in the `model` statement, is shown in Figure 2.17. The parameter estimates in this table are for the *difficulties* of each of the items. For the purposes of model identification, ACER ConQuest constrains the difficulty estimate for the last item to ensure an average difficulty of zero. This constraint has been achieved by setting the difficulty of the last item to be the negative sum of the previous items. The fact that this item is constrained is indicated by the asterisk (`*`) placed next to the parameter estimate.

Figure 2.18 shows the second table, which displays the parameter estimates, standard errors and fit statistics associated with the second term in the `model` statement, the step parameters. You will notice that the number of step parameters that has been estimated for each item is one less than the number of modelled response categories for the item. Furthermore, the last of the parameters for each item is constrained so that the sum of the parameters for an item equals zero. This is a necessary identification constraint. In the case of item 1, for example, there are three categories, 0, 1 and 2. Two values are reported, but only the first step parameter has been estimated. The second is the negative of the first. The parameter labelled as step 1, describes the transition from category 0 to 1, where the probability of being in category 1 is greater than the probability of being in category 0, while the second step describes the transition from 1 to 2. The section The Structure of ACER ConQuest Design Matrices in Chapter 3 gives a description of why an item has two fewer step parameters than it has categories, and it discusses the interpretation of these parameters.

There is a fit statistic reported for each category. This statistic provides a comparison of the expected number of students responding in the category with the observed number responding in that category.

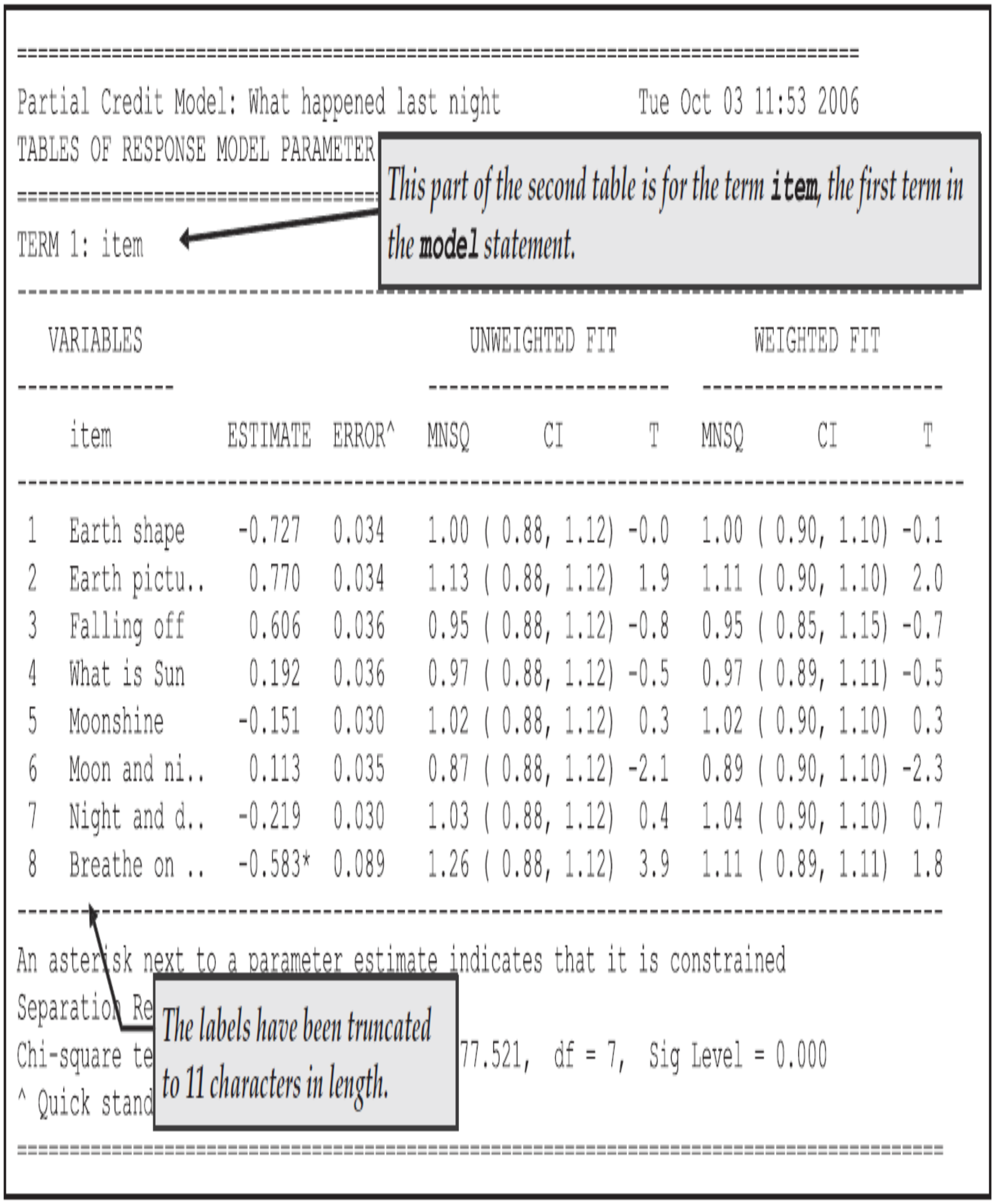

```
==============================================================================
Partial Credit Model: What happened last night          Tue Oct 03 11:53 2006
TABLES OF RESPONSE MODEL PARAMETER
=================================
TERM 1: item
------------------------------------------------------------------------------
  VARIABLES                                UNWEIGHTED FIT         WEIGHTED FIT
---------------                      ---------------------- ----------------------
    item          ESTIMATE  ERROR^  MNSQ       CI       T  MNSQ       CI       T
------------------------------------------------------------------------------
 1  Earth shape    -0.727   0.034   1.00 ( 0.88, 1.12) -0.0  1.00 ( 0.90, 1.10) -0.1
 2  Earth pictu..   0.770   0.034   1.13 ( 0.88, 1.12)  1.9  1.11 ( 0.90, 1.10)  2.0
 3  Falling off     0.606   0.036   0.95 ( 0.88, 1.12) -0.8  0.95 ( 0.85, 1.15) -0.7
 4  What is Sun     0.192   0.036   0.97 ( 0.88, 1.12) -0.5  0.97 ( 0.89, 1.11) -0.5
 5  Moonshine      -0.151   0.030   1.02 ( 0.88, 1.12)  0.3  1.02 ( 0.90, 1.10)  0.3
 6  Moon and ni..   0.113   0.035   0.87 ( 0.88, 1.12) -2.1  0.89 ( 0.90, 1.10) -2.3
 7  Night and d..  -0.219   0.030   1.03 ( 0.88, 1.12)  0.4  1.04 ( 0.90, 1.10)  0.7
 8  Breathe on ..  -0.583*  0.089   1.26 ( 0.88, 1.12)  3.9  1.11 ( 0.89, 1.11)  1.8
------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Re
Chi-square te                 77.521,  df = 7,  Sig Level = 0.000
^ Quick stand
==============================================================================
```


Figure 2.17: Parameter Estimates for the First Term in the ‘model‘ Statement

TERM 2: item*step

This part of the second table is for the term **item*step**, *the second term in the* **model** *statement.*

VARIABLES									
item	step	ESTIMATE	ERROR^	MNSQ	CI	T	MNSQ	CI	T
1 Earth shape	0			1.05	(0.88, 1.12)	0.8	1.04	(0.88, 1.12)	0.7
1 Earth shape	1	-0.234	0.095	0.97	(0.88, 1.12)	-0.5	0.98	(0.94, 1.06)	-0.6
1 Earth shape	2	0.234*		0.97	(0.88, 1.12)	-0.5	0.96	(0.93, 1.07)	-1.3
2 Earth pictu..	0			1.05	(0.88, 1.12)	0.8	1.00	(0.92, 1.08)	-0.0
2 Earth pictu..	1	-1.420	0.098	1.01	(0.88, 1.12)	0.1	1.01	(0.95, 1.05)	0.3
2 Earth pictu..	2	-0.514	0.107	1.24	(0.88, 1.12)	3.5	1.13	(0.91, 1.09)	2.5
2 Earth pictu..	3	1.934*		1.29	(0.88, 1.12)	4.2	1.01	(0.47, 1.53)	0.1
3 Falling off	0			0.84	(0.88, 1.12)	-2.6	0.89	(0.89, 1.11)	-2.1
3 Falling off	1	-2.022	0.102	0.95	(0.88, 1.12)	-0.9	0.95	(0.94, 1.06)	-1.8
3 Falling off	2	1.363	0.165	1.17	(0.88, 1.12)	2.6	1.03	(0.75, 1.25)	0.3
3 Falling off	3	0.659*		0.60	(0.88, 1.12)	-7.5	1.01	(0.58, 1.42)	0.1
4 What is Sun	0			1.03	(0.88, 1.12)	0.5	1.01	(0.92, 1.08)	0.2
4 What is Sun	1	-1.151	0.090	1.00	(0.88, 1.12)	0.1	1.00	(0.97, 1.03)	0.2
4 What is Sun	2	1.151*		0.80	(0.88, 1.12)	-3.5	0.95	(0.83, 1.17)	-0.6
5 Moonshine	0			1.07	(0.88, 1.12)	1.0	1.07	(0.91, 1.09)	1.5
5 Moonshine	1	0.308	0.094	0.97	(0.88, 1.12)	-0.4	1.00	(0.85, 1.15)	0.0
5 Moonshine	2	-0.331	0.112	1.12	(0.88, 1.12)	1.8	1.02	(0.88, 1.12)	0.4
5 Moonshine	3	0.023*		0.82	(0.88, 1.12)	-3.1	0.93	(0.90, 1.10)	-1.4
6 Moon and ni..	0			0.92	(0.88, 1.12)	-1.3	0.93	(0.92, 1.08)	-1.8
6 Moon and ni..	1	-0.748	0.090	0.97	(0.88, 1.12)	-0.4	0.98	(0.97, 1.03)	-1.6
6 Moon and ni..	2	0.748*		0.75	(0.88, 1.12)	-4.4	0.90	(0.87, 1.13)	-1.5
7 Night and d..	0	[illegible]	[illegible]	[illegible]	[illegible], 1.12)	0.6	1.02	(0.91, 1.09)	0.4
7 Night and d..	1	[illegible]	[illegible]	[illegible]	[illegible], 1.12)	-0.1	1.01	(0.85, 1.15)	0.1
7 Night and d..	2	[illegible]	[illegible]	[illegible]	[illegible], 1.12)	-0.3	1.00	(0.83, 1.17)	-0.0
7 Night and d..	3	-0.520*		1.01	(0.88, 1.12)	0.2	1.01	(0.90, 1.10)	0.2
8 Breathe on ..	0			1.23	(0.88, 1.12)	3.4	1.04	(0.89, 1.11)	0.8
8 Breathe on ..	1	1.240	0.148	1.28	(0.88, 1.12)	4.1	1.03	(0.75, 1.25)	0.3
8 Breathe on ..	2	1.735	0.337	1.11	(0.88, 1.12)	1.6	1.01	(0.36, 1.64)	0.1
8 Breathe on ..	3	-2.975*		1.23	(0.88, 1.12)	3.5	1.12	(0.89, 1.11)	2.1

These parameters are estimated for combinations of steps and items.

An asterisk next to a parameter estimate indicates that it is constrained

Figure 2.18: Parameter Estimates for the Second Term in the ‘model‘ Statement

The third table in the file (not shown here) gives the estimates of the population parameters. In this case, the mean of the latent ability distribution is –0.320, and the variance of that distribution is 0.526.

The fourth table reports the reliability coefficients. Three different reliability statistics are available (Adams, 2005). In this case just the third index (the EAP/PV reliability) is reported because neither of the maximum likelihood estimates has been computed at this stage. The reported reliability is 0.735.

The fifth table Figure 2.19 is a map of the parameter estimates and latent ability distribution. For this model, the map consists of two panels, one for the latent ability distribution and one for each of the terms in the `model` statement that do not include a *step* (in this case one). In this case the leftmost panel shows the estimated latent ability distribution and the second shows the item difficulties.

> **EXTENSION:** The headings of the panels in Figure 2.19 are preceded by a plus sign (`+`). This indicates the orientation of the parameters. A plus indicates that the facet is modelled with difficulty parameters, whereas a minus sign (`-`) indicates that the facet is modelled with easiness parameters. This is controlled by the sign that you use in the `model` statement.

Figure 2.20, the sixth table from the file `ex2a_shw.txt`, is a plot of the Thurstonian thresholds for the items. The definition of these thresholds is discussed in Computing Thresholds in Chapter 3. Briefly, they are plotted at the point where a student has a 50% chance of achieving at least the indicated level of performance on an item.

The `itanal` command in line 17 produces a file (`ex2a_itn.txt`) that contains traditional item statistics (Figure 2.21). In the previous section a multiple-choice test was analysed and the `itanal` output for multiple-choice items was described. In this example a `key` statement was not used and the items use partial credit scoring. As a consequence the `itanal` results are provided at the level of scores, rather than response categories.

> **EXTENSION:** The method used to construct the ability distribution is determined by the `estimates=` option used in the `show` statement. The `latent` distribution is constructed by drawing a set of plausible values for the students and constructing a histogram from the plausible values. Other options for the distribution are `EAP`, `WLE` and `MLE`, which result in histograms of expected a-posteriori, weighted maximum likelihood and maximum likelihood estimates, respectively. Details of these ability estimates are discussed in Latent Estimation and Prediction in Chapter 3.

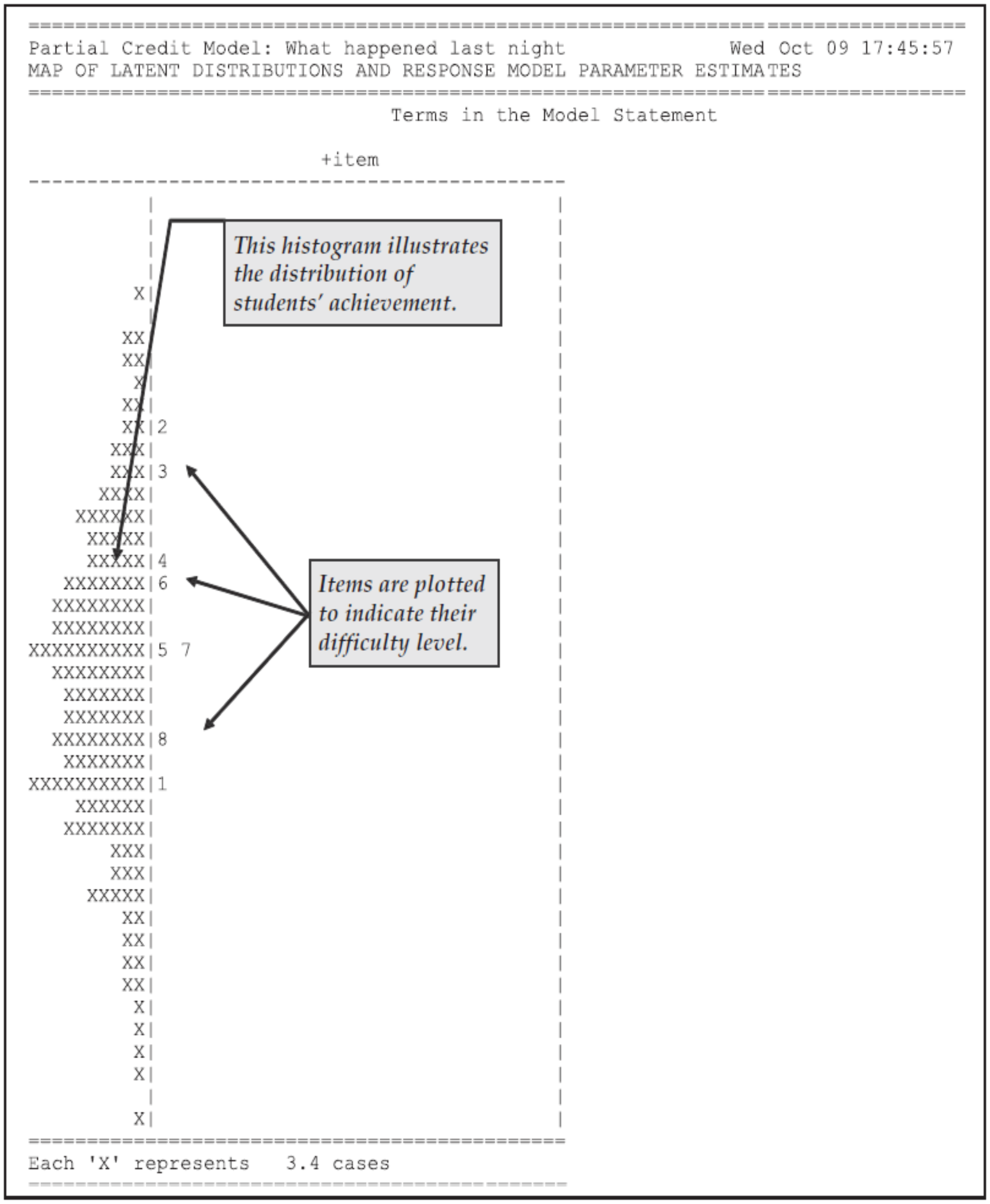


Figure 2.19: The Item and Latent Distribution Map for the Partial Credit Model

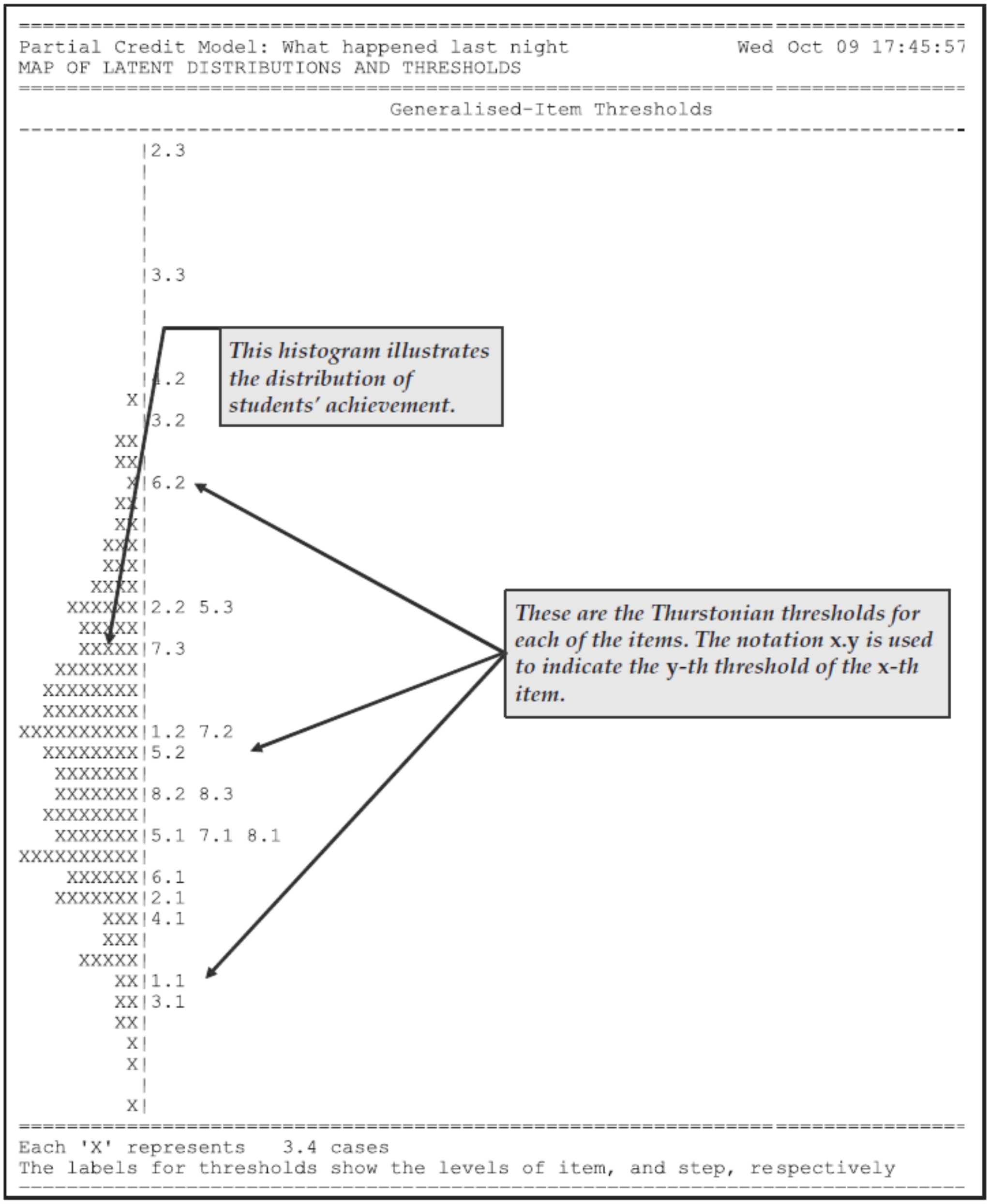
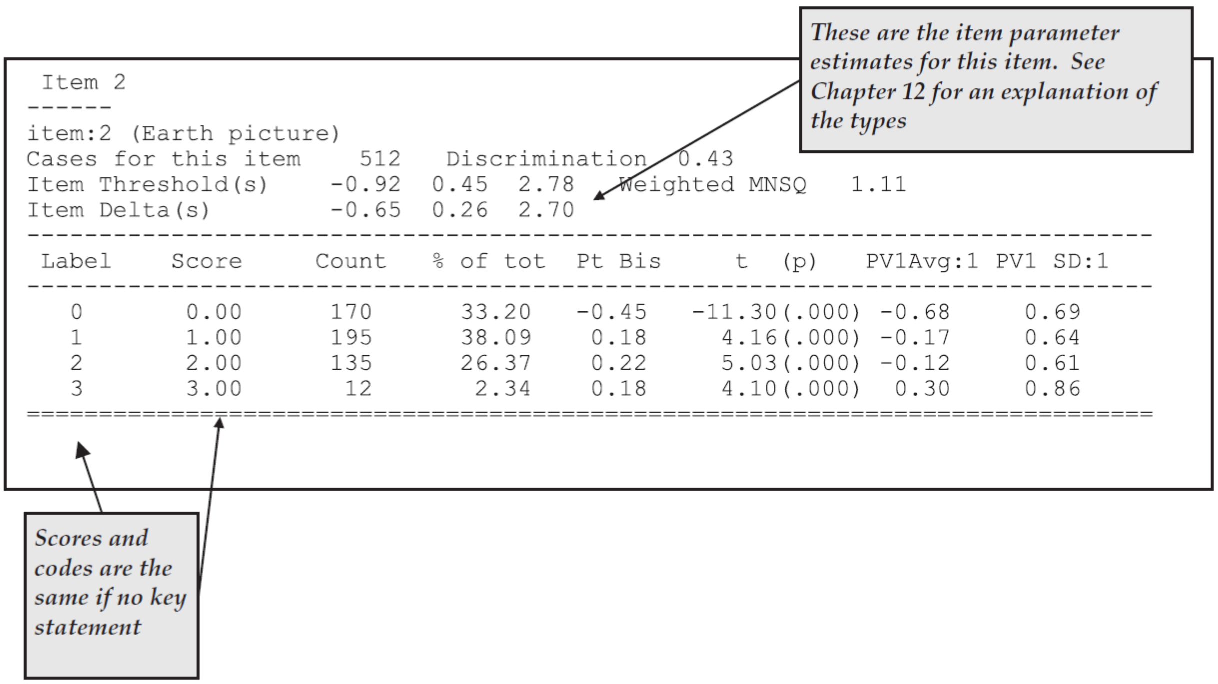


Figure 2.20: Item Thresholds for the Partial Credit Model



```
 Item 2
------
item:2 (Earth picture)
Cases for this item    512   Discrimination  0.43
Item Threshold(s)    -0.92  0.45  2.78   Weighted MNSQ   1.11
Item Delta(s)        -0.65  0.26  2.70
------------------------------------------------------------------------------
 Label     Score     Count   % of tot  Pt Bis     t  (p)    PV1Avg:1 PV1 SD:1
------------------------------------------------------------------------------
   0        0.00      170      33.20   -0.45   -11.30(.000) -0.68     0.69
   1        1.00      195      38.09    0.18     4.16(.000) -0.17     0.64
   2        2.00      135      26.37    0.22     5.03(.000) -0.12     0.61
   3        3.00       12       2.34    0.18     4.10(.000)  0.30     0.86
==============================================================================
```


Figure 2.21: Extract of Item Analysis Printout for a Partial Credit Item

The three plot commands (lines 18–20) produce the graphs shown in Figure 2.22. For illustrative purposes only plots for item 2 are requested. This item showed poor fit to the scaling model — in this case the partial credit model.

The item fit MNSQ of 1.11 indicates that this item is less discriminating than expected by the model. The first plot, the comparison of the observed and modelled expected score curves is the best illustration of this misfit. Notice how in this plot the observed curve is a little flatter than the modelled curve. This will often be the case when the MNSQ is significantly larger than 1.0.

The second plot shows the item characteristic curves, both modelled and empirical. There is one pair of curves for each possible score on the item, in this case 0, 1, 2 and 3. Note that the disparity between the observed and modelled curves for category 2 is the largest and this is consistent with the high fit statistic for this category.

The third plot is a cumulative form of the item characteristic curves. In this case three pairs of curves are plotted. The rightmost pair gives the probability of a response of 3, the next pair is for the probability of 2 or 3, and the final pairing is for the probability of 1, 2 or 3. Where these curves attain a probability of 0.5, the value on the horizontal axis corresponds to each of the three threshold parameters that are reported under the figure.

2.3.2 b) Partial Credit and Rating Scale Models: A Comparison of Fit

A key feature of ACER ConQuest is its ability to fit alternative Rasch-type models to the same data set. Here a rating scale model and a partial credit model are fit to a set of items that were designed to measure the importance placed by teachers on adequate resourcing and support to the success of bilingual education programs.

2.3.2.1 Required files

The data come from a study undertaken by Zammit (1997). The data consist of the responses of 582 teachers to the 10 items listed in Figure 2.23. Each item was presented with a Likert-style response format; and in the data file, strongly agree was coded as 1, agree as 2, uncertain as 3, disagree as 4, and strongly disagree as 5.

The files that we use are:

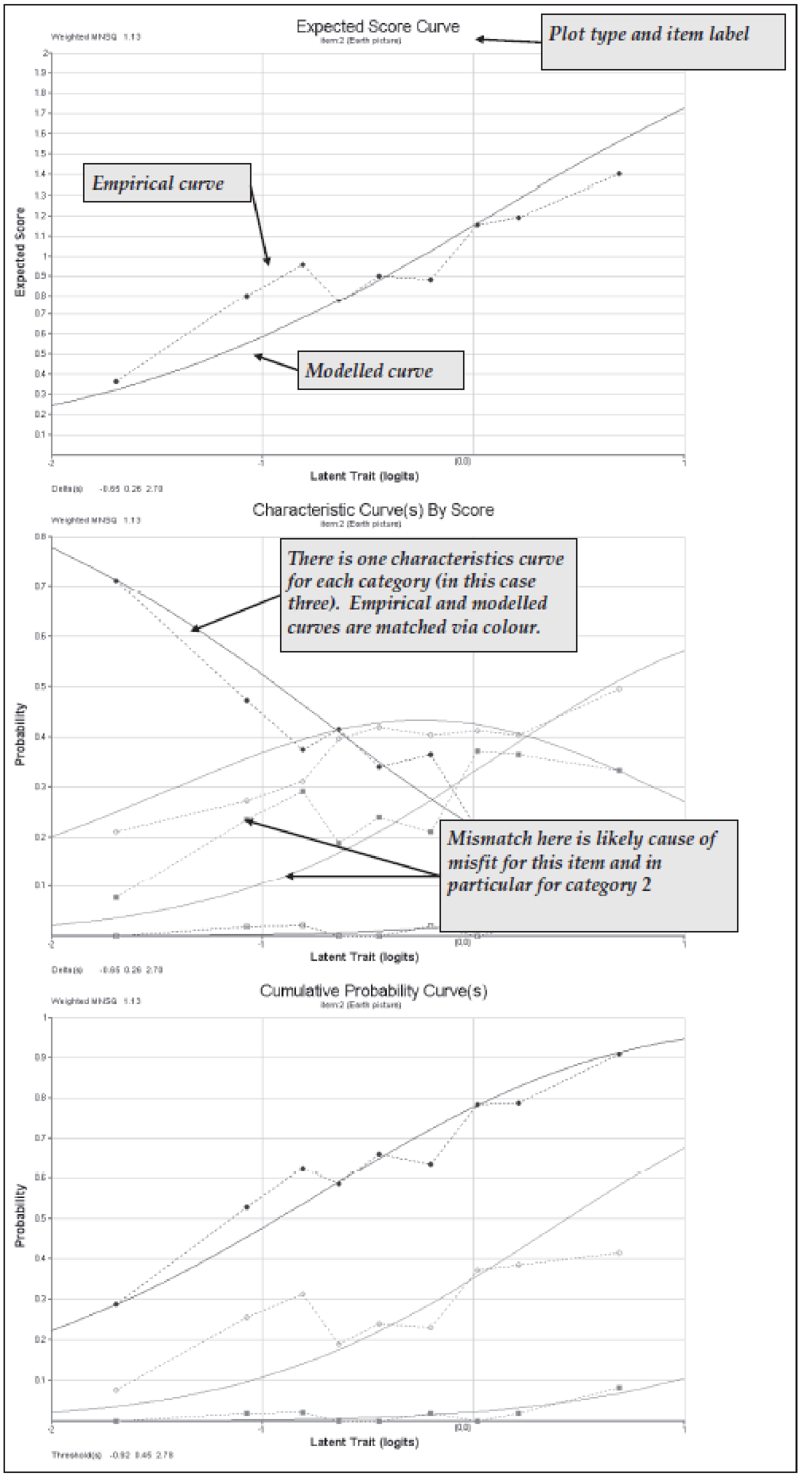


Figure 2.22: Plots for Item 2

1. A bilingual teaching program is only successful when curriculum materials are adequate.
2. A bilingual teaching program should not be implemented before there are enough bilingual teachers.
3. A bilingual teaching program requires adequate resources.
4. The staff's commitment to bilingual teaching is essential if bilingual teaching is to succeed.
5. The staff's commitment to bilingual teaching is very important in improving the students' achievement in school.
6. A bilingual program must last several years for students to achieve basic bilingual proficiency.
7. To achieve maximum success all staff should be convinced that the bilingual teaching program has achievable goals.
8. The principal's commitment to bilingual teaching is very important in improving students' achievement in school.
9. A bilingual teaching program cannot succeed without the active support of parents.
10. Bilingual teaching requires a high level of motivation from students.

Figure 2.23: Items Used in the Comparison of the Rating Scale and the Partial Credit Models

filename	content
ex2b.cqc	The command statements.
ex2b_dat.txt	The data.
ex2b_lab.txt	The variable labels for the items on the rating scale.
ex2b_shw.txt	The results of the rating scale analysis.
ex2b_itn.txt	The results of the traditional item analyses.
ex2c_shw.txt	The results of the partical credit analysis.

(The last three files are created when the command file is executed.)

2.3.2.2 Syntax

The code box below contains the contents of `ex2b.cqc`. This is the command file used in this analysis to fit a Rating Scale and then a Partial Credit Model to the same data we used in part a) of this tutorial. The list underneath the code box explains each line from the command file.

`ex2b.cqc`:

```
title Rating Scale Analysis;
datafile ex2b_dat.txt;
format responses 9-15,17-19;
codes 0,1,2;
recode (1,2,3,4,5) (2,1,0,0,0);
labels << ex2b_lab.txt;
model item + step;              /*Rating Scale*/
estimate;
show>>results/ex2b_shw.txt;
itanal>>results/ex2b_itn.txt;
reset;
title Partial Credit Analysis;
datafile ex2b_dat.txt;
format responses 9-15,17-19;
codes 0,1,2;
recode (1,2,3,4,5) (2,1,0,0,0);
labels << ex2b_lab.txt;
model item + item*step;         /*Partial Credit*/
```

```
19 estimate;
20 show>>results/ex2c_shw.txt;
```

- **Line 1**
  For this analysis, we are using the title `Rating Scale Analysis`.

- **Line 2**
  The data for this sample analysis are to be read from the file `ex2b_dat.txt`.

- **Line 3**
  The `format` statement describes the layout of the data in the file `ex2b_dat.txt`. This format indicates that the responses to the first seven items are located in columns 9 through 15 and that the responses to the next three items are located in columns 17 through 19.

- **Line 4**
  The valid codes, after recode, are 0, 1 and 2.

- **Line 5**
  The original codes of 1, 2, 3, 4, and 5 are recoded to 2, 1, and 0. Because 3, 4, and 5 are all being recoded to 0, this means we are collapsing these categories (uncertain, disagree, and strongly disagree) for the purposes of this analysis.

- **Line 6**
  A set of labels for the items is to be read from the file `ex2b_lab.txt`.

- **Line 7**
  This is the `model` statement that corresponds to the rating scale model. The first term in the `model` statement indicates that an item difficulty parameter is modelled for each item, and the second indicates that step parameters are the same for all items.

- **Line 8**
  The `estimate` statement is used to initiate the estimation of the item response model.

- **Line 9**
  Item response model results are to be written to the file `ex2b_shw.txt`.

- **Line 10**
  Traditional statistics are to be written to the file `ex2b_itn.txt`.

- **Line 11**
  The `reset` statement can be used to separate jobs that are put into a single command file. The `reset` statement returns all values to their defaults. Even though many values are the same for these analyses, we advise resetting, as you may be unaware of some values that have been set by the previous statements.

- **Lines 12-20**
  These lines replicate lines 1 to 9. The only difference is in the `model` statement (compare lines 18 and 7). In the first analysis, the second term of the `model` statement is `step`, whereas in the second analysis the second term is `item*step`. In the latter case, the step structure is allowed to vary across items, whereas in the first case, the step structure is constrained to be the same across items.

2.3.2.3 Running the Comparison of the Rating Scale and Partial Credit Models

To run this sample analysis, launch the GUI version of ACER ConQuest and open the command file `ex2b.cqc` and choose `Run`→`Run All`.

ACER ConQuest will begin executing the statements that are in the file `ex2b.cqc`; and as they are executed, they will be echoed on the screen. Firstly the rating scale model will be fit, followed by the partial credit model.

To compare the fit of the two models to these data, two tables produced by the `show` statements for each model are compared. First, the summary tables for each model are compared. These two tables are reproduced in Figure 2.24. From these tables we note that the rating scale model has used 12 parameters, and the partial credit model has used 21 parameters. For the rating scale model, the parameters are the mean and variance of the latent variable, nine item difficulty parameters, and a single step parameter. For the partial credit model, the parameters are the mean and variance of the latent variable, nine item difficulty parameters and 10 step parameters.

A formal statistical test of the relative fit of these models can be undertaken by comparing the deviance of the two models. Comparing the deviance in the summary tables, note that the rating scale model deviance is 67.58 greater than the deviance for the partial credit model. If this value is compared to a chi-squared distribution with 9 degrees of freedom, this value is significant and it can be concluded that the fit of the rating scale model is significantly worse than the fit of the partial credit model.

The difference in the fit of these two models is highlighted by comparing the contents of Figures 2.25 and 2.26.

```
================================================================================
Rating Scale Analysis                                       Tue Oct 03 16:03 2006
SUMMARY OF THE ESTIMATION
================================================================================

Estimation method was: Gauss-Hermite Quadrature with 15 nodes
Assumed population distribution was: Gaussian
Constraint was: DEFAULT
The Data File: ex2b.dat
The format:  responses 9-15,17-19
The regression model:
Grouping Variables:
The item model: item+step
Sample size: 582
Final Deviance:     9340.18030
Total number of estimated parameters: 12
The number of iterations: 23

================================================================================
Partial Credit Analysis                                     Tue Oct 03 16:02 2006
SUMMARY OF THE ESTIMATION
================================================================================

Estimation method was: Gauss-Hermite Quadrature with 15 nodes
Assumed population distribution was: Gaussian
Constraint was: DEFAULT
The Data File: ex2b.dat
The format:  responses 9-15,17-19
The regression model:
Grouping Variables:
The item model: item+item*step
Sample size: 582
Final Deviance:     9272.59748
Total number of estimated parameters: 21
The number of iterations: 27
```

The difference in the deviance is 67.58282

Figure 2.24: Summary Information for the Rating Scale and Partial Credit Analyses

Figure 2.25 shows that, in the case of the rating scale model, the step parameter fits poorly, whereas in Figure 2.26 the fit statistics for the step parameters are generally small or less than their expected value (ie the t-values are negative). In both cases, the difficulty parameter for item 2 does not fit well. An examination of the text of this item in Figure 2.23 shows that perhaps the misfit of this item can be explained by the fact that it is slightly different to the other questions in that it focuses on the conditions under which a bilingual program should be started rather than on the conditions necessary for the success of a bilingual program. Thus, although overall the partial credit model fits better than the rating scale model as discussed previously, the persistence of misfit for the difficulty parameter for this item indicates that the inclusion of this item in the scale should be reconsidered.

2.3.3 Summary

In this section, ACER ConQuest has been used to fit partial credit and rating scale models. Some key points covered were:

- The `codes` statement can be used to provide a list of valid codes.
- The `recode` statement is used to change the codes that are given in the response block (defined in the `format` statement) for the data file.
- The number of response categories modelled by ACER ConQuest for each item is the number of unique codes (after recoding) for that item.
- Response categories and item scores are *not* the same thing.
- The `model` statement can be used to fit different models to the same data.
- The deviance statistic can be used to choose between models.

2.4 The Analysis of Rater Effects

The item response models, such as simple logistic, rating scale and partial credit, that have been illustrated in the previous two sections, assume that the observed responses result from the two-way interaction between the agents of measurement[7] and the objects of measurement.[8] With the increasing importance of performance assessment, Linacre

[7]The agents of measurement are the tools that are used to stimulate responses. They are typically test items or, more generally, assessment tasks.

[8]The object of measurement is the entity that is to be measured, most commonly a student, a candidate or a research subject.

```
================================================================================
Rating Scale Analysis                                       Tue Oct 03 16:03 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
================================================================================
TERM 1: item
------------------------------------------------------------------------------------------
   VARIABLES                          UNWEIGHTED FIT                         T
---------------                       -----------------------   -----          -------
     item           ESTIMATE  ERROR^  MNSQ        CI         T    MNSQ                  T
------------------------------------------------------------------------------------------
 1   Curriculum ..    0.716   0.054   1.12 ( 0.88, 1.12)  1.9   1.12 ( 0.89, 1.11)  2.1
 2   Not Until E..    1.061   0.054   1.47 ( 0.88, 1.12)  6.8   1.51 ( 0.89, 1.11)  8.2
 3   Financial R..   -0.559   0.056   0.83 ( 0.88, 1.12) -3.1   0.88 ( 0.88, 1.12) -2.1
 4   Staff Commi..   -1.046   0.057   0.73 ( 0.88, 1.12) -5.0   0.79 ( 0.88, 1.12) -3.7
 5   Commitment ..   -0.425   0.055   0.94 ( 0.88, 1.12) -1.0   0.94 ( 0.89, 1.11) -1.1
 6   Run for som..   -0.009   0.054   1.05 ( 0.88, 1.12)  0.9   1.06 ( 0.89, 1.11)  1.1
 7   Achievable ..   -0.386   0.055   0.80 ( 0.88, 1.12) -3.6   0.78 ( 0.89, 1.11) -4.0
 8   Principals ..   -0.133   0.055   0.87 ( 0.88, 1.12) -2.2   0.87 ( 0.89, 1.11) -2.3
 9   Parents sup..    0.232   0.054   1.06 ( 0.88, 1.12)  1.1   1.07 ( 0.89, 1.11)  1.3
 10  Student mot..    0.548*  0.165   1.00 ( 0.88, 1.12)  0.1   0.98 ( 0.89, 1.11) -0.4
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.993
Chi-square test of parameter equality =    1127.17,  df = 9,  Sig Level = 0.000
^ Quick standard errors have been used
================================================================================
TERM 2: step
------------------------------------------------------------------------------------------
   VARIABLES                          UNWEIGHTED FIT                 HTED FIT
---------------                       ----------------                    ------------
     step           ESTIMATE  ERROR^  MNSQ        CI                     CI           T
------------------------------------------------------------------------------------------
       0                              2.59 ( 0.88, 1.12) 18.8   1.75 ( 0.85, 1.15)  8.0
       1             -1.184   0.029   2.39 ( 0.88, 1.12) 17.0   2.55 ( 0.88, 1.12) 18.7
       2              1.184*          1.51 ( 0.88, 1.12)  7.4   1.58 ( 0.88, 1.12)  7.9
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
^ Quick standard errors have been used
==========================================================================================
==================================================================
```

Item 2 fits badly.

The step parameters fit badly.

Figure 2.25: Response Model Parameter Estimates for the Rating Scale Model

```
================================================================================
Partial Credit Analysis                                 Tue Oct 03 16:02 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
================================================================================
TERM 1: item
------------------------------------------------------------------------------------
   VARIABLES                           UNWEIGHTED FIT            WEIGHTED FIT
---------------                   -----------------------   -----------------------
     item          ESTIMATE  ERROR^   MNSQ      CI        T    MNSQ      CI        T
------------------------------------------------------------------------------------
 1   Curriculum ..    0.753   0.055   1.11 ( 0.88, 1.12)  1.8   1.10 ( 0.89, 1.11)  1.8
 2   Not Until E..    1.068   0.053   1.41 ( 0.88, 1.12)  6.0   1.37 ( 0.89, 1.11)  6.0
 3   Financial R..   -0.524   0.058   0.82 ( 0.88, 1.12) -3.2   0.87 ( 0.88, 1.12) -2.3
 4   Staff Commi..   -1.174   0.060   0.76 ( 0.88, 1.12) -4.3   0.85 ( 0.88, 1.12) -2.7
 5   Commitment ..   -0.389   0.057   0.95 ( 0.88, 1.12) -0.9   0.95 ( 0.89, 1.11) -0.9
 6   Run for som..    0.067   0.055   1.03 ( 0.88, 1.12)  0.6   1.02 ( 0.89, 1.11)  0.3
 7   Achievable ..   -0.462   0.058   0.84 ( 0.88, 1.12) -2.7   0.86 ( 0.89, 1.11) -2.6
 8   Principals ..   -0.165   0.057   0.91 ( 0.88, 1.12) -1.5   0.94 ( 0.89, 1.11) -1.1
 9   Parents sup..    0.275   0.056   1.07 ( 0.88, 1.12)  1.2   1.07 ( 0.89, 1.11)  1.3
 10  Student mot..    0.550*  0.170   1.06 ( 0.88, 1.12)  1.0   1.06 ( 0.89, 1.11)  1.1
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.993
Chi-square test of parameter equality =    1199.35,  df = 9,  Sig Level = 0.000
^ Quick standard errors have been used
================================================================================
TERM 2: item*step
------------------------------------------------------------------------------------
        VARIABLES                           UNWEIGHTED FIT            WEIGHTED FIT
------------------------               -----------------------   -----------------------
   item           step  ESTIMATE   ERROR^ MNSQ      CI        T    MNSQ      CI        T
------------------------------------------------------------------------------------
 1 Curriculum ..    0                     2.03 ( 0.88, 1.12) 13.3   1.18 ( 0.89, 1.11)  3.0
 1 Curriculum ..    1    -1.129    0.090  0.99 ( 0.88, 1.12) -0.1   1.00 ( 0.95, 1.05)  0.0
 1 Curriculum ..    2     1.129*          0.80 ( 0.88, 1.12) -3.5   0.95 ( 0.89, 1.11) -0.9
 2 Not Until E..    0                     2.25 ( 0.88, 1.12) 15.4   1.40 ( 0.90, 1.10)  7.1
 2 Not Until E..    1    -0.626    0.093  1.04 ( 0.88, 1.12)  0.7   1.04 ( 0.94, 1.06)  1.3
 2 Not Until E..    2     0.626*          1.08 ( 0.88, 1.12)  1.2   1.08 ( 0.89, 1.11)  1.4
 3 Financial R..    0                     1.39 ( 0.88, 1.12)  5.8   0.98 ( 0.78, 1.22) -0.1
 3 Financial R..    1    -1.184    0.096  0.80 ( 0.88, 1.12) -3.5   0.87 ( 0.93, 1.07) -3.9
 3 Financial R..    2     1.184*          0.73 ( 0.88, 1.12) -5.0   0.83 ( 0.91, 1.09) -3.9
 4 Staff Commi..    0                     2.56 ( 0.88, 1.12) 18.5   0.91 ( 0.66, 1.34) -0.5
 4 Staff Commi..    1    -1.463    0.099  0.73 ( 0.88, 1.12) -5.0   0.84 ( 0.92, 1.08) -4.4
 4 Staff Commi..    2     1.463*          0.73 ( 0.88, 1.12) -5.0   0.83 ( 0.91, 1.09) -4.0
 5 Commitment ..    0                     3.18 ( 0.88, 1.12) 23.6   1.06 ( 0.80, 1.20)  0.6
 5 Commitment ..    1    -1.188    0.094  0.79 ( 0.88, 1.12) -3.8   0.85 ( 0.94, 1.06) -5.0
 5 Commitment ..    2     1.188*          0.78 ( 0.88, 1.12) -4.0   0.85 ( 0.91, 1.09) -3.5
 6 Run for som..    0                     2.08 ( 0.88, 1.12) 13.9   1.11 ( 0.85, 1.15)  1.4
 6 Run for som..    1    -1.018    0.092  0.89 ( 0.88, 1.12) -1.8   0.93 ( 0.95, 1.05) -2.7
 6 Run for som..    2     1.018*          1.00 ( 0.88, 1.12)  0.0   0.91 ( 0.90, 1.10) -2.0
 7 Achievable ..    0                     3.54 ( 0.88, 1.12) 26.2   0.97 ( 0.77, 1.23) -0.2
 7 Achievable ..    1    -1.457    0.094  0.80 ( 0.88, 1.12) -3.6   0.85 ( 0.94, 1.06) -4.7
 7 Achievable ..    2     1.457*          0.75 ( 0.88, 1.12) -4.6   0.82 ( 0.91, 1.09) -4.2
 8 Principals ..    0                     1.53 ( 0.88, 1.12)  7.6   1.05 ( 0.81, 1.19)  0.5
 8 Principals ..    1    -1.395    0.093  0.86 ( 0.88, 1.12) -2.4   0.89 ( 0.94, 1.06) -3.8
 8 Principals ..    2     1.395*          0.79 ( 0.88, 1.12) -3.8   0.86 ( 0.91, 1.09) -3.1
 9 Parents sup..    0                     1.67 ( 0.88, 1.12)  9.3   1.14 ( 0.86, 1.14)  1.9
 9 Parents sup..    1    -1.143    0.091  0.96 ( 0.88, 1.12) -0.7   0.97 ( 0.95, 1.05) -1.2
 9 Parents sup..    2     1.143*          0.91 ( 0.88, 1.12) -1.5   0.96 ( 0.90, 1.10) -0.8
 10Student mot..    0                     1.90 ( 0.88, 1.12) 11.9   1.08 ( 0.86, 1.14)  1.1
 10Student mot..    1    -1.498    0.091  0.97 ( 0.88, 1.12) -0.5   0.97 ( 0.95, 1.05) -1.0
 10Student mot..    2     1.498*          0.97 ( 0.88, 1.12) -0.5   0.98 ( 0.89, 1.11) -0.4
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
^ Quick standard errors have been used
================================================================================
```

Item 2 fits badly.

Fit looks reasonable as compared to the fit for the rating scale model.

Figure 2.26: Response Model Parameter Estimates for the Partial Credit Model

(1994) recognised that the responses that are gathered in many contexts do not result from the interaction between an object and a single agent: the agent is often a composite of more fundamental subcomponents.[9] Consider, for example, the assessment of writing, where a stimulus is presented to a student, the student prepares a piece of writing, and then a rater makes a judgment about the quality of the writing performance. Here, the object of measurement is clearly the student; but the agent is a combination of the rater who makes the judgment and the stimulus that serves as a prompt for the student's writing. The response that is analysed by the item response model is influenced by the characteristics of the student, the characteristics of the stimulus, and the characteristics of the rater. Linacre (1994) would label this a three-faceted measurement context, the three facets being the student, the stimulus and the rater.

Using an extension of the partial credit model to this multifaceted context, Linacre (1994) and others have shown that item response models can be used to identify raters who are harsher or more lenient than others, who exhibit different patterns in the way they use rating schemes, and who make judgments that are inconsistent with judgments made by other raters. This section describes how ACER ConQuest can fit a multifaceted measurement model to analyse the characteristics of a set of 16 raters who have rated a set of writing tasks using two criteria.

2.4.1 a) Fitting a Multifaceted Model

2.4.1.1 Required files

The data that we are analysing are the ratings of 8296 Year 6 students' responses to a single writing task. The data were gathered as part of a study reported in Congdon & McQueen (1997). Each of the 8296 students' writing scripts was graded by two raters, randomly chosen from a set of 16 raters; and the second rating for each script was performed blind. The random allocation of scripts to the raters, in conjunction with the very large number of scripts, resulted in links between all raters being obtained. When assessing the scripts, each rater was required to provide two ratings, one labelled OP (overall performance) and the other TF (textual features).[10] The rating of both the OP and TF was undertaken

[9] Fischer (1973) recognised that items could be described by more fundamental parameters when he proposed the linear logistic test model. Linacre (1994) extended the model to the polytomous case and recognised that the more fundamental components could be raters and such.

[10] OP (overall performance) is a judgment of the task fulfilment, particularly in terms of appropriateness for purpose and audience, conceptual complexity, and organisation of the piece. TF (textual features) focuses on control and effective use of syntactic features, such as cohesion, subordination, and verb forms, and other linguistic features, such as spelling and punctuation.

against a sixpoint scale, with the labels G, H, I, J, K and L used to indicate successively superior levels of performance. For a small number of scripts, ratings of this nature could not be made; and the code N was used to indicate this occurrence.

The files used in this sample analysis are:

filename	content
ex3a.cqc	The command statements.
ex3_dat.txt	The data.
ex3a_shw.txt	The results of the multifaceted analysis.
ex3a_itn.txt	The results of the traditional item analyses.

(The last two files are created when the command file is executed.)

The data were entered into the file `ex3_dat.txt`, using one line per student. Rater identifiers (of two characters in width) for the first and second raters who rated the writing of each student are entered in columns 17 and 18 and columns 19 and 20, respectively. Each of the two raters produced an OP and a TF rating for the script. The OP and TF ratings made by the first rater have been entered in columns 21 and 22, and the OP and TF ratings made by the second rater have been entered in columns 25 and 26.

2.4.1.2 Syntax

`ex3a.cqc` is the command file used in this tutorial for fitting one possible multifaceted model to the data outlined above. The command file is shown in the code box below, and the list underneath the code box analyzes each line of syntax.

`ex3a.cqc`:

```
Title Rater Effects Model One;
datafile ex3_dat.txt;
format rater 17-18 rater 19-20
       responses 21-22 responses 25-26 ! criteria(2);
codes G,H,I,J,K,L;
score (G,H,I,J,K,L) (0,1,2,3,4,5);
labels 1 OP !criteria;
labels 2 TF !criteria;
model rater + criteria + step;
estimate!nodes=20;
```

```
11 show !estimates=latent >> results/ex3a_shw.txt;
12 itanal >> results/ex3a_itn.txt;
```

- **Line 1**
  Gives a title for the analysis. The text supplied after the `title` command will appear on the top of any printed ACER ConQuest output.

- **Line 2**
  Indicates the name and location of the data file.

- **Lines 3-4**
  Multifaceted data can be entered into data sets in many ways. Here, two sets of ratings for each student have been included on each line in the data file, and explicit rater codes have been used to identify the raters. For each of the raters, there is a matching pair of ratings (one for OP and one for TF). The OP and TF ratings are implicitly identified by the columns in which the data are entered. The ACER ConQuest `format` statement is very flexible and can cater for many alternative data specifications. In this `format` statement, you will notice that `rater` is used twice. The first use indicates the column location of the rater code for the first rater, and the second use indicates the column location of the rater code for the second rater. This is followed by two variables indicating the location of the responses (referred to as response blocks). Each response block is two characters wide; and since the default width of a response is one column, each response block refers to two responses, an OP and a TF rating. The first response block (columns 21 and 22) will be associated with the first rater, and the second response block (columns 25 and 26) will be associated with the second rater.

  This `format` statement also includes an option, `criteria(2)`, which assigns the variable name `criteria` to the two responses that are implicitly identified by each response block. If this option had been omitted, the default variable name for the responses would be `item`.

  This `format` statement spans two lines in the command file. Command statements can be 1023 characters in length and can cover any number of lines in a command file. The semi-colon (`;`) is the separator between statements, not the *return* or *new line* characters.

- **Line 5**
  The `codes` statement restricts the list of valid response codes to G, H, I, J, K, and L. All other responses will be treated as missing-response data.

- **Line 6**
  The `score` statement assigns score levels to each of the response categories. Here, the left side of the `score` argument shows the six valid codes defined by the `codes` statement, and the right side gives six matching scores. The six distinct codes on the left indicate that the item response model will model six categories for each item; the scores on the right are the scores that will be assigned to each category.

  > **NOTE:** As discussed in the previous section, ACER ConQuest makes an important distinction between response categories and response levels (or scores). The number of item response categories that will be modelled by ACER ConQuest is determined by the number of unique codes that exist after all recodes have been performed. ACER ConQuest requires a score for each response category. This can be provided via the `score` statement. Alternatively, if the `score` statement is omitted, ACER ConQuest will treat the recoded responses as numerical values and use them as scores. If the recoded responses are not numerical values, an error will be reported.

- **Lines 7-8**
  In the previous sample analyses, variable labels were read from a file. Here the `criteria` facet contains only two levels (the OP and TF ratings), so the labels are given in the command file using `labels` command syntax. These `labels` statements have two arguments. The first argument indicates the level of the facet to which the label is to be assigned, and the second argument is the label for that level. The option gives the facet to which the label is being applied.

- **Line 9**
  The `model` statement here contains three terms; `rater`, `criteria` and `step`. This `model` statement indicates that the responses are to be modelled with three sets of parameters: a set of rater harshness parameters, a set of criteria difficulty parameters, and a set of parameters to describe the step structure of the responses.

  > **EXTENSION:** The `model` statement in this sample analysis includes main effects only. An interaction term `rater*criteria` could be added to model variation in the difficulty of the criteria across the raters. Similarly, the model specifies a single step-structure for all rater and criteria combinations. Step structures that were common across the criteria but varied with raters could be modelled by using the term `rater*step`, step structures that were common across the raters but varied with criteria could be modelled by using the term `criteria*step`, and step structures

that varied with rater and criteria combinations could be modelled by using the term `rater*criteria*step`.

- **Line 10**
  The `estimate` statement initiates the estimation of the item response model.

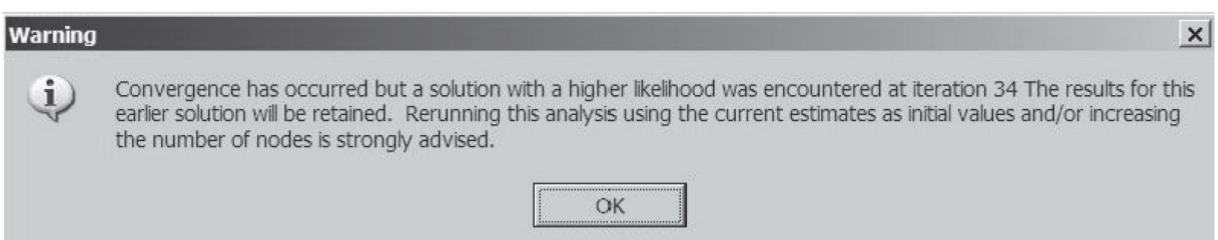
- **Line 11**
  The `show` statement produces a display of the item response model parameter estimates and saves them to the file `ex3a_shw.txt`. The option `estimates=latent` requests that the displays include an illustration of the latent ability distribution.

- **Line 12**
  The `itanal` statement produces a display of the results of a traditional item analysis. As with the `show` statement, we have redirected the results to a file (in this case, `ex3a_itn.txt`).

2.4.1.3 Running the Multifaceted Sample Analysis

To run this sample analysis, start the GUI version of ACER ConQuest and open the control file `ex3a.cqc`.

Select `Run`→`Run All`. ACER ConQuest will begin executing the statements that are in the file `ex3a.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting the multifaceted model to the data; and as it does, it will report on the progress of this estimation. Due to the large size of this data file, ACER ConQuest will take some time to perform this analysis. During estimation, ACER ConQuest reports a warning message:



As the scores of the writing test spread students far apart, as indicated by the estimated variance of the ability distribution (5.7 logits), this suggests that more nodes to cover the ability range are required in the estimation process.

To re-run ACER ConQuest with more nodes during the estimation, modify the `estimate` command as follows:

- **Line 10**
  `estimate ! nodes=30;`

The default number of nodes is 15. The above `estimate` command requests ACER ConQuest to use 30 nodes to cover the ability range.

Re-run ACER ConQuest by selecting `Run`→`Run All` from the menu. This time, ACER ConQuest no longer reports a warning for convergence problems.

After the estimation is complete, the two statements that produce output (`show` and `itanal`) will be processed. The results of the `show` statement can be found in the file `ex3a_shw.txt`, and the results of the `itanal` statement can be found in the file `ex3a_itn.txt`. On this occasion, the `show` statement will produce six tables.

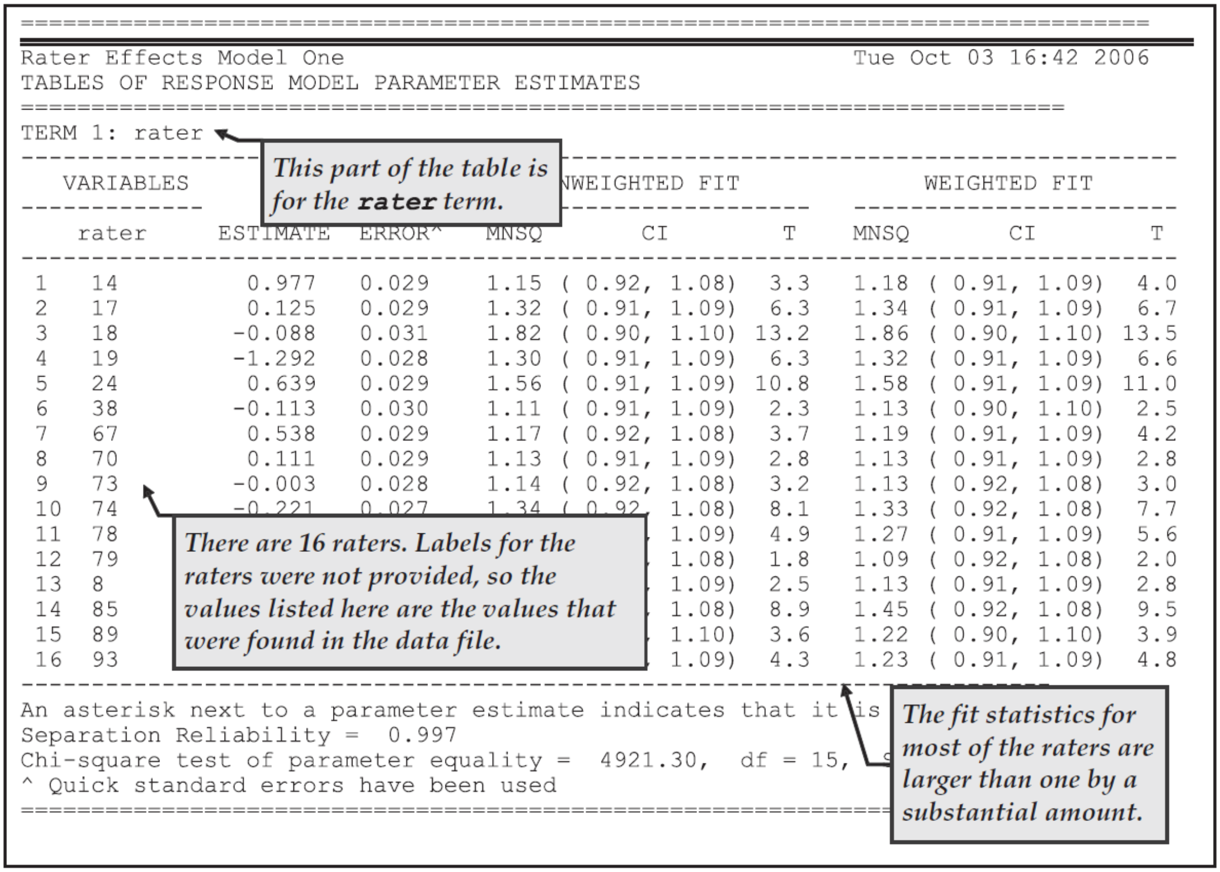
From Figure 2.27, we note that there were 16 raters and that the harshness ranges from a high of 0.977 logits for rater 14 (the first rater in the table) to a low of –1.292 for rater 19 (the fourth rater in the table). This is a range of 2.123, which appears quite large when compared to the standard deviation of the latent distribution, which is estimated to be 2.37 (the square root of the variance that is reported in the third table (the population model) in `ex3a_shw.txt`). That means that ignoring the influence of the harshness of the raters may move a student's ability estimate by as much as one standard deviation of the latent distribution. We also note that, with this model, the raters do not fit particularly well. The high mean squares (and corresponding positive t values) suggest quite a bit of unmodelled noise in the ratings.

In Figure 2.28, we note that the OP and TF difficulty estimates are very similar, differing by just 0.178 logits. This difference is significant but very small. The mean square fit statistics are less than one, suggesting that the criteria could have unmodelled dependency.

Figure 2.29 shows the step parameter estimates. The fit here is not very good, particularly for steps 1 and 4, suggesting that we should model step structures that interact with the facets. It is pleasing to note that the estimates for the steps themselves are ordered and well separated.

Figure 2.30 is the map of the parameter estimates that is provided in `ex3a_shw.txt`. The map shows how the variation between raters in their harshness is large relative to the difference in the difficulty of the two tasks. It also shows that the rater harshness estimates are well centred for the estimated ability distribution.

The file `ex3a_itn.txt` contains basic traditional statistics for this multifaceted analysis, extracts of which are shown in Figures 2.31 and 2.32.



```
================================================================================
Rater Effects Model One                                  Tue Oct 03 16:42 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
===========================================================================
TERM 1: rater
-----------------------------------------------------------------------------------
  VARIABLES                          NWEIGHTED FIT              WEIGHTED FIT
-------------                      ------------------     ------------------------
    rater     ESTIMATE  ERROR^  MNSQ        CI        T    MNSQ        CI         T
-----------------------------------------------------------------------------------
 1   14        0.977   0.029   1.15 ( 0.92, 1.08)  3.3    1.18 ( 0.91, 1.09)  4.0
 2   17        0.125   0.029   1.32 ( 0.91, 1.09)  6.3    1.34 ( 0.91, 1.09)  6.7
 3   18       -0.088   0.031   1.82 ( 0.90, 1.10) 13.2    1.86 ( 0.90, 1.10) 13.5
 4   19       -1.292   0.028   1.30 ( 0.91, 1.09)  6.3    1.32 ( 0.91, 1.09)  6.6
 5   24        0.639   0.029   1.56 ( 0.91, 1.09) 10.8    1.58 ( 0.91, 1.09) 11.0
 6   38       -0.113   0.030   1.11 ( 0.91, 1.09)  2.3    1.13 ( 0.90, 1.10)  2.5
 7   67        0.538   0.029   1.17 ( 0.92, 1.08)  3.7    1.19 ( 0.91, 1.09)  4.2
 8   70        0.111   0.029   1.13 ( 0.91, 1.09)  2.8    1.13 ( 0.91, 1.09)  2.8
 9   73       -0.003   0.028   1.14 ( 0.92, 1.08)  3.2    1.13 ( 0.92, 1.08)  3.0
 10  74       -0.221   0.027   1.34 ( 0.92, 1.08)  8.1    1.33 ( 0.92, 1.08)  7.7
 11  78                                      1.09)  4.9    1.27 ( 0.91, 1.09)  5.6
 12  79                                      1.08)  1.8    1.09 ( 0.92, 1.08)  2.0
 13  8                                       1.09)  2.5    1.13 ( 0.91, 1.09)  2.8
 14  85                                      1.08)  8.9    1.45 ( 0.92, 1.08)  9.5
 15  89                                      1.10)  3.6    1.22 ( 0.90, 1.10)  3.9
 16  93                                      1.09)  4.3    1.23 ( 0.91, 1.09)  4.8
-----------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is
Separation Reliability =  0.997
Chi-square test of parameter equality =  4921.30,  df = 15,
^ Quick standard errors have been used
=========================================================================
```


Figure 2.27: Parameter Estimates for Rater Harshness

```
TERM 2: criteria
------------------------------------------------------------------------------------
   VARIABLES                                  UNWEIGHTED FIT        WEIGHTED FIT
---------------                          -----------------------  -----------------------
   criteria     ESTIMATE   ERROR^   MNSQ        CI          T    MNSQ        CI         T
------------------------------------------------------------------------------------
 1   OP           0.089    0.010    0.97 ( 0.97, 1.03) -2.1   0.97 ( 0.97, 1.03) -1.8
 2   TF          -0.089*   0.010    1.01 ( 0.97, 1.03)  0.8   0.99 ( 0.97, 1.03) -0.4
------------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is c
====================================================================================
```

*This part of the table is for the **criteria** term.*

The criteria labels are OP and TF.

*There are only two criteria, so that effectively means one **criteria** difficulty estimate, since the average must be zero.*

The fit is less than one, suggesting dependency between these criteria.

Figure 2.28: Parameter Estimates for the Criteria

```
============
TERM 3: step
------------------------------------------------------------------------------------
   VARIABLES                                  UNWEIGHTED FIT        WEIGHTED FIT
---------------                          -----------------------  -----------------------
     step       ESTIMATE   ERROR^   MNSQ        CI          T    MNSQ        CI         T
------------------------------------------------------------------------------------
       0                            0.39 ( 0.97, 1.03)-52.1   2.45 ( 0.84, 1.16) 12.7
       1         -7.088    0.043    1.09 ( 0.97, 1.03)  5.3   1.14 ( 0.95, 1.05)  5.4
       2         -3.244    0.021    1.23 ( 0.97, 1.03) 13.6   1.09 ( 0.97, 1.03)  5.6
       3          0.613    0.015    1.11 ( 0.97, 1.03)  6.9   1.15 ( 0.97, 1.03)  9.1
       4          3.727    0.022    1.44 ( 0.97, 1.03) 25.1   1.24 ( 0.95, 1.05)  8.8
       5          5.992*            0.68 ( 0.97,                                ) 11.4
====================================================================================
```

*This part of the table is for the **step** term.*

These generalised items have six response categories, so four step parameters have been estimated.

The fit of the step parameters is poor, suggesting the need to allow an interaction between the step and the rater, the step and the criteria, or the step and both the criteria and rater.

Figure 2.29: Parameter Estimates for the Steps

```
============================================================================
Rater Effects Model One                                  Fri Oct 18 11:25:34
MAP OF LATENT DISTRIBUTIONS AND RESPONSE MODEL PARAMETER ESTIMATES
============================================================================
                 Terms in the Model (excl Step terms)
                        +rater               +criteria
---------------------------------------------------------------
              X|                      |                       |
              X|                      |                       |
             XX|                      |                       |
   4        XXX|                      |                       |
            XXX|                      |                       |
            XXX|                      |                       |
   3       XXXX|                      |                       |
         XXXXXX|                      |                       |
         XXXXXX|                      |                       |
   2     XXXXXX|                      |                       |
       XXXXXXXX|                      |                       |
      XXXXXXXXX|                      |                       |
   1  XXXXXXXXX|1                     |                       |
     XXXXXXXXXX|5                     |                       |
      XXXXXXXXX|7 15                  |                       |
   0    XXXXXXX|2 8 9 13 14           |1                      |
       XXXXXXXX|3 6 10 12             |2                      |
       XXXXXXXX|11 16                 |                       |
  -1   XXXXXXXX|                      |                       |
     XXXXXXXXXX|4                     |                       |
        XXXXXXX|                      |                       |
         XXXXXX|                      |                       |
  -2       XXXX|                      |                       |
           XXXX|                      |                       |
           XXXX|                      |                       |
  -3         XX|                      |                       |
             XX|                      |                       |
             XX|                      |                       |
  -4         XX|                      |                       |
              X|                      |                       |
              X|                      |                       |
  -5          X|                      |                       |
===============================================================
Each 'X' represents  51.4 cases
```

Figure 2.30: Map of the Parameter Estimates for the Multifaceted Model

In this analysis, the combination of the 16 raters and two criteria leads to 32 *generalised items*.[11] The statistics for each of these generalised items is reported in the file `ex3a_itn.txt`.

Figure 2.31 shows the statistics for the last generalised item, which is the combination of rater 93 (the sixteenth rater) and criterion TF (the second criterion). For this generalised item, the total number of students rated by this rater on this criteria is shown (in this case, 1002); and an index of discrimination (the correlation between students' scores on this item and their total score) is shown (in this case, 0.87). This discrimination index is very high, but it should be interpreted with care since only four generalised items are used to construct scores for each student. Thus, a student's score on this generalised item contributes 25% to their total score.

For each response category of this generalised item, the number of observed responses is reported, both as a count and as a percentage of the total number of responses to this generalised item. The point-biserial correlations that are reported for each category are computed by constructing a set of dichotomous indicator variables, one for each category. If a student's response is allocated to a category for an item, then the indicator variable for that category will be coded to 1; if the student's response is not in that category, it will be coded to 0. The point biserial is then the correlation between the indicator variable and the student's total score. It is desirable for the point biserials to be ordered in a fashion that is consistent with the category scores. However, sometimes point biserials are not ordered when a very small or a very large proportion of the item responses are in one category. This can be seen in Figure 2.31, where only seven of the 1002 cases have responses in category G.

The `itanal` statement's output concludes with a set of summary statistics (Figure 2.32). For the mean, standard deviation, variance and standard error of the mean, the scores have been scaled up so that they are reported on a scale consistent with students responding to all of the generalised items.

> **NOTE:** Traditional methods are not well suited to multifaceted measurement. If more than 10% of the response data is missing — either at random or by design (as will often be the case in multifaceted designs) — the test reliability and standard error of measurement will not be computed.

[11] Generalised item is the term that ACER ConQuest uses to refer to each of the unique combinations of the facets that are the agents of measurements.

```
Item 32
------
Rater:16 (93) criteria:2 (TF)
Cases for this item   1002   Discrimination  0.87
Item Threshold(s):    -7.61 -3.75  0.09  3.18  5.58
Item Delta(s):     -7.59 -3.75  0.11  3.23  5.49
----------------------------------------------------------------------------
Label    Score     Count   % of tot  Pt Bis     t  (p)   PV1Avg:1 PV1 SD:1
----------------------------------------------------------------------------
  G       0.00        7       0.70   -0.24    -7.83(.000) -7.20     1.40
  H       1.00      101      10.08   -0.45   -16.15(.000) -3.07     1.43
  I       2.00      369      36.83   -0.41   -14.32(.000) -1.05     1.27
  J       3.00      373      37.23    0.26     8.65(.000)  0.91     1.44
  K       4.00      117      11.68    0.46    16.39(.000)  2.85     1.34
  L       5.00       35       3.49    0.44    15.45(.000)  4.78     1.40
================================================================================
```

Figure 2.31: Extract from the Item Analysis for the Multifaceted Analysis

```
--------------------------------------------------------------------------------
In this analysis 87.51%  of the data are missing.

The following results are scaled to assume that a single response
was provided for each item.

N                                   8296
Mean                               78.86
Standard Deviation                 24.06
Variance                          578.92
Skewness                            0.20
Kurtosis                            0.54
Standard error of mean              0.26
================================================================================
```

Figure 2.32: Summary Statistics for the Multifaceted Analysis

2.4.2 b) The Multifaceted Analysis Restricted to One Criterion

In analysing these data with the multifaceted model, the fit statistics have suggested a lack of independence between the raters' judgments for the two criteria and evidence of unmodelled noise in the raters' behaviour. Here, therefore, an additional analysis is undertaken that adds some support to the hypothesis that the raters' OP and TF judgments are not independent. In this second analysis, only one criterion (OP) is analysed.

2.4.2.1 Required files

The files that we use in this sample analysis are:

filename	content
ex3b.cqc	The command statements.
ex3_dat.txt	The data.
ex3b_shw.txt	The results of the single-criterion multifaceted analysis.

(The last file is created when the command file is executed.)

2.4.2.2 Syntax

`ex3b.cqc` is the command file used in this tutorial for fitting the multifaceted model to our data, but using only one of the criteria. The code listed here is very similar to `ex3a.cqc`, the command file from the previous analysis (as shown in section 2.4.1.2). So only the differences will be discussed in the list underneath the code box.

`ex3b.cqc`:

```
Title Rater Effects Model Two;
datafile ex3_dat.txt;
format rater 17-18 rater 19-20
       responses 21 responses 25 ! criteria(1);
codes G,H,I,J,K,L;
score (G,H,I,J,K,L) (0,1,2,3,4,5);
labels 1 OP !criteria;
/*labels 2 TF !criteria;*/
model rater + criteria + step;
estimate !nodes=20;
show ! estimates=latent >> Results/ex3b_shw.txt;
```

- **Lines 1-2**
  As in the command file of the previous analysis, `ex3b.cqc`.

- **Line 3-4**
  The response blocks in the `format` statement now refer to one column only, the column that contains the OP criteria for each rater. Note that in the option we now indicate that there is just one criterion in each response block.

- **Lines 5-7**
  As in the command file of the previous analysis, `ex3b.cqc`.

- **Line 8**
  The `labels` statement for the TF criterion is now unnecessary, so we have enclosed it inside comment markers (`/*` and `*/`).

- **Lines 9-11**
  As for lines 9, 10, and 11 in `ex3a.cqc`, except the `show` statement output is directed to a different file, `ex3b_shw.txt`.

2.4.2.3 Running the Multifaceted Model for One Criterion

To run this sample analysis, start the GUI version of ACER ConQuest and open the control file `ex3b.cqc`.

Select `Run`→`Run All`.

ACER ConQuest will begin executing the statements that are in the file `ex3b.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting the multifaceted model to the data; and as it does so, it will report on the progress of the estimation. Due to the large size of this data file, ACER ConQuest will take some time to perform this analysis.

In Figures 2.33 and 2.34, the rater and step parameter estimates are given for this model from the second table in the file `ex3b_shw.txt`. The part of the table that reports on the `criteria` facet is not shown here, since there is only one criterion and it must therefore have an estimate of zero. In fact, the inclusion of the `criteria` term in the `model` statement was redundant.

A comparison of Figures 2.33 and 2.34 with Figures 2.27, 2.28, and 2.29 shows that this second model leads to an improved fit for both the `rater` and `step` parameters. It would appear that the apparent noisy behaviour of the raters, as illustrated in Figure 2.27, is a result of the redundancy in the two criteria and is not evident if a single criterion is analysed. The fit statistics for the steps are similarly improved, suggesting either that the redundancy between the criteria was influencing the step fits or that there is a `rater` by `criteria` interaction.

```
================================================================================
Rater Effects Model Two                                   Tue Oct 03 17:05 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
================================================================================
TERM 1: rater
---------------------------------------------------------------------------------
  VARIABLES                       UNWEIGHTED FIT              WEIGHTED FIT
-------------                 -----------------------   -----------------------
    rater    ESTIMATE  ERROR^  MNSQ       CI        T    MNSQ      CI         T
---------------------------------------------------------------------------------
 1   14        0.770   0.039   0.89 ( 0.92, 1.08) -2.7   0.89 ( 0.91, 1.09) -2.4
 2   17        0.070   0.039   0.97 ( 0.91, 1.09) -0.6   0.99 ( 0.91, 1.09) -0.3
 3   18       -0.039   0.041   1.50 ( 0.90, 1.10)  8.7   1.48 ( 0.90, 1.10)  8.1
 4   19       -1.320   0.037   1.03 ( 0.91, 1.09)  0.8   1.03 ( 0.92, 1.08)  0.7
 5   24        0.737   0.039   1.22 ( 0.91, 1.09)  4.7   1.22 ( 0.91, 1.09)  4.4
 6   38        0.209   0.041   0.86 ( 0.91, 1.09) -3.0   0.90 ( 0.90, 1.10) -2.0
 7   67        0.466   0.038   0.99 ( 0.92, 1.08) -0.3   1.01 ( 0.91, 1.09)  0.3
 8   70       -0.095   0.039   1.00 ( 0.91, 1.09)  0.1   0.99 ( 0.91, 1.09) -0.1
 9   73       -0.254   0.038   0.93 ( 0.92, 1.08) -1.7   0.89 ( 0.91, 1.09) -2.5
 10  74       -0.136   0.036   1.17 ( 0.92, 1.08)  4.3   1.13 ( 0.92, 1.08)  3.1
 11  78       -0.341   0.038   0.87 ( 0.91, 1.09) -3.1   0.91 ( 0.91, 1.09) -2.0
 12  79        0.124   0.038   0.83 ( 0.92, 1.08) -4.2   0.88 ( 0.91, 1.09) -2.9
 13                                                 .9   0.95 ( 0.91, 1.09) -1.2
 14                                                 .8   1.04 ( 0.92, 1.08)  1.0
 15                                                 .7   0.92 ( 0.90, 1.10) -1.6
 16  93       -0.291*  0.150   0.94 ( 0.91, 1.09) -1.3   0.98 ( 0.91, 1.09) -0.4
---------------------------------------------------------------------
================================================================================
```

The fit statistics for this model are better than the corresponding fit statistics for the previous model.

Figure 2.33: Rater Harshness Parameter Estimates

The dependency possibility can be further explored by using the model that assumed independence (the first sample analysis in this section) to calculate the expected frequencies of various pairs of OP and TF ratings and then comparing the expected frequencies with the observed frequencies of those pairs. Figure 2.35 shows a two-dimensional frequency plot of the observed and expected number of scores for pairs of values of TF and OP given by rater 85. The diagonal line shows the points where the TF and OP scores are equal. It is noted that the observed frequencies are much higher than the expected frequencies along this diagonal, indicating that rater 85 tends to give more identical scores for TF and OP than one would expect. Similar patterns are also observed for other raters. It appears that a model that takes account of the severity of the rater and the difficulty of the criteria does not fit these data well.

```
============
TERM 3: step
------------------------------------------------------------------------------------
  VARIABLES                             UNWEIGHTED FIT              WEIGHTED FIT
------------                       -----------------------    ----------------------
    step   ESTIMATE  ERROR^    MNSQ        CI        T     MNSQ        CI        T
------------------------------------------------------------------------------------
      0                        0.36 ( 0.97, 1.03)-55.3    1.45 ( 0.83, 1.17)  4.6
      1     -6.007   0.056     0.96 ( 0.97, 1.03) -2.4    1.03 ( 0.95, 1.05)  1.1
      2     -3.124   0.029     1.04 ( 0.97, 1.03)  2.6    1.02 ( 0.97, 1.03)  1.7
      3      0.766   0.020     1.03 ( 0.97, 1.03)  1.9    1.04 ( 0.97, 1.03)  2.6
      4      3.170   0.031     1.08 ( 0.97, 1.03)  5.0    1.02 ( 0.95, 1.05)  1.0
      5      5.195*            0.87 ( 0.97, 1.03) -8.6    1.28 ( 0.88, 1.12)  4.3
------------------------------------------------------------------------------------
====================================================================================
```

The fit statistics for this model are better than the corresponding fit statistics for the previous model.

Figure 2.34: Step Parameter Estimates

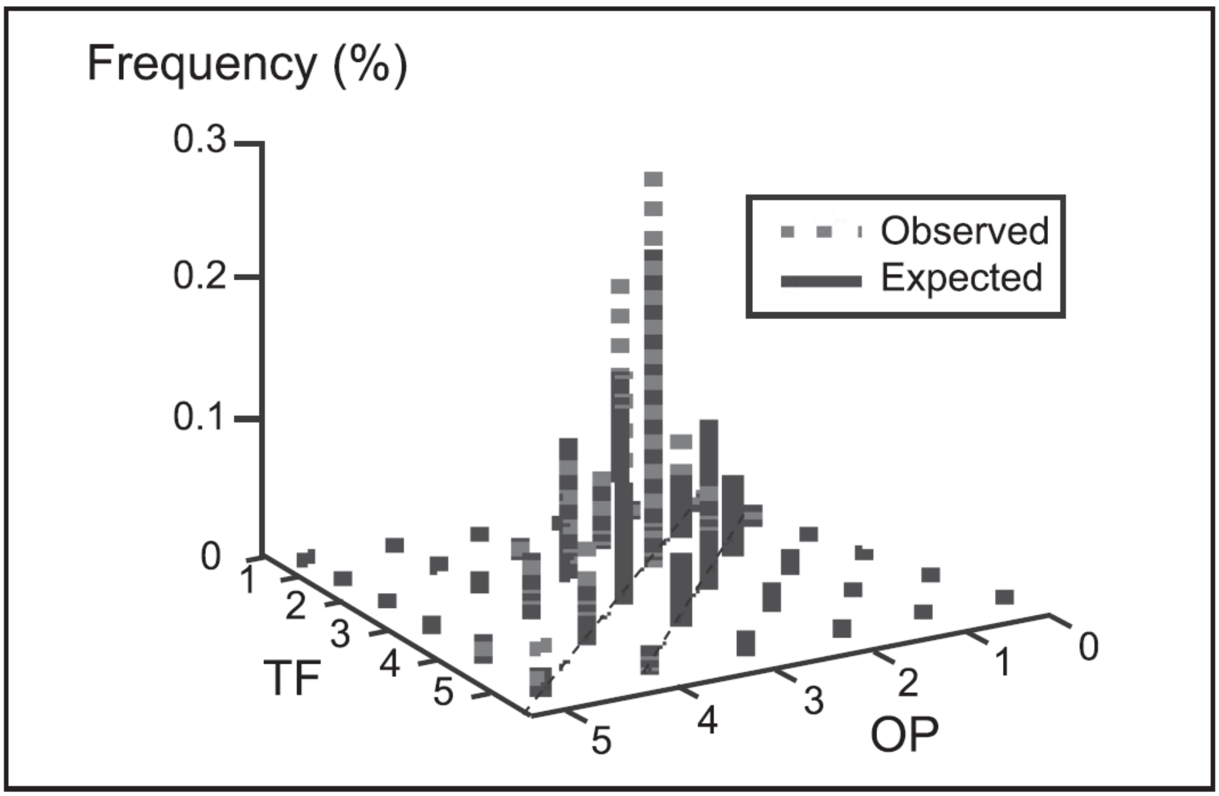


Figure 2.35: Observed Versus Expected Frequencies for Pairs of OP and TF Scores

WARNING: In section 2.3, the deviance statistic was used to compare the fit of a rating scale and partial credit model. It is not appropriate to use the deviance statistic to compare the fit of the two models fitted in this section. The deviance statistic can only be used when one model is a submodel of the other. For this to occur, the models must result in response patterns that are the same length, and each of the items must have the same number of response categories in each of the analyses (which was not the case here).

2.4.3 Summary

In this section, we have seen how to fit multifaceted models with ACER ConQuest. Our sample analysis has used only one additional facet (rater), but ACER ConQuest can analyse up to 50 facets.

Some key points we have covered in this section are:

- ACER ConQuest can be used to fit multifaceted item response models easily.
- The `format` statement is very flexible and can deal with many of the alternative ways that multifaceted data can be formatted (see the command reference in Section 4 for more examples).
- A `score` statement can be used to assign scores to the response categories that are modelled.
- We have reiterated the point that response categories and item scores are *not* the same thing.
- Fit statistics can be used to suggest alternative models that might be fitted to the data.

2.5 Many Facets and Hierarchical Model Testing

In section 2.4, the notion of additional measurement facets is introduced, and data was analysed with one additional facet, a rater facet. The number of facets that can be used with multifaceted measurement models is theoretically unlimited, although, as shall be seen in this section, the addition of each new facet adds considerably to the range of models that need to be considered.[12] A number of techniques are available for choosing between alternative models for multifaceted data. First, the deviance statistic of alternative models

[12] ACER ConQuest can model up to 50 different facets.

can be compared to provide a formal statistical test of the relative fit of models. Second, the fit statistics for the parameter estimates can be used, as was done in the previous section. Third, the estimated values of the parameters associated with a term in a model can be examined to see if that term is necessary. In this section, we illustrate these strategies for choosing between the many alternative multifaceted models that can be applied to data that have more than two facets.

The data that we are analysing in this section are simulated three-faceted data.[13] The data were simulated to reflect an assessment context in which 500 students have each provided written responses to two out of a total of four writing topics. Each of these tasks was then rated by two out of four raters against five assessment criteria. For each of the five criteria, a four-point rating scale was used with codes 0, 1, 2 and 3. This results in four sets of ratings (two essay topics by two raters' judgments) against the five criteria for each of the 500 students. In generating the data, two raters and two topics were randomly assigned to the students, and the model used assumed that the raters differed in harshness, that the criteria differed in difficulty, and that the rating structure varied across the criteria. The topics were assumed to be of equal difficulty; there were no interactions between the `topic`, `criteria` and `rater` facets; and the step structure did not vary with `rater` or `topic`.

The files used in this sample analysis are:

filename	content
ex4a.cqc	The command statements used for the first analysis.
ex4_dat.txt	The data.
ex4_lab.txt	The variable labels for the facet elements.
ex4a_prm.txt	Initial values for the item parameter estimates.
ex4a_reg.txt	Initial values for the regression parameter estimates.
ex4a_cov.txt	Initial values for the variance parameter estimates.
ex4a_shw.txt	Selected results of the first analysis.
ex4b.cqc	The command statements used for the second analysis.
ex4b_1_shw.txt and ex4b_2_shw.txt	Selected results of the second analysis.
ex4c.R	The R command file used for the third analysis.
ex4c.cqc	The ACER ConQuest command statements used for the th
ex4c_1_shw.txt through ex4c_11_shw.txt	Selected results of the third analysis.

[13]For those familiar with approach and terminology of Linacre (1994), these would be considered four-faceted data, since Linacre counts the cases as a facet, whereas we count the unique variables in the `model` statement.

(The `_prm.txt`, `_reg.txt`, `_cov.txt`, and `_shw.txt` files are created when the command file is executed.)

The data were entered into the file `ex4_dat.txt` using four lines per student, one for each rater and topic combination. For each of the lines, column 1 contains a rater code, column 3 contains a topic code and columns 5 through 9 contain the ratings of the five criteria given by the matching rater and topic combination.

2.5.1 a) Fitting a General Three-Faceted Model

In the first analysis, we fit a model that assumes main effects for all facets, the set of three two-way interactions, and a step structure that varies with `topic`, `item` and `rater`.

2.5.1.1 Syntax

`ex4a.cqc` is the command file used in the first analysis to fit one possible multifaceted model to these data. The code box below shows the contents of the file, and the list underneath the code box explains each line of syntax.

`ex4a.cqc`:

```
1  datafile ex4_dat.txt;
2  format rater 1 topic 3 responses 5-9 /
3         rater 1 topic 3 responses 5-9 /
4         rater 1 topic 3 responses 5-9 /
5         rater 1 topic 3 responses 5-9 ! criteria(5);
6  label << ex4_lab.txt;
7  set update=yes,warning=no;
8  model   rater + topic + criteria + rater*topic + rater*criteria +
9          topic*criteria + rater*topic*criteria*step;
10 export parameters >> Results/ex4a_prm.txt;
11 export reg >> Results/ex4a_reg.txt;
12 export cov >> Results/ex4a_cov.txt;
13 estimate ! nodes=10, stderr=empirical;
14 show parameters !estimates=latent,tables=1:2:4>> Results/ex4a_shw.txt;
```

- **Line 1**
  Indicates the name and location of the data file.

- **Lines 2-5**
  Multifaceted data can be entered into data sets in many ways. The ACER ConQuest `format` statement is very flexible and can cater for many alternative data specifications. Here the data are spread over four lines for each student. Each line contains a rater code, a topic code and five responses. The slash (`/`) character is used to indicate that the following data should be read from the next line of the data file. The multiple use of the terms `rater`, `topic` and `responses` allows us to read the multiple sets of ratings for each student. In this case, the term `rater` is used four times, `topic` four times and `responses` four times. Thus, the rater and topic indicated on the first line for each case will be associated with the responses on the first line, the rater and topic on the second line will be associated with the responses on the second line, and so on. More generally, if variables are repeated in a `format` statement, the n-th occurrence of `responses` will be associated with the n-th occurrence of any other variable, or the n-th occurrence of `responses` will be matched with the highest occurrence of any other variable if n is greater than the number of occurrences of that variable.

  This `format` statement also includes an option, `criteria(5)`, which assigns the variable name `criteria` to the five responses that are implicitly identified by the response block. If this option had been omitted, the default variable name for the responses would have been `item`.

- **Line 6**
  The labels for the facets in this analysis are to be read from the file `ex4_lab.txt`. The contents of this file are shown in Figure 2.36. Here we have provided labels for each of the three facets. The character string `===>` precedes the name of the facet, and the following lines contain the facet level and then the label that is to be assigned to that level.

- **Line 7**
  The `set` statement can be used to alter some of ACER ConQuest's default values. In this case, the default status of the `update` and `warnings` settings has been changed. When `update` is set to `yes`, in conjunction with the following `export` statements, updated parameter estimates will be written to a file at the completion of every iteration. This option is particularly valuable when analyses take a long time to execute. If the `update` option is set to `yes` and you have to terminate the analysis for some reason (e.g., you want to use the computer for something else and ACER ConQuest is monopolising CPU time), you can interrupt the job and then restart it at some later stage with starting values set to the most recent parameter estimates. (To use these starting values, you would have to add one or more `import` statements

```
===> rater
1  Amy
2  Beverely
3  Colin
4  David
===> topic
1  Sport
2  Family
3  Work
4  School
===> criteria
1  spelling
2  coherence
3  structure
4  grammar
5  content
```

Figure 2.36: The Labels File for the Many Facets Sample Analysis

to the command file.) Setting `warnings` to `no` tells ACER ConQuest not to report warning messages. Errors, however, will still be reported. Setting `warnings` to `no` is typically used in conjunction with setting `update` to `yes` in order to suppress the warning message that there is a file overwrite at every iteration.

- **Lines 8-9**
  The `model` statement contains seven terms: `rater`, `topic`, `criteria`, `rater*topic`, `rater*criteria`, `topic*criteria`, and `rater*topic*criteria*step`. This `model` statement indicates that seven sets of parameters are to be estimated. The first three are main effects and correspond to a set of rater harshness parameters, a set of topic difficulty parameters, and a set of criteria difficulty parameters. The next three are two-way interactions between the facets. The first of these interaction terms models a variation in rater harshness across the topics (or, equivalently, variation in topic difficulty across the raters), the second models a variation in rater harshness across the criteria, and the third represents a variation in the topic difficulties across the criteria. The final term represents a set of parameters to describe the step structure of the responses. The step structure is modelled as varying across all combinations of raters, topics and criteria.

  One additional term could be added to this model: the three-way interaction between raters, topics and criteria.

- **Lines 10-12**
  The `export` statements request that the parameter estimates be written to text files

in a simple, unlabelled format. The `export` statement can be used to produce files that are more readily read by other software. Further, the format of each export file matches the format of ACER ConQuest import files so that export files that are written by ACER ConQuest can be re-read as either anchor files or initial value files.[14]

- **Line 13**
The `estimate` statement initiates the estimation of the item response model. In this case, two options are used to change the default settings of the estimation procedures. The `nodes=10` option means that the numerical integration that is necessary in the estimation will be done with a Gauss-hermite quadrature method using 10 nodes.[15] The default number of nodes is 15, but we have chosen to reduce the number of nodes to 10 for this sample analysis, since it will reduce the processing time. Simulation results by Wu & Adams (1993) illustrate that 10 nodes will normally be sufficient for accurate estimation. The `stderr=empirical` option causes ACER ConQuest to compute the full error variance-covariance matrix for the model that has been estimated. This method provides the most accurate estimates of the asymptotic error variances that ACER ConQuest can compute. It does, however, take a considerable amount of computing time, even on very fast machines. In Estimating Standard Errors in Chapter 3, we discuss the circumstances under which it is desirable to use the `stderr=empirical` option. In this case, we have used it because of the large number of facets, each of which has only a couple of levels.

- **Line 14**
The `show` statement produces a display of the item response model parameter estimates and saves them to the file `ex4a_shw.txt`. The option `estimates=latent` requests that the displays include an illustration of the latent ability distribution. The option `tables=1:2:4` limits the output to tables 1, 2 and 4.

2.5.1.2 Running the Multifaceted Sample Analysis

To run this sample analysis, start the GUI version of ACER ConQuest and open the control file `ex4a.cqc`.

Select `Run`→`Run All`. ACER ConQuest will begin executing the statements that are in the file `ex4a.cqc`; and as they are executed, they will be echoed in the Output window. When

[14] For uses of initial value files and anchor files, see sections 2.6 and 2.9.

[15] See Estimation in Chapter 3 for further explanation of the estimation methods that are used in ACER ConQuest.

ACER ConQuest reaches the `estimate` statement, it will begin fitting the multifaceted model to the data; and as it does so, it will report on the progress of this estimation. This analysis will take around 700 iterations to converge, and the calculation of the standard errors may take a considerable amount of time. After the estimation is complete, the output from the `show` statement can be found in the file `ex4a_shw.txt`. Figures 2.37 and 2.38 are extracts from the second table in this file.

Figure 2.37 shows the parameter estimates for the three main effects: `rater`, `topic` and `criteria`. Notice that the separation reliability for the `topic` is close to zero and that the variation between the topic parameter estimates is not significant. This result suggests that the `topic` term might be deleted from the model because the topics do not vary in their difficulty. (Thus, ACER ConQuest has confirmed the model we used in our data simulation.)

Figure 2.38 shows the parameter estimates for one of the three two-way interaction terms. The results reported in this figure suggest that there is no interaction between the `topic` and `criteria`. (Again, ACER ConQuest has confirmed the model we used in our data simulation.) The results for the two remaining two-way interaction terms are not reported here; however, if you examine them in the file `ex4a_shw.txt` you will see that, although the effects are statistically significant, they are very small and we could probably ignore them.

2.5.2 b) The Fit of Two Additional Alternative Models

Many submodels of the model analysed with the command file in Figure `ex4a.cqc` (discussed in Section 2.5.1.1) can be fitted to these data. As we mentioned above, the model that was actually used in the generation of these data can be fitted by replacing the `model` statement in `ex4a.cqc` with `model rater + criteria + criteria*step`.

The file `ex4b.cqc` contains statements that will fit this submodel and an even simpler model (`rater + step`). The item response model parameter estimates that are obtained from the first of these models are saved to the file `ex4b_1_shw.txt` and shown in Figure 2.39. As would be expected, the fit for each of the parameters is good.

The other important thing to note about Figure 2.39 is the values of the parameter estimates. When the data in `ex4_dat.txt` were generated, the rater parameters were set at –1.0, –0.5, 0.5 and 1.0 and the criteria parameters were set at –1.2, –0.6, 0, 0.6 and 1.2.

Figure 2.40, an excerpt of `ex4b_2_shw.txt`, shows the item parameter estimates when the `model` statement is changed to `model rater + step`, which assumes that there is no variation between the criteria in difficulty, a simplification that we know does not hold

```
================================================================================
ConQuest: Generalised Item Response Modelling Software     Wed Oct 04 12:23 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
================================================================================
TERM 1: rater
---------------------------------------------------------------------------------------
   VARIABLES                              UNWEIGHTED FIT          WEIGHTED FIT
---------------                        -----------------------  -----------------------
    rater           ESTIMATE  ERROR    MNSQ       CI        T    MNSQ       CI        T
---------------------------------------------------------------------------------------
 1   Amy             -0.871   0.042    1.00 ( 0.81, 1.19) -0.0   0.99 ( 0.74, 1.26) -0.1
 2   Beverely        -0.537   0.035    1.09 ( 0.82, 1.18)  1.0   1.03 ( 0.78, 1.22)  0.3
 3   Colin            0.452   0.030    0.98 ( 0.81, 1.19) -0.2   0.98 ( 0.80, 1.20) -0.2
 4   David            0.956*           1.09 ( 0.81, 1.19)  0.9   1.08 ( 0.79, 1.21)  0.7
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.997
Chi-square test of parameter equality =     894.87,  df = 3,  Sig Level = 0.000
================================================================================
TERM 2: topic
---------------------------------------------------------------------------------------
   VARIABLES                              UNWEIGHTED FIT          WEIGHTED FIT
---------------                        -----------------------  -----------------------
    topic           ESTIMATE  ERROR    MNSQ       CI        T    MNSQ       CI        T
---------------------------------------------------------------------------------------
 1   Sport           -0.023   0.031    1.00 ( 0.81, 1.19) -0.0   0.97 ( 0.79, 1.21) -0.3
 2   Family           0.016   0.033    1.09 ( 0.81, 1.19)  0.9   1.08 ( 0.79, 1.21)  0.7
 3   Work             0.005   0.031    1.00 ( 0.81, 1.19)  0.1   0.99 ( 0.78, 1.22) -0.1
 4   School           0.002*           0.98 ( 0.81, 1.19) -0.1   0.99 ( 0.79, 1.21) -0.1
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.000
Chi-square test of parameter equality =       0.82,  df = 3,  Sig Level = 0.845
================================================================================
TERM 3: criteria
---------------------------------------------------------------------------------------
   VARIABLES                              UNWEIGHTED FIT          WEIGHTED FIT
---------------                        -----------------------  -----------------------
   criteria         ESTIMATE  ERROR    MNSQ       CI        T    MNSQ       CI        T
---------------------------------------------------------------------------------------
 1   spelling        -1.046   0.048    1.01 ( 0.88, 1.12)  0.2   0.98 ( 0.84, 1.16) -0.3
 2   coherence       -0.569   0.037    1.03 ( 0.88, 1.12)  0.6   1.08 ( 0.84, 1.16)  1.0
 3   structure       -0.051   0.035    0.96 ( 0.88, 1.12) -0.6   0.93 ( 0.86, 1.14) -1.0
 4   grammar          0.551   0.029    1.09 ( 0.88, 1.12)  1.4   1.09 ( 0.86, 1.14)  1.2
 5   content          1.116*           1.05 ( 0.88, 1.12)  0.8   1.07 ( 0.87, 1.13)  1.1
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.997
Chi-square test of parameter equality =    1078.20,  df = 4,  Sig Level = 0.000
```

Figure 2.37: The Parameter Estimates for Rater Harshness, Topic Difficulty and Criterion Difficulty

TERM 6: topic*criteria

VARIABLES						UNWEIGHTED FIT			WEIGHTED FIT		
topic		criteria		ESTIMATE	ERROR	MNSQ	CI	T	MNSQ	CI	T
1	Sport	1	spelling	0.057	0.069	0.87	(0.81, 1.19)	-1.4	0.92	(0.74, 1.26)	-0.6
2	Family	1	spelling	-0.031	0.074	0.90	(0.81, 1.19)	-1.0	0.98	(0.75, 1.25)	-0.1
3	Work	1	spelling	-0.091	0.073	1.08	(0.81, 1.19)	0.8	0.95	(0.73, 1.27)	-0.3
4	School	1	spelling	0.065*		0.95	(0.81, 1.19)	-0.5	1.03	(0.74, 1.26)	0.3
1	Sport	2	coherence	-0.045	0.055	1.18	(0.81, 1.19)	1.9	1.19	(0.74, 1.26)	1.4
2	Family	2	coherence	0.050	0.057	1.13	(0.81, 1.19)	1.3	1.10	(0.74, 1.26)	0.8
3	Work	2	coherence	0.003	0.053	1.05	(0.81, 1.19)	0.6	1.04	(0.74, 1.26)	0.3
4	School	2	coherence	-0.008*		0.80	(0.81, 1.19)	-2.2	0.93	(0.74, 1.26)	-0.6
1	Sport	3	structure	0.014	0.051	0.93	(0.81, 1.19)	-0.7	0.99	(0.79, 1.21)	-0.1
2	Family	3	structure	-0.018	0.054	1.03	(0.81, 1.19)	0.4	0.99	(0.78, 1.22)	-0.1
3	Work	3	structure	0.015	0.051	1.08	(0.81, 1.19)	0.8	1.03	(0.77, 1.23)	0.3
4	School	3	structure	-0.012*		0.95	(0.81, 1.19)	-0.4	0.88	(0.78, 1.22)	-1.2
1	Sport	4	grammar	-0.029	0.047	1.07	(0.81, 1.19)	0.7	1.07	(0.79, 1.21)	0.6
2	Family	4	grammar	-0.016	0.050	1.15	(0.81, 1.19)	1.5	1.08	(0.78, 1.22)	0.7
3	Work	4	grammar	0.050	0.048	0.95	(0.81, 1.19)	-0.5	0.97	(0.78, 1.22)	-0.2
4	School	4	grammar	-0.004*		1.12	(0.81, 1.19)	1.3	1.16	(0.79, 1.21)	1.4
1	Sport	5	content	0.002*		1.02	(0.81, 1.19)	0.2	0.96	(0.80, 1.20)	-0.3
2	Family	5	content	0.015*		0.96	(0.81, 1.19)	-0.3	1.02	(0.79, 1.21)	0.2
3	Work	5	content	0.023*		1.15	(0.81, 1.19)	1.5	1.15	(0.79, 1.21)	1.4
4	School	5	content	-0.041*		0.89	(0.81, 1.19)	-1.2	0.91	(0.80, 1.20)	-0.9

An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability = 0.000
Chi-square test of parameter equality = 5.66, df = 12, Sig Level = 0.932

Figure 2.38: Parameter Estimates for the `topic*criteria` Interaction

```
================================================================================
ConQuest: Generalised Item Response Modelling Software     Wed Oct 04 15:00 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
================================================================================
TERM 1: rater
------------------------------------------------------------------------------------------
   VARIABLES                                UNWEIGHTED FIT             WEIGHTED FIT
---------------                         -----------------------   -----------------------
    rater            ESTIMATE   ERROR   MNSQ        CI        T   MNSQ        CI         T
------------------------------------------------------------------------------------------
 1   Amy              -0.999   0.029    1.01 ( 0.81, 1.19)  0.2   0.98 ( 0.75, 1.25) -0.1
 2   Beverely         -0.550   0.025    1.05 ( 0.82, 1.18)  0.6   1.01 ( 0.78, 1.22)  0.1
 3   Colin             0.518   0.024    0.98 ( 0.81, 1.19) -0.2   0.99 ( 0.80, 1.20) -0.0
 4   David             1.032*           1.05 ( 0.81, 1.19)  0.5   1.03 ( 0.79, 1.21)  0.3
------------------------------------------------------------------------------------------
An asterisk next to a parameter esti                    it is constrained
Separation Reliability =  0.999
Chi-square test of parameter equalit                   = 3,                0.000
================================================================================
TERM 2: criteria
------------------------------------------------------------------------------------------
   VARIABLES                                UNWEIGHTED FIT             WEIGHTED FIT
---------------                         -----------------------   -----------------------
   criteria          ESTIMATE   ERROR   MNSQ        CI        T   MNSQ        CI         T
------------------------------------------------------------------------------------------
 1   spelling         -1.192   0.039    1.07 ( 0.88, 1.12)  1.1   1.01 ( 0.84, 1.16)  0.2
 2   coherence        -0.591   0.031    1.07 ( 0.88, 1.12)  1.1   1.08 ( 0.84, 1.16)  1.0
 3   structure         0.007   0.028    0.94 ( 0.88, 1.12) -1.0   0.92 ( 0.86, 1.14) -1.2
 4   grammar           0.617   0.026    1.07 ( 0.88, 1.12)  1.1   1.06 ( 0.86, 1.14)  0.9
 5   content           1.158*           1.03 ( 0.88, 1.12)  0.5   1.05 ( 0.87, 1.13)  0.8
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.998
Chi-square test of parameter equality =    1865.48,  df = 4,  Sig Level = 0.000
================================================================================
TERM 3: criteria*step
------------------------------------------------------------------------------------------
          VARIABLES                         UNWEIGHTED FIT             WEIGHTED FIT
---------------------                   -----------------------   -----------------------
   criteria     step  ESTIMATE  ERROR    MNSQ        CI        T   MNSQ        CI         T
------------------------------------------------------------------------------------------
 1   spelling    0                      0.44 ( 0.88, 1.12)-11.2   0.87 ( 0.72, 1.28) -0.9
 1   spelling    1   -0.362    0.116    1.01 ( 0.88, 1.12)  0.2   1.01 ( 0.79, 1.21)  0.1
 1   spelling    2   -0.226    0.110    1.07 ( 0.88, 1.12)  1.1   1.03 ( 0.87, 1.13)  0.4
 1   spelling    3    0.588*            1.08 ( 0.88, 1.12)  1.3   1.05 ( 0.87, 1.13)  0.8
 2   coherence   0                      1.66 ( 0.88, 1.12)  8.7   1.08 ( 0.81, 1.19)  0.8
 2   coherence   1    0.614    0.107    0.75 ( 0.88, 1.12) -4.2   0.90 ( 0.79, 1.21) -1.0
 2   coherence   2   -0.303    0.118    0.97 ( 0.88, 1.12) -0.5   0.99 ( 0.85, 1.15) -0.1
 2   coherence   3   -0.311*            1.00 ( 0.88, 1.12)  0.0   1.06 ( 0.86, 1.14)  0.9
 3   structure   0                      1.01 ( 0.88, 1.12)  0.1   0.95 ( 0.84, 1.16) -0.5
 3   structure   1   -0.198    0.075    0.81 ( 0.88, 1.12) -3.1   0.90 ( 0.85, 1.15) -1.3
 3   structure   2   -0.163    0.082    0.90 ( 0.88, 1.12) -1.7   0.92 ( 0.87, 1.13) -1.3
 3   structure   3    0.361*            0.89 ( 0.88, 1.12) -1.7   0.92 ( 0.87, 1.13) -1.3
 4   grammar     0                      1.52 ( 0.88, 1.12)  7.1   1.07 ( 0.86, 1.14)  1.0
 4   grammar     1    0.116    0.069    0.91 ( 0.88, 1.12) -1.5   0.92 ( 0.86, 1.14) -1.1
 4   grammar     2    0.104    0.086    1.00 ( 0.88, 1.12)  0.0   0.99 ( 0.86, 1.14) -0.1
 4   grammar     3   -0.220*            0.99 ( 0.88, 1.12) -0.2   0.97 ( 0.87, 1.13) -0.4
 5   content     0                      1.15 ( 0.88, 1.12)  2.3   1.07 ( 0.87, 1.13)  1.1
 5   content     1   -0.314    0.060    1.02 ( 0.88, 1.12)  0.3   1.02 ( 0.87, 1.13)  0.3
 5   content     2    0.077    0.077    1.02 ( 0.88, 1.12)  0.3   1.05 ( 0.86, 1.14)  0.7
 5   content     3    0.237*            0.94 ( 0.88, 1.12) -0.9   0.99 ( 0.86, 1.14) -0.1
-----------------------------------------------------------------
```

Parameter estimates are close to their generating values.

Fit statistics look good.

Figure 2.39: Parameter Estimates for `model rater + criteria + criteria*step;`

for these data. The fact that this model is not appropriate for the data can be easily identified by the fact that the deviance has increased significantly from the deviance for the model that was fit in Figure 2.39 (as shown in the first table generated by the `show` statement). This observation is discussed in detail in the next section, *A Sequence of Models*. From Figure 2.40, however, we note that the fit statistics, at least in the case of the rater parameters, are smaller than they should be.

When lower than expected fit statistic values are found, it is generally a result of unmodelled dependencies in the data. In the previous section, we saw that low fit was probably due to an unmodelled dependency between the two criteria, OP and TF. Here the low fit suggests that there is an unmodelled consistency between the rater judgments. The judgments across raters are more consistent than the model expects, and this has arisen because an element of consistency between judgments in the ratings can be traced to the variance in the criteria difficulties, a variation that is not currently being modelled.

```
================================================================================
ConQuest: Generalised Item Response Modelling Software     Wed Oct 04 15:00 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
=============================================================================
TERM 1: rater
------------------------------------------------------------------------------------
   VARIABLES                           UNWEIGHTED FIT             WEIGHTED FIT
---------------                    -----------------------   -----------------------
    rater        ESTIMATE  ERROR   MNSQ        CI         T   MNSQ        CI         T
------------------------------------------------------------------------------------
 1   Amy           -0.669   0.022    0.84 ( 0.81, 1.19) -1.7   0.87 ( 0.74, 1.26) -1.0
 2   Beverely      -0.322   0.019    0.90 ( 0.82, 1.18) -1.1   0.84 ( 0.78, 1.22) -1.5
 3   Colin          0.333   0.018    0.88 ( 0.81, 1.19) -1.3   0.86 ( 0.79, 1.21) -1.4
 4   David          0.658*           0.93 ( 0.81, 1.19) -0.7   0.94 ( 0.79, 1.21) -0.6
------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.998
Chi-square test of parameter equality =    1518.90,  df = 3
============================================================
TERM 2: step
------------------------------------------------------------------------------------
   VARIABLES                           UNWEIGHTED FIT             WEIGHTED FIT
---------------                    -----------------------   -----------------------
     step        ESTIMATE  ERROR   MNSQ        CI         T   MNSQ        CI         T
------------------------------------------------------------------------------------
       0                             1.01 ( 0.88, 1.12)  0.2   0.99 ( 0.85, 1.15) -0.2
       1           0.415   0.033    0.94 ( 0.88, 1.12) -1.0   0.92 ( 0.86, 1.14) -1.1
       2          -0.046   0.039    1.01 ( 0.88, 1.12)  0.2   1.03 ( 0.87, 1.13)  0.5
       3          -0.369*           1.02 ( 0.88, 1.12)  0.3   1.05 ( 0.87, 1.13)  0.7
------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
```

These fit statistics are all negative.

Figure 2.40: Parameter Estimates for `model rater + step;`

2.5.3 c) A Sequence of Models

A search for a model that provides the most parsimonious fit to these data can be undertaken in a systematic fashion by using hierarchical model fitting techniques in conjunction with the use of the Chi-squared test of parameter equality.

2.5.3.1 Syntax

We will fit the models in the hierarchy shown in Figure 2.41 using R in conjunction with conquestr. The R command file used is `ex4c.R`. Since we are only interested in the effect of using different terms in the model, all other aspects of the command file, i.e. the format of our data and the method of estimation, stay the same across all models in the hierarchy. The use of R will allow us to efficiently loop through all models of interest, by only updating/overwriting the `model` statement in the command file `ex4c.cqc` at each iteration. At the end of each iteration (i.e. after each model has been fitted) we retain the statistics of interest: Deviance and the number of parameters. These will allow us to conduct a Chi-squared test between nested models in the hierarchy, and hence decide which terms are significant.

2.5.3.2 Results

The results of all 11 fitted models are written to the files `ex4c_1_shw.txt` through `ex4c_11_shw.txt`. A summary of Deviance statistics is written to the csv file `ex4cDeviances.csv`. Figure 2.41 illustrates the hierarchy of models that are included in `ex4c.R` and summarises the fit of the models.

We can now use the Chi-squared statistics to compare any pair of nested models. Under the null hypothesis that the nested/smaller model is correct (rather than the model with more parameters), the Chi-squared statistic is distributed according to a Chi-squared distribution with degrees of freedom given by the difference in number of parameters between the two models.

Notice, as we move through the hierarchy from model (1) to model (5) and then model (9), how the fit is not significantly worsened by removing terms. This is evident in the Chi-squared statistics being relatively close to their null means (i.e. their hypothesised degrees of freedom). It may be worth to note that some of the p-values along this path are between 0.01 and 0.05, and hence fall below the common significance threshold 0.05. For example, moving from model (2) to (3) yields $P(\chi^2_{33} > 53.5) = 0.013$. However, in the interest of finding a parsimonious model one may want to adapt a stricter threshold than 0.05.

By similar reasoning, model fit does not worsen significantly following the path (1) to (3) and then (7) to (9).

Towards the bottom of the hierarchy we then encounter models which do not explain sufficient variability in the data at even the lowest signifcance levels. Here the Chi-squared statistics and their null means differ by orders of magnitude:

- Comparing models (5) and (6) (χ^2=1578.5, df=3), we note that the `rater` term is necessary—that is, there is significant variation between the raters in their harshness.
- Comparing models (9) and (10) (χ^2=172.6, df=8), we can see that the `step` parameters vary significantly with `criteria`.

2.5.4 Summary

In this section, we have seen how ACER ConQuest can be used to compare the fit of competing models that may be considered appropriate for a data set. We have seen how to use the deviance statistics, fit statistics and test of parameter equality to assist in the choice of a best fitting model.

2.6 Unidimensional Latent Regression

The term latent regression refers to the direct estimation of regression models from item response data. To illustrate the use of latent regression, consider the following typical situation. There are two groups of students, group A and group B, and it is of interest to estimate the difference in the mean achievement of the two groups. A common approach is to administer a test to the students and then use this test to produce achievement scores for all of the students. A standard procedure can then be applied, such as regression (which, in this simple case, becomes identical to a *t*-test), to examine the difference in the means of the achievement scores. Depending upon the model that is used to produce 'student scores,' this approach can result in misleading inferences about the differences in the means. Using the latent regression methods described by Adams, Wilson, & Wu (1997), ACER ConQuest avoids such problems by directly estimating the difference in the mean achievement of the groups from the item response data without first producing individual student scores.

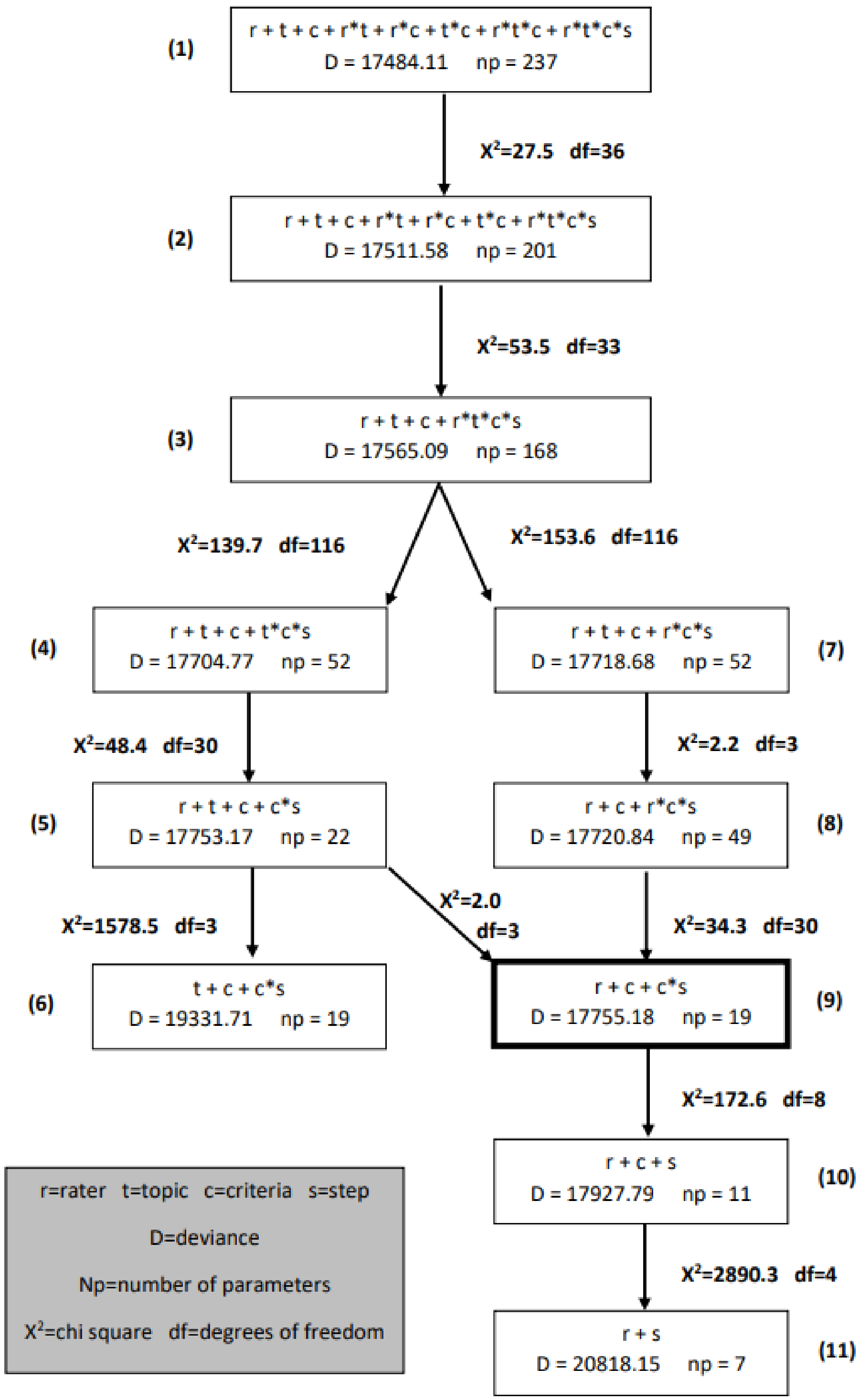


Figure 2.41: A Hierarchy of Models and Their Fit

The data used here are a subset of the data that were collected by Lokan, Lokan et al. (1996) as part of the Third International Mathematics and Science Study (TIMSS) (Beaton et al., 1996). The TIMSS data that we will be using are the mathematics achievement test data, collected from a sample of 6800 students in their first two years of secondary schooling in Australia.[16]

The TIMSS study used a sophisticated test item rotation plan that enabled achievement data to be gathered on a total of 158 test items while restricting the testing time for any individual student to 90 minutes. Details on how this was achieved are described in Adams & Gonzales (1996). In this section, we will be using the data to examine grade differences and gender differences in students' mathematics achievement as tested by the TIMSS tests.

The data set used in this sample analysis, `ex5_dat.txt`, contains 6800 lines of data, one line for each student that was tested. Columns 20 to 177 contain the item responses. The TIMSS tests consist of multiple choice, short answer and extended response questions. For the multiple choice items, the codes *1*, *2*, *3*, *4* and *5* are used to indicate the response alternatives to the items. For the short answer and extended response items, the codes *0*, *1*, *2* and *3* are used to indicate the student's score on the item. If an item was not presented to a student, the code *.* (a period) is used; if the student failed to attempt an item and that item is part of a block of non-attempts at the end of a test, then the code *R* is used. For all other non-attempts, the code *M* is used. The first 19 columns of the data set contain identification and demographic information. In this example, only the data in columns 17 through 19 are used. Column 17 contains the code *0* for male students and *1* for female students; column 18 contains the code *0* for lower grade (first year of secondary school) students and *1* for upper grade (second year of secondary school) students; and column 19 contains the product of columns 17 and 18, that is, it contains *1* for upper grade female students and *0* otherwise.

2.6.1 a) A Latent Variable t-Test

In the first sample analysis that uses these data, it is of interest to estimate the difference in achievement between the lower and upper grades. To illustrate the value of directly estimating the differences using latent regression, only the first six items are used. Later in the section, we will compare the results obtained from analysing only these six items with the results obtained from analysing all 158 items.

[16] These 6800 students were randomly selected from a larger Australian TIMSS sample of over 13 000 students in their first two years of secondary schooling.

2.6.1.1 Required files

The files used in this first sample analysis are:

filename	content
ex5a.cqc	The command statements used for the first analysis.
ex5_dat.txt	The data.
ex5_lab.txt	The variable labels for the items.
ex5a_mle.txt	Maximum likelihood ability estimates for the students.
ex5a_eap.txt	Expected a-posterior ability estimates for the students.
ex5a_shw.txt	Selected results of the analysis.
ex5a_itn.txt	The results of the traditional item analyses.

(The last four files will be created when the command file is executed.)

2.6.1.2 Syntax

The command file used in this sample analysis for a Latent Variable *t*-Test (Six Items) is `ex5a.cqc`. It is shown in the code box below, and explained line-by-line in the list that follows the code.

`ex5a.cqc`:

```
1  datafile ex5_dat.txt;
2  title     Australian TIMSS Mathematics Data--First Six Items;
3  format    gender 17 level 18 gbyl 19 responses 20-25;
4  labels  << ex5_lab.txt;
5  key  134423 ! 1;
6  regression level;
7  model item;
8  estimate !fit=no;
9  show cases ! estimate=mle >> Results/ex5a_mle.txt;
10 show cases ! estimate=eap >> Results/ex5a_eap.txt;
11 show ! tables=3 >> Results/ex5a_shw.txt;
12 itanal >> Results/ex5a_itn.txt;
```

- **Line 1**
  Indicates the name and location of the data file.

- **Line 2**
  Gives a title for this analysis. The text that is given after the command `title` will appear on the top of any printed output. If a title is not provided, the default, `ConQuest: Generalised Item Response Modelling Software`, will be used.

- **Line 3**
  The `format` statement describes the layout of the data in the file `ex5_dat.txt`. This format indicates that a code for gender is located in column 17, a code for level is located in column 18, column 19 contains the code for a variable we have called `gbyl`, and responses are to be read from columns 20 through 25. We have not given a name to the responses, so they will be referred to as `item`.

- **Line 4**
  A set of labels for the items are to be read from the file `ex5_lab.txt`.

  > **NOTE:** The file `ex5_lab.txt` contains labels for all 158 items. These are all read and held in memory by ACER ConQuest, even though we are only using the first six items in this analysis.

- **Line 5**
  The argument of the `key` statement identifies the correct response for each of the six multiple choice test items. In this case, the correct answer for item 1 is 1, the correct answer for item 2 is 3, the correct answer for item 3 is 4, and so on. The length of the `key` statement argument is six characters, which is the length of the response block given in the `format` statement. The `key` statement option indicates that each correct answer will be recoded to 1. By default, incorrect answers will be recoded to 0.

  > **NOTE:** These data contain three kinds of missing-response data. The codes for these missing-response data are *.* (a period), *M*, and *R*. In this analysis, ACER ConQuest will treat *.* as missing-response data, since it is one of the default missing-response codes. Those data coded *M* and *R* will be treated as incorrect, because these codes do not match the values in the `key` statement argument.

- **Line 6**
  The independent variables that we want to include as predictors of the latent variable are included as arguments in the `regression` statement. By including the variable `level` as the argument here, we are instructing ACER ConQuest to regress latent

ability onto `level`; and in this case, since `level` is coded *0* (lower grade) and *1* (upper grade), ACER ConQuest will estimate the difference between the means of these two groups. The `regression` statement is used to describe the ACER ConQuest population model.

- **Line 7**
  The `model` statement here contains only the term `item` because we are dealing with single-faceted dichotomous data.

- **Line 8**
  The `estimate` statement is used to initiate the estimation of the model. The `fit=no` option is included because in this sample analysis we are not concerned with the item fit and it will save time if the fit statistics are not computed.

  > **TIP:** If you want to regress the latent variable onto a categorical variable, then the categorical variable must first be appropriately recoded. For example, dummy coding or contrast coding can be used. A variable used in regression must be a numerical value, not merely a label. For example, gender would normally be coded as *0* and *1* so that the estimated regression is the estimated difference between the group means. Remember that the specific interpretation of the latent regression parameters depends upon the coding scheme that you have chosen for the categorical variable.

- **Line 9**
  The `show` statement produces a display of the results from fitting the model. Here the `cases` argument is used to request a set of ability estimates for the students. The `estimates=mle` option indicates that maximum likelihood estimates of the ability are requested, and they are redirected to the file `ex5a_mle.txt`. When case estimates are requested, both the option indicating the type of estimate and redirection to a file are required.

- **Line 10**
  As for line 9, only we are requesting expected a-posteriori ability estimates rather than maximum likelihood ability estimates be written to the file `ex5a_eap.txt`. In Latent Estimation and Prediction in Chapter 3, the difference between these two types of ability estimates is described.

- **Line 11**
  This third `show` statement writes the third results table to the file `ex5a_shw.txt`. This table contains the parameter estimates for the population model.

- **Line 12**
  The `itanal` statement produces some traditional item statistics and writes them to the file `ex5a_itn.txt`.

2.6.1.3 Running the t-Test Sample Analysis

To run this sample analysis, start the GUI version of ACER ConQuest and open the control file `Ex5a.cqc`.

Select `Run`→`Run All`. ACER ConQuest will begin executing the statements that are in the file `ex5a.cqc`; and as they are executed, they will be echoed in the Output window. When ACER ConQuest reaches the `estimate` statement, it will begin fitting Rasch's simple logistic model to the data; as it does so, it will report on the progress of the estimation. Figure 2.42 shows an extract of the information that is reported as ACER ConQuest iterates to a solution. This figure differs slightly from that shown in Figure 2.8 in that it contains two regression coefficients rather than the overall mean. The first regression coefficient is the `CONSTANT`, and the second is the regression c oefficient of the variable `level` in the regression of latent ability onto `level`.

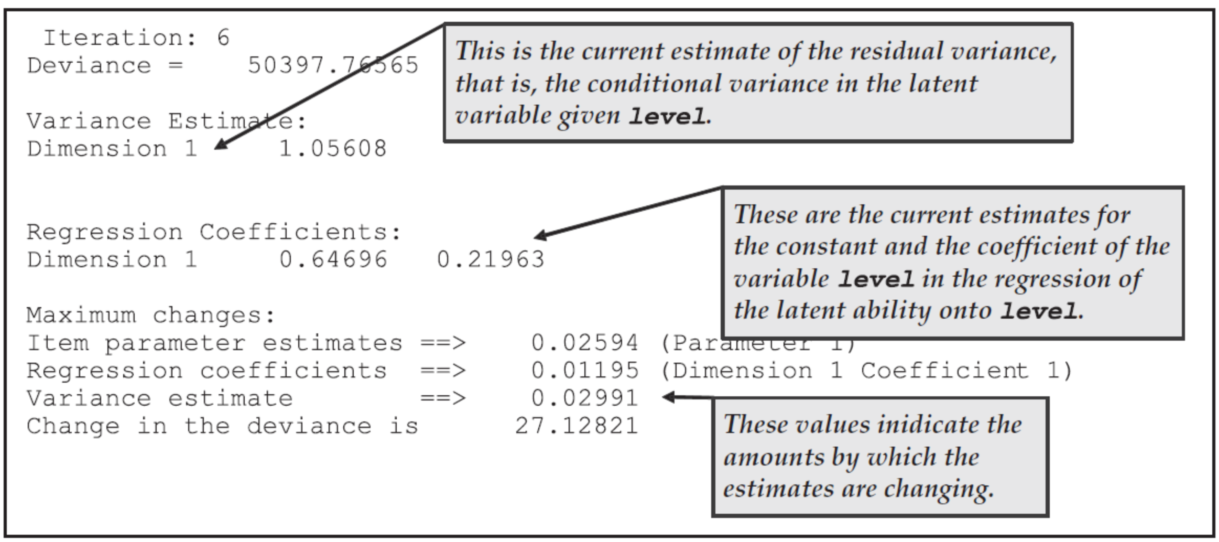


Figure 2.42: Reported Information on Estimation Progress for `ex5a.cqc`

Figure 2.43 shows the contents of the file `ex5a_shw.txt`. The values reported here are the parameter estimates for the population component of the ACER ConQuest model — in this case, a regression of the latent ability onto grade level. In these data, the `level` variable was coded as *0* for the lower grade and *1* for the upper grade, so the results shown in Figure 2.43 indicate that the estimated mean of the lower grade is 0.671 and the mean of the upper grade is 0.231 higher (mean of higher grade=0.902). The conditional variance in the latent variable is estimated to be 1.207. If an item response model is fitted without the regression variable, the estimated mean and variance of the latent ability are 0.80 and 1.219 respectively.[17]

```
===============================================================================
Australian TIMSS Mathematics Data--First Six Items        Sat Oct 28 11:33 2006
TABLES OF POPULATION MODEL PARAMETER ESTIMATES
===============================================================================
REGRESSION COEFFICIENTS

Regression Variable

CONSTANT                     0.671 ( 0.020)
level                        0.231 ( 0.027)
-----------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
===============================================
COVARIANCE/CORRELATION MATRIX

Dimension

 Dimension 1
----------------------------------
Variance                     1.207
----------------------------------
An asterisk next to a parameter estimate indicates that it is constrained

==================================

RELIABILITY COEFFICIENTS
------------------------
Dimension: (Dimension 1)
-----------------------
 MLE Person separation RELIABILITY:   0.350
 WLE Person separation RELIABILITY:  Unavailable
 EAP/PV RELIABILITY:                  0.561
```

These are the estimates for the constant and the coefficient of the variable **`level`** *in the regression of the latent ability onto* **`level`**.

This is the estimate of the residual variance, that is, the conditional variance in the latent variable given **`level`**.

Figure 2.43: Population Model Parameter Estimates

The command file `ex5a.cqc` also produces the files `ex5a_mle.txt` and `ex5a_eap.txt`. These files contain latent ability estimates for each of the 6800 students in the file

[17] The current version of ACER ConQuest does not report standardised regression coefficients or standard errors for the regression parameter estimates. Plausible values can be generated (via `show cases !estimates=plausible`) and analysed to obtain estimates of standard errors and to obtain standardised regression coefficients.

`ex5_dat.txt`. The format of these files is as follows. The file `ex5a_mle.txt` contains one line of data for each student in the sample who provided a valid response to at least one of the six items that we have analysed—in this sample, 6778 students. Columns 1 through 5 contain an identification number for the case, which is the sequence number of the student in the original data file. Columns 6 through 15 contain the total score that the student attained, columns 16 through 26 contain the maximum possible score that the student could have attained, columns 27 through 37 contain the maximum likelihood estimate of the student's latent ability, and columns 38 through 48 provide an asymptotic standard error for that ability estimate. An extract from `ex5a_mle.txt` is shown in Figure 2.44.

> **EXTENSION:** The maximum likelihood estimation method does not provide finite latent ability estimates for students who receive a score of zero or students who achieve the maximum possible score on each item. ACER ConQuest produces finite estimates for zero and maximum scorers by estimating the abilities that correspond to the scores r and M–r where M is the maximum possible score and r is an arbitrarily specified real number. In ACER ConQuest, the default value for r is 0.3. This value can be changed with the `set` command argument `zero/perfect=r`.

```
         1         2         3         4
123456789012345678901234567890123456789012345678 (col. nos.)

    1      3.00       6.00   -0.00648    0.84976
    2      4.00       6.00    0.74404    0.89598
    3      1.00       6.00   -1.71624    1.10927
    4      0.30       6.00   -3.08143    1.88317
    5      3.00       6.00   -0.00648    0.84976
    6      5.00       6.00    1.72089    1.11607
    7      5.00       6.00    1.72089    1.11607
    8      1.00       6.00   -1.71624    1.10927
    9      3.00       6.00   -0.00648    0.84976
    .         .          .          .          .
    .         .          .          .          .
```

Figure 2.44: An Extract from the MLE File `ex5a\_mle.txt`

The file `ex5a_eap.txt` contains three lines of data for each student in the sample who provided a valid response to at least one of the six items that we have analysed—in this case, 20 334 lines. The first line contains an identification number, which is the sequence number of the student in the original data file. The second line contains the expected value of the student's posterior latent ability distribution—the so-called EAP ability estimate. The third line is the variance of the student's posterior latent ability distribution; this can be used as the error variance for the EAP ability estimate. An extract from `ex5a_eap.txt` is shown in Figure 2.45.

> **WARNING:** The maximum likelihood estimate is a function of the item response data only; as such, it is not influenced by the population model. The EAP estimates are a function of both the population model and the item response model, so a change in the population model will result in a change in the EAP estimates.

```
   1     0.25916    0.68392    0.61255
   2     0.74681    0.71617    0.57515
   3    -0.69164    0.72285    0.56718
   4    -1.27239    0.80593    0.46197
   5     0.25916    0.68392    0.61255
   6     1.29044    0.75783    0.52427
   7     1.29044    0.75783    0.52427
   8    -0.69164    0.72285    0.56718
   9     0.25916    0.68392    0.61255
  10    -0.20316    0.68233    0.61434
  11    -0.20316    0.68233    0.61434
  12     0.25916    0.68392    0.61255
   .
   .
```

Figure 2.45: An Extract from the EAP File `ex5a\_eap.txt`

2.6.1.4 Comparing Latent Regression with OLS Regression

If the file `ex5a_mle.txt` is merged with the `level` variable for each case, it is possible to regress the maximum likelihood ability estimates onto `level`. Similarly, if the file `ex5a_eap.txt` is merged with the `level` variable, a regression of EAP estimates onto `level` can be carried out. The results obtained from these two regression analyses can be compared (see Figure 2.43). For the purposes of this comparison, we have also fitted a model without any regressors and added the EAP ability estimates from this run to the file `ex5a.out`, which we have provided.[18]

The results of ordinary least squares (OLS) regressions of the various estimates of latent ability onto `level` are shown in Figure 2.46.

Analysis Method	*Latent Ability*		*Regression Coefficients*		*Conditional Variance*	R^2
	Mean	*Variance*	*Constant*	*Level*		
Estimator*						
MLE	0.83	2.22	0.70	0.236	2.201	0.006
EAP with Regressor	0.80	0.68	0.67	0.231	0.664	0.019
EAP without Regressor	0.80	0.67	0.73	0.128	0.669	0.006
Direct ConQuest Estimation	0.80	1.22	0.67	0.231	1.207	0.010

* These are the ability estimates computed by ConQuest and then used as dependent variables in the analyses.

Figure 2.46: OLS Regression Results Using Alternative Latent Ability Estimates

The last row of the table contains the results produced directly by ACER ConQuest. Theoretical and simulation studies by Mislevy (Mislevy, 1984, 1985) and Adams, Wilson, & Wu (1997) indicate that the ACER ConQuest results are the 'correct' results. The results in the table show that the mean of the latent ability is reasonably well estimated from all three estimators. The slight overestimation that occurs when using the MLE estimator is likely due to the ad-hoc approach that must be applied to give finite ability estimates to those students with either zero or perfect scores. The variance is overestimated by the MLE estimator and underestimated by the two EAP estimators. The overestimation of variance from the MLE ability estimator results from the independent measurement error

[18]The file `ex5a.out` contains the EAP and maximum likelihood ability estimates merged with the `level` variable for the 6800 students. The file contains one line per student, and the fields in the file are sequence number, level, maximum likelihood ability estimate (fourth field in Figure 2.44), EAP ability estimate when `level` is used as a regression variable (third field in Figure 2.45 and EAP ability estimate when no regressor is used.

component (Wright & Stone, 1979) and a slight 'outwards' bias in the MLE estimates (Lord, 1983, 1984). The underestimation of variance from the EAP ability estimators results from the fact that the EAP estimates are 'shrunken' (Lord, 1983, 1984).

> **EXTENSION:** In section 2.9, we will discuss plausible values, the use of which enables the unbiased estimation of the parameters of any submodel of the population model that is specified in the ACER ConQuest analysis and is used to generate the plausible values.

For the regression model, we note that MLE estimates are reasonably close to the ACER ConQuest results, the EAP estimates produced with the use of the regressor give results the same as those produced by ACER ConQuest, and the EAP estimates produced without the regressor overestimate the constant term and underestimate the level effect. As was the case with the means, the difference between the MLE-based estimates and the ACER ConQuest-based estimates for the constant term is likely due to the ad-hoc treatment of zero and perfect scores when ACER ConQuest generates the maximum likelihood point estimates. The EAP estimates produced with the use of the regressor give unbiased estimates of the regression coefficients, while the estimates produced with the EAP without regressor are shrunken. The conditional variances behave in the same fashion as the (unconditional) variance of the latent ability.

None of the point estimators of students' latent abilities can be relied upon to produce unbiased results for all of the parameters that may be of interest. This is particularly true for short tests, as is the case here. When tests of 15 or more items are used, both MLE and EAP estimators will produce results similar to those produced directly by ACER ConQuest.

2.6.2 b) Avoiding the Problem of Measurement Error

The differences between the regression results that are obtained from ACER ConQuest and from the use of ordinary least squares using the various point estimates of latent ability can be avoided by using longer tests. In Section 2.6.2.2 below we present the command file `ex5b.cqc`, which reads and analyses all of the items in the file `ex5_dat.txt`.

2.6.2.1 Required files

The files that we use in this second sample analysis are:

filename	content
ex5b.cqc	The command statements used for the second analysis.
ex5_dat.txt	The data.
ex5_lab.txt	The variable labels for the items.
ex5b_prm.txt	Initial values for the item parameter estimates.
ex5b_reg.txt	Initial values for the regression parameter estimates.
ex5b_cov.txt	Initial values for the variance parameter estimates.
ex5b_shw.txt	Selected results of the analysis.
ex5b_eap.txt	Expected a-posterior ability estimates for the students.

(The last two files will be created when the command file is executed.)

2.6.2.2 Syntax

`ex5b.cqc` is the command file used in the second sample analysis for a Latent Variable *t*-Test (158 Items). The file is shown in the code box below. The list underneath the code box explains each line of commands.

ex5b.cqc:

```
datafile ex5_dat.txt;
title     Australian TIMSS Mathematics Data--All Items;
format    gender 17 level 18 gbyl 19 responses 20-176;
labels  << ex5_lab.txt;
codes 0,1,2,3,4,5,M;
recode (R) (.);
key
13442341153114122133113244235223341323324535
24322331211313511511324113143242223435112411
14141221331131232441221432233421354221332511
2253131111111111111111141!1;
key
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXX2X2XXXXX22XXXXXX2XX2XXXXXXXXXXXX2XXXX
XX2X2XX2XXXXX2XXXXX2XX2XXXXXXXX2XXXXXXXXXX2X
XXXXXX22222222222222222X2!2;
key
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

```
19 XXXXXXX3X3XXXXX33XXXXXX3XX3XXXXXXXXXXXX3XXXX
20 XX3X3XX3XXXXX3XXXXX3XX3XXXXXXXX3XXXXXXXXXX3X
21 XXXXXX33333333333333333X3!3;
22 regression level;
23 model item+item*step;
24 import init_parameters << ex5b_prm.txt;
25 import init_reg_coefficients << ex5b_reg.txt;
26 import init_covariance << ex5b_cov.txt;
27 estimate !fit=no,stderr=quick;
28 set p_nodes=1000;
29 show !table=3 >> Results/ex5b_shw.txt;
30 show cases !estimate=eap >> Results/ex5b_eap.txt;
```

- **Lines 1-4**
  As for `ex5a.cqc` (discussed in Section 2.6.1.2), except the `title` statement has been changed to indicate all items are being analysed rather than the first six and the response block in the `format` statement has been enlarged to include all 158 responses.

- **Line 5**
  In this analysis, we would like to treat the data coded `R` as missing-response data and the data coded `M` as incorrect. It is necessary therefore to make an explicit list of codes that excludes the `R`. This is in contrast to the previous sample analysis in which we did not provide a code list. In that case, all data in the file were regarded as legitimate, and those responses not matching a key were scored as incorrect.

- **Line 6**
  Here the `R` code is recoded to `.` (period), one of the default missing-response codes. Strictly speaking, this recode statement is unnecessary since the absence of the `R` in the code string will ensure that it is treated as missing-response data. It is added here as a reminder that `R` is being treated as missing-response data.

- **Lines 7-21**
  The `key` statement argument is now 158 characters long because there are 158 items. This test contains a mixture of multiple choice, short answer and extended response items, so we are using three `key` statements to deal with the fact that the short answer and extended response items are already scored. The first `key` argument contains the keys for the multiple choice items; and for short answer and extended response items, the code 1 has been entered. Any matches to this `key` argument

will be recoded to 1, as shown by the option. In other words, correct answers to multiple choice items will be recoded to 1; and for the short answer and extended response items, 1 will remain as 1. All other codes will be recoded to 0 (incorrect) after the last `key` statement and any `recode` statements have been read. The second and third `key` statements contain the character `X` for the multiple choice items and 2 and 3 respectively for the short answer and extended response items. As X does not occur in the response block of the data file, these key statements will have no effect on the multiple choice items (correct answers to which have been recoded to 1 by the first `key` statement), but the short answer and extended response items will have code 2 scored as 2 and code 3 scored as 3. While the second and third `key` statements don't change the codes, they prevent the 2 and 3 codes in the short answer and extended response items from being recoded to 0, as would have occurred if only one `key` statement were used.

> **EXTENSION:** ACER ConQuest uses the Monte Carlo method to estimate the mean and variance of the marginal posterior distributions for each case. The system value `p_nodes` (The default is 2000, and this can be changed using the command `set` with the argument `p_nodes`) governs the number of random draws in the Monte Carlo approximations of the integrals that must be computed.

> **WARNING:** For cases with extreme latent ability estimates, the variance of the marginal posterior distribution may not be estimated accurately if `p_nodes` is small. Increasing `p_nodes` will improve the variance estimates. On the other hand, for EAP estimates, moderate values of `p_nodes` are sufficient.

- **Line 22**
  As for line 6 of `ex5a.cqc` (see Section 2.6.1.2).

- **Line 23**
  This `model` statement yields the partial credit model. In the previous sample analysis, all of the items were dichotomous, so a `model` statement without the `item*step` term was used. Here we are specifying the partial credit model because it will deal with the mixture of dichotomous and polytomous items in this analysis.

- **Lines 24-26**
  This analysis takes a considerable amount of time, so initial value files are used to import a set of starting values for the item, regression and variance parameter estimates.

- **Line 27**
  In this sample analysis, we are not concerned with the properties of the items, so we are specifying the `fit=no` option to speed up the analysis.

- **Line 28**
  The `set` command is used to alter some of ACER ConQuest's default values. The `p_nodes=1000` argument requests that 1000 nodes be used when EAP estimates are produced and when plausible values are drawn. The default value for `p_nodes` is 2000. Reducing this to 1000 decreases the time necessary to compute EAP estimates.

- **Line 29**
  This `show` statement writes the population model parameter estimates (table 3) to `ex5b_shw.txt`.

- **Line 30**
  This `show` statement writes a file containing the EAP ability estimate for each case.

2.6.2.3 Running the Second t-Test Sample Analysis

To run this sample analysis, start the GUI version of ACER ConQuest and open the control file `Ex5b.cqc`.

Select `Run`→`Run All`. ACER ConQuest will begin executing the statements that are in the file `ex5b.cqc`; and as they are executed, they will be echoed in the Output Window. When ACER ConQuest reaches the `estimate` statement, it will begin fitting the partial credit model to the data. In this case, only two iterations will be necessary because the initial values that were provided in the files `ex5b_prm.txt`, `ex5b_reg.txt` and `ex5b_cov.txt` are the output of a full analysis that have been performed on a previous occasion.

Figure 2.47 shows the contents of `ex5b_shw.txt`. A comparison of the results reported here with those reported in Figure 2.43 is quite interesting. Recall that the results in Figure 2.43 are from fitting a similar latent regression model to the first six items only — the set of items taken by all students. What we note is that the variance estimates are very similar, as is the regression coefficient for `level`. In fact, this similarity is quite remarkable given that the first analysis used only six of the 158 items and approximately one-fifth of the data that were actually available. The `CONSTANT` terms are quite different. The difference between the estimates for the `CONSTANT` is due to the model identification constraint. In the previous analysis, the item response model was identified by setting the mean difficulty of the first six items to zero. In this second run, the mean difficulty of all 158 items is set to zero.

```
================================================================================
Australian TIMSS Mathematics Data--All Items             Mon Oct 30 13:36 2006
TABLES OF POPULATION MODEL PARAMETER ESTIMATES
================================================================================
REGRESSION COEFFICIENTS

Regression Variable

CONSTANT                   0.338 ( 0.020)
level                      0.275 ( 0.027)
-----------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
===============================================
COVARIANCE/CORRELATION MATRIX

Dimension

 Dimension 1
----------------------------------
Variance                   1.214
----------------------------------
An asterisk next to a parameter estimate indicates that it is constrained

==================================
RELIABILITY COEFFICIENTS
------------------------
Dimension: (Dimension 1)
-----------------------
 MLE Person separation RELIABILITY:   0.869
 WLE Person separation RELIABILITY:  Unavailable
 EAP/PV RELIABILITY:                  0.893
```

These are the estimates for the constant and the coefficient of the variable **`level`** *in the regression of the latent ability onto* **`level`**.

This is the estimate of the residual variance, that is, the conditional variance in the latent variable given **`level`**.

Figure 2.47: Population Model Parameter Estimates

2.6.2.4 Comparing Latent Regression with OLS Regression for the Second Sample Analysis

As with the previous sample analysis, we produced a file of EAP ability estimates and then merged these with the `level` variable for each case. For the purposes of this comparison, we have also fitted a model without any regressors and added the EAP ability estimates from this run to the file `ex5b.out`, which we have provided.[19] Figure 2.48 shows the results of regressing these EAP estimates onto `level` and compares the results obtained with those obtained by ACER ConQuest.

Analysis Method	*Latent Ability*		*Regression Coefficients*		*Conditional Variance*	*R^2*
	Mean	*Variance*	*Constant*	*Level*		
Estimator*						
EAP with Regressor	0.49	1.08	0.335	0.277	1.065	0.02
EAP without Regressor	0.49	1.08	0.352	0.244	1.068	0.01
Direct ConQuest Estimation	0.49	1.23	0.338	0.275	1.214	0.02

* These are the ability estimates computed by ConQuest and then used as dependent variables in the analyses.

Figure 2.48: OLS Regression Results Using Alternative Latent Ability Estimates

The mean is well estimated by the EAP latent ability estimates, but as in the previous sample analysis the variance is underestimated. The degree of underestimation is much less marked than it was in the previous sample analysis, but it is still noticeable. For the regression coefficients, we note that the EAP with regressor latent ability estimates are very close to the values produced by ACER ConQuest. The EAP without regressor values are moderately biased, again due to their shrunken nature: the `CONSTANT` term is overestimated and the difference between the levels is underestimated. The conditional variances are again under-estimated by the EAP-based ability estimates.

[19]The file `ex5b.out` contains the EAP ability estimates merged with the `level` variable for all 6800 students. The file contains one line per student and the fields in the file are sequence number, the `level` variable, EAP ability estimate when `level` is used as a regression variable, and EAP ability estimate when no regressor is used.

2.6.3 c) Latent Multiple Regression

The regressions undertaken in the last two sample analyses used a single regressor, `level`, which takes two values, 0 to indicate lower grade and 1 to indicate upper grade. This effectively meant that these two sample analyses were equivalent to two-sample *t*-tests. In ACER ConQuest, up to 200 regression variables can be used simultaneously, and the regressors can be continuous numerical values. As a final sample analysis, we will show the results of analysing the data in `ex5_dat.txt` using three regressors.

2.6.3.1 Syntax

The command file for this sample analysis (`ex5c.cqc`) is given in the code box below. The only substantive difference between `ex5b.cqc` (cf. Section 2.6.2.2) and `ex5c.cqc` is in line 19, where the variables `gender` and `gbyl` are added.

`ex5c.cqc`:

```
1  datafile ex5_dat.txt;
2  title  Australian TIMSS Mathematics Data-All Items;
3  format gender 17 level 18 gbyl 19 responses 20-176;
4  labels  << ex5_lab.txt;
5  codes 0,1,2,3,4,5,M;
6  recode (R) (.);
7  key
8  1344234115311412213311324423522334132332453524322331211313511
9  5113241131432422234351124111414122133113123244122143223342135
10 42213325112253131111111111111111141!1;
11 key
12 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX2X2XXXXX22
13 XXXXXX2XX2XXXXXXXXXXXX2XXXXXX2X2XX2XXXXX2XXXXX2XX2XXXXXXXX2XX
14 XXXXXXXX2XXXXXXX22222222222222222X2!2;
15 key
16 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX3X3XXXXX33
17 XXXXXX3XX3XXXXXXXXXXXX3XXXXXX3X3XX3XXXXX3XXXXX3XX3XXXXXXXX3XX
18 XXXXXXXX3XXXXXXX33333333333333333X3!3;
19 regression level gender gbyl;
20 model item+item*step;
21 import init_parameters << ex5c_prm.txt;
```

```
import init_reg_coefficients << ex5c_reg.txt;
import init_covariance << ex5c_cov.txt;
estimate !fit=no,stderr=quick;
show !table=3 >> Results/ex5c_shw.txt;
set p_nodes=1000;
show cases !estimate=eap >> Results/ex5c_eap.txt;
```

> **NOTE:** The ACER ConQuest population model is a regression model that assumes normality of the underlying latent variable, conditional upon the values of the regression variables. If you want to regress latent ability onto categorical variables or to specify interactions between variables, then appropriate contrasts and interaction terms must be created external to the ACER ConQuest program. For instance, in this sample analysis, we have constructed the interaction between `gender` and `level` by constructing the new variable `gbyl`, which is the product of `gender` and `level`, and adding it to the data file.

2.6.3.2 Running the Latent Multiple Regression Analysis

Figure 2.49 shows the contents of `ex5c_shw.txt`, the population model parameter estimates for this third latent multiple regression sample analysis. The results reported in the figure show that the main effects of grade (`level`) and `gender` are 0.251 and –0.030, respectively, while the interaction between `gender` and grade (`gbyl`) is 0.052. The `CONSTANT` (0.351) is the estimated mean for male students in the lower grade. The estimated mean of female students in the lower grade is 0.321 (=0.351–0.030), of male students in the upper grade is 0.602 (=0.351+0.251), and of female students in the upper grade is 0.624 (=0.351+0.251 – 0.030+0.052).

2.6.4 Summary

In this section, we have seen how to use ACER ConQuest to fit unidimensional latent regression models. Our sample analyses have been concerned with using categorical regressors, but ACER ConQuest can analyse up to 200 continuous or categorical regressors. Some key points covered in this section are:

- Secondary analyses using EAP and MLE ability estimates do not produce results that are equivalent to the 'correct' latent regression results. The errors that can be

```
================================================================================
Australian TIMSS Mathematics Data-All Items                Mon Oct 30 16:39 2006
TABLES OF POPULATION MODEL PARAMETER ESTIMATES
================================================================================
REGRESSION COEFFICIENTS

Regression Variable

CONSTANT                    0.351 ( 0.027)
level                       0.251 ( 0.037)
gender                     -0.030 ( 0.041)
gbyl                        0.052 ( 0.054)
-----------------------------------------------
An asterisk next to a parameter estimate indicate
===============================================
COVARIANCE/CORRELATION MATRIX

Dimension

 Dimension 1
----------------------------------
Variance                    1.213
----------------------------------
An asterisk next to a parameter estimate indicates that it is constrained

==================================

RELIABILITY COEFFICIENTS
------------------------
Dimension: (Dimension 1)
-----------------------
 MLE Person separation RELIABILITY:  Unavailable
 WLE Person separation RELIABILITY:  Unavailable
 EAP/PV RELIABILITY:                  0.893
```

These are the estimates for the constant and the coefficients of the variables **`level`**, **`gender`** *and* **`gbyl`** *in the regression of the latent ability onto* **`level`**, **`gender`** *and* **`gbyl`**.

This is the estimate of the residual variance, that is, the conditional variance in the latent variable given **`level`**, **`gender`** *and* **`gbyl`**.

Figure 2.49: Population Model Parameter Estimates for the Latent Multiple Regression Sample Analysis

made in a secondary analysis of latent ability estimates are greater when measurement error is large.

- The `key` command can be used with a mixture of dichotomous and polytomous items.
- The `show` command can be used to create files of ability estimates. ACER ConQuest provides both EAP and maximum likelihood ability estimates.
- The `import` command can be used to read files of initial values for parameter estimates.

2.7 Differential Item Functioning

Within the context of Rasch modelling an item is deemed to exhibit differential item functioning (DIF) if the response probabilities for that item cannot be fully explained by the ability of the student and a fixed set of difficulty parameters for that item. Through the use of its multi-faceted modelling capabilities, and more particularly its ability to model interactions between facets, ACER ConQuest provides a powerful set of tools for examining DIF.

In this section three examples are considered. In the first, ACER ConQuest is used to explore the existence of DIF with respect to gender in a short multiple-choice test. This is a traditional DIF analysis because it is applied to dichotomously scored items and examines DIF between two groups—that is, it uses a binary grouping variable. In the second example DIF is explored when the grouping variable is polytomous—in fact the grouping variable defines eight groups of students. Finally, in the third example DIF in some partial credit items is examined.

2.7.1 a) Examining Gender Differences in a Multiple Choice Test

2.7.1.1 Required files

The data used in this first example are the TIMSS data that were described in the previous section (2.6).

The files used in this example are:

filename	content
ex6a.cqc	The command lines used for the first analysis.
ex5_dat.txt	The data.
ex6_lab.txt	A file of labels for the items.
ex6a_shw.txt	Selected results from the analysis.

2.7.1.2 Syntax

The control code for analysing these data is contained in `ex6a.cqc`, as shown in the code box below. Each line of commands is explained in the list that follows the code.

`ex6a.cqc`:

```
1 datafile ex5_dat.txt;
2 title TIMSS Mathematics--First Six Items--Gender Differences;
3 format book 16 gender 17 level 18 gbyl 19 responses 20-25;
4 labels << ex6_lab.txt;
5 key 134423 ! 1;
6 model item-gender+item*gender;
7 estimate !fit=no,stderr=empirical;
8 show !table=2>> Results/ex6a_shw.txt;
9 plot icc! gins=1:2,overlay=yes,legend=yes;
10 plot icc! gins=3:4,overlay=yes,legend=yes;
11 plot icc! gins=5:6,overlay=yes,legend=yes;
12 plot icc! gins=7:8,overlay=yes,legend=yes;
13 plot icc! gins=9:10,overlay=yes,legend=yes;
14 plot icc! gins=11:12,overlay=yes,legend=yes;
```

- **Line 1**
  The data in `ex5_dat.txt` is to be used.

- **Line 2**
  Sets the title.

- **Line 3**
  Note that in this format we are reading the explicit variables `book`, `gender`, `level` and the product of `gender` and `level` from columns 16, 17, 18 and 19 respectively.

- **Line 4**
  Note that the labels file for this analysis contains labels for `book`, `gender` and `item`.

- **Line 5**
  Gives the scoring key.

- **Line 6**
  The `model` statement has three terms. These three terms involve two facets, `item` and `gender`. So, as ACER ConQuest passes over the data, it will identify all possible combinations of the `item` and `gender` variables and construct 12 (six items by 2 genders) *generalised* items. The `model` statement requests that ACER ConQuest describes the probability of correct responses to these generalised items using an `item` main effect, a `gender` main effect and an interaction between `item` and `gender`.

  The first term will yield a set of item difficulty estimates, the second term will give the mean ability of the male and female students and the third term will give an estimate of the difference in the difficulty of the items for the two gender groups. Note, a negative sign (`-`) has been used in front of the `gender` term. This ensures that the gender parameters will have the more natural orientation of a higher number corresponding to a higher mean ability.

- **Line 7**
  Two options have been included with the `estimate` command. `fit=no`, means that fit statistics will not be computed, and `stderr=empirical` means that the more time consuming (and more accurate) method will be used to calculate asymptotic standard error estimates for the items. The more accurate method has been chosen for this analysis since the comparison of estimates of some parameters to their standard errors is used in judging whether there is DIF.

- **Line 8**
  The `show` command will write table 2 to the file `ex6a_shw.txt`.

- **Lines 9-14**
  Plots the item characteristic curves for each of the six items. Because this run of ACER ConQuest uses a multi-faceted model that involves six items and two genders there are actually 12 *generalised* items that are analysed. In the `model` statement the `item` facet is mentioned first and the `gender` facet is mentioned second, as a consequence the `gender` facet reference cycles fastest in the referencing of generalised items. That is, generalised item one corresponds to item one and gender one; generalised item two corresponds to item one and gender two; generalised item three

corresponds to item two and gender one; generalised item four corresponds to item two and gender two; and so on.

Each `plot` command plots the item characteristic curves for two generalised items. For example the first command plots generalised items one and two, which corresponds to a plot of item one for the two gender groups separately. The `overlay=yes` option results in both item characteristic curves being plotted in the same graph.

2.7.1.3 Running the Test for DIF

To run this sample analysis, start the GUI version of ACER ConQuest and open the control file `Ex6a.cqc`.

Select `Run`→`Run All`. ACER ConQuest will begin executing the statements that are in the file `ex6a.cqc`; and as they are executed they will be echoed in the Output Window. When it reaches the `estimation` command ACER ConQuest will begin fitting a multi-faceted model to the dichotomous data. TThe item parameter estimates will be written to the file `ex6a_shw.txt`.

The contents of `ex6a_shw.txt` are shown in Figure 2.50. The figure contains three tables, one for each of the terms in the `model` statement.

The **first table** shows the item difficulty parameter estimates for each of the six items.

The **second table** shows the estimates for the gender differences in ability estimates. A negative sign (-) was used for the gender term in the item response model so these results indicate that the male students have performed more poorly than the female students. The actual parameter estimate for the male students is three times larger than its standard error estimate so the difference between the male and female means is obviously significant. The chi-square value of 9.63 on one degree of freedom is consistent with this finding. The conclusion that can be drawn here is that the male mean performance is lower than that of the females, this DOES NOT indicate differential item functioning. Further, the estimated difference of 0.114 is small at just over 10% of a student standard deviation.[20]

The **third table** gives the interaction between the item and gender facets. The estimate of 0.060 for item BSMMA01 and males indicates that 0.060 must be added to the difficulty of this item for male students, similarly –0.060 must be added for the females. That is,

[20]The standard deviation is around 1.1 (See section 2.6). The results reported here should not be extrapolated to the Australian TIMSS data. The significance testing done here does not take account of the design effects that exist in TIMSS due to the cluster sampling that was used, further they are based on a random selection of half of the TIMSS data set.

female students found this item to be relatively easier than did the males. The results in this table show that three items (BSMMA03, BSMMA05 and BSMMA06) are relatively easier for males than females, two items (BSMMA01 and BSMMA04) are relatively easier for females than males, and one item (BSMMA02) has the same difficulty. The significant chi-square (155.00, df=5) also shows the existence of DIF.

> **NOTE:** By including the main effect, gender, in the item response model, estimates of the mean scores for male and female students has been obtained. An alternative approach that would have achieved an identical result would have been to place the gender variable in the population model. It would not be appropriate to include gender in both the item response and the population models since this would make the model unidentified.

> **WARNING:** The current version of ACER ConQuest assumes independence between the parameter estimates when computing the chi-square test of parameter equality.

While this analysis has shown the existence of DIF in these items it is the magnitude of that DIF that will determine if the effect of that DIF is of substantive importance. For example, the first item (BSMMA01) is significantly more difficult for males than females but the difference estimate is just 0.12 logits. If all of the items exhibited DIF of this magnitude it would shift the male ability distribution by just over 10% of a student standard deviation. With just one item having this DIF, the effect is much smaller. The fourth item (BSMMA04) exhibits much more DIF. In fact if all of the items in the test had behaved like this item the estimated mean score for the males would be 0.582 logits lower than that of the females, that is more than 50% of a student standard deviation.

Figure 2.51 shows the item characteristic curves for Item 4 for males and females separately. The dark (blue) curves are for males, and the light (green) curves are for females. It can be seen that, given a particular ability level, the probability of being successful on this item is higher for females than for males, i.e., females find this item easier than males.

2.7.2 b) Examining DIF When the Grouping Variable Is Polytomous

ACER ConQuest can also be used to examine DIF when the grouping variable is polytomous, rather than dichotomous, as is the case with gender.

```
================================================================================
TIMSS Mathematics--First Six Items--Gender Differences     Mon Oct 30 21:15 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
================================================================================
TERM 1: item
-------------------------------------------------------------------------------------
   VARIABLES                                   UNWEIGHTED FIT          WEIGHTED FIT
---------------                         -----------------------   --------------------
     item            ESTIMATE  ERROR    MNSQ       CI         T    MNSQ       CI
-------------------------------------------------------------------------------------
 1   BSMMA01           0.048   0.026
 2   BSMMA02          -0.663   0.029
 3   BSMMA03          -0.368   0.027
 4   BSMMA04           0.562   0.026
 5   BSMMA05           0.918   0.025
 6   BSMMA06          -0.498*
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.998
Chi-square test of parameter equality =    2511.21,  df = 5,  Sig Level = 0.000
================================================================================
TERM 2: (-)gender
-------------------------------------------------------------------------------------
   VARIABLES                                   UNWEIGHT                          IT
---------------                         -----------                       ----------
    gender           ESTIMATE  ERROR    MNSQ       C
-------------------------------------------------------------------------------------
 1   male             -0.057   0.018
 2   female            0.057*
-------------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates
Separation Reliability Not Applicable
Chi-square test of parameter equality =        9.63,
================================================================================
TERM 3: item*gender
-------------------------------------------------------------------------------------
          VARIABLES                                     UNWEIGHTED FIT
WEIGHTED FIT
------------------------------              -----------------------   -----
     item          gender          ESTIMATE  ERROR    MNSQ       CI         T    MNSQ
-------------------------------------------------------------------------------------
 1   BSMMA01    1   male             0.060   0.026
 2   BSMMA02    1   male             0.029   0.029
 3   BSMMA03    1   male            -0.100   0.027
 4   BSMMA04    1   male             0.291   0.026
 5   BSMMA05    1   male            -0.062   0.025
 6   BSMMA06    1   male            -0.218*
 1   BSMMA01    2   female          -0.060*
 2   BSMMA02    2   female          -0.029*
 3   BSMMA03    2   female           0.100*
 4   BSMMA04    2   female          -0.291*
 5   BSMMA05    2   female           0.062*
 6   BSMMA06    2   female           0.218*
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.970
Chi-square test of parameter equality =     155.00,  df = 5,  Sig Level = 0.000
================================================================================
```

Fit tests were not requested so this area is blank.

Shows that the male students scored 0.114 lower than the female students. The fact that the parameter estimate is more than twice its standard error and the fact that the chi-square p-value is small indicate that this difference is statistically significant.

There is significant variance here so there is some evidence of DIF. Items 3, 4 and 6 look somewhat problematic.

Figure 2.50: Parameter Estimates for DIF Examination in a Multiple Choice Test

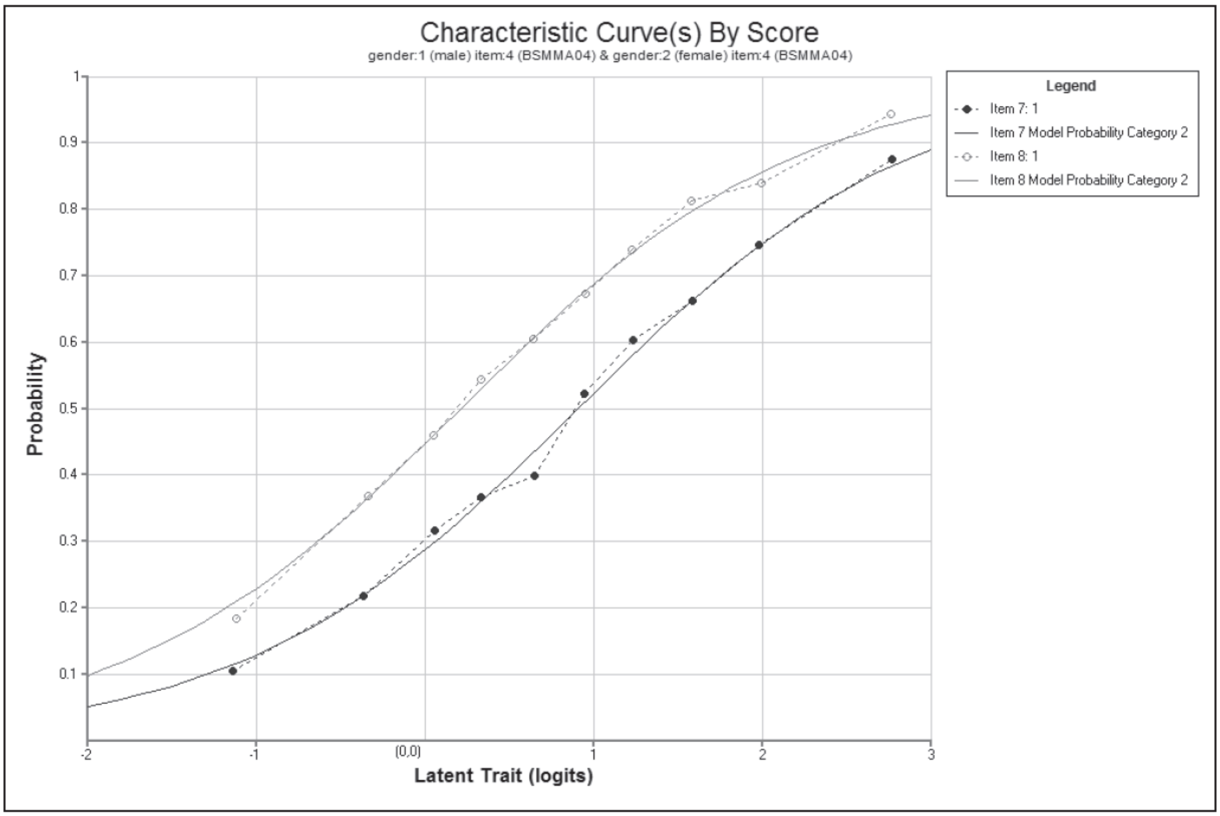


Figure 2.51: Item Characteristic Curves for Generalised Items Seven and Eight (Item 4, Males and Females)

2.7.2.1 Required files

In the TIMSS design the test items were allocated to eight different testing booklets and students were allocated one of the eight booklets at random. One way of testing whether the rotation scheme was implemented successfully is to estimate the mean ability estimates for the students who were assigned each booklet and to see if there is any evidence of DIF across the booklets.

The files that we will use in this example are:

filename	content
ex6b.cqc	The command lines used for the second analysis.
ex5_dat.txt	The data.
ex6_lab.txt	A file of labels for the items.
ex6b_shw.txt	Selected results from the analysis.

2.7.2.2 Syntax

The contents of the control file, `ex6b.cqc`, used in this analysis to examine booklet effect in a MC test, is shown in the code box below. The only command that is different here to `ex6a.cqc` (see Section 2.7.1.2) is the `model` statement, in which the variable `book` rather than `gender` is used.

`ex6b.cqc:`

```
datafile ex5_dat.txt;
title Australian TIMSS Mathematics Data--First Six Items--Booklet Differences;
format  book 16 gender 17 level 18 gbyl 19 responses 20-25;
labels  << ex6_lab.txt;
key  134423 ! 1;
model item+book+item*book;
estimate !fit=no;
show !table=2 >> Results/ex6b_shw.txt;
```

2.7.2.3 Running the Test for DIF when the Grouping Variable is Polytomous

After running this analysis using the same procedures as described for previous examples, the file `ex6b_shw.txt` will be produced, the contents of which are shown in Figure 2.52.

This figure shows that there is no statistically significant *book effect* and that there is no between booklet DIF.

2.7.3 c) DIF for Polytomous Items

As a final example on DIF, a set of polytomous items is examined.

2.7.3.1 Required files

The data were collected by Adams et al. (1991) as a part of their study of science achievement. The set of items that are analysed formed an instrument that assessed students' understanding of force and motion.

The files used in this example are:

filename	content
ex6c.cqc	The command lines used for the third set of analyses.
ex6_dat.txt	The data.
ex6c_lab.txt	The variable labels for the items on the test.
ex6c_shw.txt	The results of an analysis that includes gender by step interactions.
ex6d_shw.txt	The results from an analysis that does not include gender by step interactions.

2.7.3.2 Syntax

The control code for this example (`ex6c.cqc`) is shown in the code box below. `ex6c.cqc` is very similar to the command files of earlier examples in this section (`ex6a.cqc` and `ex6b.cqc`), so only the distinguishing aspects of `ex6c.cqc` are commented upon in the list underneath the code box.

Note that in this case the control code will actually run two ACER ConQuest analyses.

`ex6c.cqc`:

```
datafile ex6_dat.txt;
format responses 10-18 grade 118 gender 119!tasks(9);
set warnings=no;
model tasks - gender + tasks*gender + gender*tasks*step;
labels << ex6c_lab.txt;
```

```
================================================================================
Australian TIMSS Mathematics Data--First Six Items--BookletTue Oct 31 12:30 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
================================================================================
TERM 1: item
------------------------------------------------------------------------------------------
   VARIABLES                                UNWEIGHTED FIT           WEIGHTED FIT
---------------                        -----------------------  -----------------------
     item            ESTIMATE  ERROR     MNSQ        CI       T    MNSQ       CI        T
------------------------------------------------------------------------------------------
 1   BSMMA01           0.051   0.026     0.89 ( 0.97, 1.03) -6.7   0.92 ( 0.98, 1.02) -6.4
 2   BSMMA02          -0.662   0.028     1.02 ( 0.97, 1.03)  1.1   1.01 ( 0.97, 1.03)  0.5
 3   BSMMA03          -0.375   0.027     0.94 ( 0.97, 1.03) -3.6   0.96 ( 0.97, 1.03) -2.8
 4   BSMMA04           0.584   0.025     0.99 ( 0.97, 1.03) -0.3   0.99 ( 0.98, 1.02) -0.6
 5   BSMMA05           0.912   0.025     1.14 ( 0.97, 1.03)  7.8   1.10 ( 0.98, 1.02)  8.7
 6   BSMMA06          -0.509*            1.07 ( 0.97, 1.03)  4.1   1.05 ( 0.97, 1.03)  2.9
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.99
Chi-square test of parameter e                         f = 5,  Sig Level = 0.000
================================================               =========================
TERM 2: book
------------------------------------------------               ---------------------------
   VARIABLES                                                   FIT       WEIGHTED FIT
---------------                                                -----  -----------------------
     book            ESTIMATE                                       T    MNSQ       CI        T
------------------------------------------------------------------------------------------
 1   book1             0.035   0.047     1.01 ( 0.91,                        .91, 1.09)  0.5
 2   book2            -0.000   0.048     1.01 ( 0.90,                        .90, 1.10)  0.3
 3   book3            -0.026   0.048     1.01 ( 0.91,                        .91, 1.09)  0.4
 4   book4            -0.058   0.048     0.97 ( 0.91,                        .91, 1.09) -0.6
 5   book5            -0.030   0.048     0.98 ( 0.91,                        .91, 1.09) -0.5
 6   book6             0.038   0.048     0.96 ( 0.90,                        .91, 1.09) -0.7
 7   book7             0.019   0.049     0.99 ( 0.90,                        .90, 1.10)  0.0
 8   book8             0.022*            0.98 ( 0.90,                        .90, 1.10) -0.6
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.000
Chi-square test of parameter equality =       3.48,  df = 7,  Sig Level = 0.837
================================================================================
TERM 3: item*book
--------------------------------------------------------------------------------------------
          VARIABLES                              UNWEIGHTED FIT           WEIGHTED FIT
-------------------------                  -----------------------  -----------------------
     item          book   ESTIMATE  ERROR    MNSQ        CI       T    MNSQ       CI        T
--------------------------------------------------------------------------------------------
 1   BSMMA01  1    book1    0.094   0.068   0.87 ( 0.91, 1.09) -2.8   0.91 ( 0.93, 1.07) -2.8
 2   BSMMA02  1    book1   -0.026   0.074   1.05 ( 0.91, 1.09)  1.0   1.00 ( 0.91, 1.09) -0.0
 3   BSMMA03  1    book1   -0.026   0.071   0.93 ( 0.91, 1.09) -1.5   0.96 ( 0.92, 1.08) -1.0
 4   BSMMA04  1    book1    0.008   0.066   0.96 ( 0.91, 1.09) -0.9   0.96 ( 0.94, 1.06) -1.2
 5   BSMMA05  1    book1   -0.058   0.066   1.05 ( 0.91, 1.09)  1.0   1.03 ( 0.94, 1.06)  1.0
 6   BSMMA06  1    book1    0.008*          1.09 ( 0.91, 1.09)  1.9   1.05 ( 0.92, 1.08)  1.2
 1   BSMMA01  2    book2   -0.047   0.069   0.89 ( 0.90, 1.10) -2.3   0.92 ( 0.93, 1.07) -2.1
 2   BSMMA02  2    book2    0.008   0.075   0.99 ( 0.90, 1.10) -0.1   1.02 ( 0.91, 1.09)  0.4
 3   BSMMA03  2    book2    0.032   0.072   0.94 ( 0.90, 1.10) -1.3   0.95 ( 0.92, 1.08) -1.1
 4   BSMMA04  2    book2   -0.035   0.067   0.92 ( 0.90, 1.10) -1.7   0.94 ( 0.94, 1.06) -2.0
 5   BSMMA05  2    book2    0.077   0.068   1.09 ( 0.90, 1.10)  1.9   1.06 ( 0.94, 1.06)  1.8
 6   BSMMA06  2    book2   -0.035*          1.09 ( 0.90, 1.10)  1.7   1.05 ( 0.91, 1.09)  1.2
 ..........................................................................................
 ..........................................................................................
 1   BSMMA01  8    book8    0.089*          0.92 ( 0.90, 1.10) -1.7   0.93 ( 0.93, 1.07) -2.1
 2   BSMMA02  8    book8    0.022*          1.07 ( 0.90, 1.10)  1.5   1.04 ( 0.91, 1.09)  0.8
 3   BSMMA03  8    book8   -0.087*          0.97 ( 0.90, 1.10) -0.6   0.99 ( 0.92, 1.08) -0.2
 4   BSMMA04  8    book8   -0.022*          1.02 ( 0.90, 1.10)  0.4   1.01 ( 0.94, 1.06)  0.4
 5   BSMMA05  8    book8   -0.018*          1.16 ( 0.90, 1.10)  3.1   1.11 ( 0.94, 1.06)  3.5
 6   BSMMA06  8    book8    0.017*          1.07 ( 0.90, 1.10)  1.4   1.04 ( 0.92, 1.08)  0.9
--------------------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.240
Chi-square test of parameter equality =      44.95,  df = 35,  Sig Level = 0.121
================================================================================
```

These parameter estimates are small, relative to their standard errors.

The book parameter estimates are not significantly different from zero.

Figure 2.52: Parameter Estimates for DIF Examination Across Booklets

```
6  estimate;
7  show!table=1:2 >> Results/ex6c_shw.txt;
8  plot expected! gins=1:2,overlay=yes,legend=yes;
9  reset;
10
11 datafile ex6_dat.txt;
12 format responses 10-18 grade 118 gender 119!tasks(9);
13 set warnings=no;
14 model tasks-gender+tasks*gender+tasks*step;
15 labels << ex6c_lab.txt;
16 estimate!fit=no,stderr=empirical;
17 show!table=1:2 >> Results/ex6d_shw.txt;
18 plot expected! gins=1:2,overlay=yes,legend=yes;
```

- **Line 4**
  This `model` includes four terms. Two main effects, `tasks` and `gender`, give the difficulty of each of the tasks and the means of the two gender groups. The interaction `tasks*gender` models the variation in difficulty of the task between the two genders and finally the `gender*tasks*step` term models differing step structures for each task and gender.

  > **EXTENSION:** In this example randomly chosen students from both an upper and lower grade responded to all of the tasks so the use of grade as a regressor is not necessary to produce consistent estimates of the item response model parameters.
  >
  > If the sub-samples of students who respond to specific test tasks were systematically different in their latent ability distribution then the use of a regressor will be necessary to produce consistent parameter estimates for the item response model (Mislevy & Sheehan, 1989).

- **Line 9**
  The `reset` command separates sets of analyses to be run.

- **Line 14**
  This `model` command is similar to the previous one in that it has four terms. The difference is that the final term does not include variation between males and females in the task's step structure. Comparing the fit of this model to the model given by line 4, we can assess the need for a step structure that is different for male and female students.

2.7.3.3 Running the Analysis

After this analysis is run using the same procedures as described for previous examples, the files `ex6c_shw.txt` and `ex6d_shw.txt` will be produced. An extract of `ex6c_shw.txt` is given in Figure 2.53, it shows that there is no difference between the overall performance of male and female students and that there is no interaction between gender and task difficulty. In this figure the parameter estimates for the term `gender*tasks*step` are not shown because the easiest way to test whether the step structure is the same for the male and female students is to compare the deviance of the two models that were fitted by the code in `ex6c.cqc`.

The results reported in Figure 2.53 show that the model with a step structure that is invariant to gender does not fit as well as the model with a step structure that varies with gender. The conclusion that can be drawn from these analyses is that while the overall male and female performance is equivalent, as are the difficulty parameters for each of the tasks it appears that male and female students have differing step structures. A closer examination of the difference in the step structures between male and female students would appear to be required.

To illustrate the differences between these two models, the expected score curves have been plotted for the first two generalised items for each model. The plots are shown in Figure 2.54. The first plot shows the expected score curves when a different step structure is used for male and female students, while the second plot shows the expected score curves when a common step structure is used. In the second plots the curves are parallel, in the sense that they have the same shape but are just displaced on the horizontal axes. In the first plots the expected curves take a different shape, and in fact cross.

2.7.4 Summary

In this section we have illustrated how ACER ConQuest can be used to examine DIF with dichotomous items and polytomous items, and how DIF can be explored where the grouping variable is polytomous.

Some key points covered in this section are:

- Modelling DIF can be done through adding an item-by-facet interaction term in the `model` statement.
- Item characteristic curves can be plotted with the overlay option.
- A comparison of model fit can be carried out using the deviance statistic.

```
================================================================================
ConQuest: Generalised Item Response Modelling Software     Sun Jan 09 09:20 2005
SUMMARY OF THE ESTIMATION
================================================================================

Estimation method was: Gauss-Hermite Quadrature with 15 nodes
Assumed population distribution was: Gaussian
Constraint was: DEFAULT
The Data File: ex6.dat
The format:  responses 10-18 grade 118 gender 119!tasks(9)
The regression model:
Grouping Variables:
The item model: tasks-gender+tasks*gender+gender*tasks*step
Sample size: 2564
Final Deviance:     18834.53602
Total number of estimated parameters: 55
The number of iterations: 131
Termination criteria:  Max iterations=1000, Parameter Change= 0.00010
                       Deviance Change= 0.00010
Iterations terminated because the deviance convergence              reached
Random number generation seed:    1.00000
Number of nodes used when drawing PVs: 2000
Number of nodes used when computing fit: 1000
Nu             ble values to draw: 5
Ma             of iterations without a deviance impro
Ma             of Newton steps in M-step: 10
Value for obtaining finite MLEs for zero/perfects:

================================================================================
ConQuest: Generalised Item Response Modelling Software     Sun Jan 09 09:20 2005
SUMMARY OF THE ESTIMATION
================================================================================

Estimation method was: Gauss-Hermite Quadrature with 15 nodes
Assumed population distribution was: Gaussian
Constraint was: DEFAULT
The Data File: ex6.dat
The format:  responses 10-18 grade 118 gender 119!tasks(9)
The regression model:
Grouping Variables:
The item model: tasks-gender+tasks*gender+tasks*step
Sample size: 2564
Final Deviance:     18919.36178
Total number of estimated parameters: 37
The number of iterations: 111
Termination criteria:  Max iterations=1000, Parameter Change= 0.00010
                       Deviance Change= 0.00010
Iterations terminated because the deviance convergence criteria was reached
Random number generation seed:    1.00000
Number of nodes used when drawing PVs: 2000
Number of nodes used when computing fit: 1000
Number of plausible values to draw: 5
Maximum number of iterations without a deviance improvement: 100
Maximum number of Newton steps in M-step: 10
Value for obtaining finite MLEs for zero/perfects:    0.30000
```

The deviance is 84.83 larger

There are 14 fewer parameters in the second model

Figure 2.53: The Summary Tables for the Two Polytomous DIF Analyses

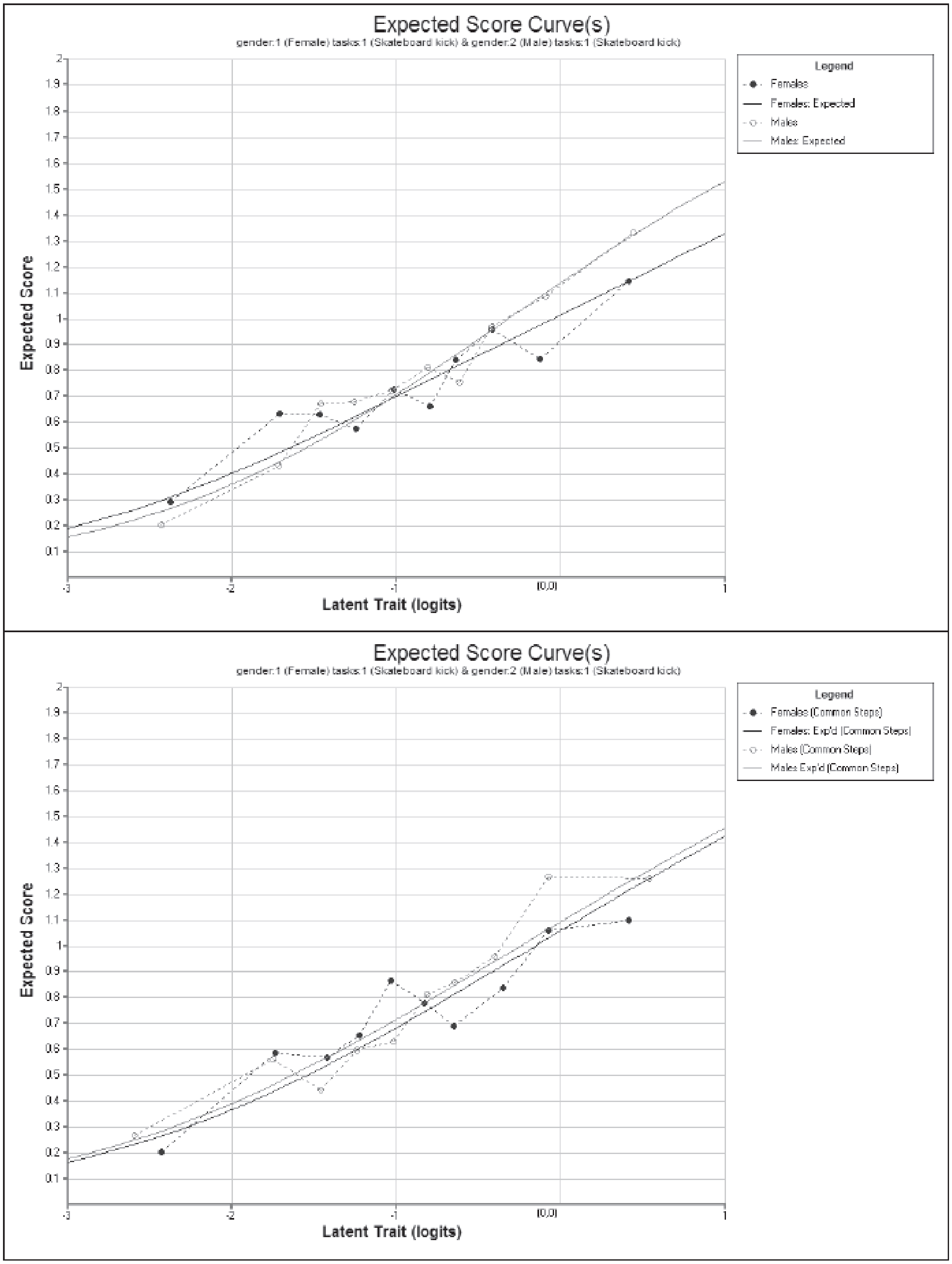


Figure 2.54: Output from Analysis of DIF in Polytomous Items

- Expected score curves are useful for polytomous items.
- Different steps structures can be specified using the `model` statement.

2.8 Multidimensional Models

ACER ConQuest analyses are not restricted to models that involve a single latent dimension. ACER ConQuest can be used for the analysis of sets of items that are designed to produce measures on up to 30 latent dimensions.[21] In this section, multidimensional models are fitted to data that were analysed in previous sections using a one-dimensional model. In doing so, we are able to use ACER ConQuest to explicitly test the unidimensionality assumption made in the previous analyses. We are also able to illustrate the difference between derived estimates and ACER ConQuest's direct estimates of the correlation between latent variables. In this section, we also introduce the two different approaches to estimation (quadrature and Monte Carlo) that ACER ConQuest offers; and in the latter part of the section, we discuss and illustrate two types of multidimensional tests: multidimensional between-item and multidimensional within-item tests.

2.8.1 Example A: Fitting a Two-Dimensional Model

In the first sample analysis in this section, the data used in section 2.2 is re-analysed. In that section, we described a data set that contained the responses of 1000 students to 12 multiple choice items, and the data were analysed as if they were from a unidimensional set of items. This was a bold assumption, because these data are actually the responses of 1000 students to six mathematics multiple choice items and six science multiple choice items.

2.8.1.1 Required files

The files used in this sample analysis are:

[21] Although ACER ConQuest will permit the analysis of up to 30 dimensions, our simulation studies suggest that there may be moderate bias in the estimates of the latent covariance matrix for models with more than eight dimensions (Volodin & Adams, 1995).

filename	content
ex7a.cqc	The command statements.
ex1_dat.txt	The data.
ex1_lab.txt	The variable labels for the items on the multiple choice test.
ex7a_shw.txt	The results of the Rasch analysis.
ex7a_itn.txt	The results of the traditional item analyses.
ex7a_eap.txt	The EAP ability estimates for the students.
ex7a_mle.txt	The maximum likelihood ability estimates for the students.

2.8.1.2 Syntax

The contents of the command file `ex7a.cqc` are shown in the code box below, and explained line-by-line in the list that follows the figure.

`ex7a.cqc:`

```
1  datafile ex1_dat.txt;
2  format id 1-5  responses 12-23;
3  labels << ex1_lab.txt;
4  key acddbcebbacc ! 1;
5  score (0,1) (0,1) ()! items(1-6);
6  score (0,1) () (0,1)! items(7-12);
7  model item;
8  estimate ;
9  show !estimates=latent,tables=1:2:3:9>> Results/ex7a_shw.txt;
10 itanal >> Results/ex7a_itn.txt;
11 show cases !estimates=eap >> Results/ex7a_eap.txt;
12 show cases !estimates=mle >> Results/ex7a_mle.txt;
```

- **Line 1**
  Indicates the name and location of the data file. Any name that is valid for the computer you are using can be used here.
- **Line 2**
  The `format` statement describes the layout of the data in the file `ex1_dat.txt`.
- **Line 3**
  Reads a set of item labels from the file `ex1_lab.txt`.

- **Line 4**
  Recodes the correct responses to 1 and all other values to 0.

- **Lines 5-6**
  The fact that a multidimensional model is to be fitted is indicated by the `score` statement syntax. In our previous uses of the `score` statement, the argument has had two lists, each in parentheses—a *from* list and a *to* list. The effect of those `score` statements was to assign the scores in the *to* list to the matching codes in the *from* list. If a multidimensional model is required, additional *to* lists are added. The arguments of the two `score` statements here each contain three lists. The first is the *from* list and the next two are *to* lists, one for each of two dimensions. The first six items are scored on dimension one; hence, the second *to* list in the first `score` statement is empty. The second six items are scored on the second dimension; hence, the first *to* list in the second `score` statement is empty.

- **Line 7**
  The simple logistic model is used.

- **Line 8**
  The model will be estimated using default settings.

  > **NOTE:** The default settings will result in a Gauss-Hermite method that uses 15 nodes for each latent dimension when performing the integrations that are necessary in the estimation algorithm. For a two-dimensional model, this means a total of $15 \times 15 = 225$ nodes. The total number of nodes that will be used increases exponentially with the number of dimensions, and the amount of time taken per iteration increases linearly with the number of nodes. In practice, we have found that 5000 nodes is a reasonable upper limit on the total number of nodes that can be used.

- **Line 9**
  This `show` statement writes tables 1, 2, 3, and 4 into the file `ex7a_shw.txt`. Displays of the ability distribution will represent the distribution of the latent variable.

- **Line 10**
  The `itanal` statement writes item statistics to the file `ex7a_itn.txt`.

- **Line 11**
  This `show` statement writes a file containing EAP ability estimates for the students on both estimated dimensions.

- **Line 12**
  This `show` statement writes a file containing maximum likelihood ability estimates for the students on both estimated dimensions.

2.8.1.3 Running the Two-Dimensional Sample Analysis

To run this sample analysis, start the GUI version of ACER ConQuest and open the control file `Ex7a.cqc`.

Select `Run`→`Run All`. ACER ConQuest will begin executing the statements that are in the file `ex7a.cqc`; and as they are executed, they will be echoed in the Output Window. When ACER ConQuest reaches the `estimate` statement, it will begin fitting a multidimensional form of Rasch's simple logistic model to the data. As it does so, it will report on the progress of the estimation.

Figure 2.55 is a sample of the information that will be reported by ACER ConQuest as it iterates to find the parameter estimates.

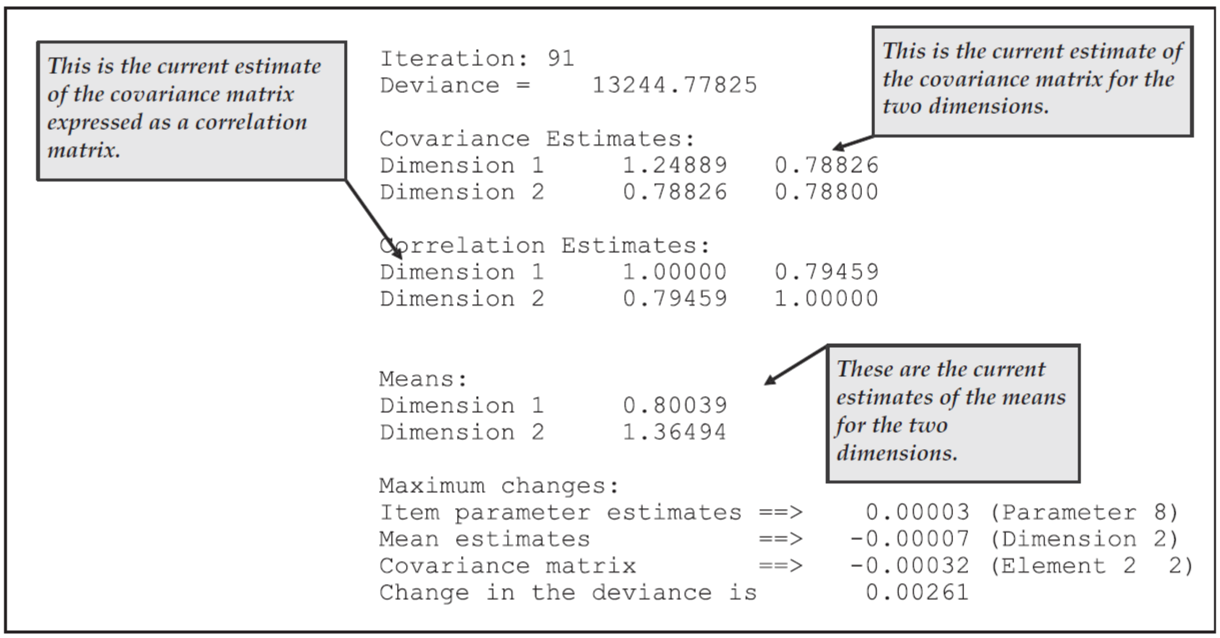


Figure 2.55: Reported Information on Estimation Progress for `ex7a.cqc`

In Figure 2.56, we have reported the first table (table 1) from the file `ex7a_shw.txt`. From this figure, we note that the multidimensional model has estimated 15 parameters; they are made up of 10 item difficulty parameters, the means of the two latent dimensions, and the three unique elements of the variance-covariance matrix. Ten item parameters are used to describe the 12 items because identification constraints are applied to the last item on each dimension.

The deviance for this model is 13244.73. If we refer back to Figure 2.9, we note that a unidimensional model when fitted to these data required the estimation of 13 parameters — 11 item difficulty parameters, one mean, and one variance — and the deviance was 13274.88. As the unidimensional model is a submodel of the two-dimensional model, the difference between the deviance of these two models is distributed as a chi-square with two degrees of freedom. Given the estimated difference of 30.1 in the deviance, we conclude that the unidimensional model does not fit these data as well as the two-dimensional model does.

```
================================================================================
ConQuest: Generalised Item Response Modelling Software     Tue Oct 31 16:05 2006
SUMMARY OF THE ESTIMATION
================================================================================

Estimation method was: Gauss-Hermite Quadrature with 225 nodes
Assumed population distribution was: Gaussian
Constraint was: DEFAULT
The Data File: ex1.dat
The format:  id 1-5 responses 12-23
The regression model:
Grouping Variables:
The item model: item
Sample size: 1000
Final Deviance:    13244.72950
Total number of estimated parameters: 15
The number of iterations: 140
Termination criteria:  Max iterations=1000, Parameter Change= 0.00010
                       Deviance Change= 0.00010
Iterations terminated because the convergence criteria were reached
Random number generation seed:    1.00000
Number of nodes used when drawing PVs: 2000
Number of nodes used when computing fit: 200
Number of plausible values to draw: 5
Maximum number of iterations without a deviance improvement: 100
Maximum number of Newton steps in M-step: 10
Value for obtaining finite MLEs for zero/perfects:    0.30000
================================================================================
```

The deviance for this model is 29.7 less than that for the one-dimensional model that was fitted to the same data in Chapter 3. This value follows a chi-square distribution with two degrees of freedom. It follows that we would reject the hypothesis that the undimensional model fits these data as well as the two-dimensional model.

Figure 2.56: Summary Information for the Two-Dimensional Model

Figure 2.57 shows the second table (table 2) that is produced by the first `show` statement. It contains the item difficulty estimates and the fit statistics. It is interesting to note that the fit statistics reported here are almost identical to those reported for the unidimensional model. Note also that two of the item parameters are constrained. For identification purposes, the mean of the item parameters on each dimension is constrained to be zero. This is achieved by choosing the difficulty of the last item on each dimension to be equal to the negative sum of the difficulties of the other items on the dimension. As an alternative approach, it is possible to use the `lconstraints` argument of the `set` command to force the means of the latent variables to be set at zero and to allow all item parameters to be free.

```
================================================================================
ConQuest: Generalised Item Response Modelling Software     Tue Oct 31 16:05 2006
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
==========================================================================
TERM 1: item
----------------------------------------------------------------------------------
   VARIABLES                          UNWEIGHTED FIT              WEIGHTED FIT
-------------                    -----------------------   -----------------------
     item     ESTIMATE  ERROR^   MNSQ       CI         T   MNSQ       CI         T
----------------------------------------------------------------------------------
 1   BSMMA01    0.056   0.055    0.87 ( 0.91, 1.09) -3.0   0.91 ( 0.94, 1.06) -2.8
 2   BSMMA02   -0.515   0.057    1.10 ( 0.91, 1.09)  2.2   1.02 ( 0.92, 1.08)  0.5
 3   BSMMA03   -0.354   0.056    0.91 ( 0.91, 1.09)                              6
 4   BSMMA04    0.555   0.054    0.99 ( 0.91, 1.09)                              8
 5   BSMMA05    0.917   0.054    1.17 ( 0.91, 1.09)                              9
 6   BSMMA06   -0.659*  0.123    1.00 ( 0.91, 1.09)                              2
 7   BSMSA07   -0.079   0.052    1.04 ( 0.91, 1.09)                              2
 8   BSMSA08   -0.014   0.052    1.10 ( 0.91, 1.09)  2.2   1.06 ( 0.92, 1.08)  1.5
 9   BSMSA09   -0.648   0.056    0.91 ( 0.91, 1.09) -2.0   0.97 ( 0.88, 1.12) -0.6
 10  BSMSA10   -0.079   0.052    1.08 ( 0.91, 1.09)  1.8   1.03 ( 0.92, 1.08)  0.7
 11  BSMSA11   -0.186   0.053    0.91 ( 0.91, 1.09) -2.2   0.96 ( 0.91, 1.09) -0.9
 12  BSMSA12    1.005*  0.119    0.99 ( 0.91, 1.09) -0.2   0.99 ( 0.95, 1.05) -0.3
--------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Separation Reliability =  0.987
Chi-square test of parameter equality =     673.95,  df = 10,  Sig Level = 0.000
^ Quick standard errors have been used
================================================================================
```

Constraints are applied to make the mean of the item difficulties on each dimension zero.

Figure 2.57: Item Parameter Estimates for the Two-Dimensional Model

Figure 2.58 shows the estimates of the population parameters as they appear in the third table (table 3) in file `ex7a_shw.txt`.

The first panel of the table shows that the estimated mathematics mean is 0.800 and the estimated science mean is 1.363.

> **NOTE:** This does not mean that this sample of students is more able in science than in mathematics. The origin of the two scales has been set by

making the mean of the item difficulty parameters on each dimension zero, and no constraints have been placed upon the variances. Thus, these are two separate dimensions; they do not have a common unit or origin.

The second panel of the table shows the variances, covariance and correlation for these two dimensions. The correlation between the mathematics and science latent variables is 0.774. Note that this correlation is effectively corrected for any attenuation caused by measurement error.

```
================================================================================
ConQuest: Generalised Item Response Modelling Software     Tue Oct 31 16:05 2006
TABLES OF POPULATION MODEL PARAMETER ESTIMATES
================================================================================
REGRESSION COEFFICIENTS

                                          Dimension
                         --------------------------------
Regression Variable          Dimension 1       Dimension 2

CONSTANT                   0.800 ( 0.035)    1.363 ( 0.028)
-----------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
=================================================================
COVARIANCE/CORRELATION MATRIX

                              Dimension
                           ---------------
                             1        2

 Dimension 1                        0.790
 Dimension 2               0.802
-------------------------------------------
Variance                   1.245    0.779
-------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Values below the diagonal are correlations and values above are covariances
================================================================================
```

These are the estimated means for the two dimensions.

This is the estimated correlation between the dimensions.

This is the estimated covariance between the dimensions.

These are the estimated variances for the two dimensions.

Figure 2.58: Population Parameter Estimates for the Two-Dimensional Model

Figure 2.59 is the last table (table 4) from the file `ex7a_shw.txt`. The left panel shows a representation of the latent mathematics ability distribution, and the right panel indicates the difficulty of the mathematics items. In the unidimensional equivalent of this figure, the items are plotted so that a student with a latent ability estimate that corresponded to the level at which the item was plotted would have a 50% chance of success on that item. For the multidimensional case, each item is assigned to a single dimension. A student whose latent ability estimate on that dimension is equal to the difficulty estimate for the item would have a 50% chance of success on that item.

EXTENSION: If quadrature-based estimation is used, the computation time needed to fit multidimensional models increases rapidly as additional dimensions are added. This can be alleviated somewhat by reducing the number of nodes being used, although reducing the number of nodes by too much will affect the accuracy of the parameter estimates. With this particular sample analysis, the use of 10 nodes per dimension results in variance estimates that are greater than those obtained using 20 nodes per dimension and the deviance is somewhat higher. If 30 nodes per dimension are used, the results are equivalent to those obtained with 20 nodes.

If you want to explore the possibility of using quadrature with less than 20 nodes per dimension, then we recommend fitting the model with a smaller number of nodes (e.g., 10) and then gradually increasing the number of nodes, noting the impact that the increased number of nodes has on parameter estimates, most importantly the variance. When you reach a point where increasing the number of nodes does not change the parameter estimates, including the variance, then you can have some confidence that an appropriate number of nodes has been chosen.

2.8.1.4 Comparing the Latent Correlation with Other Correlation Estimates

The last two `show` statements in `ex7a.cqc` (see Section 2.8.1.2) produced files of students' EAP and maximum likelihood ability estimates respectively. From these files we are able to compute the product moment correlations between the various ability estimates. In a run not reported here, we also fitted separate unidimensional models to the mathematics and science items and from those analyses produced EAP ability estimates. The various correlations that can be computed between mathematics and science are reported in Figure 2.60.[22]

The estimates based on the raw score, unidimensional EAP, and MLE, which are all similar, indicate a correlation of about 0.40 between mathematics and science. All three estimates are attenuated substantially by measurement error. As the estimated KR-20 reliability of each of these dimensions is 0.58 and 0.43 respectively, an application of

[22] The file `ex7a.out` (provided with the samples) contains the data used in computing the results shown in Figure 2.60. The fixed-format file contains eight fields in this order: mathematics raw score, science raw score, mathematics MLE, science MLE, mathematics EAP from the joint calibration, science EAP from the joint calibration, mathematics EAP from separate calibrations, and science EAP from separate calibrations.

```
================================================================================
ConQuest: Generalised Item Response Modelling Software     Tue Oct 31 16:05 2006
MAP OF LATENT DISTRIBUTIONS AND RESPONSE MODEL PARAMETER ESTIMATES
================================================================================
        Dimension        Terms in the Model (excl Step terms)
--------------------
Dimension 1                                   +item
-------------------------------------------------------------------
                          |
                         X|
                         X|
                         X|
                        XX|
   3                     X|
                       XXX|
                      XXXX|
                     XXXXX|
                    XXXXXX|
                  XXXXXXXX|
   2              XXXXXXXX|
                 XXXXXXXXX|
             XXXXXXXXXXXXX|
                XXXXXXXXXX|
            XXXXXXXXXXXXXX|
              XXXXXXXXXXXX|
             XXXXXXXXXXXXX|
   1        XXXXXXXXXXXXXX|
           XXXXXXXXXXXXXXX|5
              XXXXXXXXXXXX|
         XXXXXXXXXXXXXXXXX|4
         XXXXXXXXXXXXXXXXX|
           XXXXXXXXXXXXXXX|
               XXXXXXXXXXX|
   0          XXXXXXXXXXXX|1
               XXXXXXXXXXX|
                 XXXXXXXXX|3
                    XXXXXX|2
                  XXXXXXXX|6
                     XXXXX|
                     XXXXX|
  -1                   XXX|
                       XXX|
                       XXX|
                        XX|
                        XX|
                         X|
  -2                      |
===================================================================
Each 'X' represents   3.5 cases

===================================================================
```

This is the estimated latent mathematics ability distribution.

These are the mathematics items plotted at their difficulty estimates.

Figure 2.59: Map of the Latent Variables for the Two-Dimensional Model

Analysis Method	*Correlation*
Estimates*	
Raw Scores	0.396
Multidimensional EAP	0.933
Unidimensional EAP	0.399
MLE	0.394
Direct ConQuest Estimation	**0.774**

* These are ability estimates computed by ConQuest and then used to estimate the correlation.

Figure 2.60: Comparison of Some Correlation Estimates with the Latent Ability Estimates

the standard 'correction for attenuation' formula yields estimated correlations of about 0.80.[23] This value is in fairly close agreement with the ACER ConQuest estimate. The correlation of 0.933 between the EAP estimates derived from the two-dimensional analysis is a dramatic overestimation of the correlation between these two variables and should not be used. This overestimation occurs because the EAP estimates are 'shrunken' towards each other. The degree of shrinkage is a function of the reliability of measurement on the individual dimensions; so if many items are used for each dimension, then all of the above indices will be in agreement.

> **EXTENSION:** It is possible to recover the ACER ConQuest estimate of the latent ability correlation from the output of a multidimensional analysis by using plausible values instead of EAP estimates. Plausible values can be produced through the use of the `cases` argument and the `estimates=latent` option of the `show` command. Plausible values are discussed in section 2.9.

2.8.2 Example B: Higher-Dimensional Item Response Models

ACER ConQuest can be used to fit models of up to 15 or more dimensions, and we have routinely used it with up to eight dimensions. When analysing data with three or more dimensions, a Monte Carlo approach to the calculation of the integrals should be used.

2.8.2.1 Required files

In this sample analysis, we fit a five-dimensional model to some performance assessment data that were collected in Australia as part of the TIMSS study (Lokan et al., 1996). The data consist of the responses of 583 students to 28 items that belong to five different performance assessment tasks. These data are quite sparse because each student was only required to undertake a small subset of the tasks, but every task appears at least once with every other task.

The files that will be used in this sample analysis are:

[23] Here we are using the KR-20 index that is reported by ACER ConQuest at the end of the printout from an `itanal` analysis.

filename	content
ex7b.cqc	The command statements.
ex7b.dat.txt	The data.
ex7b.lab.txt	The variable labels for the items.
ex7b.prm.txt	The estimates of the item response model parameters.
ex7b.reg.txt	The estimates of the regression coefficients for the population model.
ex7b.cov.txt	The estimates of the variance-covariance matrix for the population model.
ex7b.shw.txt	The results of the Rasch analysis.

2.8.2.2 Syntax

The command file `ex7b.cqc` is used in this Tutorial to fit a Higher-Dimensional Item Response Model. It is shown in the code box below, and each line of syntax is detailed in the list below the code.

`ex7b.cqc:`

```
title Australian Performance Assessment Data;
datafile ex7b.dat.txt;
format responses 1-28;
codes 0,1,2,3;
labels << ex7b.lab.txt;
recode (2) (1) !items(9,10);
recode (3) (2) !items(25);
score (0,1,2,3) (0,1,2,3) ( ) ( ) ( ) ( ) ! items (1-6);
score (0,1,2,3) ( ) (0,1,2,3) ( ) ( ) ( ) ! items (7-13);
score (0,1,2,3) ( ) ( ) (0,1,2,3) ( ) ( ) ! items (14-17);
score (0,1,2,3) ( ) ( ) ( ) (0,1,2,3) ( ) ! items (18-25);
score (0,1,2,3) ( ) ( ) ( ) ( ) (0,1,2,3) ! items (26-28);
model item+item*step;
export parameters >> ex7b.prm;
export reg_coefficients >>ex7b.reg;
export covariance >> ex7b.cov;
import init_parameters << ex7b.prm.txt;
import init_reg_coefficients <<ex7b.reg.txt;
import init_covariance << ex7b.cov.txt;
estimate ! method = montecarlo, nodes = 2000, conv = .005, stderr = quick;
show ! tables=1:2:3:4:9, estimates = latent >> Results/ex7b.shw.txt;
```

- **Line 1**
  Gives the title.

- **Line 2**
  Gives the name of the data file to be analysed. In this case, the data are contained in the file `ex7b.dat.txt`.

- **Line 3**
  The `format` statement indicates that there are 28 items, and they are in the first 28 columns of the data file.

- **Line 4**
  Restricts the valid codes to `0`, `1`, `2` or `3`.

- **Line 5**
  A set of labels for the items are to be read from the file `ex7b.lab.txt`.

- **Lines 6-7**
  If a gap occurs in the scores in the response data for an item, then the next higher score for that item must be recoded downwards to close the gap. For example, in this data set, by coincidence, no response to item 9 or item 10 was scored as 1; all responses to these two items were scored as 0 or 2. To fill the gap between 0 and 2, the 2 has been recoded to 1 by the first `recode` statement. Similarly, for item 25, none of the response data is equal to 2, so 3 must be recoded to 2 to fill the gap.

  > **NOTE:** The model being fitted here is a partial credit model. Therefore, all score categories between the highest category and the lowest category must contain data. If this is not the case, then some parameters will not be identified. If `warnings` is not set to `no`, then ACER ConQuest will flag those parameters that are not identified and will indicate that recoding of the data is necessary. If `warnings` is set to `no`, then the parameters that are not identified due to null categories will not be reported. If a rating scale model were being fitted to these data, then recoding would not be necessary because all of the step parameters would be identified.

- **Lines 8-12**
  The model that we are fitting here is five dimensional, so the `score` statements contain six sets of parentheses as their arguments, one for the *from* codes and five for the *to* codes. The option of the first `score` statement gives the items to be assigned to the first dimension, the option of the second `score` statement gives the items to be allocated to the second dimension, and so on.

- **Line 13**
  The model we are using is the partial credit model.

- **Line 14**
  We want to update the export files of parameter estimates (see lines 15 through 17) every iteration, without warnings.

- **Lines 15-17**
  Request that item, regression and covariance parameter estimates be written to the files `ex7b.prm.txt`, `ex7b.reg.txt`, and `ex7b.cov.txt` respectively.

- **Lines 18-20**
  Initial values of item, regression and covariance parameter estimates are to be read from the files `ex7b.prm.txt`, `ex7b.reg.txt`, and `ex7b.cov.txt` respectively.

- **Line 21**
  This `estimate` statement has three arguments: `method=montecarlo` requests that the integrals that are computed in the estimation be approximated using Monte Carlo methods; `nodes=2000` requests 2000 nodes be used in computing integrals; and `converge=.005` requests that the estimation be terminated when the largest change in any parameter estimate between successive iterations becomes less than 0.005.

EXTENSION: Wilson & Masters (1993) discuss a method of dealing with data that have 'null' categories of the type we observe in these data for items 9, 10 and 25. Their approach can be implemented easily in ACER ConQuest by using a `score` statement that assigns a score of 2 to the category 1 of items 9 and 10 and a score of 3 to the category 2 of item 25, after recoding has been done to close the gaps.

NOTE: We have used the same names for the initial value and export files. These files must already exist so that, before the estimation commences, initial values can be read from them. After each iteration, the values in these files are then updated with the current parameter estimates. Importing and exporting doesn't happen until the `estimate` statement is executed; thus, the order of the `import` and `export` statements is irrelevant, so long as they precede the `estimate` statement.

2.8.2.3 Running a Higher-Dimensional Sample Analysis

To run this sample analysis, launch the console version of ACER ConQuest by typing the command (on Windows) `ConQuestConsole.exe ex7b.cqc`.

ACER ConQuest will begin executing the statements that are in the file `ex7b.cqc;` and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting a multidimensional form of Rasch's simple logistic model to the data. As it does so, it will report on the progress of the estimation.

Figures 2.61, 2.62 and 2.63 show three of the tables (2, 3 and 4) that are written to `ex7b_shw.txt`.

In Figure 2.61, note that five items have their parameter estimates constrained. These are the five items that are listed as the last item on each of the dimensions. Their values are constrained to ensure that the mean of the item parameters for each dimension is zero.

> **EXTENSION:** As an alternative to identifying the model by making the mean of the item parameters on each dimension zero (default behaviour), the `lconstraints=cases` argument of the `set` command can be used to have the mean of each latent dimension set to zero as an alternative constraint. If this were done, all item parameters would be estimated, but the mean of each of the latent dimensions would be zero.

Figure 2.62 shows the population parameter estimates, which in this case consist of means for each of the dimensions and the five-by-five variance-covariance matrix of the latent dimensions.

Figure 2.63 is a map of the five latent dimensions and the item difficulties. For the purposes of this figure, we have omitted the rightmost panel, which shows the item step-parameter estimates.

2.8.3 Within-Item and Between-Item Multidimensionality

The two preceding sample analyses in this section are examples of what Wang (1995) would call *between-item multidimensionality* (see also Adams, Wilson, & Wang (1997)). To assist in the discussion of different types of multidimensional models and tests, Wang introduced the notions of *within-item* and *between-item multidimensionality.* A test is regarded as multidimensional between-item if it is made up of several unidimensional subscales. A

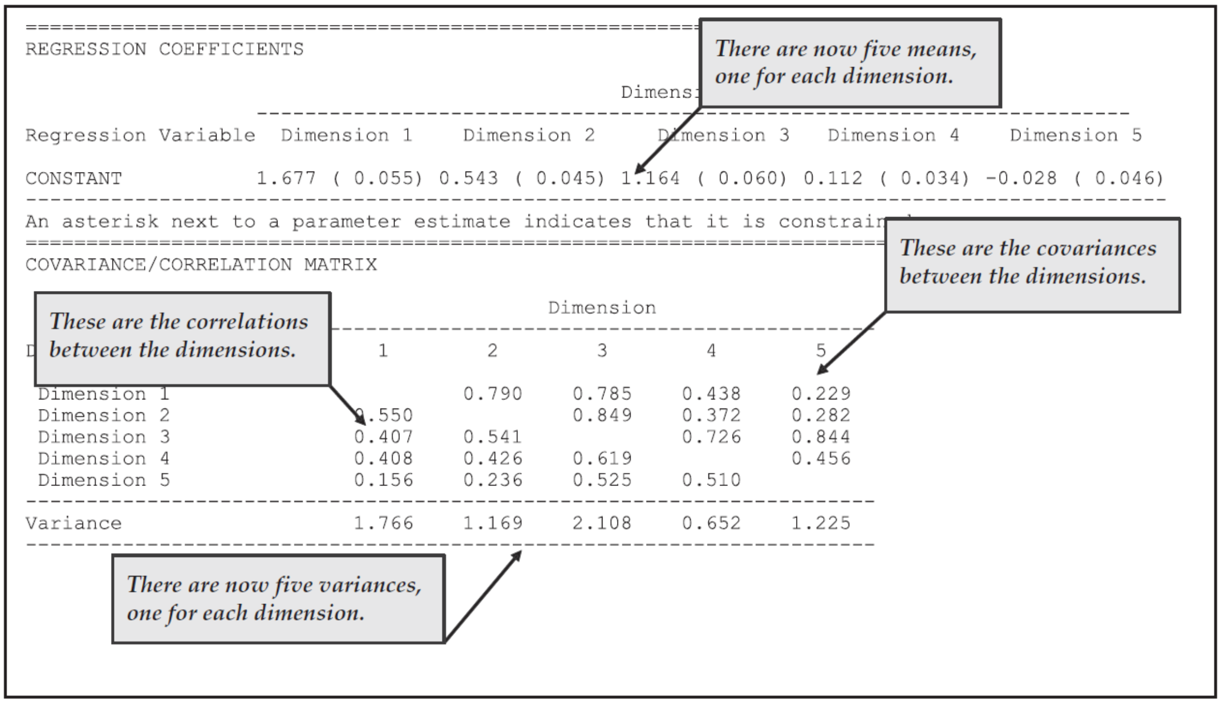
Australian Performance Assessment Data Sun Feb 18 10:13 2007
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES

TERM 1: item

VARIABLES				UNWEIGHTED FIT			WEIGHTED FIT		
	item	ESTIMATE	ERROR^	MNSQ	CI	T	MNSQ	CI	T
1	bspm11	-1.126	0.132	1.65	(0.80, 1.20)	5.4	1.18	(0.47, 1.53)	0.7
2	bspm12	0.024	0.128	1.01	(0.80, 1.20)	0.2	0.93	(0.79, 1.21)	-0.6
3	bspm13	-0.579	0.121	0.90	(0.80, 1.20)	-1.0	0.90	(0.57, 1.43)	-0.5
4	bspm14	0.395	0.104	1.05	(0.80, 1.20)	0.5	1.01	(0.80, 1.20)	0.1
5	bspm15a	-0.836	0.138	0.73	(0.80, 1.20)	-2.8	0.86	(0.69, 1.31)	-0.9
6	bspm15b	2.122*	0.280	0.96	(0.80, 1.20)	-0.4	1.00	(0.86, 1.14)	-0.0
7	bspm21	-2.804	0.114	0.18	(0.80, 1.20)	-12.7	0.77	(0.30, 1.70)	-0.7
8	bspm22	0.702	0.080	1.02	(0.80, 1.20)	0.3	1.03	(0.83, 1.17)	0.3
9	bspm23	-1.426	0.105	0.70	(0.80, 1.20)	-3.2	0.84	(0.73, 1.27)	-1.2
10	bspm24	-0.301	0.099	0.93	(0.80, 1.20)	-0.7	0.91	(0.85, 1.15)	-1.2
11	bspm25	0.622	0.082	0.97	(0.80, 1.20)	-0.2	1.01	(0.83, 1.17)	0.1
12	bspm26a	1.580	0.074	1.15	(0.79, 1.21)	1.4	1.08	(0.81, 1.19)	0.9
13	bspm26b	1.628*	0.229	[illegible]	[illegible]	[illegible]	1.15	(0.80, 1.20)	1.4
14	bspm31	-0.034	0.074	[illegible]	[illegible]	[illegible]	1.04	(0.78, 1.22)	0.4
15	bspm32	-0.749	0.077	[illegible]	[illegible]	[illegible]	0.87	(0.73, 1.27)	-1.0
16	bspm33	-0.182	0.074	[illegible]	[illegible]	[illegible]	0.96	(0.77, 1.23)	-0.3
17	bspm34	0.965*	0.130	[illegible]	[illegible]	[illegible]	1.15	(0.79, 1.21)	1.3
18	bspm41	-1.181	0.109	1.11	(0.80, 1.20)	1.1	1.06	(0.75, 1.25)	0.5
19	bspm42	-0.698	0.093	0.79	(0.80, 1.20)	-2.1	0.95	(0.81, 1.19)	-0.5
20	bspm43	-0.516	0.104	1.43	(0.80, 1.20)	3.8	1.31	(0.83, 1.17)	3.4
21	bspm44	-1.205	0.110	1.27	(0.80, 1.20)	2.4	1.15	(0.75, 1.25)	1.2
22	bspm45a	0.221	0.069	0.82	(0.80, 1.20)	-1.8	0.87	(0.82, 1.18)	-1.4
23	bspm45b	0.356	0.111	0.98	(0.80, 1.20)	-0.2	0.98	(0.83, 1.17)	-0.2
24	bspm45c	-0.212	0.083	0.84	(0.80, 1.20)	-1.6	0.85	(0.82, 1.18)	-1.8
25	bspm46	3.235*	0.260	0.59	(0.80, 1.20)	-4.7	0.89	(0.39, 1.61)	-0.3
26	bspm51	-0.323	0.066	1.10	(0.80, 1.20)	1.0	1.07	(0.81, 1.19)	0.7
27	bspm52	-0.245	0.076	0.83	(0.80, 1.20)	-1.8	0.85	(0.83, 1.17)	-1.8
28	bspm53	0.568*	0.100	1.15	(0.80, 1.20)	1.5	1.09	(0.82, 1.18)	0.9

Five item parameters are constrained, one for each dimension.

Figure 2.61: Item Parameter Estimates for a Five-Dimensional Sample Analysis



```
================================================================
REGRESSION COEFFICIENTS

                                                Dimens
                   -----------------------------------------------------------------------
Regression Variable  Dimension 1     Dimension 2     Dimension 3   Dimension 4    Dimension 5

CONSTANT           1.677 ( 0.055) 0.543 ( 0.045) 1.164 ( 0.060) 0.112 ( 0.034) -0.028 ( 0.046)
----------------------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrain
==========================================================================
COVARIANCE/CORRELATION MATRIX

                                         Dimension
                          -----------------------------------------------
D                             1         2         3         4         5

 Dimension 1                          0.790     0.785     0.438     0.229
 Dimension 2               0.550                0.849     0.372     0.282
 Dimension 3               0.407     0.541                0.726     0.844
 Dimension 4               0.408     0.426     0.619                0.456
 Dimension 5               0.156     0.236     0.525     0.510
----------------------------------------------------------------------
Variance                   1.766     1.169     2.108     0.652     1.225
----------------------------------------------------------------------
```


Figure 2.62: Population Model Parameter Estimates for the Five-Dimensional Sample Analysis

```
==========================================================================================
Australian Performance Assessment Data                      Thu Dec 19 17:26:55
MAP OF LATENT DISTRIBUTIONS AND RESPONSE MODEL PARAMETER ESTIMATES
==========================================================================================
                       Dimension                                   Terms in the Model Statement
--------------------------------------------------
          1         2         3         4         5                      +item
------------------------------------------------------------------------------------------
   5           |                             |         |
               |                             |         |
               |                             |         |
              X|                             |         |
              X|         |        X|         |         |
   4          X|         |        X|         |         |
             XX|         |       XX|         |         |
             XX|         |        X|         |         |
            XXX|         |        X|         |         |
            XXX|         |        X|         |         |25
   3        XXX|        X|      XXX|         |         |
            XXX|        X|      XXX|         |         |
          XXXXX|        X|      XXX|         |         |
          XXXXX|        X|     XXXX|         |         |
          XXXXX|        X|      XXX|         |        X|
   2      XXXXX|     XXXX|    XXXXX|        X|        X|6
          XXXXX|      XXX|    XXXXX|        X|       XX|
         XXXXXX|     XXXX|    XXXXX|        X|       XX|12 13
         XXXXXX|    XXXXX|   XXXXXX|      XXX|       XX|
         XXXXXX|   XXXXXX|    XXXXX|      XXX|      XXX|
   1       XXXX|   XXXXXX|    XXXXX|     XXXX|     XXXX|
          XXXXX|  XXXXXXX|      XXX|  XXXXXXX|     XXXX|17
           XXXX|  XXXXXXX|     XXXX|  XXXXXXX|   XXXXXX|8 11
            XXX|  XXXXXXX|     XXXX| XXXXXXXX| XXXXXXXX|28
           XXXX|  XXXXXXX|    XXXXXXXXXXXXXXX|  XXXXXXX|4 22 23
            XXX|    XXXXX|    XXXXX|XXXXXXXXX|  XXXXXXX|2
   0        XXX|    XXXXX|     XXXX| XXXXXXXX|    XXXXX|14 16
             XX|   XXXXXX|      XXX| XXXXXXXX|   XXXXXX|10 24 26 27
             XX|     XXXX|       XX|  XXXXXXX|   XXXXXX|20
              X|     XXXX|      XXX|   XXXXXX|    XXXXX|3 15 19
              X|      XXX|       XX|   XXXXXX|    XXXXX|5
  -1          X|       XX|       XX|      XXX|     XXXX|1
               |        X|       XX|       XX|     XXXX|18 21
               |        X|       XX|       XX|     XXXX|9
               |        X|        X|        X|      XXX|
               |        X|        X|        X|        X|
  -2           |         |        X|         |        X|
               |         |        X|         |        X|
               |         |         |         |        X|
               |         |         |         |        X|
               |         |         |         |         |7
  -3           |         |         |         |         |
==========================================================================================
Each 'X' represents   5.9 cases
==========================================================================================
```

There are now five estimated latent distributions.

Figure 2.63: Variable Map for the Five-Dimensional Sample Analysis

test is considered multidimensional within-item if any of the items relates to more than one latent dimension.

Multidimensional Between-Item Models
Tests that contain several subscales, each measuring related but distinct latent dimensions, are very commonly encountered in practice. In such tests, each item belongs to only one particular subscale, and there are no items in common across the subscales. In the past, item response modelling of such tests has proceeded by either applying a unidimensional model to each of the scales separately or by ignoring the multidimensionality and treating the test as unidimensional. Both of these methods have weaknesses that make them less desirable than undertaking a joint, multidimensional calibration (Adams, Wilson, & Wang, 1997). In the preceding sample analyses in this section, we have illustrated the alternative approach of fitting a multidimensional model to the data.

Multidimensional Within-Item Models
If the items in a test measure more than one latent dimension and some of the items require abilities from more than one dimension, then we call the test within-item multidimensional.

The distinction between the within-item and between-item multidimensional models is illustrated in Figure 2.64.

In the left of Figure 2.64, we have depicted a between-item multidimensional test that consists of nine items measuring three latent dimensions. On the right of Figure 2.64, we have depicted a within-item multidimensional test with nine items and three latent dimensions.

2.8.4 Example C: A Within-Item Multidimensional Model

As a final sample analysis in this section, we show how ACER ConQuest can be used to estimate a within-item multidimensional model like that illustrated in Figure 2.64.

For the purpose of this sample analysis, we use simulated data that consist of the responses of 2000 students to nine dichotomous questions. These items are assumed to assess three different latent abilities, with the relationship between the items and the latent abilities as depicted in Figure 2.64. The generating value for the mean for each of the latent abilities was zero, and the generating covariance between the latent dimensions was:

$$\sum = \begin{bmatrix} 1.00 & 0.00 & 0.58 \\ 0.00 & 1.00 & 0.58 \\ 0.58 & 0.58 & 1.00 \end{bmatrix}$$

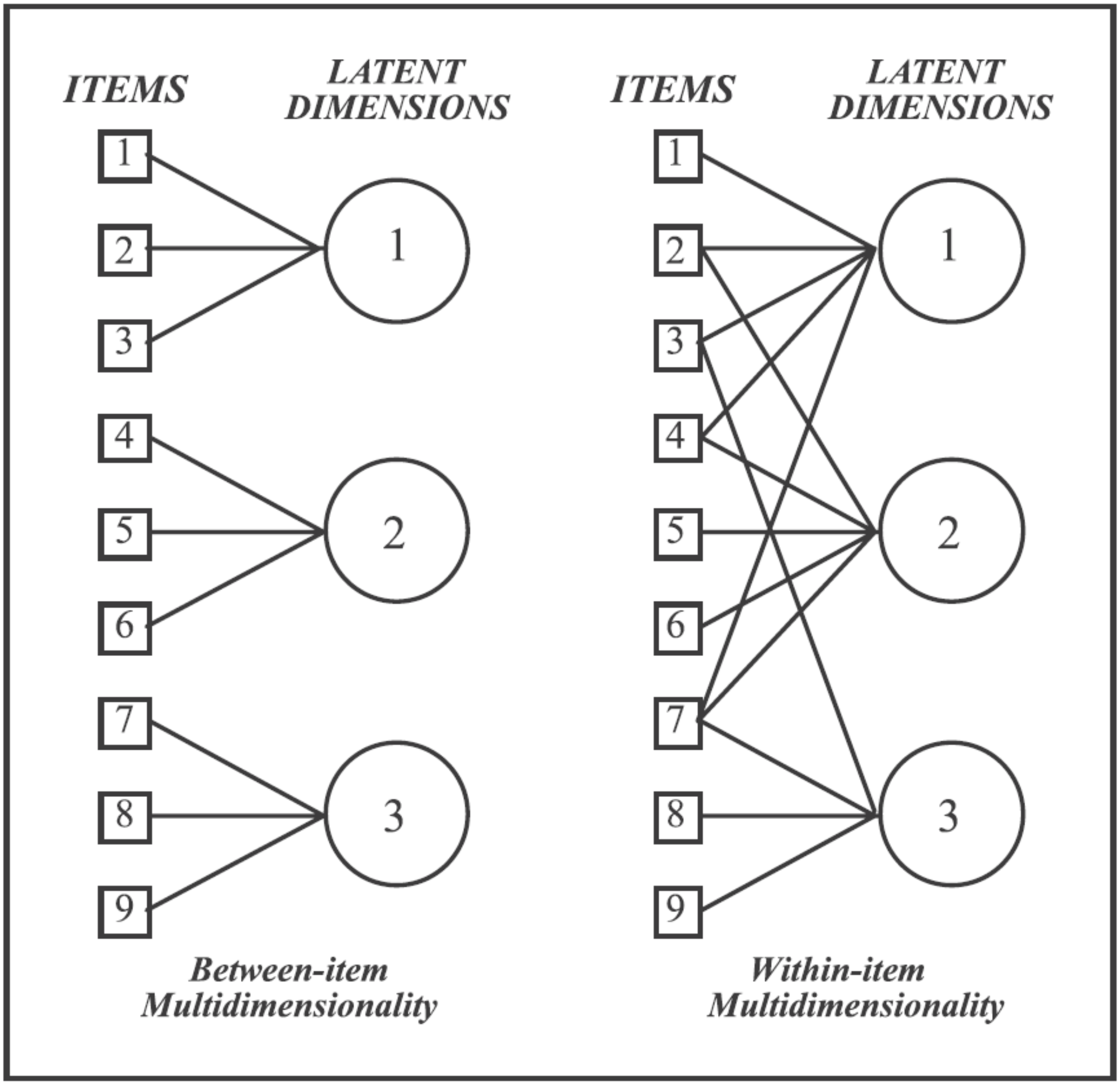


Figure 2.64: A Graphical Representation of Within-Item and Between-Item Multidimensionality

The generating item difficulty parameters were –0.5 for items 1, 4 and 7; 0.0 for items 2, 5 and 8; and 0.5 for items 3, 6 and 9.

2.8.4.1 Required files

The files that we use in this sample analysis are:

filename	content
ex7c.cqc	The command statements used to fit the model.
ex7c_dat.txt	The data.
ex7c_prm.txt	Item parameter estimates.
ex7c_reg.txt	Regression coefficient estimates.
ex7c_cov.txt	Covariance parameter estimates.
ex7c_shw.txt	Selected results of the analysis.

2.8.4.2 Syntax

`ex7c.cqc` is the command file necessary for fitting the Within-Item Multidimensional Model. It is shown in the code block below, and commented upon in the list underneath the embedded command file.

This command file actually runs two analyses. The first is used to obtain an approximate solution that is used as initial values for the second analysis, which is used to produce a more accurate solution.

`ex7c.cqc`:

```
datafile ex7c_dat.txt;
format responses 1-9;
set lconstraints=cases,update=yes,warnings=no;
score (0,1) (0,1) (   ) (   ) ! items(1);
score (0,1) (0,1) (0,1) (   ) ! items(2);
score (0,1) (0,1) (   ) (0,1) ! items(3);
score (0,1) (0,1) (0,1) (   ) ! items(4);
score (0,1) (   ) (0,1) (   ) ! items(5);
score (0,1) (   ) (0,1) (   ) ! items(6);
score (0,1) (0,1) (0,1) (0,1) ! items(7);
score (0,1) (   ) (   ) (0,1) ! items(8);
```

```
12 score (0,1) (   ) (   ) (0,1) ! items(9);
13 model items;
14 export parameters >> ex7c_prm.txt;
15 export reg_coefficients >> ex7c_reg.txt;
16 export covariance >> ex7c_cov.txt;
17 estimate !method=montecarlo,nodes=200,conv=.01,fit=no,stderr=none;
18 reset;
19 datafile ex7c_dat.txt;
20 format responses 1-9;
21 set lconstraints=cases,update=yes,warnings=no;
22 score (0,1) (0,1) (   ) (   ) ! items(1);
23 score (0,1) (0,1) (0,1) (   ) ! items(2);
24 score (0,1) (0,1) (   ) (0,1) ! items(3);
25 score (0,1) (0,1) (0,1) (   ) ! items(4);
26 score (0,1) (   ) (0,1) (   ) ! items(5);
27 score (0,1) (   ) (0,1) (   ) ! items(6);
28 score (0,1) (0,1) (0,1) (0,1) ! items(7);
29 score (0,1) (   ) (   ) (0,1) ! items(8);
30 score (0,1) (   ) (   ) (0,1) ! items(9);
31 model items;
32 import init_parameters << ex7c_prm.txt;
33 import init_reg_coefficients << ex7c_reg.txt;
34 import init_covariance << ex7c_cov.txt;
35 export parameters >> ex7c_prm.txt;
36 export reg_coefficients >> ex7c_reg.txt;
37 export covariance >> ex7c_cov.txt;
38 estimate !method=montecarlo,nodes=1000;
39 show !tables=1:2:3 >> Results/ex7c_shw.txt;
```

- **Line 1**
  Read data from the file `ex7c_dat.txt`.

- **Line 2**
  The responses are in columns 1 through 9.

- **Line 3**
  Set `update` to `yes` and `warnings` to `no` so that current parameter estimates are written to a file at every iteration. This statement also sets `lconstraints=cases`,

which should be used if ACER ConQuest is being used to estimate models that have within-item multidimensionality.

> **EXTENSION:** ACER ConQuest can be used to estimate within-item multidimensional models without the use of `lconstraints=cases`. This will, however, require the user to define his or her own design matrices. A description of how to construct design matrices is found in section 2.10, Importing Design Matrices. Sample analyses that use user-defined design matrices are provided in section 3.1.7, Design Matrices.

- **Lines 4-12**
  These `score` statements describe how the items 'load' on each of the latent dimensions. The first item, for example, has scores on dimension one but not dimensions two or three. The second item is scored on the first and second dimensions, the third on the first and third, and so on.

- **Line 13**
  The items are all dichotomous, so we are using the simple logistic model.

- **Lines 14-16**
  The item, regression and covariance parameter estimates will each be written to a file. The combination of the `update` argument in the `set` statement (line 3) and these `export` statements means that these files will be updated at every iteration.

  > **NOTE:** The implicit variable names `item` and `items` are synonymous in ACER ConQuest, so you may use either in ACER ConQuest statements.

- **Line 17**
  In this estimation, we are using the Monte Carlo integration method with 200 nodes and a convergence criterion of 0.01. This analysis is undertaken to provide initial values for the more accurate analysis that follows.

- **Line 18**
  Resets all system values so that a new analysis can be undertaken.

- **Lines 19-31**
  As for lines 1 through 13.

- **Lines 32-34**
  Initial values for all of the parameter estimates are read from the files that were created in the previous analysis.

- **Lines 35-37**
  As for lines 14 through 16.

- **Line 38**
  The Monte Carlo method of estimation is used with 1000 nodes and the default convergence criterion of 0.001.

- **Line 39**
  Tables 1, 2 and 3 are written to `ex7c_shw.txt`.

2.8.4.3 Running the Within-Item Multidimensional Sample Analysis

To run this sample analysis, launch the console version of ACER ConQuest by typing the command (on Windows) `ConQuestConsole.exe ex7c.cqc`.

ACER ConQuest will begin executing the statements that are in the file `ex7c.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting a within-item three-dimensional form of Rasch's simple logistic model to the data, using 200 nodes and a convergence criterion of 0.01 with the Monte Carlo method. ACER ConQuest will then proceed to the second analysis. This analysis begins with the provisional estimates provided by the first analysis and uses 1000 nodes with the default convergence criterion of 0.0001. The `show` statement at the end of the command file will produce three output tables. The second and third of these are reproduced in Figures 2.65 and 2.66. The results in these tables show that ACER ConQuest has done a good job in recovering the generating values for the parameters.

2.8.5 Summary

In this section, we have seen how ACER ConQuest can be used to fit multidimensional item response models. Models of two, three and five dimensions have been fit.

Some key points covered in this section are:

- The `score` statement can be used to indicate that a multidimensional item response model should be fit to the data.
- The fitting of a multidimensional model as an alternative to a unidimensional model can be used as an explicit test of the fit of data to a unidimensional item response model.

```
==========================================================================
ConQuest: Generalised Item Response Modelling Software  Sun Feb 18 11:14 2007
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
====================================================================
TERM 1: items
-----------------------------------------------------------------------------
   VARIABLES                         UNWEIGHTED FIT          WEIGHTED FIT
-------------                 ----------------------- -----------------------
     item    ESTIMATE  ERROR^ MNSQ       CI        T  MNSQ       CI        T
-----------------------------------------------------------------------------
 1   1        -0.380   0.049  0.99 ( 0.94, 1.06) -0.2 1.00 ( 0.96, 1.04) -0.2
 2   2        -0.009   0.026  1.04 ( 0.94, 1.06)  1.2 1.02 ( 0.95, 1.05)  0.9
 3   3         0.496   0.029  1.03 ( 0.94, 1.06)  1.0 1.03 ( 0.95, 1.05)  1.0
 4   4        -0.529   0.028  1.01 ( 0.94, 1.06)  0.2 1.01 ( 0.94, 1.06)  0.4
 5   5         0.028   0.049  1.00 ( 0.94, 1.06) -0.0 1.00 ( 0.96, 1.04) -0.0
 6   6         0.402   0.050  1.00 ( 0.94, 1.06)  0.1 1.00 ( 0.96, 1.04) -0.0
 7   7        -0.510   0.022  1.03 ( 0.94, 1.06)  0.9 1.00 ( 0.93, 1.07)  0.1
 8   8         0.085   0.049  1.01 ( 0.94, 1.06)  0.2 1.00 ( 0.96, 1.04)  0.3
 9   9         0.528   0.050  1.02 ( 0.94, 1.06)  0.5 1.01 ( 0.96, 1.04)  0.4
--------------------------------------------------------------------
==============================================================================
```

The fit statistics look good (not surprising since the data were simulated to fit this model).

Figure 2.65: Item Parameter Estimates for a Within-Item Three-Dimensional Sample Analysis

```
============================================                  ====================
ConQuest: Generalised Item Response Model                     Feb 18 11:14 2007
TABLES OF POPULATION MODEL PARAMETER ESTI
=========================================                     =======================
REGRESSION COEFFICIENTS

                                              Dimension
                         ---------------------------------------------------
Regression Variable          Dimension 1       Dimension 2       Dimension 3

CONSTANT                   0.000*            0.000*            0.000*
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
================================================================================
COVARIANCE/CORRELATION MATRIX

                                  Dimension
                         ---------------------------
Dimension                    1         2         3

 Dimension 1                        0.098     0.642
 Dimension 2               0.100              0.580
 Dimension 3               0.667    0.550
----------------------------------------------------
Variance                   0.897    1.077     1.033
----------------------------------------------------
```

The means are all zero because of the constraint used to identify the model.

The estimated covariances are close to their generating values of 0.00 and 0.58.

The estimated variances are close to their generating values of 1.00.

Figure 2.66: Population Parameter Estimates for a Within-Item Three-Dimensional Sample Analysis

- The secondary analysis of latent ability estimates does not produce results that are equivalent to the ‘correct’ latent regression results. The errors that can be made in a secondary analysis of latent ability estimates are greater when measurement error is large.
- ACER ConQuest offers two approximation methods, quadrature and Monte Carlo, for computing the integrals that must be computed in marginal maximum likelihood estimation. The quadrature method is generally the preferred approach for problems of three or fewer dimensions, while the Monte Carlo method is preferred for higher dimensions.
- ACER ConQuest can be used to fit models that are multidimensional between-item or multidimensional within-item. Fitting multidimensional within-items requires the use of `lconstraints=cases`, unless an imported design matrix is used.

2.9 Multidimensional Latent Regression

In section 2.8, we illustrated how ACER ConQuest can be used to fit multidimensional item response models; and in section 2.6, we illustrated how ACER ConQuest can be used to estimate latent regression models. In this section, we bring these two functions together, using ACER ConQuest to fit multidimensional latent regression models.

In parts a) and b) of this section, we fit multidimensional latent regression models of two and five dimensions. Some output that is standard for regression analysis is not available in this version of ACER ConQuest; but in part c) we illustrate how plausible values can be drawn. The plausible values can be analysed, using traditional regression techniques, to produce further regression statistics.

The data we are analysing were collected by Adams et al. (1991) as part of their study of science achievement in Victorian schools. In their study, Adams et al. used a battery of multiple choice and extended response written tests.

The data set contains the responses of 2564 students to the battery of tests; all of the items have been prescored. The multiple choice items are located in columns 50 through 114, and the extended response test that we will use is located in columns 1 through 9. If students were administered a test but did not respond to an item, a code of 9 has been entered into the file. If a student was not administered an item, then the file contains a blank character. We will be treating the 9 as an incorrect response and the blanks as missing-response data. The student’s grade code is located in column 118, the gender code is located in column 119, and the indicator of socio-economic status is in columns

122 through 127.[24] The gender variable is coded 0 for female and 1 for male, the grade variable is coded 1 for the lower grade and 2 for the upper grade, and the socio-economic indicator is a composite that represents a student's socio-economic status.

2.9.1 a) Fitting a Two-Dimensional Latent Regression

In this sample analysis, we will consider ability as assessed by the multiple choice test as one latent outcome and ability as assessed by the first of the extended response tests as a second latent outcome. Then we will regress these two outcomes onto three background variables: student grade, student gender and an indicator of socio-economic status.

2.9.1.1 Required files

The files that will be used in this sample analysis are:

filename	content
ex8a.cqc	The command statements that we use.
ex8a_no_regressors.cqc	The command statements to estimate the variance of latent variables.
ex6_dat.txt	The data.
ex8a_prm.txt	An initial set of item parameter estimates.
ex8a_reg.txt	An initial set of regression coefficient estimates.
ex8a_cov.txt	An initial set of variance-covariance parameter estimates.
ex8a_shw.txt	The population model parameter estimates.

2.9.1.2 Syntax

This sample analysis uses the command file `ex8a.cqc` to conduct a Two-Dimensional Latent Regression. `ex8a.cqc` is shown in the code box below, and explained line-by-line in the list underneath the figure.

ex8a.cqc:

```
datafile ex6_dat.txt;
format responses 1-9,50-114 grade 118 gender 119 ses 122-127!tasks(74);
model tasks+tasks*step;
```

[24]See Adams et al. (1991) for how the socio-economic indicator was constructed.

```
4  recode (9) (0);
5  score (0,1,2,3,4) (0,1,2,3,4) ( ) !tasks(1-9);
6  score (0,1,2,3,4) ( ) (0,1,2,3,4) !tasks(10-74);
7  regression grade,gender,ses;
8  export parameters >> ex8a_prm.txt;
9  export reg_coefficients >> ex8a_reg.txt;
10 export covariance >> ex8a_cov.txt;
11 import init_parameters << ex8a_prm.txt;
12 import init_reg_coefficients << ex8a_reg.txt;
13 import init_covariance << ex8a_cov.txt;
14 set update=yes,warnings=no;
15 estimate!fit=no,converge=.002,stderr=quick;
16 show ! tables=3 >> Results/ex8a_shw.txt;
```

- **Line 1**
  We are analysing data in the file `ex6_dat.txt`.

- **Line 2**
  The `format` statement is reading 74 responses; assigning the label `tasks` to those responses; and reading `grade`, `gender` and `ses` data. The column specifications for the responses are made up of two separate response blocks. The first nine items are read from columns 1 through 9 (these are the extended response items that we are using), and the remaining 65 items are read from columns 50 through 114 (these are the multiple choice items).

- **Line 3**
  We are using the partial credit model because the items are a mixture of polytomous and dichotomous items.

- **Line 4**
  A code of `9` has been used for missing-response data caused by the student not responding to an item. We want to treat this as though it were identical to an incorrect response, so we recode it to `0`.

- **Lines 5-6**
  We use two `score` statements, one for each dimension. The first statement scores the first nine tasks on the first dimension, and the second statement scores the remaining 65 tasks on the second dimension.

- **Line 7**
  This `regression` statement specifies a population model that regresses the two latent variables onto `grade`, `gender` and `ses`.

- **Lines 8-10**
  These `export` statements result in the parameter estimates being written to the files `ex8a_prm.txt`, `ex8a_reg.txt` and `ex8a_cov.txt`. In conjunction with the `set` statement (line 14), these `export` statements result in updated parameter estimates being written to these files after each iteration.

- **Lines 11-13**
  Initial values of all parameter estimates are read from the files `ex8a_prm.txt`, `ex8a_reg.txt` and `ex8a_cov.txt`. These initial values have been provided to speed up the analyses.

- **Line 14**
  In conjunction with the `export` statements (lines 8 through 10), this `set` statement results in updated parameter estimates being written to the files after each iteration, and it turns off warning messages.

- **Line 15**
  Begins estimation of the model. The options turn off calculation of the fit tests and instruct estimation to terminate when the change in the parameter estimates from one iteration to the next is less than 0.002.

- **Line 16**
  Writes the estimates of the population model parameter estimates to `ex8a_shw.txt`.

2.9.1.3 Running the Two-Dimensional Latent Regression

To run this sample analysis, launch the console version of ACER ConQuest by typing the command (on Windows) `ConQuestConsole.exe ex8a.cqc`.

ACER ConQuest will begin executing the statements that are in the file `ex8a.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting the two-dimensional latent multiple regression. This particular sample analysis will converge after a single iteration, because we have provided very accurate initial values.

> **NOTE:** If you run this sample analysis without the initial values, it will take in excess of 1000 iterations to converge. While fitting multidimensional models can take a substantial amount of computing time, this particular analysis will take an unusually large number of iterations because of the sparse nature of the data set. In these data, just 40% of the students responded to items on the first dimension; and the first 50 multiple choice items were responded to by only 25% of the sample. All students responded to the last 15 items.

In Figure 2.67, we report the parameter estimates for the population model used in this analysis. In this case, we have two sets of four regression coefficients — a constant and one for each of the three regressors. The conditional variance-covariance matrix is also reported.

All of the results reported here are in their natural metrics (logits). For example, on the first dimension, the difference between the performances of the lower grade and upper grades is 0.700 logits, the male students outperform the female students by 0.072, and a unit increase in the socio-economic status indicator predicts an increase of 0.366 logits in the latent variable. For the second dimension, the difference between the performances of the lower grade and upper grades is 1.391 logits, the male students outperform the female students by 0.229, and a unit increase in the socio-economic status indicator predicts an increase of 0.479 logits in the latent variable.[25]

To aid in the interpretation of these results, it is useful to fit a model without the regressors to obtain estimates of the variance of the two latent variables in this model, the multiple choice items and the extended response item. The command file `ex8a_no_regressors.cqc` is provided with the samples for this purpose. If this command file is executed, it will provide estimates of 0.601 (extended response) and 1.348 (multiple choice) for the variances of the two latent variables.

In Figure 2.68, we report the R^2 for each of the dimensions in the latent regression, and we report the grade, gender and socio-economic status (SES) regression coefficients as effect sizes that have been computed by dividing the estimate of the regression coefficients by the unconditional standard deviation of the respective latent variables.

The results in the table show that the regression model explains marginally more variance for the multiple choice items than it does for the extended response items. Interestingly, the grade and SES effects are similar for the item types, but the gender effect is larger for

[25]The current version of ACER ConQuest does not report standardised regression coefficients or standard errors for the regression parameter estimates. Plausible values can be generated (as explained later in this section) and analysed to obtain estimates of standard errors and to obtain standardised coefficients.

```
================================================================================
ConQuest: Generalised Item Response Modelling Software      Mon Jan 06 06:48:23
TABLES OF POPULATION MODEL PARAMETER ESTIMATES
================================================================================
REGRESSION COEFFICIENTS

                             Dimension
                         ------------------
Regression Variable          1        2

CONSTANT                  -1.431   -1.727
grade                      0.700    1.391
gender                     0.072    0.229
ses                        0.366    0.479
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
================================================================================
COVARIANCE/CORRELATION MATRIX

                             Dimension
                         ------------------
Dimension                    1        2

    1                               0.411
    2                      0.747
--------------------------------------------------------------------------------
Variance                   0.410    0.738
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Values below the diagonal are correlations and values above are covariances
================================================================================
```

These are the two sets of regression coefficients, one for each latent dimension.

This is the conditional covariance/correlation matrix for the two-dimensional latent space.

Figure 2.67: Population Parameter Estimates for a Two-Dimensional Latent Multiple Regression

the multiple choice items. For the extended response items, the gender difference is 9% of a student standard deviation, whereas for the multiple choice it is 19.7%.

Estimate	Extended Response	Multiple Choice
R^2	31.8%	45.3%
Grade Effect	0.903	1.198
Gender Effect	0.090	0.197
SES Effect	0.472	0.407

Figure 2.68: Effect Size Estimates for the Two-Dimensional Latent Multiple Regression

EXTENSION: The model fitted in `ex8b.cqc` has the item response model parameters anchored at the values that were obtained from the model that is fit with `ex8a.cqc`. In general, the item response parameter estimates obtained from fitting a model with regressors will produce item parameter estimates that have smaller standard errors, although the gain in efficiency is generally very small. More importantly, there are occasions when item response model parameters estimated without the use of regressors will be inconsistent. This data set provides such a case, because some of the multiple choice items were administered only to students in the upper grade, while others were administered only to students in the lower grade. Readers interested in this issue are referred to Mislevy & Sheehan (1989) and Adams, Wilson, & Wu (1997).

2.9.2 b) Five-Dimensional Multiple Regression - Unconditional Model

In the Adams et al. (1991) battery of tests, four extended response tests and a set of 15 multiple choice were administered to students in both the upper and lower grades. In this higher-dimensional sample analysis, we are interested in grade, gender and SES effects for the five latent dimensions that are assumed to be assessed by these instruments. First, we will run an unconditional model (using the command file `ex8b.cqc`, described in Section 2.9.2.1) to obtain initial values for a conditional model. Then we will run the conditional model and will also have ACER ConQuest draw plausible values, using the command file `ex8c.cqc` (shown in Section 2.9.3.1).

Because of the high dimensionality, the analysis that is required here is best undertaken with Monte Carlo integration; and as this will need a large number of nodes, the model without regressors (the unconditional model) is fitted in two stages. In the first stage, a small number of nodes with a moderate convergence criterion is used to produce initial values. In the second stage, the initial values are read back into an analysis that uses more nodes and a more stringent convergence criteria.

2.9.2.1 Syntax

The contents of the command file for this tutorial (`ex8b.cqc`), are shown in the code box located below. `ex8b.cqc` is used to fit the Five-Dimensional Latent Unconditional Model to the dataset `ex6_dat.txt`. The list underneath the code box describes each line of syntax.

`ex8b.cqc`:

```
datafile ex6_dat.txt;
format responses 1-18,31-49,100-114 grade 118 gender 119 ses 122-127
       !tasks(52);
model tasks+tasks*step;
recode (9) (0);
score (0,1,2,3,4) (0,1,2,3,4) ( ) ( ) ( ) ( ) ! tasks(1-9);
score (0,1,2,3,4) ( ) (0,1,2,3,4) ( ) ( ) ( ) ! tasks(10-18);
score (0,1,2,3,4) ( ) ( ) (0,1,2,3,4) ( ) ( ) ! tasks(19-28);
score (0,1,2,3,4) ( ) ( ) ( ) (0,1,2,3,4) ( ) ! tasks(29-37);
score (0,1,2,3,4) ( ) ( ) ( ) ( ) (0,1,2,3,4) ! tasks(38-52);
export reg_coefficient >> ex8b_reg.txt;
```

```
12 export covariance >> ex8b_cov.txt;
13 export parameters >> ex8b_prm.txt;
14 set update=yes,warnings=no;
15 estimate!fit=no,method=montecarlo,nodes=400,conv=.01,stderr=none;
16 reset;
17 datafile ex6_dat.txt;
18 format responses 1-18,31-49,100-114 grade 118 gender 119 ses 122-127
19         !tasks(52);
20 model tasks+tasks*step;
21 recode (9) (0);
22 score (0,1,2,3,4) (0,1,2,3,4) ( ) ( ) ( ) ( ) ! tasks(1-9);
23 score (0,1,2,3,4) ( ) (0,1,2,3,4) ( ) ( ) ( ) ! tasks(10-18);
24 score (0,1,2,3,4) ( ) ( ) (0,1,2,3,4) ( ) ( ) ! tasks(19-28);
25 score (0,1,2,3,4) ( ) ( ) ( ) (0,1,2,3,4) ( ) ! tasks(29-37);
26 score (0,1,2,3,4) ( ) ( ) ( ) ( ) (0,1,2,3,4) ! tasks(38-52);
27 export parameters >> ex8b_prm.txt;
28 import init_reg_coefficient << ex8b_reg.txt;
29 import init_covariance << ex8b_cov.txt;
30 import init_parameters << ex8b_prm.txt;
31 set update=yes,warnings=no;
32 estimate!method=montecarlo,nodes=2000,conv=.002,stderr=quick;
33 show !tables=1:3:5 >> Results/ex8b_shw.txt;
```

- **Line 1**
  We are using the data in `ex6_dat.txt`.

- **Lines 2-3**
  The responses to the four extended response instruments administered to all the students are in columns 1 through 18 and 31 through 49; and the responses to the 15 multiple choice items administered to all the students are in columns 100 through 114. Columns 19 through 30 contain the responses to an instrument that was administered to the lower grade students only, and columns 50 through 99 contain the responses to multiple choice items that were administered to students in one of the grades only. We have decided not to include those data in these analyses.

- **Line 4**
  We are using the partial credit model.

- **Line 5**

Any code of `9` (item not responded to by the student) will be recoded to `0` and therefore scored as `0`.

- **Lines 6-10**
  These five `score` statements allocate the items that make up the five instruments to the five different dimensions.

- **Lines 11-14**
  The `export` statements, in conjunction with the `set` statement, ensure that the parameter estimates are written to the files `ex8b_reg.txt`, `ex8b_cov.txt` and `ex8b_prm.txt` after each iteration. This is useful if you want to use the values generated by the final iteration as initial values in a further analysis, as we will do here.

- **Line 15**
  Initiates the estimation of a partial credit model using the Monte Carlo method to approximate multidimensional integrals. This estimation is done with 400 nodes, a value that will probably lead to good estimates of the item parameters, but the latent variance-covariance matrix may not be well estimated.[26] We are using 400 nodes here to obtain initial values for input into the second analysis that uses 2000 nodes. We have specified `fit=no` because we will not be generating any displays and thus have no need for this data at this time. We are also using a convergence criteria of just 0.01, which is appropriate for the first stage of a two-stage estimation.

- **Line 16**
  The `reset` statement resets all variables to their initial values and is used to separate distinct analyses that are in a single command file.

- **Lines 17-26**
  As for lines 1 through 10 above.

- **Line 27**
  We are exporting only the item response model parameter estimates.

- **Lines 28-30**
  Initial values for all of the parameter estimates are being read from the files that were written in the previous analysis.

[26]Simulation studies (Volodin & Adams, 1995) suggest that 1000 to 2000 nodes may be needed for accurate estimation of the variance-covariance matrix.

- **Line 31**
  Used in conjunction with line 27 to ensure that the item response model parameter estimates are written after each iteration.

- **Line 32**
  The estimation method is Monte Carlo, but this time we are using 2000 nodes and a convergence criterion of 0.002. This should be sufficient to produce accurate estimates for all of the parameters.

- **Line 33**
  Writes selected tables to the output file `ex8b_shw.txt`.

2.9.2.2 Running the Five-Dimensional Latent Unconditional Sample Analysis

To run this sample analysis, launch the console version of ACER ConQuest by typing the command (on Windows) `ConQuestConsole.exe ex8b.cqc`.

ACER ConQuest will begin executing the statements in the file `ex8b.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the first `estimate` statement, it will begin fitting the five-dimensional model using a 400-node Monte Carlo integration. ACER ConQuest will then proceed to analyse the data again using a 2000-node Monte Carlo integration, reading initial values from the export files produced by the previous 400-node analysis.

Figure 2.69 shows the estimated population parameters for the unconditional fivedimensional latent space. The analysis shows that the correlation between these latent dimensions is moderately high but unlikely to be high enough to justify the use of a unidimensional model.

> **NOTE:** If you run this sample analysis without the initial values, it will take in excess of 1000 iterations to converge. While fitting multidimensional models can take a substantial amount of computing time, this particular analysis will take an unusually large number of iterations because of the sparse nature of the data set. In these data, just 40% of the students responded to items on the first dimension; and the first 50 multiple choice items were responded to by only 25% of the sample. All students responded to the last 15 items.

```
================================================================================
ConQuest: Generalised Item Response Modelling Software      Fri Feb 21 16:15:34
TABLES OF POPULATION MODEL PARAMETER ESTIMATES
================================================================================
REGRESSION COEFFICIENTS

                                            Dimension
                          ------------------------------------------------------
Regression Variable           1         2         3         4         5

CONSTANT                   -0.374    -0.897    -0.092    -0.967     0.584
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
================================================================================
COVARIANCE/CORRELATION MATRIX

                                            Dimension
                          ------------------------------------------------------
                              1         2         3         4         5

                                      0.456     0.417     0.368     0.680
    2                        0.740              0.396     0.323     0.589
    3                        0.679    0.636               0.315     0.490
    4                        0.736    0.636     0.624               0.543
                             0.807    0.690     0.578     0.784
--------------------------------------------------------------------------------
                             0.608    0.626     0.618     0.411     1.166
--------------------------------------------------------------------------------
                       parameter estimate indicates that it is constrained
Values below the diagonal are correlations and values above are covariances
================================================================================
```

These are the means for each of the latent dimensions.

This is the unconditional variance-covariance matrix for the five-dimensional latent space.

This is the unconditional correlation matrix for the five-dimensional latent space.

These are the unconditional variances for each of the latent dimensions.

Figure 2.69: Population Parameter Estimates for the Unconditional Five-Dimensional Model

2.9.3 c) Five-Dimensional Multiple Regression - Conditional Model

2.9.3.1 Syntax

`ex8c.cqc` is the command file for fitting the five-dimensional latent regression model (the conditional model). It is given in the code box below. `ex8c.cqc` is very similar to the command file used for the unconditional analysis (`ex8b.cqc`, see Section 2.9.2.1). So the description of `ex8c.cqc` underneath the code embedding will focus only on the differences.

`ex8c.cqc`:

```
1  datafile ex6_dat.txt;
2  format responses 1-18,31-49,100-114 grade 118 gender 119 ses 122-127!tasks(52);
3  regression grade,gender,ses;
4  model tasks+tasks*step;
5  recode (9) (0);
6  score (0,1,2,3,4) (0,1,2,3,4) ( ) ( ) ( ) ( ) ! tasks(1-9);
7  score (0,1,2,3,4) ( ) (0,1,2,3,4) ( ) ( ) ( ) ! tasks(10-18);
8  score (0,1,2,3,4) ( ) ( ) (0,1,2,3,4) ( ) ( ) ! tasks(19-28);
9  score (0,1,2,3,4) ( ) ( ) ( ) (0,1,2,3,4) ( ) ! tasks(29-37);
10 score (0,1,2,3,4) ( ) ( ) ( ) ( ) (0,1,2,3,4) ! tasks(38-52);
11 import init_covariance <<ex8b_cov.txt;
12 import anchor_parameters <<ex8b_prm.txt;
13 estimate!method=montecarlo,nodes=2000,conv=.002,iter=3,stderr=quick;
14 show cases !estimates=latent >> Results/ex8c_pls.txt;
15 show cases !estimates=eap >> Results/ex8c_eap.txt;
16 show !tables=1:3:5>> Results/ex8c_shw.txt;
```

- **Line 3**
  The third statement in this command file specifies the regression variables that are to be used in the model (in this case, `grade`, `gender` and `ses`).

- **Line 11**
  This `import` statement uses the estimated unconditional variance-covariance matrix as an initial value. This is done in this sample analysis so that the analysis will be performed more quickly.

- **Line 12**
  This `import` statement requests that item response model parameter values be read from the file `ex8b_prm.txt` (created by the five-dimensional unconditional model) and be anchored at the values specified in that file. This means that, in this analysis, we will not be estimating item parameters.

  > **WARNING:** The current version of ACER ConQuest is unable to estimate both item response model parameters and population model parameters in a conditional model (that is, a model with regressors) when the Monte Carlo method is used. This will not usually be a severe limitation because you can generally obtain consistent estimates of the item parameters by fitting an unconditional model and then entering those estimates as anchored values in a conditional model.

- **Line 13**
  The estimation will be done with the Monte Carlo method, using 2000 nodes and a convergence criterion of 0.002.

- **Lines 14-15**
  These `show` statements result in plausible values and expected a-posteriori estimates being written to the files `ex8c_pls.txt` and `ex8c_eap.txt` respectively.

- **Line 16**
  The final `show` statement requests tables 1, 3 and 5 be written to file `ex8c_shw.txt`.

2.9.3.2 Running the Five-Dimensional Latent Regression Sample Analysis

To run this sample analysis, launch the console version of ACER ConQuest by typing the command (on Windows) `ConQuestConsole.exe ex8c.cqc`.

ACER ConQuest will begin executing the statements in the file `ex8c.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting the five-dimensional model, using a 2000-node Monte Carlo integration. The `show` statements will then be executed, producing files of plausible values, expected a-posteriori ability estimates and output tables. Extracts from the first two files are shown in Figures 2.70 and 2.71.

> **NOTE:** The expected a-posteriori and plausible value files contain values for all cases on all dimensions—even for latent dimensions on which the cases have

not responded to any questions. If there are dimensions for which one or more cases have not made any response, then maximum likelihood ability estimates of the latent variable cannot be calculated.

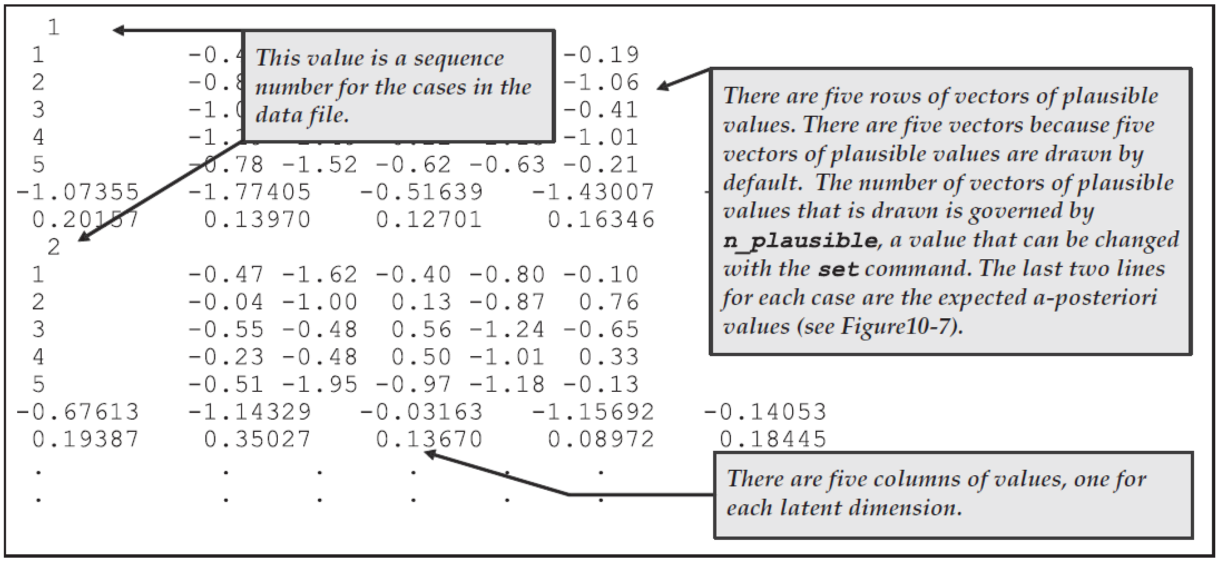


Figure 2.70: Extract from the File of Plausible Value

27

Figure 2.72 shows the estimates of the parameters of the population model. It contains estimates of the four regression coefficients for each of the latent dimensions and the estimate of the conditional variance-covariance matrix between the dimensions. This variance-covariance matrix is also expressed as a correlation matrix.

In Figure 2.73, the estimates of the regression coefficients have been divided by the estimate of the unconditional standard deviation of the respective latent variables to provide effect size estimates. Combining the unconditional results that were obtained from analysing the data with the command file `ex8b.cqc` and were reported in Figure 2.69 with the latent regression results produced using the command file `ex8c.cqc` and reported in Figure 2.72, we obtain the effect size estimates reported in Figure 2.73. Additional analyses of this latent regression model can be obtained by merging the EAP ability estimates and the plausible values with the background variables (such as gender or grade) and undertaking conventional analyses.

[27] The EAP values in Figures 2.70 and 2.71 are not the same, because ACER ConQuest selects a different random number generator seed each time EAP values are generated.

```
  1
-1.09143   -1.759                                   0   -0.52452
 0.17039    0.135                                   2    0.19183
  2
-0.67869   -1.12420   -0.02075   -1.15043   -0.12802
 0.19116    0.32623    0.13262    0.08459    0.17645
  3
-1.04968   -1.55069   -0.58795   -1.61956   -0.84419
 0.11481    0.35155    0.35113    0.17914    0.19273
  .          .          .          .          .
  .          .          .          .
```

This value is a sequence number for the cases in the file.

There are two rows of values. The first contains the EAP ability estimates for each dimension, and the second contains the error variance of the posterior distributions.

There are five columns of values, one for each latent dimension.

Figure 2.71: Extract from the File of Expected A-posteriori Values

```
=================================================================================
ConQuest: Generalised Item Response Modelling Software      Thu Jan 09 11:31:07
TABLES OF POPULATION MODEL PARAMETER ESTIMATES
=================================================================================
REGRESSION COEFFICIENTS

                                        Dimension
                         ---------------------------------------------------
Regression Variable          1          2          3          4          5

CONSTANT                  -1.459     -1.959     -0.921     -1.951     -1.459
grade                      0.726      0.705      0.556      0.701      1.334
gender                     0.100      0.076      0.047     -0.009      0.248
ses                        0.367      0.265      0.260      0.207      0.461
---------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
=================================================================================
COVARIANCE/CORRELATION MATRIX

                                        Dimension
                         ---------------------------------------------------
Di                           1          2          3          4

                                      0.290      0.305      0.188      0.365
                           0.652                 0.284      0.166      0.311
   3                       0.656      0.550                 0.179      0.264
   4                       0.582      0.462      0.476                 0.266
                           0.722      0.553      0.449      0.653
---------------------------------------------------------------------------------
                           0.400      0.493      0.542      0.261      0.638
---------------------------------------------------------------------------------
                   a parameter estimate indicates that it is constrained
Values below the diagonal are correlations and values above are covariances
=================================================================================
```

These are the regression coefficients for each of the latent dimensions.

This is the conditional variance-covariance matrix for the five-dimensional latent space.

This is the conditional correlation matrix for the five-dimensional latent space.

These are the conditional variances for each of the latent dimensions.

Figure 2.72: Population Model Parameter Estimates for the Five-Dimensional Latent Regression

Estimate	Force & Motion	Light & Sight	Matter	Earth & Space	Multiple Choice
R^2	34%	21%	12%	36%	45%
Grade Effect	1.188	1.134	0.900	0.586	1.143
Gender Effect	0.160	0.125	0.083	-0.039	0.213
SES Effect	0.600	0.425	0.417	0.511	0.396

Figure 2.73: Effect Size Estimates for the Five-Dimensional Latent Multiple Regression

2.9.4 Summary

In this section, we have seen how ACER ConQuest can be used to fit multidimensional latent regression models. The fitting of multidimensional latent regression models brings together two sets of functionality that we have demonstrated in previous sections: the facility to estimate latent regression models and the facility to fit multidimensional item response models.

2.10 Importing Design Matrices

In this section, we provide sample analyses in which the model is described through a design matrix, rather than through a `model` statement. In each of the other sample analyses in this manual, a `model` statement is used to specify the form of the model, and ACER ConQuest then automatically builds the appropriate design matrix. While the `model` statement is very flexible and allows a diverse array of models to be specified, it does not provide access to the full generality of the model that is available when a design matrix is directly specified rather than built with a `model` statement.

Contexts in which the importation of design matrices are likely to be useful include:

- *Imposing Parameter Equality Constraints:* On some occasions, you may wish to constrain the values of one or more item parameters to the same value. For example, you may want to test the hypothesis of the equality of two or more parameters.

- *Mixing Rating Scales:* Under some circumstances, you may need to analyse a set of items that contain subsets of items, each of which use different rating scales. These subsets could be assessing the same latent variable, or they could be assessing different latent variables and a multidimensional analysis may be undertaken.

- *Mixing Faceted and Non-faceted Data:* A set of item responses may include a mix of objectively scored items (for example, multiple choice items) and some items that required the use of raters. Under these circumstances, the rater facet would not apply to the objectively scored items.

- *Modelling Within-item Multidimensionality:* ACER ConQuest can only automatically generate design matrices for within-item multidimensional tests if the mean of the latent variables is set to zero. Within-item multidimensional tests that do not have this constraint can, however, be analysed if a design matrix is imported.

In this section, we will provide two sample analyses in which a design matrix is imported so that a model that cannot be described by a `model` statement can be fitted. The first sample analysis (a)) illustrates the use of an imported design to model a mixture of two rating scales. The second (b)) shows how within-item multidimensionality without setting the means of the latent variables to zero can be accommodated.

The data we analyse in this section were collected as part of the SEPUP study (Roberts et al., 1997). It consists of the responses of 721 students to a set of 18 items that used two different rubrics. Items 1, 2, 3, 6, 10, 12, 13, 16, 17 and 18 used one rubric, and items 4, 5, 7, 8, 9, 11, 14, and 15 used an alternative rubric.

2.10.1 a) Mixing Rating Scales

In this sample analysis, we fit a sequence of three models to these data. First, we fit a rating scale model that imposes a common rating structure on all of the items. Then we use an imported design matrix to fit a model that uses two rating scales, one for the items that used the first rubric and one for the items that used the second rubric. We then fit a partial credit model.

2.10.1.1 Required files

The files used in this sample analysis are:

filename	content
ex9a.cqc	The command statements that we use.
ex9a_dat.txt	The data.
ex9a_des.txt	The design matrix imported to fit the mixture of rating scales.
ex9a_1_shw.txt	The results of the rating scale analysis.
ex9a_2_shw.txt	The results of the mixture of two rating scales.
ex9a_3_shw.txt	The results of the partial credit analysis.

2.10.1.2 Syntax

The command file used to fit the model in this section (`ex9a.cqc`) is shown in the code box below. In the text that follows the figure, each line of syntax is explained.

`ex9a.cqc`:

```
1 datafile ex9a_dat.txt;
2 format responses 5-9,12,15,17,18,24-32;
3 codes 1 2 3 4 5;
4 score (1 2 3 4 5) (0 1 2 3 4);
5 model item + step;
6 estimate;
7 show >> Results/ex9a_1_shw.txt;
8 reset;
9 datafile ex9a_dat.txt;
10 format responses 5-9,12,15,17,18,24-32;
11 codes 1 2 3 4 5;
12 score (1 2 3 4 5) (0 1 2 3 4);
13 model item + step;
14 import designmatrix << ex9a_des.txt;
15 estimate;
16 show >> Results/ex9a_2_shw.txt;
17 reset;
18 datafile ex9a_dat.txt;
19 format responses 5-9,12,15,17,18,24-32;
20 codes 1 2 3 4 5;
21 score (1 2 3 4 5) (0 1 2 3 4);
22 model item + item*step;
23 estimate;
24 show >> Results/ex9a_3_shw.txt;
```

- **Line 1**
  The data file is `ex9a_dat.txt`.

- **Line 2**
  The `format` statement describes the locations of the 18 items in the data file.

- **Line 3**
  The codes 1, 2, 3, 4 and 5 are valid.

- **Line 4**
  A `score` statement is used to assign scores to the codes. As this is a unidimensional analysis, a `recode` statement could have been used as an alternative to this `score` statement.

- **Line 5**
  This `model` statement results in a rating scale model that is applied to all items.

- **Line 6**
  Commences the estimation.

- **Line 7**
  Writes some results to the file `ex9a_1_shw.txt`.

- **Line 8**
  Resets all system values at their defaults so that a new analysis can be started.

- **Lines 9-12**
  As for lines 1 through 4 above.

- **Lines 13-14**
  These two lines together result in a model being fitted that uses a mixture of two rating scales. The `model` statement must be supplied even when a model is being imported. This `model` statement allows ACER ConQuest to identify the generalised items that are to be analysed with the imported model. In this case, we need ACER ConQuest to identify 18 items, so we simply use a `model` statement that will generate a standard rating scale model for the 18 items. The second line imports the design that is in the file `ex9a_des.txt`. This matrix will replace the design matrix that is automatically generated by ACER ConQuest in response to the `model` statement. The contents of the imported design are illustrated and described in Figure 2.74.

- **Lines 15-17**
  Estimates the model and writes results to `ex9a_2_shw.txt` and resets the system values.

- **Lines 18-24**
  This set of commands is the same as for lines 1 through 7, except that we are fitting a partial credit rather than a rating scale model and writing to the file `ex9a_3_shw.txt`.

NOTE: The number of rows in the imported design matrix must correspond to the number of rows that ACER ConQuest is expecting. ACER ConQuest determines this using a combination of the `model` statement and an examination of the data. The `model` statement indicates which combinations of facets will be used to define generalised items. ACER ConQuest then examines the data to find all of the different combinations; and for each combination, it finds the number of categories.

The best strategy for manually building a design matrix usually involves running ACER ConQuest, using a `model` statement to generate a design matrix, and then exporting the automatically generated matrix, using the `designmatrix` argument of the `export` statement. The exported matrix can then be edited as needed.

2.10.1.3 Running the Mixture of Rating Scales

To run this sample analysis, launch the console version of ACER ConQuest by typing the command (on Windows) `ConQuestConsole.exe ex9a.cqc`.

ACER ConQuest will begin executing the statements that are in the file `ex9a.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the first `estimate` statement, it will begin fitting the rating scale model to the data. The results will be written to the file `ex9a_1_shw.txt`. ACER ConQuest will then proceed to analyse the imported model, writing results to the file `ex9a_2_shw.txt`; and then the partial credit model will be fitted, writing the results to `ex9a_3_shw.txt`.

In Figure 2.75, the fit of this sequence of models is compared using the deviance statistic. Moving from the rating scale to the mixture improves the deviance by 50.42 and requires an additional three parameters; this is clearly significant. The improvement between the mixture and partial credit model is 160.3, and the partial credit model requires 48 additional parameters. This improvement is also significant, although the amount of improvement per parameter is considerably less than that obtained in moving from the rating scale to the mixture of two rating scales. An examination of the parameter fit statistics in the files `ex9a_1_shw.txt`, `ex9a_2_shw.txt` and `ex9a_3_shw.txt` leads to the same conclusions as does the examination of Figure 2.75.

When a model is imported, the ACER ConQuest output will only be provided in an abbreviated form with all parameters listed in one Table. The output produced for the mixture of rating scales is shown in Figure 2.76.

2.10.2 b) Within-Item Multidimensionality

As a second sample analysis that uses an imported design matrix, we will return to the within-item multidimensional sample analysis that was used in section 2.8. In section

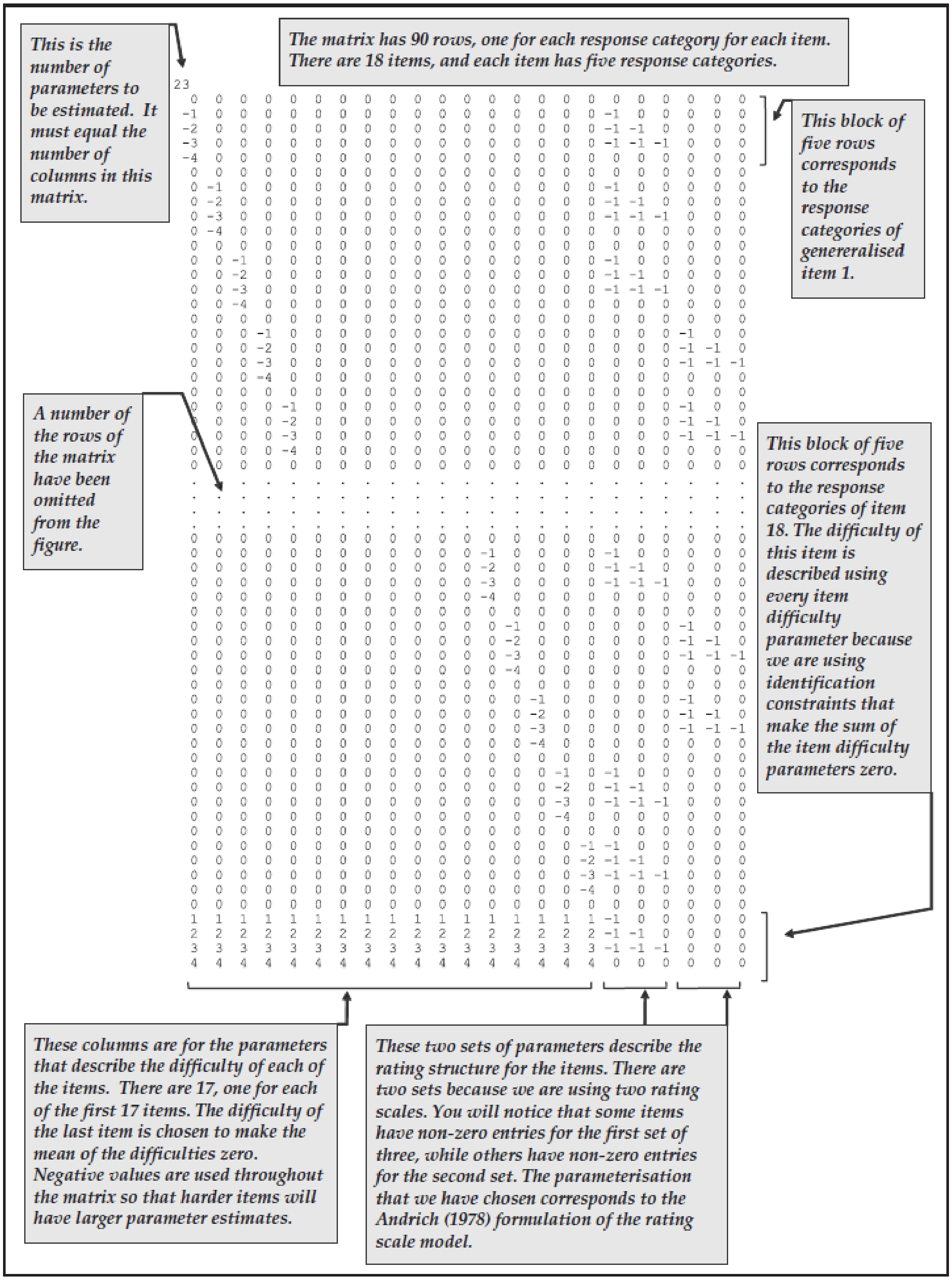


Figure 2.74: The Imported Design Matrix for Mixing Two Rating Scale

Model	Deviance	Estimated Parameters	Difference: Deviance	Difference: Estimated Parameters
Rating Scale Model	9380.95	22		
Mixture of Two Rating Scales	9330.50	25	50.42	3
Partial Credit Model	9170.22	73	160.28	48

Figure 2.75: Deviance Statistics for the Three Models Fitted to the SEPUP Data

```
==============================================================================
Parameter Estimates
--------------------------------------------------------------------------------
   VARIABLES                             UNWEIGHTED FIT            WEIGHTED FIT
                                     ----------------------   ----------------------
              ESTIMATE    ERROR^  MNSQ        CI         T   MNSQ        CI         T
--------------------------------------------------------------------------------
Parameter 1  -0.01670  0.07419  0.79 ( 0.85, 1.15) -3.0   0.82 ( 0.86, 1.14) -2.7
Parameter 2   0.4161                     5, 1.15) -4.0   0.79 ( 0.85, 1.15) -3.0
Parameter 3   0.246                      5, 1.15) -4.5   0.71 ( 0.85, 1.15) -4.3
Parameter 4  -0.0 32                     5, 1.15) -5.0   0.68 ( 0.86, 1.14) -4.9
Parameter 5  -1. 312                     5, 1.15) -3.7   0.75 ( 0.84, 1.16) -3.3
Parameter 6    .22591  0.06270  1.16 ( 0.88, 1.12)  2.5   1.17 ( 0.88, 1.12)  2.7
Parameter 7   0.53499  0.06487  0.99 ( 0.88, 1.12) -0.1   0.99 ( 0.88, 1.12) -0.2
Parameter 8  -1.47939  0.06376  0.73 ( 0.88, 1.12) -4.9   0.73 ( 0.88, 1.12) -4.9
Parameter 9  -1.55998  0.06390  0.87 ( 0.88, 1.12) -2.3   0.87 ( 0.88, 1.12) -2.3
Parameter 10 -0.56111  0.05987  1.00 ( 0.85, 1.15)  0.1   0.99 ( 0.85, 1.15) -0.1
Parameter 11  0.01046  0.06360  1.12 ( 0.85, 1.15)  1.6   1.08 ( 0.85, 1.15)  1.0
Parameter 12  0.41289  0.07914  0.81 ( 0.85, 1.15) -2.7   0.84 ( 0.85, 1.15) -2.2
Parameter 13  0.34666  0.07870  0.77 ( 0.85, 1.15) -3.3   0.77 ( 0.85, 1.15) -3.2
Parameter 14  1.44429  0.08260  0.78 ( 0.79, 1.21) -2.3   0.85 ( 0.78, 1.22) -1.3
Parameter 15  1.60987  0.08355  0.82 ( 0.79, 1.21) -1.8   0.92 ( 0.78, 1.22) -0.7
Parameter 16 -0.05226  0.07534  0.36 ( 0.79, 1.21) -8.0   0.40 ( 0.78, 1.22) -7.1
Parameter 17 -0.00939  0.07552  0.39 ( 0.79, 1.21) -7.6   0.38 ( 0.78, 1.22) -7.4
Parameter 18 -1.97045  0.05359  1.63 ( 0.90, 1.10) 10.0   1.48 ( 0.89, 1.11)  7.3
Parameter 19 -1.46062  0.05894  1.27 ( 0.90, 1.10)  4.6   1.52 ( 0.88, 1.12)  7.0
Parameter 20  0.70711  0.10329  1.96 ( 0.90, 1.10) 14.3   1.55 ( 0.78, 1.22)  4.3
Parameter 21 -3.08112  0.06452  1.41 ( 0.90, 1.10)  6.9   1.36 ( 0.89, 1.11)  5.7
Parameter 22 -1.61459  0.05710  1.26 ( 0.90, 1.10)  4.6   1.22 ( 0.90, 1.10)  3.9
Parameter 23  0.66223  0.07707  1.36 ( 0.90, 1.10)  6.1   1.32 ( 0.87, 1.13)  4.3
```

The parameters are simply listed by number.

For this particular model, the last six values are step parameters. They don't fit well!

Figure 2.76: Unlabelled Output that is Produced when a Design Matrix is Imported

2.8, we used `lconstraints=cases`, since this enabled ACER ConQuest to automatically generate a design matrix for the model. If the model is to be identified by applying constraints to the item parameters, then ACER ConQuest cannot automatically generate the design matrix for withinitem multidimensional models.[28]

2.10.2.1 Required files

The files used in this sample analysis are:

filename	content
ex9b.cqc	The command statements.
ex7_dat.txt	The data.
ex9b_des.txt	The design matrix imported to fit the within-item multidimensional model.
ex9b_prm.txt	Initial values for the item parameter estimates.
ex9b_reg.txt	Initial values for the regression parameter estimates.
ex9b_cov.txt	Initial values for the covariance parameter estimates.
ex9b_shw.txt	The results of the rating scale analysis.

2.10.2.2 Syntax

The command file for this sample analysis is `ex9b.cqc` (as shown in the code box below). As this command file is very similar to `ex7c.cqc` (which was discussed in Section 2.8.4.2), the list below the embedded code will only highlight the differences between `ex9b.cqc` and `ex7c.cqc`.

`ex9b.cqc`:

```
datafile ex7_dat.txt;
format responses 1-9;
set update=yes,warnings=no;
score (0,1) (0,1) (   ) (   ) ! items(1);
score (0,1) (0,1) (0,1) (   ) ! items(2);
score (0,1) (0,1) (   ) (0,1) ! items(3);
score (0,1) (0,1) (0,1) (   ) ! items(4);
score (0,1) (   ) (0,1) (   ) ! items(5);
```

[28] This would be necessary if a latent regression model were being estimated.

```
9  score (0,1) (   ) (0,1) (   ) ! items(6);
10 score (0,1) (0,1) (0,1) (0,1) ! items(7);
11 score (0,1) (   ) (   ) (0,1) ! items(8);
12 score (0,1) (   ) (   ) (0,1) ! items(9);
13 model items;
14 import designmatrix << ex9b_des.txt;
15 export parameters >> ex9b_prm.txt;
16 export reg_coefficients >> ex9b_reg.txt;
17 export covariance >> ex9b_cov.txt;
18 estimate !method=montecarlo,nodes=200,conv=.01;
19 reset;
20 datafile ex7_dat.txt;
21 format responses 1-9;
22 set update=yes,warnings=no;
23 score (0,1) (0,1) (   ) (   ) ! items(1);
24 score (0,1) (0,1) (0,1) (   ) ! items(2);
25 score (0,1) (0,1) (   ) (0,1) ! items(3);
26 score (0,1) (0,1) (0,1) (   ) ! items(4);
27 score (0,1) (   ) (0,1) (   ) ! items(5);
28 score (0,1) (   ) (0,1) (   ) ! items(6);
29 score (0,1) (0,1) (0,1) (0,1) ! items(7);
30 score (0,1) (   ) (   ) (0,1) ! items(8);
31 score (0,1) (   ) (   ) (0,1) ! items(9);
32 model items;
33 import designmatrix << ex9b_des.txt;
34 import init_parameter << ex9b_prm.txt;
35 import init_reg_coefficients << ex9b_reg.txt;
36 import init_covariance << ex9b_cov.txt;
37 export parameters >> ex9b_prm.txt;
38 export reg_coefficients >> ex9b_reg.txt;
39 export covariance >> ex9b_cov.txt;
40 estimate !method=montecarlo,nodes=1000;
41 show >> Results/ex9b_shw.txt;
```

- **Lines 3 & 22**
  Note that these `set` statements do not include `lconstraints=cases`, as did the `set` statements in the command file `ex7c.cqc`, shown in Section 2.8.4.2 (lines 3 and 21). Thus, the means for the latent dimensions will not be constrained, and identification

of the model must be assured through the design for the item parameters. ACER ConQuest cannot automatically generate a correct design for a within-item multidimensional model without `lconstraints=cases`, so an imported design is necessary.

- **Lines 14 & 33**
  These `import` statements request that a user-specified design be imported from the file `ex9b_des.txt` to replace the design that ACER ConQuest has automatically generated.[29] The contents of the imported design are shown in Figure 2.77. A full explanation of how designs can be prepared for within-item multidimensional models is beyond the scope of this manual. The interested reader is referred Design Matrices in section 3.1 and to Volodin & Adams (1995).

- **Line 41**
  The `show` statement cannot produce individual tables when an imported design matrix is used.

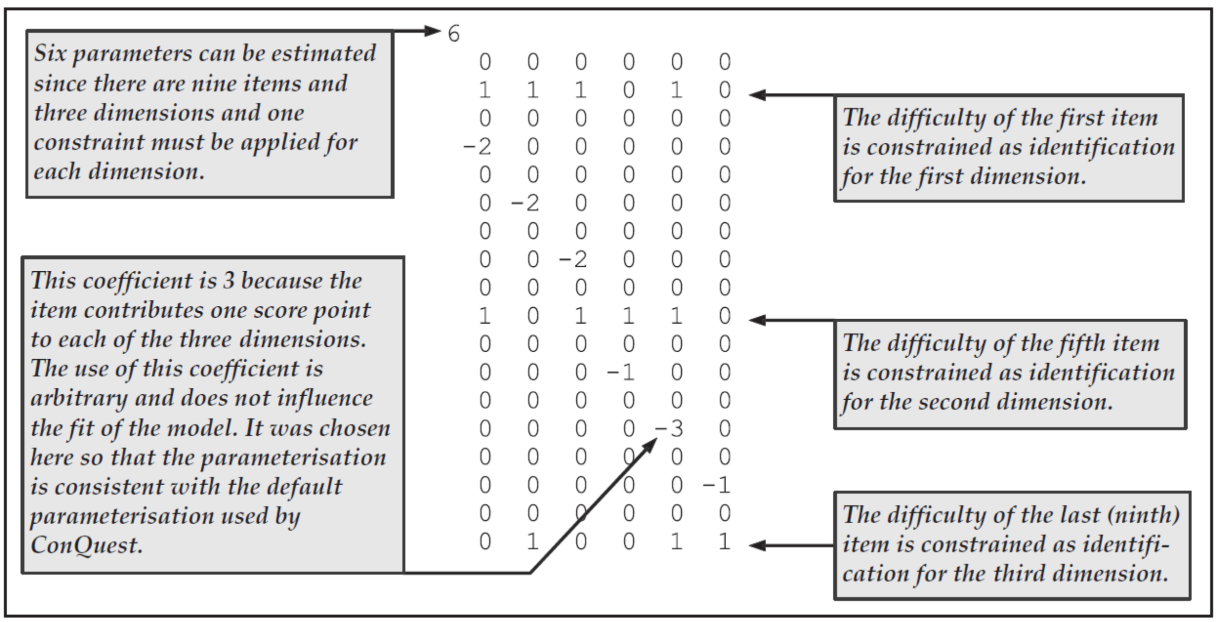


Figure 2.77: Design Matrix Used to Fit a Three-Dimensional Within-Item Model

[29] ACER ConQuest will attempt to build a design for within-item multidimensional models, but this design will be incorrect if `lconstraints=cases` is not used.

2.10.2.3 Running the Within-Item Multidimensional Sample Analysis with an Imported Design Matrix

To run this sample analysis, launch the console version of ACER ConQuest by typing the command (on Windows) `ConQuestConsole.exe ex9b.cqc`.

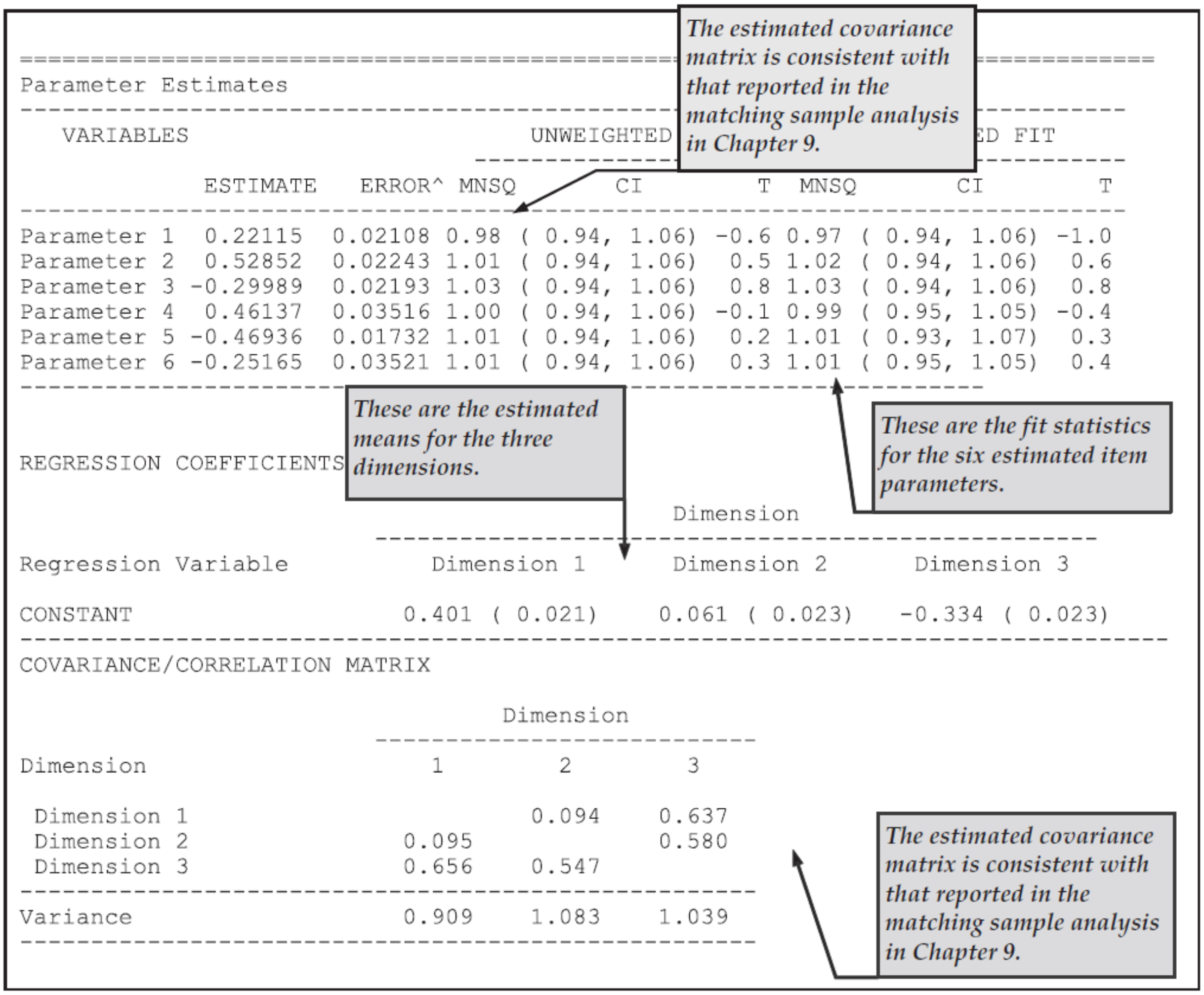
ACER ConQuest will begin executing the statements that are in the file `ex9b.cqc`; and as they are executed, they will be echoed on the screen. As with the corresponding sample analysis in section 2.8, this sample analysis will fit a within-in three-dimensional form of Rasch's simple logistic model, first approximately, using 200 nodes, and then more accurately, using 1000 nodes.

The results obtained from this analysis are shown in Figure 2.78.

> **EXTENSION:** The multidimensional item response model given in section 3.1 is written as:
>
> $f(x;\xi|\theta) = \psi(\theta,\xi)\exp[x'(B\theta + A\xi)]$
>
> with $\theta \sim MVN(\mu, \sum)$.
>
> If θ is rewritten as $\theta^* + \mu$ with $\theta^* \sim MVN(0, \sum)$, then it can be shown that two models, one described with the design matrices A and B and one descrived with design matrices A^* and B^*, are equivalent if
>
> $B^*\mu^* + A^*\xi^* = B\mu + A\xi$
>
> A small amount of matrix algebra can be used to show that the results reported in Figures 2.65 and 2.78 satisfy this condition.

2.10.3 Summary

In this section, we have seen how design matrices can be imported to fit models for which ACER ConQuest cannot automatically generate a correct design. Imported designs can be used to fit models that have equality constraints imposed on parameters, models that involve the mixtures of rating scales, models that require the mixing of faceted and non-faceted data, and within-item multidimensional models that do not set the means of the latent variables to zero.



```
========================================================================================
Parameter Estimates
-------------------------------------------------------------------------------------
   VARIABLES                           UNWEIGHTED                          ED FIT
                                    ---------------------------------------------------
            ESTIMATE   ERROR^ MNSQ          CI          T  MNSQ         CI          T
-------------------------------------------------------------------------------------
Parameter 1  0.22115  0.02108 0.98 ( 0.94, 1.06) -0.6 0.97 ( 0.94, 1.06) -1.0
Parameter 2  0.52852  0.02243 1.01 ( 0.94, 1.06)  0.5 1.02 ( 0.94, 1.06)  0.6
Parameter 3 -0.29989  0.02193 1.03 ( 0.94, 1.06)  0.8 1.03 ( 0.94, 1.06)  0.8
Parameter 4  0.46137  0.03516 1.00 ( 0.94, 1.06) -0.1 0.99 ( 0.95, 1.05) -0.4
Parameter 5 -0.46936  0.01732 1.01 ( 0.94, 1.06)  0.2 1.01 ( 0.93, 1.07)  0.3
Parameter 6 -0.25165  0.03521 1.01 ( 0.94, 1.06)  0.3 1.01 ( 0.95, 1.05)  0.4
------------------------------------------------------------------------------

REGRESSION COEFFICIENTS

                                            Dimension
                         ---------------------------------------------------
Regression Variable            Dimension 1       Dimension 2       Dimension 3

CONSTANT                     0.401 ( 0.021)    0.061 ( 0.023)   -0.334 ( 0.023)
----------------------------------------------------------------------------------
COVARIANCE/CORRELATION MATRIX

                                    Dimension
                         --------------------------
Dimension                      1         2         3

 Dimension 1                           0.094     0.637
 Dimension 2                 0.095               0.580
 Dimension 3                 0.656     0.547
----------------------------------------------------
Variance                     0.909     1.083     1.039
----------------------------------------------------
```


Figure 2.78: Output from the Three-Dimensional Within-Item Sample Analysis with Imported Design

2.11 Modelling multiple choice items with the two-parameter logistic model

The Rasch's simple logistic model specifies the probability of a correct response in a given item as a function of on the individual's ability and the difficulty of the item. The model assumes that all items have equal discrimination power in measuring the latent trait by fixing the slope parameter to ´1´ (Rasch, 1980). The two-parameter logistic model (2PL) is a more general model that estimates a discrimination parameter for each item. In ACER ConQuest we refer to these additional parameters as scoring parameters, or scores. In the 2PL, items have different levels of difficulty and also different capabilities to discriminate among individuals of different proficiency (Birnbaum, 1968). Thus, the 2PL model 'frees' the slope of each parameter, allowing different discrimination power for each item. This tutorial exemplifies how to fit a 2PL model for dichotomously scored data in ACER ConQuest. The actual form the model that is fit for dichotomous data is provided as equation (3) in *Note 6: Score Estimation and Generalised Partial Credit Models*.

2.11.1 Required files

The files used in this sample analysis are:

filename	content
ex10.cqc	The command statements.
ex1_dat.txt	The data.
ex1_lab.txt	The variable labels for the items on the multiple choice test.
ex10_shw.xlsx	The results of the two-parameter analysis.
ex10_itn.xlsx	The results of the traditional item analyses.

(The last two files are created when the command file is executed.)

The data used in this tutorial comes from a 12-item multiple-choice test that was administered to 1000 students. The data have been entered into the file `ex1_dat.txt`, using one line per student. A unique student identification code has been entered in columns 1 through 5, and the students' responses to each of the items have been recorded in columns 12 through 23. The response to each item has been allocated one column; and the codes `a`, `b`, `c` and `d` have been used to indicate which alternative the student chose for each item. If a student failed to respond to an item, an `M` has been entered into the data file. An extract from the data file is shown in Figure 2.79 [30].

[30] In Figure 2.79, each column of the data file is labelled so that it can be easily referred to in the text.

```
                 1         2
12345678901234567890123(column numbers)
40016     acdabaeadacd
655       acdcccecbaca
31140     eccdbcebbacb
.         .
.         .
50321     dabcMcebdaca
30782     acddbcebbacc
.         .
.         .
```

Figure 2.79: Extract from the Data File `ex1\_dat.txt`

In this sample analysis, the generalised model for dichotomously-scored items will be fitted to the data. Traditional item analysis statistics are generated.

2.11.2 Syntax

`ex10.cqc` is the command file used in this tutorial to analyse the data; the file is shown in the code box below. Each line of commands in `ex10.cqc` is detailed in the list underneath the command file.

`ex10.cqc`:

```
Datafile ex1_dat.txt;
Format id 1-5 responses 12-23;
Labels << ex1_lab.txt;
set lconstraints=cases;
Key acddbcebbacc ! 1;
Model item!scoresfree;
Estimate;
Show  !filetype=xlsx  >> Results/ex10_shw.xlsx;
```

The actual ACER ConQuest data file does not have any column labels.

```
9  Itanal!filetype=xlsx  >> Results/ex10_itn.xlsx;
10 Plot icc! filesave=yes >> Results/ex10_;
11 Plot mcc! legend=yes,filesave=yes >> Results/ex10_;
12 plot icc! gins=all,raw=no,overlay=yes,filesave=yes >> Results/ex10_;
```

- **Line 1**
  The `datafile` statement indicates the name and location of the data file. Any file name that is valid for the operating system you are using can be used here.

- **Line 2**
  The `format` statement describes the layout of the data in the file `ex1_dat.txt`. This `format` statement indicates that a field that will be called `id` is located in columns 1 through 5 and that the `responses` to the items are in columns 12 through 23 of the data file. Every `format` statement must give the location of the responses. In fact, the explicit variable responses must appear in the `format` statement or ACER ConQuest will not run. In this particular sample analysis, the responses are those made by the students to the multiple choice items; and, by default, `item` will be the implicit variable name that is used to indicate these responses. The levels of the `item` variable (that is, item 1, item 2 and so on) are implicitly identified through their location within the set of responses (called the response block) in the `format` statement; thus, in this sample analysis, the data for item 1 is located in column 12, the data for item 2 is in column 13, and so on.

- **Line 3**
  The `labels` statement indicates that a set of labels for the variables (in this case, the items) is to be read from the file `ex1_lab.txt`. An extract of `ex1_lab.txt` is shown in Figure 2.80. (This file must be text only; if you create or edit the file with a word processor, make sure that you save it using the text only option.) The first line of the file contains the special symbol `===>` (a string of three equals signs and a greater than sign) followed by one or more spaces and then the name of the variable to which the labels are to apply (in this case, `item`). The subsequent lines contain two pieces of information separated by one or more spaces. The first value on each line is the level of the variable (in this case, `item`) to which a label is to be attached, and the second value is the label. If a label includes spaces, then it must be enclosed in double quotation marks (" "). In this sample analysis, the label for item 1 is `BSMMA01`, the label for item 2 is `BSMMA02`, and so on.

- **Line 4**
  The `set` statement specifies new values for a range of ACER ConQuest system

```
===> item
1      BSMMA01
2      BSMMA02
3      BSMMA03
4      BSMMA04
5      BSMMA05
6      BSMMA06
```

Figure 2.80: Contents of the Label File ex1_lab.txt

variables. In this case, the use of the `lconstraints` argument is setting the identification constraints to `cases`. Therefore, the constraints will be set through the population model by forcing the means of the latent variables to be set to zero and allowing all item parameters (difficulty and discrimination) to be free. The use of `cases` as the identification constraint is required when estimating a 2PL.

- **Line 5**
  The `key` statement identifies the correct response for each of the multiple choice test items. In this case, the correct answer for item 1 is `a`, the correct answer for item 2 is `c`, the correct answer for item 3 is `d`, and so on. The length of the argument in the `key` statement is 12 characters, which is the length of the response block given in the `format` statement. If a `key` statement is provided, ACER ConQuest will recode the data so that any response `a` to item 1 will be recoded to the value given in the `key` statement option (in this case, `1`). All other responses to item 1 will be recoded to the value of the `key_default` (in this case, `0`). Similarly, any response `c` to item 2 will be recoded to `1`, while all other responses to item 2 will be recoded to `0`; and so on.

- **Line 6**
  The `model` statement must be provided before any traditional or item response analyses can be undertaken. In this example, the argument for the `model` statement is the name of the variable that identifies the response data that are to be analysed (in this case, `item`). The option `scoresfree` indicates that a score is to be estimated for each scoring category. In this case the data are dichotomously coded, so the resulting model is the 2PL model.

- **Line 7**

The `estimate` statement initiates the estimation of the item response model.

- **Line 8**
  The `show` statement produces a sequence of tables that summarise the results of fitting the item response model. The option `filetype` sets the format of the results file, in this case an Excel file. The redirection symbol (`>>`) is used so that the results will be written to the file `ex10_shw.xlsx` in your current directory.

- **Line 9**
  The `itanal` statement produces a display of the results of a traditional item analysis. As with the `show` statement, the results are redirected to a file (in this case, `ex10_itn.xlsx`).

- **Line 10**
  The `plot icc` statement will produce 12 item characteristic curve plots, one for each item. The plots will compare the modelled item characteristic curves with the empirical item characteristic curves. The option `filesave` indicates that the resulting plot will be saved into a file in your working directory. The redirection symbol (`>>`) is used so that the plots will be written to png files named `ex10_`. The name of the file will be completed with 'item X' where the X represents the number of the item (e.g.`ex10_item7`). Note that the `plot` command is not available in the console version of ACER ConQuest.

- **Line 11**
  The `plot mcc` statement will produce 12 category characteristic curve plots, one for each item. The plots will compare the modelled item characteristic curves with the empirical item characteristic curves (for correct answers) and will also show the behaviour of the distractors. As with the `plot icc` statement, the results are redirected to a file (in this case, `ex10_`). Note that this command is not available in the console version of ACER ConQuest.

- **Line 12**
  The `plot icc` statement will produce 12 item characteristic curve plots, one for each item. The option `gins=all` indicates that one plot is provided for each listed generalised item. The use of the `raw=no` option prevents the display of the raw data in the plot. The `overlay=yes` option allows the requested plots to be shown in a single window. As with the previous `plot` statements, the resulting plots are saved to png files in the working directory.

2.11.3 Running the two-parameter model

To run this sample analysis, start the GUI version. Open the file `ex10.cqc` and choose `Run`→`Run All`. ACER ConQuest will begin executing the statements that are in the `cqc file`; and as they are executed they will be echoed in the Output Window. When it reaches the estimation command ACER ConQuest will begin fitting the two-parameter model to the data. After the estimation is completed, the two statements that produce Excel files output (`show` and `itanal`) will be processed. The `show` statement will produce an Excel file (`ex10_shw.xlsx`) with nine tabs summarising the results of fitting the item response model. The `itanal` statement will produce an Excel file (`ex10_itn.xlsx`) with one tab showing items statistics. In the case of the GUI version, the `plot` statements will produce 25 plots altogether. 12 plots will contain the item characteristic curve by score category for each of the items in the data. 12 plots will contain the item characteristic curve by response category for each of the items in the data. The last `plot` statement will produce one plot with the ICC by score category for all items.

2.11.4 Results of fitting the two parameter model

As mentioned above, the show file will contain nine tabs. The first tab in the `ex10_shw.xlsx` file shows a summary of the estimation. An extract is shown in Figure 2.81. The table indicates the data set that set that was analysed and provides summary information about the model fitted (e.g.the number of parameters estimated, the number of iterations that the estimation took, the reason for the estimation termination).

The second tab in the `ex10_shw.xlsx` Excel file gives the parameter difficulty estimates for each of the items along with their standard errors and some diagnostics tests of fit (Figure 2.82 [31]). The difficulty parameter estimates the "delta" values in equation (3) of *Note 6: Score Estimation and Generalised Partial Credit Models*. The last column in the table (*2PL scaled estimate*) shows the two-parameter scaled estimate of the item. Each value in this column is the delta value divided by the estimate of the score and is a common *alternative expression* of item difficulty for 2PL models. At the bottom of the table an item separation reliability and chi-squared test of parameter equality are reported.

The sixth and seventh tabs provide the item map of the item difficulty parameters (not shown here). The first of these maps provides an item difficulty plot according to the estimate displayed in the *2PL scaled estimate* column in Figure 2.82. The second map is based on the unscaled estimate (*estimate* column in Figure 2.82).

[31] Note: the tables in Figs. 2.82–2.84 show decimal commas in the parameter estimates. Different versions of Excel might render decimal marks differently (e.g., as 'dot').

```
================================================================================
ConQuest: Generalised Item Response Modelling Software    Wed Jul 08 09:21 2015
SUMMARY OF THE ESTIMATION
==========================================================Build: Jul  8 2015===

Estimation method was: Gauss-Hermite Quadrature with 15 nodes
No node filtering
Assumed population distribution was: Gaussian
Constraint was: CASES
The Data File: ex1.dat
The format:  id 1-5 responses 12-23
No case weights
The regression model:
Grouping Variables:
The item model: item
Two parameter logistic model
Cases in file: 1000  Cases in estimation: 1000
Final Deviance:    13154.79108
Akaike Information Criterion (AIC): 13202.79108
Total number of estimated parameters: 24
The number of iterations: 23
Termination criteria:  Max iterations=1000, Parameter Change= 0.00010
                       Deviance Change= 0.00010
Iterations terminated because the deviance convergence criteria was reached
Random number generation seed:    1.00000
Number of nodes used when drawing PVs: 2000
Number of nodes used when computing fit: 200
Number of plausible values to draw: 5
Maximum number of iterations without a deviance improvement: 100
Maximum number of Newton steps in M-step: 10
Value for obtaining finite MLEs for zero/perfects:    0.30000

key 1 scored as 1: acddbcebbacc
================================================================================
```

Figure 2.81: Summary of estimation Table

==
ConQuest: Generalised Item Response Modelling Software Fri Jul 24 14:34 2015
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
===Build: Jul 22 2015===

TERM 1: item

VARIABLES				UNWEIGHTED FIT				WEIGHTED FIT				2PL SCALED
item		ESTIMATE	ERROR^	MNSQ	Confidence Interval		T	MNSQ	Confidence Interval		T	ESTIMATE
1	BSMMA01	-0,985	0,125	1,00	0,91	1,09	0,1	1,00	0,91	1,09	0,0	-0,491
2	BSMMA02	-1,276	0,096	1,04	0,91	1,09	0,8	1,00	0,92	1,08	0,0	-1,260
3	BSMMA03	-1,284	0,113	1,02	0,91	1,09	0,5	1,01	0,92	1,08	0,2	-0,882
4	BSMMA04	-0,252	0,082	0,99	0,91	1,09	-0,1	1,00	0,94	1,06	-0,1	-0,214
5	BSMMA05	0,097	0,067	1,00	0,91	1,09	0,0	1,00	0,97	1,03	0,1	0,203
6	BSMMA06	-1,412	0,100	1,01	0,91	1,09	0,1	1,00	0,92	1,08	0,1	-1,406
7	BSMSA07	-1,369	0,089	1,01	0,91	1,09	0,2	1,00	0,92	1,08	0,0	-1,998
8	BSMSA08	-1,237	0,080	1,00	0,91	1,09	-0,1	1,00	0,92	1,08	0,0	-2,956
9	BSMSA09	-2,076	0,128	0,97	0,91	1,09	-0,6	1,00	0,88	1,12	0,0	-2,068
10	BSMSA10	-1,338	0,086	0,99	0,91	1,09	-0,2	1,00	0,92	1,08	-0,1	-2,283
11	BSMSA11	-1,694	0,118	0,97	0,91	1,09	-0,6	1,01	0,91	1,09	0,2	-1,417
12	BSMSA12	-0,357	0,075	1,00	0,91	1,09	-0,1	1,00	0,95	1,05	0,0	-0,409

A parameter estimate in italics indicates that it is constrained
Separation Reliability = 0.975
Chi-square test of parameter equality = 1788.60, df = 12, Sig Level = 0.000
^ Empirical standard errors have been used
==

Figure 2.82: Item Parameter Estimates

For the purpose of this Tutorial, the tab of interest in the `ex10_shw.xlsx` Excel file is the `scores` tab. Here, the item discrimination parameters are presented (Figure 2.83). The *score* column displays the different score assigned to the correct response in each item (discrimination parameter). The error associated to the estimate is also presented.

The item analysis is shown on the `ex10_itn.xlsx` output file. The `itanal` output includes a table showing classical difficulty, discrimination, and point-biserial statistics for each item. Figure 2.84 shows the results for items 2 and 3. The 2PL discrimination estimate for each is shown in the *score* column. Summary results, including coefficient alpha for the test as a whole, are printed at the end of the spreadsheet.

Figure 2.85 shows plots that were produced by the `plot icc` and the `plot mcc` command for items 1 and item 5. In the left panel, the ICC plot shows a comparison of the empirical item characteristic curve (the broken line, which is based directly upon the observed data) with the modelled item characteristic curve (the smooth line).

The right panel shows a matching plot produced by the `plot mcc command`. In addition to showing the modelled curve and the matching empirical curve, this plot shows the characteristics of the incorrect responses — the distractors. In particular it shows the

===
ConQuest: Generalised Item Response Modelling Software Fri Jul 24 14:34 2015
TABLE(S) OF GIN SCORES
===Build: Jul 22 2015===

GIN	Score	Error	GIN Labels		
1.1	2,003814	0,22667	item	1	BSMMA01
2.1	1,012438	0,122578	item	2	BSMMA02
3.1	1,455735	0,158226	item	3	BSMMA03
4.1	1,173713	0,127218	item	4	BSMMA04
5.1	0,4782	0,087131	item	5	BSMMA05
6.1	1,004396	0,124657	item	6	BSMMA06
7.1	0,685228	0,108473	item	7	BSMSA07
8.1	0,418354	0,097187	item	8	BSMSA08
9.1	1,003912	0,141661	item	9	BSMSA09
10.1	0,586227	0,104229	item	10	BSMSA10
11.1	1,195995	0,142907	item	11	BSMSA11
12.1	0,873232	0,106369	item	12	BSMSA12

Average Score 0.99094

Figure 2.83: Score estimates for each item

proportion of students in each of a sequence of ten ability groupings[32] that responded with each of the possible responses.

The second `plot icc` command of the `ex10.cqc` file produces the plot shown in Figure 2.86. Here all ICCs are plotted in the same window, which allows the graphical comparison of the different discrimination capabilities of each item.

2.11.5 Summary

This tutorial shows how ACER ConQuest can be used to analyse a multiple-choice test with the 2PL model. Some key points covered in this tutorial are:

- the need to `set lconstraints` to `cases` when estimation of discrimination parameters is required.
- the `model` statement allows the estimation of different slopes (discrimination) for each item through the `scoresfree` option.
- the `itanal` statement provides information about the discrimination estimate for each item.

[32]Ten ability groupings is a default setting that can be altered.

Item 2

item:2 (BSMMA02)

Cases for this item 1000 Item-Rest Cor. 0.31 Item-Total Cor. 0.47

Item Threshold(s): NOT AVAILABLE Weighted MNSQ 1.00

Item Delta: -1.28 2-PL scaled delta: -1.26 Slope: 1.01

Label	Score	Count	% of tot	Pt Bis	t	sig	PV1Avg:1	PV1 SD:1
M	0	5	0,5	-0,08	-2,51	0,012	-0,928	1,339
a	0	59	5,9	-0,17	-5,3	0,000	-0,762	0,984
b	0	152	15,2	-0,19	-6,22	0,000	-0,626	0,972
c	1,01	743	74,3	0,31	10,13	0,000	0,185	0,882
d	0	41	4,1	-0,1	-3,12	0,002	-0,574	1,041

Scores assigned to each of the codes

Item slope value

Item 3

item:3 (BSMMA03)

Cases for this item 1000 Item-Rest Cor. 0.41 Item-Total Cor. 0.56

Item Threshold(s): NOT AVAILABLE Weighted MNSQ 1.01

Item Delta: -1.28 2-PL scaled delta: -0.88 Slope: 1.46

Label	Score	Count	% of tot	Pt Bis	t	sig	PV1Avg:1	PV1 SD:1
M	0	9	0,9	-0,11	-3,47	0,001	-0,919	0,815
a	0	117	11,7	-0,26	-8,61	0,000	-0,879	0,863
b	0	119	11,9	-0,21	-6,66	0,000	-0,736	0,875
c	0	38	3,8	-0,11	-3,54	0,000	-0,848	0,854
d	1,46	717	71,7	0,41	14,03	0,000	0,279	0,843

Figure 2.84: Item Analysis Results

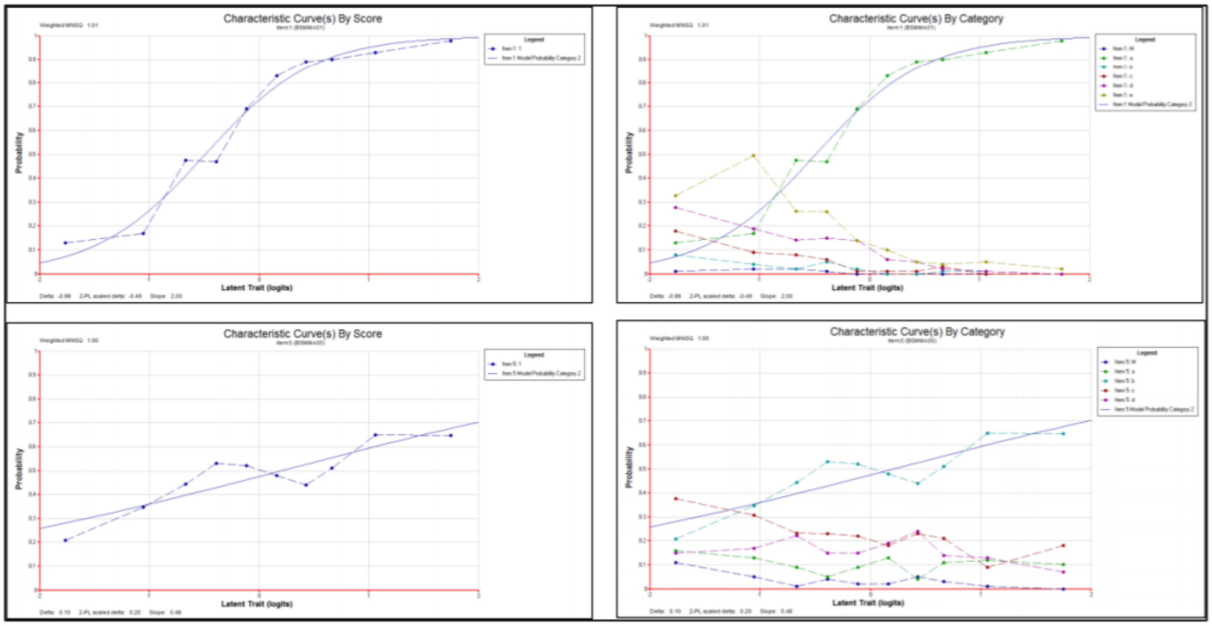


Figure 2.85: Plots for item 1 and item 5

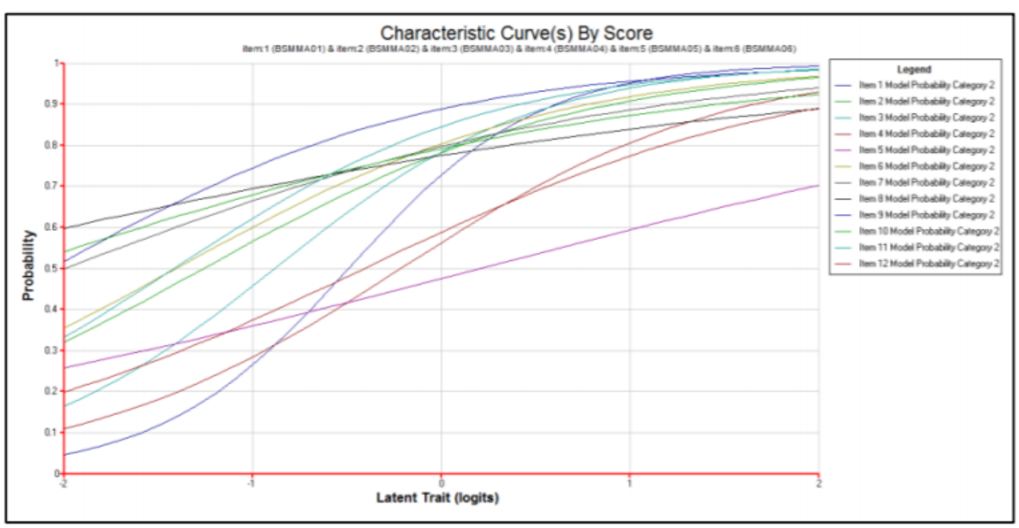


Figure 2.86: Item Characteristic Curve plot for all items in the data set

- the `plot` statement allows the graphical comparison of the discrimination power of each item.

2.12 Modelling Polytomous Items with the Generalised Partial Credit and Bock Nominal Response Models

As discussed in *Note 6: Score Estimation and Generalised Partial Credit Models*, ACER ConQuest can estimate scoring parameters for a wide range of models with polytomous data where item responses are categorical values, including multidimensional forms of the two-parameter family of models such as the multidimensional generalised partial credit models (Muraki, 1992). In addition, ACER ConQuest can also estimate scoring parameters for models with polytomous data where item responses are in the form of nominal categories, such as Bock's nominal response model (Bock, 1972). In this tutorial, the use of ACER ConQuest to fit the generalised partial credit and Bock nominal response models is illustrated through two sets of sample analyses. Both analyses use the same cognitive items: in the first the generalised partial credit model is fitted to the data; and in the second, the Bock nominal response model is fitted.

The data for this tutorial are the responses of 515 students to a test of science concepts related to the Earth and space previously used in the Tutorial *Modelling Polytomously Scored Items with the Rating Scale and Partial Credit Models*.

The data have been entered into the file `ex2a_dat.txt`, using one line per student. A unique identification code has been entered in columns 2 through 7, and the students' response to each of the items has been recorded in columns 10 through 17. In this data, the upper-case alphabetic characters `A`, `B`, `C`, `D`, `E`, `F`, `W`, and `X` have been used to indicate the different kinds of responses that students gave to these items. The code `Z` has been used to indicate data that cannot be analysed. For each item, these codes are scored (or, more correctly, mapped onto performance levels) to indicate the level of quality of the response. For example, in the case of the first item (the item in column 10), the response coded `A` is regarded as the best kind of response and is assigned to level 2, responses `B` and `C` are assigned to level 1, and responses `W` and `X` are assigned to level 0. An extract of the file `ex2a_dat.txt` is shown in Figure 2.87.

```
         1         2
12345678901234567890123 (column numbers)
```

```
2110104ZHWBDCBBCBEABBBB
2110106ZEACDBXBCXXXXXXX
2110109ZHBWBBBWCAXAXXXX
2110113ZIWBWBXWCXXXABBB
2110115ZHWBFBBWCWXAWAXX
2110121ZHWWEBWBBCAABABA
2110123YIBWWBEWBWXABABB
2110305ZHCBABABAABACCCA
2110313YBBCFBDBCXXXXXXX
 .               .
 .               .
```

Figure 2.87: Extract from the Data File `ex2a\_dat.txt`

2.12.1 a) Fitting the Generalised Partial Credit Model

2.12.1.1 Required files

The files used in this sample analysis are:

filename	content
ex11a.cqc	The command statements.
ex2a_dat.txt	The data.
ex2a_lab.txt	The variable labels for the items on the partial credit test.
ex11a_shw.txt	The results of the generalised partial credit analysis.
ex11a_itn.txt	The results of the traditional item analyses.

(The last two files are created when the command file is executed.)

2.12.1.2 Syntax

`ex11a.cqc` is the command file used to fit the Generalised Partial Credit Model in this tutorial. It is shown in the code box below, and each line of the command file is explained in the list underneath the code.

`ex11a.cqc`:

```
1  Title Generalised Partial Credit Model: What happened last night;
2  data ex2a_dat.txt;
3  format name 2-7 responses 10-17;
4  labels << ex2a_lab.txt;
5  codes 3,2,1,0;
6  set lconstraints=cases;
7  recode (A,B,C,W,X) (2,1,1,0,0)            !items(1);
8  recode (A,B,C,W,X) (3,2,1,0,0)            !items(2);
9  recode (A,B,C,D,E,F,W,X) (3,2,2,1,1,0,0,0)!items(3);
10 recode (A,B,C,W,X) (2,1,0,0,0)            !items(4);
11 recode (A,B,C,D,E,W,X) (3,2,1,1,1,0,0)    !items(5);
12 recode (A,B,W,X) (2,1,0,0)                !items(6);
13 recode (A,B,C,W,X) (3,2,1,0,0)            !items(7);
14 recode (A,B,C,D,W,X) (3,2,1,1,0,0)        !items(8);
15 model item + item*step!scoresfree;
16 estimate;
17 show !estimates=latent >> Results/ex11a_shw.txt;
18 itanal >> Results/ex11a_itn.txt;
19 plot expected >> Results/ex11a_expected_;
20 plot mcc >> Results/ex11a_mcc_;
```

- **Line 1**
  Gives a title for this analysis. The text supplied after the command `title` will appear on the top of any printed ACER ConQuest output. If a title is not provided, the default, *ConQuest: Generalised Item Response Modelling Software*, will be used.

- **Line 2**
  Indicates the name and location of the data file. Any name that is valid for the operating system you are using can be used here.

- **Line 3**
  The `format` statement describes the layout of the data in the file `ex2a_dat.txt`. This format indicates that a field called `name` is located in columns 2 through 7 and that the responses to the items are in columns 10 through 17 (the response block) of the data file.

- **Line 4**
  A set of labels for the items are to be read from the file `ex2a_lab.txt`. If you take a look at these labels, you will notice that they are quite long. ACER ConQuest

labels can be of any length, but most ACER ConQuest printouts are limited to displaying many fewer characters than this. For example, the tables of parameter estimates produced by the `show` statement will display only the first 11 characters of the labels.

- **Line 5**
  The `codes` statement is used to restrict the list of codes that ACER ConQuest will consider valid. This meant that any character in the response block defined by the format statement—except a blank or a period (`.`) character (the default missing-response codes) — was considered valid data. In this sample analysis, the valid codes have been limited to the digits `0`, `1`, `2` and `3`; any other codes for the items will be treated as missing response data. It is important to note that the `codes` statement refers to the codes *after* the application of any recodes.

- **Line 6**
  The `lconstraints=cases` argument of the `set` command is used to have the mean of each latent dimension set to zero, rather than the mean of the item parameters on each dimension set to zero (e.g., `lconstraints=items`). All item parameters are still estimated, but the mean of each of the latent dimensions is set to zero.

- **Lines 7-14**
  The eight `recode` statements are used to collapse the alphabetic response categories into a smaller set of categories that are labelled with the digits `0`, `1`, `2` and `3`. Each of these `recode` statements consists of three components. The first component is a list of codes contained within parentheses. These are codes that will be found in the data file `ex2a_dat.txt`, and these are called the *from* codes. The second component is also a list of codes contained within parentheses, these codes are called the *to* codes. The length of the *to* codes list must match the length of the *from* codes list. When ACER ConQuest finds a response that matches a *from* code, it will change (or recode) it to the corresponding *to* code. The third component (the option of the recode command) gives the levels of the variables for which the recode is to be applied. Line 11, for example, says that, for item `6`, `A` is to be recoded to `2`, `B` is to be recoded to `1`, and `W` and `X` are both to be recoded to `0`. Any codes in the response block of the data file that do not match a code in the *from* list will be left untouched. In these data, the `Z` codes are left untouched; and since `Z` is not listed as a valid code, all such data will be treated as missing-response data. When ACER ConQuest models these data, the number of response categories that will be assumed for each item will be determined from the number of distinct codes for that item. Item 1 has three distinct codes (`2`, `1` and `0`), so three categories will be modelled; item 2 has

four distinct codes (`3`, `2`, `1` and `0`), so four categories will be modelled.

- **Line 15**
  The `model` statement for these data contains two terms (`item` and `item*step`) and will result in the estimation of two sets of parameters. The term `item` results in the estimation of a set of item difficulty parameters, and the term `item*step` results in a set of item step-parameters that are allowed to vary across the items. The option `scoresfree` results in the estimation of an additional set of item scores that are allowed to vary across the items. This is the generalised partial credit model.

  In the section The Structure of ACER ConQuest Design Matrices, there is a description of how the terms in the `model` statement specify different versions of the item response model. In addition, *Note 6: Score Estimation and Generalised Partial Credit Models* describes how ACER ConQuest estimates the score parameters in models such as the generalised partial credit model.

- **Line 16**
  The `estimate` statement is used to initiate the estimation of the item response model.

- **Line 17**
  The `show` statement produces a display of the item response model parameter estimates and saves them to the file `ex11a_shw.txt`. The option `estimates=latent` requests that the displays include an illustration of the latent ability distribution.

- **Line 18**
  The `itanal` statement produces a display of the results of a traditional item analysis. As with the `show` statement, the results have been redirected to a file (in this case, `ex11a_itn.txt`).

- **Lines 19-20**
  The `plot` statements produce two displays for each item in the test. The first requested plot is a comparison of the observed and the modelled expected score curve, while the second is a comparison of the observed and modelled item characteristics curves by category.

2.12.1.3 Running the Generalised Partial Credit sample analysis

To run this sample analysis, start the GUI version. Open the file `ex11a.cqc` and choose `Run`→`Run All`.

ACER ConQuest will begin executing the statements that are in the file `ex11a.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting the generalised partial credit model to the data, and as it does so it will report on the progress of the estimation.

After the estimation is complete, the two statements that produce output (`show` and `itanal`) will be processed. The `show` statement will produce seven separate tables. All of these tables will be in the file `ex11a_shw.txt`. The contents of the first table were discussed in the Tutorial *A Dichotomously Scored Multiple Choice Test*, and the contents of the second one in the Tutorial *Modelling Polytomously Scored Items with the Rating Scale and Partial Credit Models*. The third table (not shown here) gives the estimates of the population parameters. In this case, the mean of the latent ability distribution was constrained to 0.000, and the variance of that distribution constrained to 1.000.

The fourth table reports the reliability coefficients. Three different reliability statistics are available (Adams, 2005). In this case just the third index (the EAP/PV reliability) is reported because neither of the maximum likelihood estimates has been computed at this stage. The reported reliability is 0.746.

The fifth table was also discussed in the Tutorial *Modelling Polytomously Scored Items with the Rating Scale and Partial Credit Models*, and is a map of the parameter estimates and latent ability distribution. However, with the exception of predicted probability maps, item maps are not applicable for models with estimated scores. The sixth table, which contains information related to the item score estimates produced by the `scoresfree` argument in the `model` statement, is shown in Figure 2.88. The score parameter estimates are reported for each category of each generalised item, although for the generalised partial credit model ACER ConQuest only estimates a single parameter for each item, shown in the final (seventh) table of the show file, discussed later.

For the first item, two score estimates have been reported, corresponding to the codes (1, 2) that this item can take in the data (code 0 will always be scored as zero). For the second item, three score estimates have been reported, corresponding to the codes (1, 2, 3) that this item can take in the data.

Figure 2.89 shows the seventh table, which displays the Tau parameter estimates for each item and associated standard errors. This estimate is applied to each category of each generalised item to estimate the score parameter estimates that were produced in the previous table. If you compare the sixth and seventh tables, you will notice that the first score estimate for each item in the sixth table is the same as the Tau estimate for that item in the seventh table. The second score estimate (corresponding to category 2) is then double the Tau value, the third score estimate (corresponding to category 3) is triple the

```
================================================================================
Generalised Partial Credit Model: What happened last night Tue Jul 21 10:18 2015
TABLE(S) OF GIN SCORES
===========================================================Build: Jun 24 2015===
GIN Number   Score           Error              GIN Labels
--------------------------------------------------------------------------------
1.1          0.771           0.107             item 1  Earth shape
1.2          1.541           0.214             item 1  Earth shape
2.1          0.427           0.077             item 2  Earth picture
2.2          0.853           0.155             item 2  Earth picture
2.3          1.280           0.232             item 2  Earth picture
3.1          1.088           0.148             item 3  Falling off
3.2          2.175           0.296             item 3  Falling off
3.3          3.263           0.444             item 3  Falling off
4.1          0.922           0.127             item 4  What is Sun
4.2          1.843           0.254             item 4  What is Sun
5.1          0.726           0.097             item 5  Moonshine
5.2          1.452           0.195             item 5  Moonshine
5.3          2.178           0.292             item 5  Moonshine
6.1          1.405           0.183             item 6  Moon and night
6.2          2.810           0.367             item 6  Moon and night
7.1          0.654           0.088             item 7  Night and day
7.2          1.307           0.176             item 7  Night and day
7.3          1.961           0.264             item 7  Night and day
8.1          0.483           0.066             item 8  Breathe on moon
8.2          0.966           0.133             item 8  Breathe on moon
8.3          1.450           0.199             item 8  Breathe on moon
--------------------------------------------------------------------------------
   Average Score      1.40736
================================================================================
```

Figure 2.88: Item Score Parameters Estimated by the Generalised Partial Credit Model

Tau value, and so on. Regardless of how many categories each item has, only a single Tau parameter is estimated by the model. This Tau parameter is an estimate of each item's *discrimination*.

```
================================================================================
Generalised Partial Credit Model: What happened last night Tue Jul 21 10:18 2015
TABLE OF TAU VALUES
===========================================================Build: Jun 24 2015===
 Tau 1       0.771           0.107                item Earth shape
 Tau 2       0.427           0.077                item Earth picture
 Tau 3       1.088           0.148                item Falling off
 Tau 4       0.922           0.127                item What is Sun
 Tau 5       0.726           0.097                item Moonshine
 Tau 6       1.405           0.183                item Moon and night
 Tau 7       0.654           0.088                item Night and day
 Tau 8       0.483           0.066                item Breathe on moon
--------------------------------------------------------------------------------
      Average Tau      0.80930
================================================================================
```

Figure 2.89: Tau Parameters Estimated by the Generalised Partial Credit Model

The `itanal` command in line 18 produces a file (`ex11a_itn.txt`) that contains traditional item statistics (Figure 2.90). In this example a `key` statement was not used and the items use partial credit scoring. As a consequence the `itanal` results are provided at the level of scores, rather than response categories. As you can see in the output, the scores reported are those estimated by the model, not the codes that the response categories are assigned in the data. For the generalised partial credit model, the difference between the scores assigned to consecutive response categories is the same for all categories that item has, and corresponds to the Tau value estimated for that item in the show file. In this case, you can see in Figure 2.89 that the Tau value for item 2 is 0.427, which is equal to the difference between the scores assigned to consecutive categories shown in Figure 2.90.

The `plot` commands in line 19 and 20 produce the graphs shown in Figure 2.91. For illustrative purposes only plots for item 1 and 2 are shown. The second item showed poor fit to the scaling model — in this case the generalised partial credit model.

The second item's Tau value of 0.427 indicates that this item is less discriminating than the first item (Tau=0.771). The comparison of the observed and modelled expected score curves (the plots appearing on the left of the figure) is the best illustration of this lower discrimination. Notice how for the second item's plot the observed curve is a little flatter than the modelled curve. This will often be the case when the item discrimination is low.

The plots appearing on the right of the figure show the item characteristic curves, both modelled and empirical. There is one pair of curves for each possible score on the item.

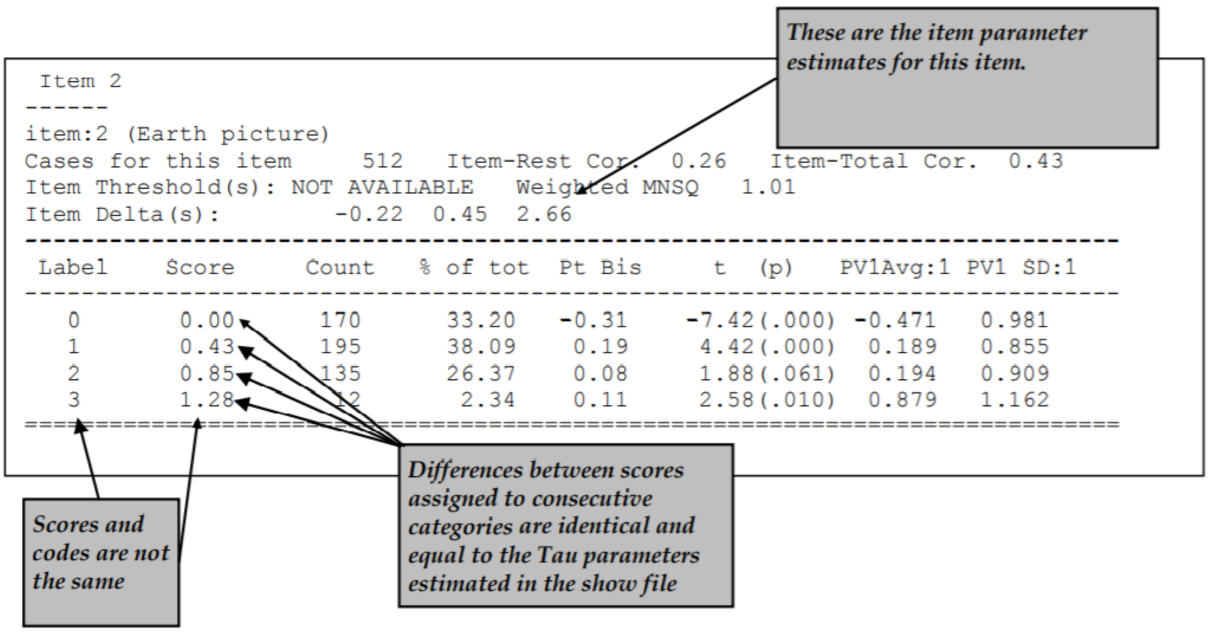


Figure 2.90: Extract of Item Analysis Printout for a Polytomously Scored Item Estimated with the Generalised Partial Credit Model

Note that for item 2 the disparity between the observed and modelled curves for category 2 is the largest. The second part of this tutorial will demonstrate how ACER ConQuest can estimate scores for each category of each item in the model, to determine how well each category score fits the scaling model.

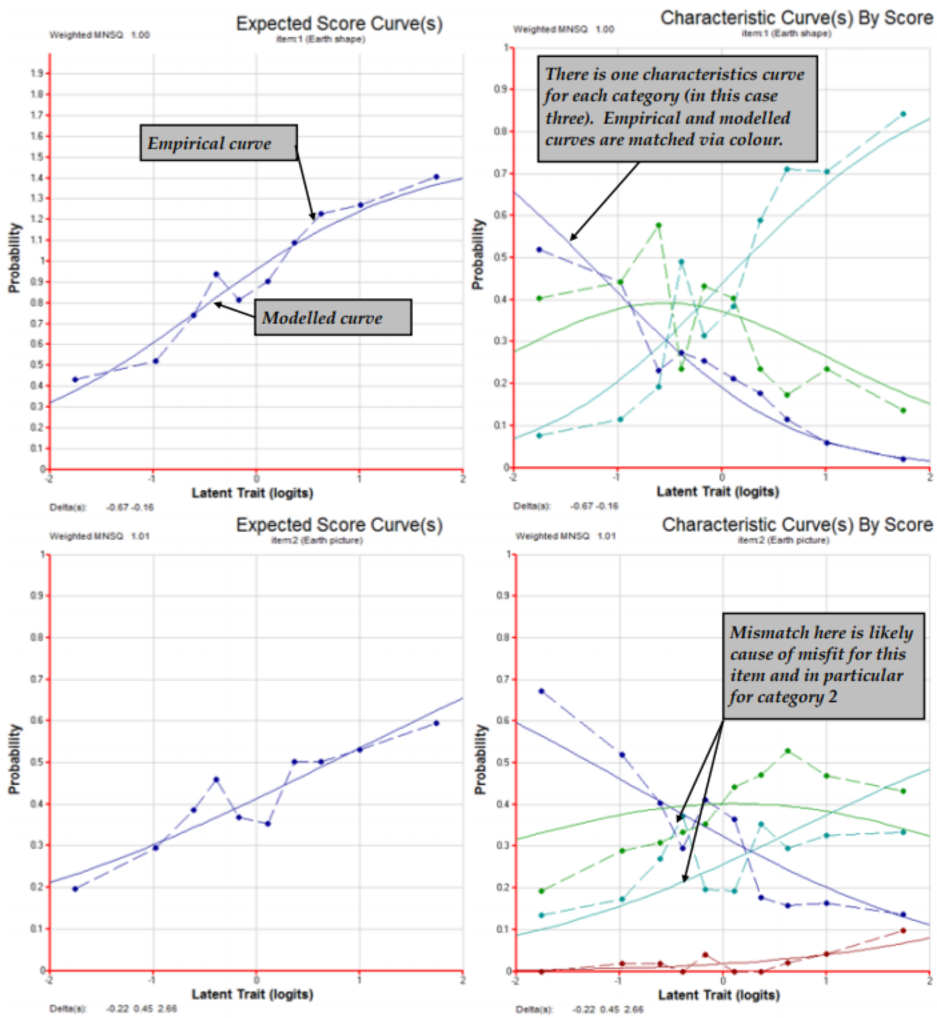


Figure 2.91: Plots for Items 1 and 2

2.12.2 b) Bock's Nominal Response Model

In the second sample analysis of this tutorial, the Bock nominal response model is fitted to the same data used in the previous analysis, to illustrate the differences between the two models.

2.12.2.1 Required files

The files that we use are:

filename	content
ex11b.cqc	The command statements.
ex2a_dat.txt	The data.
ex2a_lab.txt	The variable labels for the items on the test.
ex11b_shw.txt	The results of the nominal response analysis.
ex11b_itn.txt	The results of the traditional item analyses.

(The last two files are created when the command file is executed.)

2.12.2.2 Syntax

The command file for fitting the Bock nominal response model to the data is `ex11b.cqc`; it is shown in the code box below. In the list following the code box each line of commands is explained in detail.

`ex11b.cqc`:

```
Title Bock Nominal Response Analysis: What happened last night;
datafile ex2a_dat.txt;
format name 2-7 responses 10-17;
labels << ex2a_lab.txt;
codes 3,2,1,0;
set lconstraints=cases;
recode (A,B,C,W,X) (2,1,1,0,0)           !items(1);
recode (A,B,C,W,X) (3,2,1,0,0)           !items(2);
recode (A,B,C,D,E,F,W,X) (3,2,2,1,1,0,0,0)!items(3);
recode (A,B,C,W,X) (2,1,0,0,0)           !items(4);
recode (A,B,C,D,E,W,X) (3,2,1,1,1,0,0)    !items(5);
```

```
12 recode (A,B,W,X) (2,1,0,0)                   !items(6);
13 recode (A,B,C,W,X) (3,2,1,0,0)               !items(7);
14 recode (A,B,C,D,W,X) (3,2,1,1,0,0)           !items(8);
15 model item + item*step!bock;
16 estimate;
17 show !estimates=latent >> Results/ex11b_shw.txt;
18 itanal >> Results/ex11b_itn.txt;
19 plot expected >> Results/ex11b_expected_;
20 plot mcc >> Results/ex11b_mcc_;
```

- **Line 1** For this analysis, we are using the title `Bock Nominal Response Analysis: What happened last night`.
- **Lines 2-14** The commands in these lines are exactly the same as for the generalised partial credit model analysis (see above).
- **Line 15** The `model` statement for these data is exactly the same as for the generalised partial credit model analysis. The option `bock` results in the estimation of an additional set of item category scores that are allowed to vary across each of the categories of each of the items. This is the Bock nominal response model.
- **Lines 16-20** The commands in these lines are exactly the same as for the generalised partial credit model analysis (see above), however the names of the `show` and traditional item (`itanal`) analysis files have been changed to `ex11b_shw.txt` and `ex11b_itn.txt`, respectively.

2.12.2.3 Running the Bock Nominal Response Sample Analysis

To run this sample analysis, start the GUI version. Open the file `ex11b.cqc` and choose `Run`→`Run All`.

ACER ConQuest will begin executing the statements that are in the file `ex11b.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting the Bock nominal response model to the data, and as it does so it will report on the progress of the estimation.

After the estimation is complete, the two statements that produce output (`show` and `itanal`) will be processed. The `show` statement will again produce seven separate tables. All of these tables will be in the file `ex11b_shw.txt`, and are the same as those described in the generalised partial credit model (see above).

The important difference between this model and the generalised partial credit model is illustrated in the sixth and seventh tables in the show file. The sixth table, contains information related to the item score estimates produced by the bock option in the `model` statement, is shown in Figure 2.92. The score parameter estimates are reported for each category of each item, and in this case ACER ConQuest estimates a single parameter for each category of each item (rather than a single parameter for each item, as was the case for the generalised partial credit model).

As with the generalised partial credit model, two score estimates have been reported for the first item, corresponding to the codes (1, 2) that this item can take in the data (code 0 will always be scored as zero). For the second item, three score estimates have been reported, corresponding to the codes (1, 2, 3) that this item can take in the data.

```
================================================================================
Partial Credit Model: What happened last night           Tue Jul 21 09:25 2015
TABLE(S) OF GIN SCORES
=================================================================Build: Jun 24 2015===
GIN Number   Score          Error              GIN Labels
--------------------------------------------------------------------------------
1.1          0.527          0.172             item 1  Earth shape
1.2          1.564          0.216             item 1  Earth shape
2.1          0.939          0.159             item 2  Earth picture
2.2          0.753          0.163             item 2  Earth picture
2.3          1.831          0.435             item 2  Earth picture
3.1          1.612          0.233             item 3  Falling off
3.2          1.740          0.330             item 3  Falling off
3.3          3.408          0.571             item 3  Falling off
4.1          0.764          0.151             item 4  What is Sun
4.2          2.119          0.300             item 4  What is Sun
5.1          0.372          0.201             item 5  Moonshine
5.2          0.994          0.224             item 5  Moonshine
5.3          2.511          0.357             item 5  Moonshine
6.1          1.220          0.205             item 6  Moon and night
6.2          2.932          0.399             item 6  Moon and night
7.1          0.660          0.203             item 7  Night and day
7.2          1.337          0.253             item 7  Night and day
7.3          1.924          0.254             item 7  Night and day
8.1          0.798          0.256             item 8  Breathe on moon
8.2          1.406          0.510             item 8  Breathe on moon
8.3          1.466          0.197             item 8  Breathe on moon
--------------------------------------------------------------------------------
   Average Score      1.47022
================================================================================
```

Figure 2.92: Item Score Parameters Estimated by Bock's Nominal Response Model

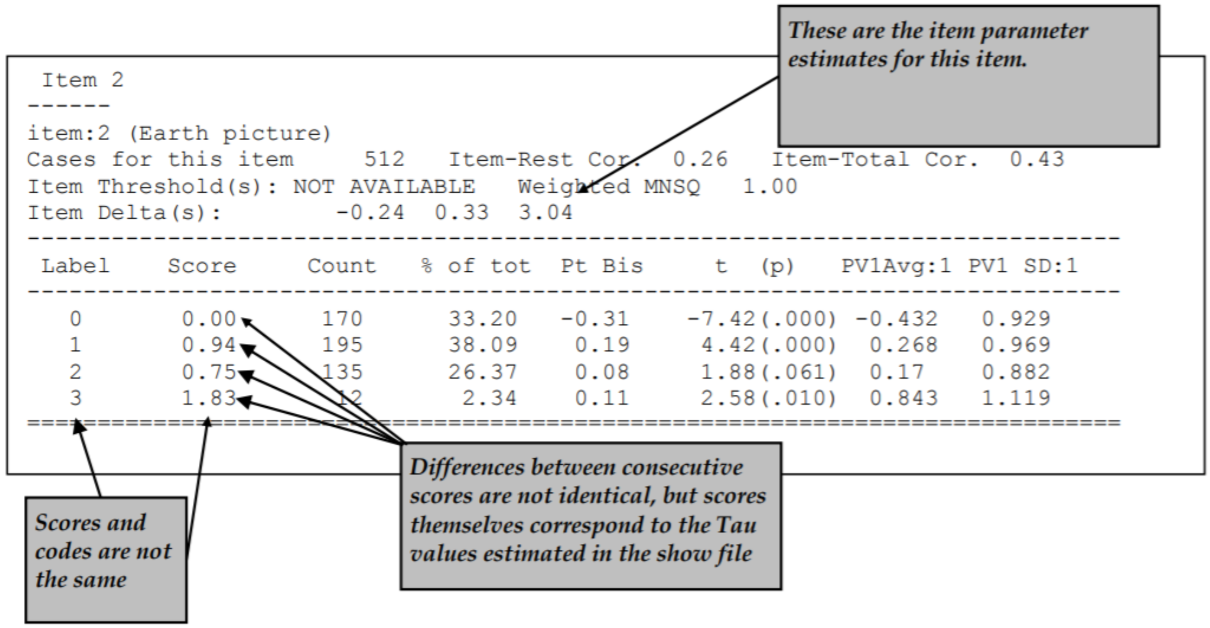
Figure 2.93 shows the seventh table, which displays the Tau parameter estimates for each item and associated standard errors, as it did for the generalised partial credit model. However, you will notice that there are more values in this table than there was for the generalised partial credit model. This is because ACER ConQuest is estimating score parameters for each category of each item individually. Consequently, there is a one-to-one correspondence between the values in this table and those that were reported in the previous table. These Tau parameters provide an estimate of each item category's *discrimination*.

```
================================================================================
Partial Credit Model: What happened last night            Tue Jul 21 09:25 2015
TABLE OF TAU VALUES
===========================================================Build: Jun 24 2015===
 Tau 1       0.527           0.172                item Earth shape
 Tau 2       0.939           0.159                item Earth picture
 Tau 3       1.612           0.233                item Falling off
 Tau 4       0.764           0.151                item What is Sun
 Tau 5       0.372           0.201                item Moonshine
 Tau 6       1.220           0.205                item Moon and night
 Tau 7       0.660           0.203                item Night and day
 Tau 8       0.798           0.256                item Breathe on moon
 Tau 9       1.564           0.216                item Earth shape step 1
 Tau 10      0.753           0.163                item Earth picture step 1
 Tau 11      1.831           0.435                item Earth picture step 2
 Tau 12      1.740           0.330                item Falling off step 1
 Tau 13      3.408           0.571                item Falling off step 2
 Tau 14      2.119           0.300                item What is Sun step 1
 Tau 15      0.994           0.224                item Moonshine step 1
 Tau 16      2.511           0.357                item Moonshine step 2
 Tau 17      2.932           0.399                item Moon and night step 1
 Tau 18      1.337           0.253                item Night and day step 1
 Tau 19      1.924           0.254                item Night and day step 2
 Tau 20      1.406           0.510                item Breathe on moon step 1
 Tau 21      1.466           0.197                item Breathe on moon step 2
--------------------------------------------------------------------------------
       Average Tau      1.47022
================================================================================
```

Figure 2.93: Tau Parameters Estimated by Bock's Nominal Response Model

The `itanal` command in line 18 produces a file (`ex11b_itn.txt`) that contains traditional item statistics (Figure 2.94). In this example, as with the generalised partial credit example, a `key` statement was not used and the items use partial credit scoring. As a consequence the `itanal` results are provided at the level of scores, rather than response categories. As you can see in the output, the scores reported are those estimated by the model, not the codes that the response categories are assigned in the data. These scores correspond to the Tau values estimated in the show file in Figure 2.93, as well as the score values in Figure 2.92, as the Tau and score parameters are identical in the Bock nominal response model.

As you can see in both the show file and the traditional item statistics, the category scores estimated by ACER ConQuest can differ quite substantially to the codes that were manually allocated to the data values. In an example with ordinal response data such



```
 Item 2
------
item:2 (Earth picture)
Cases for this item     512   Item-Rest Cor.  0.26   Item-Total Cor.  0.43
Item Threshold(s): NOT AVAILABLE   Weighted MNSQ   1.00
Item Delta(s):        -0.24  0.33  3.04
------------------------------------------------------------------------------
 Label    Score     Count   % of tot  Pt Bis     t  (p)    PV1Avg:1 PV1 SD:1
------------------------------------------------------------------------------
   0       0.00      170      33.20   -0.31    -7.42(.000) -0.432   0.929
   1       0.94      195      38.09    0.19     4.42(.000)  0.268   0.969
   2       0.75      135      26.37    0.08     1.88(.061)  0.17    0.882
   3       1.83       12       2.34    0.11     2.58(.010)  0.843   1.119
==============================================================================
```


Figure 2.94: Extract of Item Analysis Printout for a Polytomous Item Estimated with Bock's Nominal Response Model

as this, the order of the category scores estimated by ACER ConQuest should match the order of the codes that were in the data (so that a code of 2 gets a higher score than a code of 1). You can see in this example that this is not the case for item 2. The scores estimated by ACER ConQuest for codes 1, 2 and 3 are 0.939, 0.753, and 1.831 respectively. As the score estimated for code 2 is less than that estimated for code 1, this points to a problem in the coding of the original data.

The `plot` commands in lines 19 and 20 produce the graphs shown in Figure 2.95. For illustrative purposes only plots for item 1 and 2 are shown. These graphs show a similar picture to what was shown in the generalised partial credit example. The disparity between the observed and modelled item characteristic curves for category 2 of item 2 that was noted in the generalised partial credit example is still observed here, and supported by the discrepancy between the scores estimated for this item in the show file and traditional item statistics.

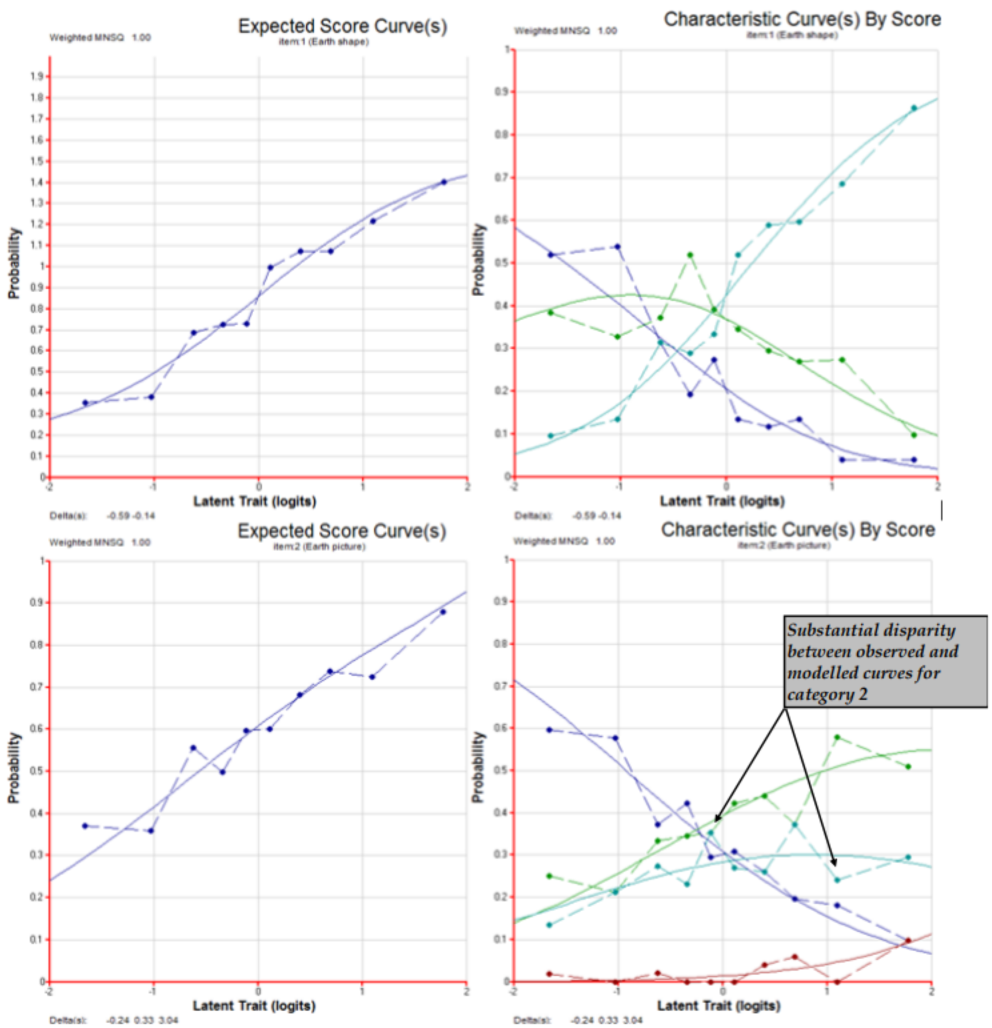


Figure 2.95: Plots for Items 1 and 2

2.12.3 Summary

In this tutorial, ACER ConQuest has been used to fit the generalised partial credit and Bock nominal response models. Some key points covered were:

- The `scoresfree` option in the `model` statement can be used to estimate a single parameter for each item in a given dataset which is used to determine scores that each item category receives (generalised partial credit model).
- The `bock` option in the `model` statement can be used to estimate a score for each category of each item in a given dataset (bock nominal response model).
- The score parameters estimated by ACER ConQuest can be used to determine item fit (generalised partial credit model) as well as item category fit (bock nominal response model).

2.13 The use of Matrix Variables in examining DIF

The purpose of this tutorial is to illustrate the use of matrix variables. Matrix variables are internal (matrix valued) objects that can be created by various ACER ConQuest procedures, or read into ACER ConQuest and then manipulated. For example the `estimate` command can create matrix variables that store the outcomes of the estimation[33]. Matrix variables can be manipulated, saved or plotted.

In this Tutorial we show how subsets of the data can be analysed to evaluate differential item functioning. In this case we analyse differences between male and female students. We show how the results can be stored as matrix variables and how those matrices can be manipulated and plotted.

2.13.1 Required files

The files used in this sample analysis are:

[33]For a list of commands that can produce matrix variables and the content of those variables see the section Matrix Objects Created by Analysis Commands.

filename	content
ex12.cqc	The command statements.
ex5_dat.txt	The data.
ex6_lab.txt	The variable labels for the items on the multiple choice test.

The `ex5_dat.txt` file contains achievement for 6800 students. Each line in the file represents one tested student. The first 19 columns of the data set contain identification and demographic information for each student. Columns 20 to 176 contain student responses to multiple-choice, and short and extended answer items. For the multiple-choice items, the codes `1`, `2`, `3`, `4` and `5` are used to indicate the response alternatives to the items. For the short answer and extended response items, the codes `0`, `1`, `2` and `3` are used to indicate the student's score on the item. If an item was not presented to a student, the code `.` (dot/period) is used; if the student failed to attempt an item and that item is part of a block of non-attempts at the end of a test, then the code `R` is used. For all other non-attempts, the code `M` is used. More information about the `ex5_dat.txt` file can be found in the Tutorial *Unidimensional Latent Regression*. An extract from the data file is shown in Figure 2.96[34].

```
         1         2         3
1234567890123456789012345678901234567 (column numbers)

27       2201   71100431315413433123233......142M13
30       2201   85111121133434514..................
......   ......  ......
42       2201   82010232243......143421532132......
......   ......  ......
44       2201   85111134141411531..................
45       2201   86010532333......M41221............
47       2201   88010534345423221..................
```

Figure 2.96: Extract from the Data File `ex5\_dat.txt`

In this example, only data from columns 16 to 25 are used. Column 16 contains the code for the booklet that each student responded; the range is 1 to 8. Column 17 contains the

[34]In Figure 2.96, each column of the data file is labelled so that it can be easily referred to in the text. The actual ACER ConQuest data file does not have any column labels.

code 0 for male students and 1 for female students. Column 18 contains the code 0 for lower grade (first year of secondary school) students and 1 for upper grade (second year of secondary school) students. Column 19 contains the product of columns 17 and 18, that is, it contains 1 for upper grade female students and 0 otherwise. Columns 20 to 25 contain the student responses to the first six items in the database. These six items are dichotomously scored.

In this sample analysis, the simple logistic model will be fitted to the data to analyse differences in item difficulty between boys and girls using graphic displays.

2.13.2 Syntax

Below we show the contents of the command file used in this analysis, `ex12.cqc`, followed by a description of each line of syntax.

`ex12.cqc`:

```
datafile ex5_dat.txt;
title TIMSS Mathematics--First Six Items;
set lconstraints=cases;
format book 16 gender 17 level 18 gbyl 19 responses 20-25;
labels << ex6_lab.txt;
key 134423 ! 1;
model item;
keepcases 0! gender;
estimate!matrixout=male;
reset;

datafile ex5_dat.txt;
title TIMSS Mathematics--First Six Items;
set lconstraints=cases;
format book 16 gender 17 level 18 gbyl 19 responses 20-25;
labels << ex6_lab.txt;
key 134423 ! 1;
model item;
keepcases 1! gender;
estimate!matrixout=female;
```

```
/* create data to plot an identity line */
compute itemparams=male_itemparams->female_itemparams;
let identityx=matrix(2:1);
let identityy=matrix(2:1);
compute identityx[1,1]=min(itemparams);
compute identityy[1,1]=min(itemparams);
compute identityx[2,1]=max(itemparams);
compute identityy[2,1]=max(itemparams);

/* plot the relationship */
scatter identityx,identityy!join=yes,seriesname=identity;
scatter male_itemparams,female_itemparams!overlay=yes,
                                          legend=yes,
                                          xmax=1,
                                          xmin=-2,
                                          ymax=1,
                                          ymin=-2,
                                          seriesname=male vs female,
                                          title=Comparison of Item Parameter Estimates
                                          subtitle=Male versus Female;


/* centre the item parameter estimates for both groups on zero
 and compute differences */
compute male_itemparams=male_itemparams-sum(male_itemparams)/rows(male_itemparams);
compute female_itemparams=female_itemparams-sum(female_itemparams)/rows(female_itempar
compute difference=male_itemparams-female_itemparams;

/* extract the standard errors from the error covariance matrix */
let var_male=matrix(6:1);
let var_female=matrix(6:1);
for (i in 1:6)
{
    compute var_male[i,1]=male_estimatecovariances[i,i];
    compute var_female[i,1]=female_estimatecovariances[i,i];
};

/* create data to plot upper and low 95% CI on Wald test */
```

```
61 let upx=matrix(2:1);
62 let upy=matrix(2:1);
63 let downx=matrix(2:1);
64 let downy=matrix(2:1);
65 compute upx[1,1]=1;
66 compute upy[1,1]=1.96;
67 compute upx[2,1]=rows(difference);
68 compute upy[2,1]=1.96;
69 compute downx[1,1]=1;
70 compute downy[1,1]=-1.96;
71 compute downx[2,1]=rows(difference);
72 compute downy[2,1]=-1.96;
73 compute item=counter(rows(difference));
74
75 /* calculate SE of difference and Wald test */
76 compute se_difference=sqrt(var_male+var_female);
77 compute wald=difference//se_difference;
78
79 /* plot standard differences */
80 scatter upx,upy!join=yes,seriesname=95 PCT CI Upper;
81 scatter downx,downy!join=yes,overlay=yes,seriesname=95 PCT CI Lower;
82 scatter item,wald!join=yes,
83                   overlay=yes,
84                   legend=yes,
85                   seriesname=Wald Values,
86                   title=Wald Tests by Item,
87                   subtitle=Male versus Female;
```

- **Line 1**
  The `datafile` statement indicates the name and location of the data file. Any file name that is valid for the operating system you are using can be used here.

- **Line 2**
  The `title` statement specifies the title that is to appear at the top of any printed ACER ConQuest output.

- **Line 3**
  The `set` statement specifies new values for a range of ACER ConQuest system

variables. In this case, the use of the `lconstraints` argument is setting the identification constraints to `cases`. Therefore, the constraints will be set through the population model by forcing the means of the latent variables to be set to zero and allowing all item parameters (difficulty and discrimination) to be free.

- **Line 4**
  The `format` statement describes the layout of the data in the file `ex5_dat.txt`. This `format` statement indicates the name of the fields and their location in the data file. For example, the field called `book` is located in column 16 and the field called `gender` is located in column 17. The `responses` to the six items used in this tutorial are in columns 20 through 25 of the data file.

- **Line 5**
  The `labels` statement indicates that a set of labels for the variables (in this case, the items) is to be read from the file `ex6_lab.txt`. An extract of `ex6_lab.txt` is shown in Figure 2.97. (This file must be text only; if you create or edit the file with a word processor, make sure that you save it using the text only option.)

  The first line of the file contains the special symbol `===>` (a string of three equals signs and a greater than sign) followed by one or more spaces and then the name of the variable to which the labels are to apply (in this case, `item`). The subsequent lines contain two pieces of information separated by one or more spaces. The first value on each line is the level of the variable (in this case, `item`) to which a label is to be attached, and the second value is the label. If a label includes spaces, then it must be enclosed in double quotation marks (`" "`). In this sample analysis, the label for item 1 is `BSMMA01`, the label for item 2 is `BSMMA02`, and so on.

- **Line 6**
  The `key` statement identifies the correct response for each of the multiple choice test items. In this case, the correct answer for item 1 is `1`, the correct answer for item 2 is `3`, the correct answer for item 3 is `4`, and so on. The length of the argument in the `key` statement is 6 characters, which is the length of the response block given in the `format` statement.

  If a `key` statement is provided, ACER ConQuest will recode the data so that any response `1` to item 1 will be recoded to the value given in the key statement option (in this case, `1`). All other responses to item 1 will be recoded to the value of the `key_default` (in this case, 0). Similarly, any response `3` to item 2 will be recoded to `1`, while all other responses to item 2 will be recoded to 0; and so on.

- **Line 7**

```
===> book
1      book1
2      book2
3      book3
.      .
8      book8
===> gender
0      male
1      female
===> level
0      "lower grade"
1      "upper grade"
===> item
1      BSMMA01
2      BSMMA02
3      BSMMA03
.      .
.      .
```

Figure 2.97: Contents of the Label File ex6_lab.txt

The `model` statement must be provided before any traditional or item response analyses can be undertaken. In this example, the argument for the `model` statement is the name of the variable that identifies the response data that are to be analysed (in this case, `item`). By omitting the option statement we are fitting a rasch model where scores for each item are fixed.

- **Line 8**
  The `keepcases` statement specifies a list of values for explicit variables that if not matched will be dropped from the analysis. The `keepcases` command can use two possible types of matching:

  1. EXACT matching occurs when a code in the data is compared to a keep code value using an exact string match. A code will be treated as a keep value if the code string matches the keep string exactly, including leading or trailing blank characters. Values placed in double quotes are matched with this approach.
  2. The alternative is TRIM matching, which first trims leading and trailing spaces from both the *keep* string and the *code* string and then compares the results. Values not in quotes are matched with this approach. To ensure TRIM matching of a blank or a period character, the words `blank` and `dot` are used. The list of codes should be followed by the name of the explicit variables where these codes are to be found. If there is more than one variable, they should be comma separated.

  In this case, we are keeping the code `0` for the variable `gender`, therefore modelling only males' responses. All cases with value 1 in this variable will be excluded from the analysis. By using the `keepcases` command we estimate separate item parameters for these two groups of students, producing separate matrix variables for males and females. We then use these matrix variables to evaluate DIF.

- **Line 9**
  The `estimate` statement initiates the estimation of the item response model. The `matrixout` option indicates that a set of matrices with prefix `male_` will be created to hold the results. This matrix will be stored in the temporary workspace. Any existing matrices with matching names will be overwritten without warning.

  The Matrices produced by `estimate` depend upon the options chosen. The list of matrices is found in Figure 2.98 and their content is described in the section Matrix Objects Created by Analysis Commands. You can see these matrices using the `print` command or using the workspace menu in the GUI mode.

Default	itemparams	history	
If `method=jml` or `abilities=yes` in conjunction with an MML method	mle	wle	
If `abilities=yes` in conjunction with an MML method	pvs		
If `ifit=yes`	Itemfit		
If `pfit=yes`	casefit		
If `stderr=empirical`	estimatecovariances		
If `stderr=quick`	Itemerrors	regressionerrors	covarianceerrors

Figure 2.98: Matrices created by the estimate command

- **Line 10**
  The `reset` command resets ACER ConQuest system values to their default values, except for tokens and variables. The command is used here to erase the effects of previously issued commands.

- **Lines 12-20**
  This set of commands is exactly the same to that mentioned above, with the exception of the last two (`estimate` and `keepcases`). In this part of the `ex12.cqc` file, we are modelling responses for females. Therefore, the `keepcases` statement instructs ACER ConQuest to keep in the analysis only those cases where the value of the variable gender equals `1`. A set of matrices named with the prefix `female_` will hold the results of the estimated model (`estimate` statement).

In **Lines 23-42** of `ex12.cqc`, data is extracted from the two matrices created above with the `estimate` statement. The data is used to create an identity line and then plotted to show differences in item difficulty for males and females.

- **Line 24**
  The `compute` command takes the `male_itemparams` and the `female_itemparams` object from the matrices created with the `estimate` statements. By using the `->` operator these two matrices are concatenated in a new matrix named `itemparams`. The new matrix contains six rows and two columns. The rows, one for each item, contain the estimated item location parameters (difficulty) and the columns correspond to student gender, male and female. For a list of compute command operators and functions see section 4.8.

- **Lines 25-26**
  The two `let` statements define two empty matrices, `identityx` and `identityy`, each with two rows and one column. These matrices allow us to draw the identity line in the scatter plot created below.

- **Lines 27-30**
  The `compute` statements fill the two newly created matrices with the minimum and maximum values observed in the matrix `itemparams`. Both matrices are filled with the same values.

- **Line 33**
  The `scatter` statement produces a scatter plot of two variables. In this case, `identityx` and `identityy`. The `join` option indicates that the two points are to be joined by a line; in this case, the identity line. The `seriesname` option defines the text to be used as a series name. The plot is displayed as a separate window in the screen and is shown in Figure 2.99.

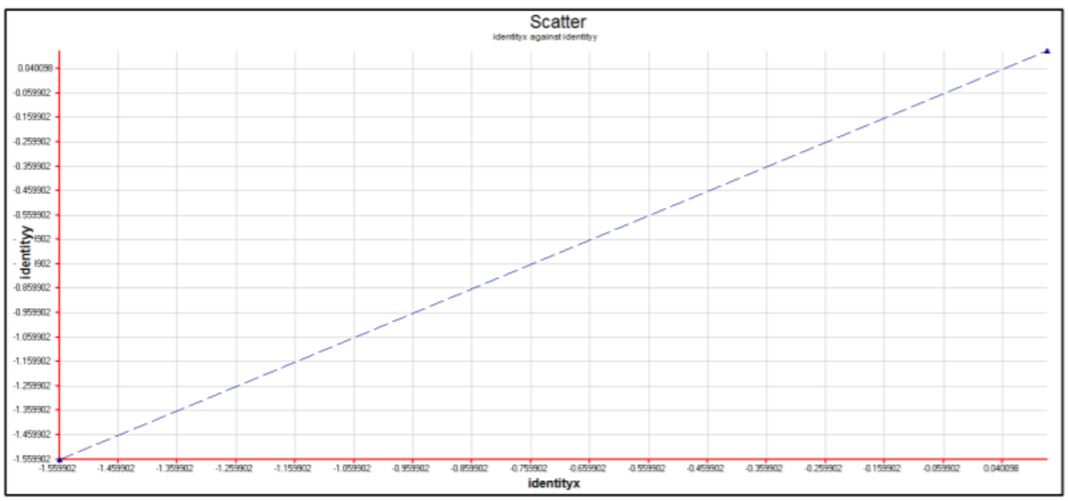


Figure 2.99: Scatter plot for the identity line

- **Lines 34-42**
  The second `scatter` statement produces a scatter plot of the item parameters for males and females (Figure 2.100). The `overlay` option allows the resulting plot to be overlayed on the existing active plot. In this case, results will be overlayed with the identity line shown in Figure 2.99. The option `legend` indicates that legend is displayed. The `xmax`, `xmin`, `ymax` and `ymin` options set the maximum and minimum

values for the horizontal and vertical axes of the plot, respectively and overwrite the values on the previous plot. The `seriesname` option specifies the text to be used as series name. The `title` and `subtitle` options specify the text to be used as title and subtitle of the plot.

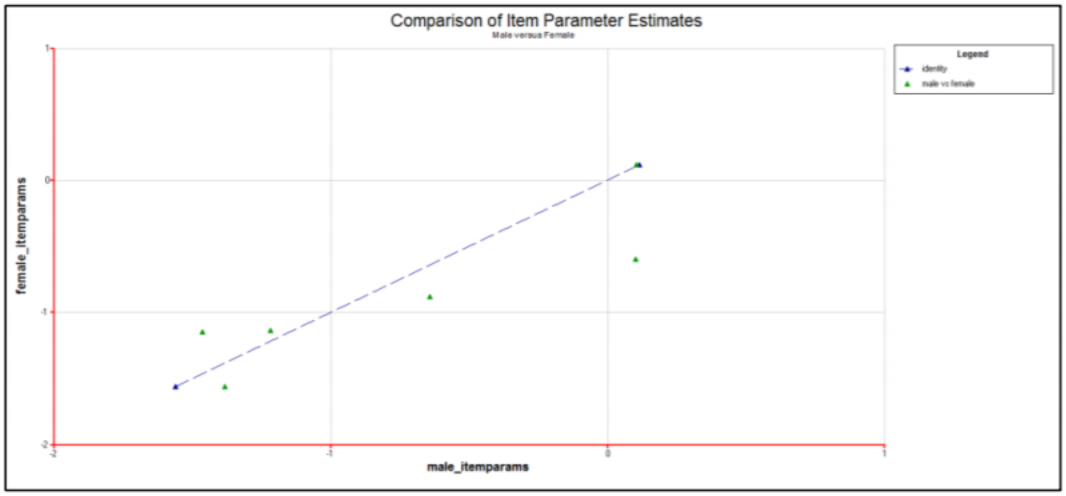


Figure 2.100: Scatter plot of item parameters for males and females

The set of statements in **Lines 45-87** of `ex12.cqc` centres the item parameters for both groups on zero and computes the difference between them for each item. With these results and the standard errors from the covariance matrix, a scatter plot is produced to display the Wald test of differences between the two groups (Engle, 1984). The plot also includes 95% confident levels for the Wald test.

- **Lines 47-49**
  The `compute` statement centres the item parameters (e.g.`male_itemparams`) by subtracting the mean of the item difficulties (e.g.`sum(male_itemparams)/rows(male_itemparams)`) to each item. A matrix with the centred values of item parameters is computed for each group. The difference of item difficulties between the two groups is also computed and stored in a new matrix named `difference`.

- **Lines 52-53**
  The `let` statements create two 6 by 1 empty matrices — one for each group.

- **Lines 54-58**
  The `for` statement fills the above created matrices with the values of the estimate

error variance for each item. These values are found in the diagonal of the estimates error variance-covariance matrix that is produced in the `estimate` statement (rows 9 and 18 in the command file `ex12.cqc`.).

- **Lines 61-64**
  The `let` statements create four 2 by 1 empty matrices, `upx`, `upy`, `downx`, and `downy` so we can plot the confidence interval lines in the plot.

- **Lines 65-72**
  The `compute` statements fill the matrices with the following values.

  - The element in the first row and column (i.e. `[1,1]`) of the matrices `upx` and `downx` with the number `1`.
  - The element in the second row and first column (i.e., `[2,1]`) of the matrices `upx` and `downx` with the number of rows of the difference matrix (i.e., 6).
  - The first and second rows of the matrices `upy` and `downy` with the number `1.96` and `-1.96`, respectively.

- **Line 73**
  The `compute` statement creates a variable named `item`. The function `counter` creates a matrix with the same number of rows as the difference matrix (i.e., 6) and 1 column, filled with integers running from 1 to 6. This serves for producing the horizontal axis in the scatter plot described in the last `scatter` statement in `ex12.cqc`.

- **Lines 76-77**
  The `compute` statements define two 6 by 1 matrices: `se_difference` and `wald`. The row values in the first of these matrices correspond to the square root (`sqrt`) of the sum of variances for each item between groups (`var_male+var_female`). By using the `//` operator, the values in the Wald matrix are computed as the division of each element in the `difference` matrix by the matching element in the `se_difference` matrix. The Wald test can be used to test for standard differences in item parameters between two groups, males and females in this case.

- **Line 80**
  The `scatter` statement produces a scatter plot of the `upx` and `upy` matrix variables. The plot is displayed on a new window. The values 1 and 6 in the horizontal axis and the value 1.96 in the vertical axis. The option `join` specifies a line that joins the points in the horizontal axis. The `seriesname` option defines the text to be used as series name.

- **Line 81**
  The `scatter` statement produces a scatter plot of the `downx` and `downy` matrix variables. The values 1 and 6 in the horizontal axis and the value -1.96 in the vertical axis. The option `join` specifies a line that joins the points in the horizontal axis. The `overlay` option indicates that the resulting plot is overlayed with the active plot produced by the previous `scatter` statement. The `seriesname` option defines the text to be used as series name.

- **Lines 82-87**
  The last `scatter` statement produces a scatter plot of the `item` and `wald` matrix variables (Figure 2.101). The `item` matrix, with values from 1 to 6 is displayed in the horizontal axis. And the `wald` matrix in the vertical axis. The plot is overlayed with the active plot produced by the two previous `scatter` statements by using the option `overlay`. The legend is set to be displayed by using the option `legend`. The name of the new series added to the plot is set with the `seriesname` option. The `title` and `subtitle` are also specified with the corresponding options.

  To avoid having a large number of decimal places in the values of the Wald test you have two options. One is to specify the upper and lower values of the vertical axis using the `ymax` and `ymin` options in the `scatter` statement. Another is to manipulate the graph via the `PlotQuest` window menus. The second approach is the one we used in Figure 2.101.

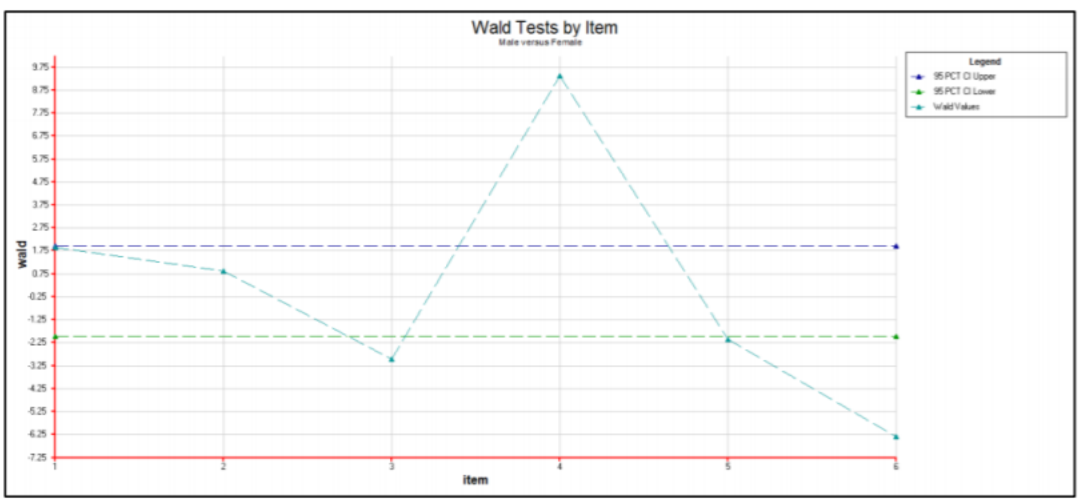


Figure 2.101: Wald test for standardised differences in item estimates between males and females

2.13.3 Running the Analysis

To run this sample analysis, start the GUI version. Open the file `ex12.cqc` and choose `Run`→`Run All`.

ACER ConQuest will begin executing the statements that are in the file `ex12.cqc`; and as they are executed they will be echoed in the Output Window. When it reaches the `estimate` command ACER ConQuest will begin fitting the two-parameter model to the data.

After the estimation is completed, the `scatter` statements will produce two plots that will be displayed in new windows. The first of these plots contains a comparison of the item parameter estimates for males and females, and also displays the identity line. The second plot contains the Wald test of standardized differences in item parameters for these two groups, along with the 95% confidence intervals.

As mentioned above, the first plot produced by the `ex12.cqc` file contains a comparison of the item estimates for males and females, along with the identity line. The plot is shown in Figure 2.100. According to the plot, there seems to be some variation in item difficulties for these two groups of students. An item where difference is more noticeable and thus of particular interest is item four (the one in the low right corner). Other items showing some degree of variability between the two groups are items three and six (the two on the left bottom corner).

The plot in Figure 2.101 allows us to determine whether the differences observed in the previous plot are statistically significant. In fact, items three, four and six are those where the Wald values fall considerable outside of the confidence interval, showing presence of DIF between the males and females. Wald values for items one and two are within the confidence interval, which indicates that although these items have different difficulty parameters for males and females, the difference is not statistically significant. Wald value of item five is just outside of the confidence interval; a close inspection of the item to investigate DIF is recommended.

2.13.4 Summary

This tutorial shows how ACER ConQuest matrix variables can be used to evaluate Differential Item Functioning (DIF) between two groups. Some key points covered in this tutorial are:

- the use of the `keepcases` command allows the estimation of item parameters separately for different groups.

- the use of the `matrixout` option in the `estimate` statement allows holding the results for each group in separate matrix variables.
- the use of operators and functions associated to the `compute` statement provide the opportunity to manipulate matrix variables created through the `estimate` command and compute new variables.
- the `scatter` statement allows the graphical comparison of the item parameters for different groups of students.

2.14 Modelling Pairwise Comparisons using the Bradley-Terry-Luce (BTL) Model

2.14.1 Background[35]

ACER ConQuest can be used to fit a logistic pairwise comparison model, also known as the Bradley-Terry-Luce (BTL) model (Bradley & Terry, 1952; Luce, 2005). Discussed in *Note 2: Pairwise Comparisons*, pairwise comparison is an approach to estimate a single parameter based on paired comparisons. The paired comparisons may be subjective (e.g., subjective rankings of two objects) or objective (e.g., winner in a paired game). The pairwise comparison approach is useful because there are situations where it is easier to make judgements between two objects than it is to rank all objects at once. It is easier to discriminate between two objects than to differentiate among a large set of objects and place them on an interval scale.

There are also situations where direct ranking may not be feasible (for example if there are a large number of objects to rank). In the example used in this tutorial, a sports tournament, estimating team strengths using the BTL model requires data on each team's performance against a set of opponents with each game treated as a pairwise comparison having a dichotomous outcome (win or lose).

In the original Bradley-Terry (1952) model, the probability of success (or higher rank) of an object in the pair is given as:

$$P_{ij} = \frac{\delta_i}{\delta_i + \delta_j} \tag{2.1}$$

[35] This is an updated document based on the original, authored by Alvin Vista and Ray Adams, 12 October 2015.

where P_{ij} denotes the probability that object i is ranked higher than object j (or that i wins over j), and δ is the scale location parameter for objects i and j. It can be shown that for any pair (i, j) if one wins the other loses, as shown in the derivation below (Glickman, 1999):

$$\begin{aligned} P_{ij} + P_{ji} &= \frac{\delta_i}{\delta_i + \delta_j} + \frac{\delta_j}{\delta_j + \delta_i} \\ &= \frac{\delta_i + \delta_j}{\delta_i + \delta_j} \\ &= 1 \end{aligned} \tag{2.2}$$

Reparametrising the model in terms of the fixed pair i, j where $x_{ij} = 1$ if i is ranked higher and $x_{ij} = 0$ if i is ranked lower, we have the BTL model as presented in *Note 2*:

$$P(X_{ij} = 1; \delta_i, \delta_j) = \frac{exp(x_{ij}\delta_i - (1 - x_{ij})\delta_j)}{1 + exp(\delta_i - \delta_j)} \tag{2.3}$$

2.14.2 Required files

The data for the sample analysis are the game results of 16 teams over 2,123 games. The data is formatted such that the outcome (1=win, 0=loss) refers to the team designated as object i, and entered as the first of the pair.

The files used in this sample analysis are:

filename	content
ex13.cqc	The command statements.
ex13_dat.txt	The data.
ex13_ObjectLocations.png	The Wright Map plot displaying the object locations graphically.
ex13_shw.txt	The results of the pairwise comparison, showing the parameter estimates
ex13_res.csv	The residuals (difference between observed and predicted (probability i v

(The last three files are created when the command file is executed.)

The data have been entered into the file `ex13_dat.txt`, using one line per game. The data is in fixed format, the teams designated as object i have been recorded in columns 1 through 13, while teams designated as object j have been recorded in columns 14 through 26. The value for the outcome is indicated in column 38. An extract of the file `ex13_dat.txt` is shown in Figure 2.102.

2.14.3 Syntax

The contents of the command file for this sample analysis (`ex13.cqc`) are shown in the code box below. Each of the command statements is explained in the list underneath the command file.

ex13.cqc:

```
1 title  Pairwise Analysis of Australian Football League;
2 data ex13_dat.txt;
3 format team1 1-13 team2 14-26 responses 38;
4 model team1-team2 ! type = pairwise;
5 estimate! stderr=quick;
6 plot wrightmap ! order = value, estimate = wle, rout = Results/wm >> Results/ex13_;
7 show >> Results/ex13_shw.txt;
8 show residuals !filetype=csv >> Results/ex13_res.csv;
9 show parameters! filetype=xlsx >> Results/ex13_prm.xlsx;
```

- **Line 1**
  gives a `title` for this analysis. The text supplied after the command `title` will appear on the top of any printed ACER ConQuest output. If a title is not provided, the default, `ConQuest: Generalised Item Response Modelling Software`, will be used.

```
         1         2         3
12345678901234567890123456789012345678 (column numbers)

St Kilda     West Coast           1
St Kilda     Sydney               1
St Kilda     West Coast           0
St Kilda     Sydney               0
 .               .
 .               .
```

Figure 2.102: Extract from the Data File ex2a.dat

- **Line 2**
  indicates the name and location of the data file. Any name that is valid for the operating system you are using can be used here.

- **Line 3**
  The `format` statement describes the layout of the data in the file `ex13_dat.txt`. This `format` indicates that a field called `team1` is located in columns 1 through 13 and that `team2` is located in columns 14 through 26; the outcomes of each pairwise comparison are in column 38 of the data file.

- **Line 4**
  The `model` statement for the pairwise analysis, showing which two objects are being compared (`team1` and `team2`).

- **Line 5**
  The `estimate` statement is used to initiate the estimation of the item response model. The `estimate` statement requires that quick standard errors (`stderr=quick`) are used for pairwise comparisons.

- **Line 6**
  The `plot` statement will display the item locations graphically on a Wright Map. The `order=value` option is available for Wright Maps and displays the objects ordered by their scale location parameters (in this case, the team strength). The Wright Map only displays weighted likelihood parameter estimates (`estimates=wle`) in pairwise comparisons.

- **Line 7**
  The `show` statement produces a display of the item response model parameter estimates and saves them to the file `ex13_shw.txt`. The show file output is different in pairwise comparisons compared to the usual ACER ConQuest 1PL and 2PL model outputs. The show file only provides a list of the parameter estimates and their standard errors. Population parameters and traditional item statistics are not applicable with the pairwise model.

- **Line 8**
  The `show residuals` statement requests residuals for each fixed pair-outcome combination. These results are written to the file `ex13_res.csv` and are only available for weighted likelihood estimates.

2.14.4 Running the Analysis

To run this sample analysis, start the GUI version. Open the file `ex13.cqc` and choose `Run`→`Run All`. ACER ConQuest will begin executing the statements that are in the file `ex13.cqc`; and as they are executed, they will be echoed on the screen. When ACER ConQuest reaches the `estimate` statement, it will begin fitting the BTL model to the data, and as it does so it will report on the progress of the estimation.

After the estimation is complete, the outputs will be produced. The first `show` statement will produce a summary output and one table that shows the parameter estimates of each team and the standard errors of these parameter estimates. This output is in the file `ex13_shw.txt` (by default, ACER ConQuest will add an appropriate file extension to all outputs). The parameter estimates are in logits and placed on an interval scale, thereby allowing for evaluating the relative differences between the teams using a uniform unit of measurement. The location parameters are constrained to a mean of zero.

Figure 2.103 shows the location parameter estimates for each of the 16 teams. Results show that Geelong is the strongest team while Richmond is the weakest.

```
================================================================================
================================================================================
Pairwise Analysis of Australian Football League          Mon Sep 28 10:24 2015
IMPORTED MODEL:
=======================================================Build: Aug 18 2015===
Parameter Estimates
------------------------------------------------------------------------------------------
   VARIABLES                                UNWEIGHTED FIT          WEIGHTED FIT
                                        ----------------------  ----------------------
                     ESTIMATE   ERROR     MNSQ       CI       T    MNSQ      CI       T
------------------------------------------------------------------------------------------
Adelaide             0.16433  0.12594
Brisbane             0.36658  0.12626
Carlton             -0.40261  0.12938
Collingwood         -0.01258  0.12563
Essendon             0.17664  0.12631
Fremantle           -0.33520  0.13002
Geelong              0.40631  0.12807
Hawthorn            -0.05592  0.12678
Melbourne           -0.35280  0.12896
Nth Melbourne        0.08000  0.12633
Port Adelaide        0.26925  0.12617
Richmond            -0.47970  0.13194
St Kilda             0.00922  0.12627
Sydney               0.18382  0.12586
West Coast          -0.01327  0.12581
Wstn Bulldogs       -0.00408  0.12651
--------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
================================================================================
```

Figure 2.103: Table of item parameter esitmates

The `show residuals` statement produces an Excel file `ex13_res.csv`. Figure 2.104 shows the contents of the residuals table in `ex13_res.csv`. These are the residuals for each game and can be interpreted as prediction errors for each game based on the estimated team strengths.

Similar to the interpretation of residuals in regression, where $r_{ij} = Y_{ij} - P_{ij}$. That is, the residual r_{ij} for a particular game for a particular pair i, j is the difference between the observed outcome Y_{ij} (1 if i actually won, 0 if i lost) and the predicted outcome P_{ij} (the probability that i wins over j).

This residuals table can be summarised (filtered or sorted) by team1, team2, and magnitude of residual value to assess the predictive power of the model and check unusually high prediction errors for some teams.

The `plot` command produce the plot shown in Figure 2.105, which shows all the teams plotted against the location parameter estimate axis (i.e., team strength). The `order=value` option arranges the teams based on their parameter value for easier comparison and ranking. The plot also presents visually which teams have similar strengths as well as the relative differences in strength among the teams.

2.14.5 Summary

In this tutorial, ACER ConQuest has been used to fit the BTL model for a pairwise comparison analysis. Some key points covered were:

- The `pairwise` option in the `model` statement can be used to estimate a BTL model given dataset which contains paired comparisons and dichotomous outcomes for each comparison.
- The object location parameters estimated by ACER ConQuest can be used for ordinal comparison data to determine the location of an object on an interval scale.
- The plots visually show the relative locations of the objects and can be used to visually represent the rankings.

Comparison	Obs	Exp	Res	Object 1	Object 2
1	1	0.505615	0.494385	St Kilda	West Coast
2	1	0.50332	0.49668	St Kilda	Wstn Bulldogs
3	1	0.456449	0.543551	St Kilda	Sydney
4	1	0.50332	0.49668	St Kilda	Wstn Bulldogs
5	1	0.456449	0.543551	St Kilda	Sydney
6	1	0.50332	0.49668	St Kilda	Wstn Bulldogs
7	1	0.505615	0.494385	St Kilda	West Coast
8	1	0.50332	0.49668	St Kilda	Wstn Bulldogs
9	1	0.505615	0.494385	St Kilda	West Coast
10	1	0.456449	0.543551	St Kilda	Sydney
11	0	0.505615	-0.505615	St Kilda	West Coast
12	0	0.456449	-0.456449	St Kilda	Sydney
13	0	0.50332	-0.50332	St Kilda	Wstn Bulldogs
14	0	0.50332	-0.50332	St Kilda	Wstn Bulldogs
15	1	0.456449	0.543551	St Kilda	Sydney
16	0	0.505615	-0.505615	St Kilda	West Coast
17	0	0.456449	-0.456449	St Kilda	Sydney
18	0	0.50332	-0.50332	St Kilda	Wstn Bulldogs
19	1	0.505615	0.494385	St Kilda	West Coast
20	1	0.456449	0.543551	St Kilda	Sydney
21	1	0.50332	0.49668	St Kilda	Wstn Bulldogs
22	1	0.50332	0.49668	St Kilda	Wstn Bulldogs
23	0	0.505615	-0.505615	St Kilda	West Coast
24	1	0.456449	0.543551	St Kilda	Sydney
25	1	0.50332	0.49668	St Kilda	Wstn Bulldogs
26	0	0.505615	-0.505615	St Kilda	West Coast
27	0	0.456449	-0.456449	St Kilda	Sydney
28	1	0.50332	0.49668	St Kilda	Wstn Bulldogs
29	1	0.456449	0.543551	St Kilda	Sydney
30	0	0.505615	-0.505615	St Kilda	West Coast
31	1	0.456449	0.543551	St Kilda	Sydney
32	0	0.50332	-0.50332	St Kilda	Wstn Bulldogs

Figure 2.104: Extract of table of residuals for each paired comparison

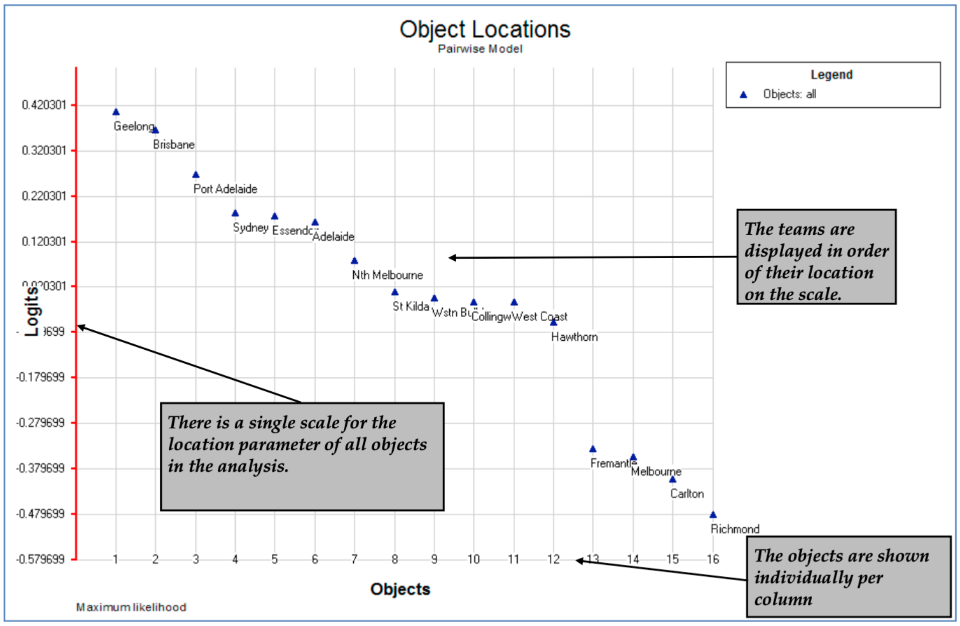


Figure 2.105: Wright Map of location parameter estimates of all teams

2.15 Fitting IRTree response models

2.15.1 Background

This tutorial demonstrates how to fit IRTree models with ACER ConQuest. We will fit one of the models that was applied to intelligence test data by De Boeck & Partchev (2012). These researchers applied a variety of IRTree models to investigate the relationship between so-called fast and slow intelligence, with responses to items partitioned into fast and slow responses respectively. Multiple latent variables were specified to model the speed and accuracy processes.

It is worth noting that IRTree models can also be implemented to investigate and model response processes from a single set of item responses, without the need for auxiliary timing or telemetry data. Prominent examples include modelling response styles for Likert-type questionnaire items, and modelling missing data mechanisms in educational assessments (Jeon & De Boeck, 2015).

In this tutorial we apply an IRTree model to data from a reading comprehension assessment targeted at approximately Grade 3 level. While this is a distinct construct from the intelligence tests just mentioned, reading comprehension is itself complex, and responding to multiple-choice reading comprehension questions can involve a multitude of mental processes. Therefore we were open to the possibility that response speed could be a proxy for the utilisation of different strategies and mental processes within items and within persons. We can use IRTrees to investigate whether differences in item and person parameters are observed for differentially speeded responses. At the same time we can also investigate the relationships between the latent trait pertaining to speed and the latent traits pertaining to 'fast' and 'slow' reading comprehension abilities. For more detail on how IRTrees can be used to investigate these substantive matters, the interested reader is referred to DiTrapani et al. (2016).

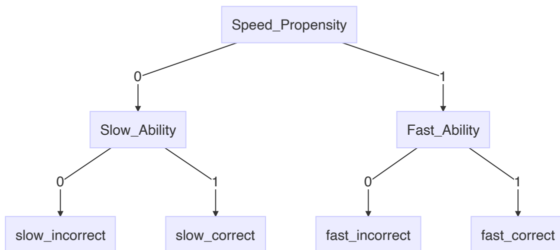
2.15.1.1 IRTree formulation

The IRTree models formulated here can be specified as multidimensional IRT models (see 2.8). We chose to specify a multidimensional Rasch (Rasch, 1960) model, consistent with the formulation in De Boeck & Partchev (2012).

One of the first steps in setting up an IRTree analysis from wide-format response files is to recode the data in a manner consistent with the conditional dependencies in the given IRTree. Responses to each individual item need to be separated into responses to multiple pseudo items that each correspond to a different node in the tree. These nodes play the role of latent variables in the multidimensional model, for which separate sets of item parameters and person parameters can be estimated.

In the case of fast and slow abilities, we first specify a latent variable node that represents the propensity to respond quickly. We use within-item median response times to separate responses into either fast or slow responses, much as De Boeck & Partchev (2012) did, in order to assign scores (1 for fast, 0 for slow) to pseudo items for this node. Then, for fast responses and slow responses in turn, we create pseudo items for the correctness of the response. A depiction of the corresponding response tree follows:

```
## file:////private/var/folders/hd/rbbr87hs06ncxdxr0kc_y9sh1q1dg1/T/RtmpR3cG3q/file15c3a
```


Speed_Propensity
0
1
Slow_Ability
Fast_Ability
0
1
0
1
slow_incorrect
slow_correct
fast_incorrect
fast_correct

It follows that for any item where a student responds quickly: they will have a value of 1 for that item's corresponding 'speed' pseudo item that loads onto the Speed Propensity node; they will have a value of either 0 or 1 for that item's 'fast and accurate' pseudo item that loads onto the Fast Ability node; and, they will have a value of 'missing' for the corresponding 'slow and accurate' pseudo item that loads onto the Slow Ability node. For a student who responds slowly to an item: they will have a value of 0 for the 'speed' pseudo item; they will have a value of either 0 or 1 for the 'slow and accurate' pseudo item; and, they will have a value of 'missing' for the 'fast and accurate' pseudo item. This yields a total of four possible response categories per item from the original response time and response accuracy information. Usefully, the conditional independence specified between the nodes makes it possible to compute the probability that a given student will respond in any one of the four end-point categories, conditional on their propensity to respond quickly and their estimated fast and slow abilities.

2.15.1.2 Recoding responses

The process of constructing an appropriately structured and recoded data file from the original response and response time information is as follows.

We first read in the item responses and the item response times for the 32 reading comprehension multiple-choice questions for a sample of 2000 students.

We then create the 'speed' pseudo items by assigning all items' responses to either fast or slow. This produces 32 such columns, one per item.

Next we create the two sets of accuracy-related pseudo items that are conditional on speed. This produces a further 32 columns for the fast responses and the slow responses respectively. Therefore there will be a total of 96 pseudo item columns in the data file to be analysed.

We can check whether the recoding is consistent with the implied conditional independence in the tree diagram. For item 10 in the sequence, we inspect the response category frequencies for fast and slow accuracies, conditional on response speed.

We can see that each of these accuracy pseudo items has half of its responses (1000) as 0 or 1 values, and half of its values as missing ('NA' is the default code for missing in the R program that was used to prepare the data). This is consistent with the within-item median split that was used for all 'speed' pseudo items. This shows that the data for this particular item have been partitioned appropriately for the estimation of the two separate latent abilities.

Table 2.1: Item 10 fast pseudo item score frequencies for slow responses (top row) and fast responses (bottom row)

Response speed	0	1	NA
Slow	0	0	1000
Fast	116	884	0

Table 2.2: Item 10 slow pseudo item score frequencies for slow responses (top row) and fast responses (bottom row)

Response speed	0	1	NA
Slow	267	733	0
Fast	0	0	1000

2.15.2 Required files

The files that we will use in this example are:

filename	content
ex15.cqc	The command lines used for the analysis.
ex15_data.csv	The pseudo item scored responses.
ex15_anch.dat	Anchor values for item parameters for the speed node.
ex15_shw.txt	Selected results from the analysis.

2.15.3 Syntax

The command syntax for an IRTree model is equivalent to that for a conventional between-item compensatory multidimensional model. The differences, as noted, relate more to the restructuring of the underlying response data.

The contents of the command file ex15.cqc are shown in the code box below, and explained line-by-line in the list that follows the figure.

2.15.3.1 Multidimensional Rasch formulation of an IRTree model

`ex15.cqc:`

```
1  title speed accuracy tree;
2  data ex15_data.csv ! filetype = csv,columnlabels = yes, responses = n_speed_01 to n_slow
3  codes 0 1;
4  score (0,1) (0,1) () () ! items(1-32);
5  score (0,1) () (0,1) () ! items(33-64);
6  score (0,1) () () (0,1) ! items(65-96);
7  import anchor_parameters << ex15_anch.dat;
8  model item;
9  estimate;
10 show ! estimates=latent  >> ex15_shw.txt;
```

- **Line 1**
  This is the title of the analysis.

- **Line 2**
  Indicates the name and location of the data file. Here we use a CSV file, as indicated following the *filetype* command. We retain column labels and we use these in conjunction with the *responses* command to name the range of columns (pseudo item responses) to include in the analysis.

- **Line 3**
  We use the codes statement to restrict the valid codes to 0 and 1. Importantly, we omit missing or blank values so that these are definitely treated as missing in the analysis.

- **Line 4-6**
  The model that we are fitting here is three dimensional, so the score statements contain four sets of parentheses as their arguments, one for the 'from' codes and three for the 'to' codes. The option of the first score statement gives the items to be assigned to the first dimension, the option of the second score statement gives the items to be allocated to the second dimension, and so on. Recall that we have a 'speed' dimension, followed by two ability dimensions, one fast and one slow. Earlier, we intentionally constructed the pseudo item response columns in blocks that corresponded to each of these latent variables. The sequential nature of these three blocks of items can be seen in parentheses after the term *! items* at the end of each of the lines 4-6.

- **Line 7**
  We choose to anchor all item parameter values corresponding to the speed node to

zero. This is because we have used a within-item median response time split (and we have very fine-grained timing data down to milliseconds), meaning that exactly half of the responses will be classified as fast and half will be classified as slow. Retaining the constraint that the mean of the item parameter values per dimension is zero, this implies that every item parameter in the speed dimension has the same relative location along the speed trait, which is necessarily zero. Anchoring is not strictly necessary, but it simplifies the estimation process since fewer parameters need to be estimated.

- **Line 8**
  The simple logistic model is used for each dimension.

- **Line 9**
  The model is estimated using the default settings.

- **Line 10**
  The show statement produces a sequence of tables that summarise the results of fitting the item response model.

2.15.4 Running the analysis

To run this sample analysis, start the GUI version. Open the file ex15.cqc and choose Run -> Run All.

Alternatively, you can launch the console version of ACER ConQuest, by typing the command (on Windows) `ConQuestConsole.exe ex15.cqc`.

For conquestr (Cloney & Adams, 2022) users, you can also call ACER ConQuest and run ex15.cqc using the command `conquestr::ConQuestCall('ex15.cqc')`.

By inspecting the show file output (Figure 2.106), we can see what parameters were estimated and we can draw several salient conclusions about the relationship between fast and slow responses in the context of our reading comprehension assessment.

In this analysis 71 parameters were estimated. They are:

a. the mean and variance of the three latent nodes that are being measured (making 6 parameters);
b. the covariance values between the three latent nodes (making 3 parameters); and

```
================================================================================
speed accuracy tree                                         Thu Aug 18 13:10 2022
SUMMARY OF THE ESTIMATION
==============================================================Version: 5.22.3===

DATA SPECIFICATIONS

The Data File: data\ex15_dat.csv
The format:  responses 1-96 (a1)
Cases in file: 2000  Weighted number of cases in estimation: 2000


MODEL SPECIFICATIONS

Xsi increment max:    1.00000
FacOldXsi:    0.00000
Assumed population distribution was: Gaussian
Location constraint was: DEFAULT
Scale constraint was: Not applicable
No case weights
The regression model:
Grouping Variables:
The item model: item
Slopes are fixed
Total number of estimated parameters: 71
Random number generation seed:    2.00000
Number of nodes used when drawing PVs: 2000
Number of plausible values to draw: 5
Maximum number of iterations without a deviance improvement: 100
Maximum number of Newton steps for each parameter in M-step: 10
Value for obtaining finite MLEs for zero/perfects:    0.30000


ESTIMATION SPECIFICATIONS AND FIT

Estimation method was: Gauss-Hermite Quadrature with 3375 nodes
No node filtering
Final Deviance:                               149706.20811
Akaike Information Criterion (AIC):           149848.20811
Akaike Information Criterion Corrected (AICc): 149843.40618
Bayesian Information Criterion (BIC):         150245.87218
The number of iterations: 148
Termination criteria:  Max iterations=1000,Parameter Change= 0.00100
                       Deviance Change= 0.00010
Iterations terminated because the convergence criteria were reached
```

Figure 2.106: Summary of estimation information

c. 62 item difficulty parameters. Recall that we did not estimate any parameters for the speed dimension, and, following the usual convention of Rasch modelling, the mean of the item difficulty parameters within each dimension has been made zero, so that a total of 31 parameters is required to describe the difficulties of the 32 items.

Several interesting observations can be made in relation to the covariance/correlation matrix and the variance values for each dimension:

- Most notably, the latent correlation between fast and slow abilities is extremely close to one. This is reassuring, in that the test appears to measure the same underlying ability irrespective of whether students are responding quickly or slowly as defined by the within-item median split.
- There is a weak negative correlation between the propensity to respond quickly and the two (ostensibly one) latent abilities. This implies that the propensity to respond quickly tends to be associated with slightly poorer performance.
- The variance of the latent ability measured from fast responses is larger than that for the latent ability measured from slow responses. There appears to be more information in the fast responses than in the slow responses.

We see from the first 32 item parameters that the anchoring process has been applied as intended. The parameters for all pseudo items mapped to the speed node are equal to zero. Interestingly, this dimension has a comparable level of reliability to the ability dimensions. The items also appear to fit the model reasonably well according to the fit statistics shown above.

Looking now at the two sets of item parameters for the fast and slow ability traits, some interesting differences can be observed. While most items have similar relative difficulty between the two sets, some individual items differ more noticeably in their relative difficulties when they are responded to quickly as opposed to slowly. For example, the fifth item in the sequence (which appears at lines 37 and 69) differs more than is the case for most items, with the difference being almost one logit (though note that these scales are not directly comparable). This item appears to be easier when it is responded to more slowly. An obvious follow up activity is to qualitatively review this item in light of these findings, and to attempt to devise some hypotheses for the differential item difficulty under fast and slow responses.

```
================================================================================
speed accuracy tree                                          Thu Aug 18 13:10 2022
TABLES OF POPULATION MODEL PARAMETER ESTIMATES
=============================================================Version: 5.22.3===
REGRESSION COEFFICIENTS

                                          Dimension
                         ---------------------------------------------------
Regression Variable          Dimension_1      Dimension_2      Dimension_3

CONSTANT                   0.007 ( 0.023)    0.934 ( 0.030)    0.657 ( 0.024)
---------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
=================================================================================
UNCONDITIONAL COVARIANCE/CORRELATION MATRIX

                                         Dimension
                         ------------------------------------------------------
Dimension                    1                 2                 3

Dimension_1                                 -0.308            -0.202
Dimension_2               -0.221                               1.420
Dimension_3               -0.182             0.978
-------------------------------------------------------------------------------
Variance                   1.065  ( 0.034)   1.818  ( 0.057)   1.160  ( 0.037)
-------------------------------------------------------------------------------
An asterisk next to a parameter estimate indicates that it is constrained
Values below the diagonal are correlations and values above are covariances
===============================================================================

RELIABILITY COEFFICIENTS
------------------------

Dimension: (Dimension_1)
-----------------------
 MLE Person separation RELIABILITY:  Unavailable
 WLE Person separation RELIABILITY:  Unavailable
 EAP/PV RELIABILITY:                  0.866
Dimension: (Dimension_2)
-----------------------
 MLE Person separation RELIABILITY:  Unavailable
 WLE Person separation RELIABILITY:  Unavailable
 EAP/PV RELIABILITY:                  0.856
Dimension: (Dimension_3)
-----------------------
 MLE Person separation RELIABILITY:  Unavailable
 WLE Person separation RELIABILITY:  Unavailable
 EAP/PV RELIABILITY:                  0.849
```

Figure 2.107: Summary of latent variable distributions

```
================================================================================
speed accuracy tree                                         Thu Aug 18 16:32 2022
TABLES OF RESPONSE MODEL PARAMETER ESTIMATES
==============================================================Version: 5.22.3===
TERM 1: item
------------------------------------------------------------------------------------------
TERM 1: item

  VARIABLES                              UNWEIGHTED FIT            WEIGHTED FIT
---------------                       -----------------------  -----------------------
    item          ESTIMATE  ERROR^    MNSQ      CI          T    MNSQ      CI          T
------------------------------------------------------------------------------------------
 1   n_speed_01      0.000*           1.15 ( 0.94, 1.06)  4.7  1.12 ( 0.97, 1.03)  6.4
 2   n_speed_02      0.000*           1.03 ( 0.94, 1.06)  0.9  1.03 ( 0.97, 1.03)  1.9
 3   n_speed_03      0.000*           1.07 ( 0.94, 1.06)  2.3  1.05 ( 0.97, 1.03)  2.9
 4   n_speed_04      0.000*           0.96 ( 0.94, 1.06) -1.3  0.98 ( 0.97, 1.03) -1.2
 5   n_speed_05      0.000*           1.10 ( 0.94, 1.06)  3.0  1.09 ( 0.97, 1.03)  4.8
 6   n_speed_06      0.000*           0.91 ( 0.94, 1.06) -2.9  0.94 ( 0.97, 1.03) -3.3
 7   n_speed_07      0.000*           0.98 ( 0.94, 1.06) -0.5  0.99 ( 0.97, 1.03) -0.5
 8   n_speed_08      0.000*           1.10 ( 0.94, 1.06)  3.1  1.08 ( 0.97, 1.03)  4.2
 9   n_speed_09      0.000*           0.96 ( 0.94, 1.06) -1.3  0.97 ( 0.97, 1.03) -1.7
 10  n_speed_10      0.000*           1.08 ( 0.94, 1.06)  2.4  1.05 ( 0.97, 1.03)  2.9
 11  n_speed_11      0.000*           1.10 ( 0.94, 1.06)  3.1  1.09 ( 0.97, 1.03)  4.9
 12  n_speed_12      0.000*           0.98 ( 0.94, 1.06) -0.7  0.99 ( 0.97, 1.03) -0.6
 13  n_speed_13      0.000*           0.93 ( 0.94, 1.06) -2.3  0.95 ( 0.97, 1.03) -2.6
 14  n_speed_14      0.000*           0.95 ( 0.94, 1.06) -1.5  0.97 ( 0.97, 1.03) -1.7
 15  n_speed_15      0.000*           0.87 ( 0.94, 1.06) -4.2  0.91 ( 0.97, 1.03) -5.3
 16  n_speed_16      0.000*           1.06 ( 0.94, 1.06)  1.9  1.06 ( 0.97, 1.03)  3.5
 17  n_speed_17      0.000*           0.95 ( 0.94, 1.06) -1.6  0.97 ( 0.97, 1.03) -1.8
 18  n_speed_18      0.000*           0.93 ( 0.94, 1.06) -2.4  0.95 ( 0.97, 1.03) -2.8
 19  n_speed_19      0.000*           0.92 ( 0.94, 1.06) -2.5  0.95 ( 0.97, 1.03) -2.9
 20  n_speed_20      0.000*           0.95 ( 0.94, 1.06) -1.7  0.96 ( 0.97, 1.03) -2.0
 21  n_speed_21      0.000*           1.20 ( 0.94, 1.06)  6.0  1.15 ( 0.97, 1.03)  8.1
 22  n_speed_22      0.000*           0.95 ( 0.94, 1.06) -1.7  0.97 ( 0.97, 1.03) -1.9
 23  n_speed_23      0.000*           0.97 ( 0.94, 1.06) -1.1  0.98 ( 0.97, 1.03) -1.3
 24  n_speed_24      0.000*           0.94 ( 0.94, 1.06) -1.9  0.95 ( 0.97, 1.03) -2.7
 25  n_speed_25      0.000*           1.00 ( 0.94, 1.06) -0.1  1.00 ( 0.97, 1.03) -0.1
 26  n_speed_26      0.000*           0.96 ( 0.94, 1.06) -1.3  0.97 ( 0.97, 1.03) -1.4
 27  n_speed_27      0.000*           1.10 ( 0.94, 1.06)  3.2  1.09 ( 0.97, 1.03)  4.9
 28  n_speed_28      0.000*           0.91 ( 0.94, 1.06) -2.8  0.94 ( 0.97, 1.03) -3.4
 29  n_speed_29      0.000*           0.94 ( 0.94, 1.06) -2.1  0.95 ( 0.97, 1.03) -2.8
 30  n_speed_30      0.000*           0.94 ( 0.94, 1.06) -1.9  0.96 ( 0.97, 1.03) -2.1
 31  n_speed_31      0.000*           0.92 ( 0.94, 1.06) -2.4  0.94 ( 0.97, 1.03) -3.2
 32  n_speed_32      0.000*           0.98 ( 0.94, 1.06) -0.6  0.99 ( 0.97, 1.03) -0.5
 33  n_fast_01      -2.650   0.068    0.50 ( 0.91, 1.09)-13.9  0.91 ( 0.78, 1.22) -0.8
 34  n_fast_02      -0.376   0.056    1.23 ( 0.91, 1.09)  4.8  1.10 ( 0.92, 1.08)  2.2
 35  n_fast_03       0.188   0.054    0.94 ( 0.91, 1.09) -1.4  0.95 ( 0.93, 1.07) -1.3
 36  n_fast_04      -0.217   0.056    0.98 ( 0.91, 1.09) -0.5  0.93 ( 0.92, 1.08) -1.8
 37  n_fast_05      -0.606   0.056    0.56 ( 0.91, 1.09)-11.8  0.75 ( 0.92, 1.08) -6.9
 38  n_fast_06       0.418   0.054    1.14 ( 0.91, 1.09)  3.1  1.09 ( 0.93, 1.07)  2.4
 39  n_fast_07      -0.385   0.056    0.91 ( 0.91, 1.09) -2.1  0.91 ( 0.92, 1.08) -2.1
 40  n_fast_08      -0.603   0.057    0.66 ( 0.91, 1.09) -8.7  0.78 ( 0.91, 1.09) -5.3
```

Figure 2.108: Summary of speed pseudo item parameters

```
33  n_fast_01      -2.650   0.068    0.50 ( 0.91, 1.09)-13.9   0.91 ( 0.78, 1.22) -0.8
34  n_fast_02      -0.376   0.056    1.23 ( 0.91, 1.09)  4.8   1.10 ( 0.92, 1.08)  2.2
35  n_fast_03       0.188   0.054    0.94 ( 0.91, 1.09) -1.4   0.95 ( 0.93, 1.07) -1.3
36  n_fast_04      -0.217   0.056    0.98 ( 0.91, 1.09) -0.5   0.93 ( 0.92, 1.08) -1.8
37  n_fast_05      -0.606   0.056    0.56 ( 0.91, 1.09)-11.8   0.75 ( 0.92, 1.08) -6.9
38  n_fast_06       0.418   0.054    1.14 ( 0.91, 1.09)  3.1   1.09 ( 0.93, 1.07)  2.4
39  n_fast_07      -0.385   0.056    0.91 ( 0.91, 1.09) -2.1   0.91 ( 0.92, 1.08) -2.1
40  n_fast_08      -0.603   0.057    0.66 ( 0.91, 1.09) -8.7   0.78 ( 0.91, 1.09) -5.3
41  n_fast_09      -0.939   0.059    0.55 ( 0.91, 1.09)-12.0   0.77 ( 0.90, 1.10) -4.7
42  n_fast_10      -1.413   0.063    0.45 ( 0.91, 1.09)-15.5   0.81 ( 0.86, 1.14) -2.8
43  n_fast_11       0.886   0.053    1.25 ( 0.91, 1.09)  5.3   1.17 ( 0.94, 1.06)  4.8
44  n_fast_12       1.255   0.054    1.44 ( 0.91, 1.09)  8.7   1.23 ( 0.93, 1.07)  6.1
45  n_fast_13       0.157   0.053    1.19 ( 0.91, 1.09)  3.9   1.18 ( 0.94, 1.06)  5.3
46  n_fast_14       0.826   0.053    1.54 ( 0.91, 1.09) 10.4   1.36 ( 0.94, 1.06) 10.2
47  n_fast_15       0.298   0.054    0.92 ( 0.91, 1.09) -1.7   0.93 ( 0.93, 1.07) -2.0
48  n_fast_16      -0.030   0.055    0.71 ( 0.91, 1.09) -7.2   0.80 ( 0.93, 1.07) -5.7
49  n_fast_17       0.510   0.054    1.11 ( 0.91, 1.09)  2.4   1.08 ( 0.93, 1.07)  2.3
50  n_fast_18      -0.493   0.056    0.82 ( 0.91, 1.09) -4.2   0.91 ( 0.92, 1.08) -2.2
51  n_fast_19       0.143   0.054    0.97 ( 0.91, 1.09) -0.8   0.99 ( 0.93, 1.07) -0.3
52  n_fast_20      -0.106   0.056    1.14 ( 0.91, 1.09)  3.0   1.12 ( 0.92, 1.08)  3.0
53  n_fast_21      -0.350   0.055    0.71 ( 0.91, 1.09) -7.3   0.86 ( 0.93, 1.07) -4.2
54  n_fast_22      -0.031   0.055    0.71 ( 0.91, 1.09) -7.2   0.83 ( 0.93, 1.07) -5.0
55  n_fast_23       0.848   0.053    0.93 ( 0.91, 1.09) -1.6   0.93 ( 0.94, 1.06) -2.4
56  n_fast_24       0.722   0.053    1.35 ( 0.91, 1.09)  7.1   1.21 ( 0.94, 1.06)  6.2
57  n_fast_25      -1.046   0.058    0.94 ( 0.91, 1.09) -1.4   0.98 ( 0.90, 1.10) -0.3
58  n_fast_26       0.487   0.054    1.05 ( 0.91, 1.09)  1.0   1.02 ( 0.93, 1.07)  0.7
59  n_fast_27       1.309   0.055    1.43 ( 0.91, 1.09)  8.6   1.22 ( 0.93, 1.07)  5.5
60  n_fast_28       0.375   0.054    1.01 ( 0.91, 1.09)  0.3   1.00 ( 0.93, 1.07)  0.0
61  n_fast_29      -0.056   0.054    0.89 ( 0.91, 1.09) -2.7   0.96 ( 0.93, 1.07) -1.1
62  n_fast_30       0.858   0.054    0.88 ( 0.91, 1.09) -2.7   0.87 ( 0.93, 1.07) -4.0
63  n_fast_31      -0.042   0.054    1.31 ( 0.91, 1.09)  6.3   1.17 ( 0.93, 1.07)  4.4
64  n_fast_32       0.061*           0.88 ( 0.91, 1.09) -2.7   0.96 ( 0.93, 1.07) -1.1
65  n_slow_01      -2.568   0.067    0.71 ( 0.91, 1.09) -7.2   0.95 ( 0.78, 1.22) -0.4
66  n_slow_02       0.047   0.052    1.11 ( 0.91, 1.09)  2.5   1.09 ( 0.94, 1.06)  3.0
67  n_slow_03      -0.206   0.053    0.88 ( 0.91, 1.09) -2.8   0.91 ( 0.94, 1.06) -3.0
68  n_slow_04       0.186   0.052    0.99 ( 0.91, 1.09) -0.2   0.99 ( 0.95, 1.05) -0.3
69  n_slow_05      -1.552   0.061    0.67 ( 0.91, 1.09) -8.4   0.86 ( 0.87, 1.13) -2.1
70  n_slow_06       0.369   0.052    1.17 ( 0.91, 1.09)  3.5   1.10 ( 0.95, 1.05)  3.6
71  n_slow_07      -0.296   0.053    1.09 ( 0.91, 1.09)  2.0   1.04 ( 0.94, 1.06)  1.1
72  n_slow_08      -0.732   0.055    0.77 ( 0.91, 1.09) -5.6   0.87 ( 0.92, 1.08) -3.4
73  n_slow_09      -0.510   0.054    1.00 ( 0.91, 1.09) -0.0   0.96 ( 0.93, 1.07) -1.3
74  n_slow_10      -0.815   0.055    0.79 ( 0.91, 1.09) -5.0   0.89 ( 0.93, 1.07) -3.2
75  n_slow_11       0.954   0.051    1.07 ( 0.91, 1.09)  1.5   1.03 ( 0.95, 1.05)  1.3
76  n_slow_12       1.519   0.052    1.19 ( 0.91, 1.09)  4.1   1.09 ( 0.94, 1.06)  3.1
77  n_slow_13       0.369   0.052    1.06 ( 0.91, 1.09)  1.3   1.05 ( 0.94, 1.06)  1.7
78  n_slow_14       1.082   0.052    1.19 ( 0.91, 1.09)  4.1   1.12 ( 0.95, 1.05)  4.7
79  n_slow_15       0.226   0.052    1.06 ( 0.91, 1.09)  1.2   1.04 ( 0.94, 1.06)  1.4
80  n_slow_16      -0.746   0.056    0.77 ( 0.91, 1.09) -5.6   0.88 ( 0.91, 1.09) -2.8
```

Figure 2.109: Summary of fast and slow item parameters

2.15.5 Summary

In this section we have illustrated how ACER ConQuest can be used to fit IRTree models. Specifically, we fit a one-parameter IRTree model and interpreted various features of the output.

Some key points that were covered include:

- IRTree models can be specified as multidimensional models with an appropriate restructuring and recoding of response data.
- A variety of response processes can be investiagted using IRTree models.
- Insights can be gained into how item parameters and latent abilities change under different processes.
- Insights can be gained into how different latent abilities relating to different aspects of a response process are related to one another.
- It is possible to carry out model comparisons that compare different assumptions about response processes and associated latent traits. As an example, it is possible to show that a model with the latent correlation between fast and slow abilities constrained to equal one has a marginally lower BIC than the unconstrained model shown in this tutorial. This provides some support for the claim that fast and slow abilities can be treated as equivalent for this reading comprehension test (though Log-Likelihood and AIC slightly prefer the unconstrained model, so the statistical support is slightly equivocal).

Chapter 3

Technical Matters

3.1 The Generalised Rasch Model

The model fitted by ACER ConQuest is a generalised multidimensional Rasch item response model coupled with a multivariate regression model. We call these two components of the model the item response model and the population model respectively. As we have illustrated in previous sections, the model allows ACER ConQuest to be used for two important types of analyses.

First, the general specification of the item response model allows us to use one model to fit a wide variety of Rasch models. In the unidimensional case, this includes the simple logistic model (Wright & Panchapakesan, 1969), the linear logistic model (Fischer, 1973), the rating scale and partial credit models (Andrich, 1978; Glas, 1989; Masters, 1982), the ordered partition model (Wilson, 1992), and multifaceted models (Linacre, 1994). Furthermore, multidimensional dichotomous and polytomous response models, such as Kelderman's LOGIMO model (Kelderman & Rijkes, 1994), Rasch's multidimensional model (Rasch, 1980), and the models of Whitely (1980), Andersen (1985) and Embretson (1991), can also be shown to be special cases of the generalised multidimensional Rasch model.

Second, the combination of the item response and population models allows ACER ConQuest to be used to undertake latent regression. The term latent regression refers to the direct estimation of regression models from item response data. To illustrate the use of latent regression, consider the following typical situation. We have two groups of students, group A and group B, and we are interested in estimating the difference in the mean achievement of the two groups. If we follow standard practice, we will administer a common test to the students and then use this test to produce achievement scores for all of the students. We could then follow a standard procedure, such as regression (which, in this simple case, becomes identical to a t-test), to examine the difference in the means. Depending upon the method that is used to construct the achievement scores, this approach can result in misleading inferences about the differences in the means. Using the latent regression methods described by Adams, Wilson, & Wu (1997), ACER ConQuest avoids such problems by directly estimating the difference in the achievement of the groups from the response data.

3.1.1 The Item Response Model

The item response model fitted by ACER ConQuest is the multidimensional random coefficients multinomial logit model that was described by Adams, Wilson, & Wang (1997). For ease of explanation, we will first describe the unidimensional form of the model.

3.1.1.1 The Unidimensional Random Coefficients Multinomial Logit Model

Assume that I items are indexed $i = 1, ..., I$, with each item admitting $K_i + 1$ response alternatives $k = 0, 1, ..., K_i$. Use the vector-valued random variable $X_i' = (X_{i1}, X_{i2}, ..., X_{iK_i})$, where

$$X_{ij} = \begin{cases} 1 & \text{if the response to item } i \text{ is in category } j \\ 0 & \text{otherwise} \end{cases}, \tag{3.1}$$

to indicate the $K_i + 1$ possible responses to item i.

A response in category zero is denoted by a vector of zeroes. This effectively makes the zero category a reference category and is necessary for model identification. The choice of this as the reference category is arbitrary and does not affect the generality of the model. We can also collect the X_i together into the single vector $X' = (X_1', X_2', ..., X_I')$, which we call the response vector (or pattern). Particular instances of each of these random variables are indicated by their lower case equivalents: x, x_i and x_{ik}.

The items are modelled through a vector $\xi' = (\xi_1, \xi_2, ..., \xi_P)$ of P parameters. Linear combinations of these are used in the response probability model to describe the empirical characteristics of the response categories of each item. These linear combinations are defined by design vectors $a_{ik}'(i = 1, ..., I; k = 1, ..., K_i)$, each of length P, which can be collected to form a design matrix $A' = (a_{11}, a_{12}, ..., a_{1K_1}, a_{21}, a_{22}, ..., a_{2K_2}, ..., a_{I1}, ..., a_{IK_I})$.

Adopting a very general approach to the definition of items, in conjunction with the imposition of a linear model on the item parameters, allows us to write a general model that includes the wide class of existing Rasch models mentioned above and to develop new types of Rasch models (for example, the item bundles models of Wilson & Adams (1995)).

An additional feature of the model is the introduction of a scoring function that allows the specification of the score or 'performance level' that is assigned to each possible response to each item. To do this, we introduce the notion of a response score b_{ij}, which gives the performance level of an observed response in category j of item i. The b_{ij} can be collected in a vector as $b' = (b_{11}, b_{12}, ..., b_{1K_1}, b_{21}, b_{22}, ..., b_{2K_2}, ..., b_{I1}, ..., b_{IK_I})$. By definition, the score for a response in the zero category is zero, but other responses may also be scored zero.

In the majority of Rasch model formulations, there has been a one-to-one match between the category to which a response belongs and the score that is allocated to the response. In the simple logistic model, for example, it has been standard practice to use the labels 0 and 1 to indicate both the categories of performance and the scores. A similar practice has been followed with the rating scale and partial credit models, where each different possible response is seen as indicating a different level of performance, so that the category indicators 0, 1, 2 etc. that are used serve both as scores and labels. The use of b as a scoring function allows a more flexible relationship between the qualitative aspects of a response and the level of performance that it reflects. Examples of where this is applicable are given in Kelderman & Rijkes (1994) and Wilson (1992).

Letting θ be the latent variable, $b_{i0} \equiv 0$ and $a_{i0} \equiv 0$, the item response probability model is written as

$$Pr(X_{ij} = 1; A, b, \xi|\theta) = \frac{exp(b_{ij}\theta + a'_{ij}\xi)}{\sum_{k=0}^{K_i} exp(b_{ik}\theta + a'_{ij}\xi)}, \tag{3.2}$$

and a response vector probability model as

$$f(x; \xi|\theta) = \Psi(\theta, \xi) exp[x'(b\theta + A\xi)], \tag{3.3}$$

with

$$\Psi(\theta, \xi) = \{\sum_{z \in \Omega} exp[z'(b\theta + A\xi)]\}^{-1}, \tag{3.4}$$

where Ω is the set of all possible response vectors.

3.1.1.2 The Multidimensional Random Coefficients Multinomial Logit Model

The multidimensional form of the model is a straightforward extension of the unidimensional model. It assumes that a set of D latent traits underlies the individuals' responses. The D latent traits define a D-dimensional latent space, and the individuals' positions in the D-dimensional latent space are represented by the vector $\theta = (\theta_1, \theta_2, ..., \theta_D)$. The scoring function of response category k in item i now corresponds to a D-by-1-column vector rather than to a scalar as in the unidimensional model. A response in category k in dimension d of item i is scored b_{ikd}. The scores across D dimensions can be collected into a column vector $b'_{ik} = (b_{ik1}, b_{ik2}, ..., b_{ikD})$ then collected into the scoring submatrix for item i, $B'_i = (b_{i1}, b_{i2}, ..., b_{iK_i})$ and then collected into a scoring matrix $B' = (B'_1, B'_2, ..., B'_I)$ for the whole test. If the item parameter vector, ξ, and the design matrix, A, are defined as they were in the unidimensional model, $b_{i0} \equiv 0$ and $a_{i0} \equiv 0$, the probability of a response in category k of item i is modelled as

$$Pr(X_{ij} = 1; A, B, \xi|\theta) = \frac{exp(b'_{ij}\theta + a'_{ij}\xi)}{\sum_{k=0}^{K_i} exp(b'_{ik}\theta + a'_{ik}\xi)}. \tag{3.5}$$

And for a response vector we have

$$f(x; \xi|\theta) = \Psi(\theta, \xi) exp[x'(B\theta + A\xi)], \tag{3.6}$$

with

$$\Psi(\theta, \xi) = \left\{ \sum_{z \in \Omega} exp[z'(B\theta + A\xi)] \right\}^{-1}. \tag{3.7}$$

The difference between the unidimensional model and the multidimensional model is that the ability parameter is a scalar, θ, in the former, and a D-by-1-column vector, θ, in the latter. Likewise, the scoring function of response k to item i is a scalar, b_{ik}, in the former, whereas it is a D-by-1-column vector, b_{ik}, in the latter.

For the purposes of the identification of (3.6), certain constraints must be placed on the design matrices A and B. Volodin & Adams (1995) show that the following are necessary and sufficient conditions for the identification of (3.6).

- *Proposition One:*
  If D is the number of latent dimensions, P is the length of the parameter vector, ξ, $K_i + 1$ is the number of response categories for item i, and $K = \sum_{i=1}^{I} K_i$, then model (3.6) if applied to the set of items I can only be identified if $P + D \leq K$.

- *Proposition Two:*
  If D is the number of latent dimensions and P is the length of the parameter vector, ξ, then model (3.6) can only be identified if $rank(A) = P$, $rank(B) = D$ and $rank(BA) = P + D$.

- *Proposition Two:*
  If D is the number of latent dimensions, P is the length of the parameter vector, ξ, $K_1 + 1$ is the number of response categories for item i, and $K = \sum_{i=1}^{I} K_i$, then model (3.6) if applied to the set of items I can only be identified if and only if $rank([BA]) = P + D \leq K$.

3.1.2 The Population Model

The item response model is a conditional model, in the sense that it describes the process of generating item responses conditional on the latent variable, θ. The complete definition of the model, therefore, requires the specification of a density, $f_\theta(\theta; \alpha)$, for the latent variable, θ. We use α to symbolise a set of parameters that characterise the distribution of θ. The most common practice when specifying unidimensional marginal item response models is to assume that the students have been sampled from a normal population with mean μ and variance σ^2. That is:

$$f_\theta(\theta; \alpha) \equiv f_\theta(\theta; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left[-\frac{(\theta-\mu)^2}{2\sigma^2}\right] \tag{3.8}$$

or equivalently

$$\theta = \mu + E \tag{3.9}$$

where $E \sim N(0, \sigma^2)$.

Adams, Wilson, & Wu (1997) discuss how a natural extension of (3.8) is to replace the mean, μ, with the regression model $Y_n'\beta$, where Y_n is a vector of U fixed and known values for student n and β is the corresponding vector of regression coefficients. For example, Y_n could be constituted of student variables, such as gender, socio-economic status, or major. Then the population model for student n becomes

$$\theta_n = Y_n'\beta + E_n, \tag{3.10}$$

where we assume that the E_n are independently and identically normally distributed with mean zero and variance σ^2 so that (3.10) is equivalent to

$$f_\theta(\theta_n; Y_n, \beta, \sigma^2) = (2\pi\sigma^2)^{-\frac{1}{2}} exp\left[-\frac{1}{2\sigma^2}(\theta_n - Y_n'\beta)'(\theta_n - Y_n'\beta)\right], \tag{3.11}$$

a normal distribution with mean $Y_n'\beta$ and variance σ^2. If (3.11) is used as the population model, then the parameters to be estimated are β, σ^2 and ξ.

The model takes the generalisation one step further by applying it to the vector-valued θ rather than the scalar-valued θ, resulting in the multivariate population model

$$f_\theta(\theta_n; Y_n, \gamma, \Sigma) = (2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}} exp\left[-\frac{1}{2}(\theta_n - \gamma Y_n)'\Sigma^{-1}(\theta_n - \gamma Y_n)\right], \tag{3.12}$$

where γ is a D-by-U matrix of regression coefficients, Σ is a D-by-D variance-covariance matrix and Y_n is a U-by-1 vector of fixed variables. If (3.12) is used as the population model, then the parameters to be estimated are γ, Σ and ξ.

3.1.3 Estimation

ACER ConQuest uses maximum likelihood methods to provide estimates of γ, Σ and ξ. Combining the conditional item response model (3.6) and the population model (3.12), we obtain the unconditional, or marginal, item response model

$$f_x(x; \xi, \gamma, \Sigma) = \int_\theta f_x(x; \xi|\theta) f_\theta(\theta; \gamma, \Sigma) d\theta, \tag{3.13}$$

and it follows that the likelihood is

$$\Lambda = \prod_{n=1}^{N} f_x(x_n; \xi, \gamma, \Sigma), \tag{3.14}$$

where N is the total number of sampled students. Throughout this section we have assumed unit weights for each sampled unit. The implemented algorithms are more general and allow for the use of weights. For simplicity the weight factor has been omitted from the equations.

Differentiating with respect to each of the parameters and defining the marginal posterior as

$$h_\theta(\theta_n; Y_n, \xi, \gamma, \Sigma | x_n) = \frac{f_x(x_n; \xi | \theta_n) f_\theta(\theta_n; Y_n, \gamma, \Sigma)}{f_x(x_n; Y_n, \xi, \gamma, \Sigma)} \tag{3.15}$$

provides the following system of likelihood equations:

$$A' \sum_{n=1}^{N} \left[x_n - \int_{\theta_n} E_z(z|\theta_n) h_\theta(\theta_n; Y_n, \xi, \gamma, \Sigma | x_n) d\theta_n \right] = 0, \tag{3.16}$$

$$\hat{\gamma} = \left(\sum_{n=1}^{N} \bar{\theta_n} Y_n' \right) \left(\sum_{n=1}^{N} Y_n Y_n' \right)^{-1}, \tag{3.17}$$

$$\hat{\Sigma} = \frac{1}{N} \sum_{n=1}^{N} \int_{\theta_n} (\theta_n - \gamma Y_n)(\theta_n - \gamma Y_n)' h_\theta(\theta_n; Y_n, \xi, \gamma, \Sigma | x_n) d\theta_n, \tag{3.18}$$

where

$$E_z\left(z|\theta_n\right) = \Psi\left(\theta_n, \xi\right) \sum_{z \in \Omega} z \exp\left[z'\left(b\theta_n + A\xi\right)\right]; \tag{3.19}$$

and

$$\overline{\theta}_n = \int_{\theta_n} \theta_n h_\theta\left(\theta_n; Y_n, \xi, \gamma, \Sigma | x_n\right) d\theta_n. \tag{3.20}$$

The system of equations defined by (3.16), (3.17) and (3.18) is solved using an EM algorithm (Dempster et al., 1977) following the approach of Bock & Aitkin (1981).

3.1.3.1 Quadrature and Monte Carlo Approximations

The integrals in equations (3.16), (3.17) and (3.18) are approximated numerically using either quadrature or Monte Carlo methods. In each case, we define a set of Q D-dimensional vectors Θ_q, $q = 1, 2, ..., Q$ (which we call nodes); and for each node Θ_q we define a corresponding weight $W_q(\gamma, \Sigma)$. The marginal item response probability (3.13) is then approximated using

$$f_x(x;\xi,\gamma,\Sigma) \approx \sum_{q=1}^{Q} f_x\left(x;\xi|\Theta_q\right) W_q(\gamma,\Sigma); \tag{3.21}$$

and the marginal posterior (3.15) is approximated using

$$h_\Theta\left(\Theta_q; Y_n,\xi,\gamma,\Sigma|x_n\right) \approx \frac{f_x\left(x_n;\xi|\Theta_q\right) W_q(\gamma,\Sigma)}{\sum_{p=1}^{Q} f_x\left(x_n;\xi|\Theta_p\right) W_p(\gamma,\Sigma)} \tag{3.22}$$

for $q = 1, ..., Q$.

The EM algorithm then proceeds as follows:

1. Prepare a set of nodes and weights depending upon $\gamma^{(t)}$ and $\Sigma^{(t)}$ which are the estimates of γ and Σ at iteration t.

2. Calculate the discrete approximation of the marginal posterior density of Θ_n, given x_n at iteration t, using

$$h_\Theta\left(\Theta_q; Y_n,\xi^{(t)},\gamma^{(t)},\Sigma^{(t)}|x_n\right) = \frac{f_x\left(x_n;\xi^{(t)}|\Theta_q\right) W_q\left(\gamma^{(t)},\Sigma^{(t)}\right)}{\sum_{p=1}^{Q} f_x\left(x_n;\xi^{(t)}|\Theta_p\right) W_p\left(\gamma^{(t)},\Sigma^{(t)}\right)}, \tag{3.23}$$

   where $\xi^{(t)}$, $\gamma^{(t)}$ and $\Sigma^{(t)}$ are estimates of ξ, γ and Σ at iteration t.

3. Use the Newton-Raphson method to solve the following to produce estimates of $\xi^{(t+1)}$

$$A' \sum_{n=1}^{N} \left[x_n - \sum_{q=1}^{Q} E_z\left(z|\Theta_q\right) h_\Theta\left(\Theta_q; Y_n,\xi^{(t)},\gamma^{(t)},\Sigma^{(t)}|x_n\right)\right] = 0. \tag{3.24}$$

4. Estimate $\gamma^{(t+1)}$ and $\Sigma^{(t+1)}$, using

$$\gamma^{(t+1)} = \left(\sum_{n=1}^{N} \overline{\Theta}_n Y_n'\right)\left(\sum_{n=1}^{N} Y_n Y_n'\right)^{-1} \tag{3.25}$$

and

$$\Sigma^{(t+1)} = \frac{1}{N}\sum_{n=1}^{N}\sum_{q=1}^{Q}\left(\Theta_q - \gamma^{(t+1)}Y_n\right)\left(\Theta_q - \gamma^{(t+1)}Y_n\right)' \cdot h_{\Theta}\left(\Theta_q; Y_n, \xi^{(t)}, \gamma^{(t)}, \Sigma^{(t)}|x_n\right), \tag{3.26}$$

where

$$\overline{\Theta}_n = \sum_{q=1}^{Q}\Theta_q h_{\Theta}\left(\Theta_q; Y_n, \xi^{(t)}, \gamma^{(t)}, \Sigma^{(t)}|x_n\right). \tag{3.27}$$

5. Return to step 1.

The difference between the quadrature and Monte Carlo methods lies in the way the nodes and weights are prepared. For the quadrature case, we begin by choosing a fixed set of Q points, $(\Theta_{d1}, \Theta_{d2}, ..., \Theta_{dQ})$, for each latent dimension d and then define a set of Q^D nodes that are indexed $q = 1, ..., Q^D$ and are given by the Cartesian coordinates

$$\Theta_q = \left(\Theta_{1j_1}, \Theta_{2j_2}, \dots, \Theta_{Dj_D}\right)$$
$$\text{with} \quad j_1 = 1, \dots, Q; \quad j_2 = 1, \dots, Q; \quad j_D = 1, \dots, Q.$$

The weights are then chosen to approximate the continuous multivariate latent population density (3.12). That is,

$$W_q = K(2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\exp\left[-\frac{1}{2}\left(\Theta_q - \gamma Y_n\right)'\Sigma^{-1}\left(\Theta_q - \gamma Y_n\right)\right], \tag{3.28}$$

where K is a scaling factor to ensure that the sum of the weights is 1.

For further information on the quadrature approach to estimating the model, see Adams, Wilson, & Wang (1997); and for further information on the Monte Carlo estimation method, see Volodin & Adams (1995).

In the Monte Carlo case, the nodes are drawn at random from the standard multivariate normal distribution; and at each iteration, the nodes are rotated, using standard methods, so that they become random draws from a multivariate normal distribution with mean γY_n and variance Σ. In the Monte Carlo case, the weight for all nodes is $1/Q$.

3.1.3.1.1 Adjusted Monte Carlo The above Monte Carlo estimation procedure is improved by what we call *Adjusted Monte Carlo*. In this variation nodes are not drawn from prior multivariate normal distribution but rather from a *different* multivariate normal distribution with mean and covariance that are our best available estimates of the true posterior mean and covariance of student n. As a result weights for nodes become ratios of prior density over proxy posterior density (estimated at these nodes). We refer to the subsection Drawing Plausible Values where this approach is presented in more detail. Adjusted Monte Carlo estimation can be used when all item parameters are fixed (anchored), and is therefore suited for estimating case posteriors for drawing Plausible Values or for estimating population parameters.

3.1.3.2 Estimating Standard Errors

Computation of standard errors of parameter estimates varies by estimation method employed. This section describes the estimation procedure for the estimation methods implemented in ACER COnQuest.

3.1.3.2.1 Under Joint Maximum Likelihood Asymptotic standard errors for the parameter estimates are estimated using the observed Fisher's Information. The current implementation assumes an independence between all parameter estimates and provides estimates for unconstrained parameters only (Wright & Masters, 1982). In ACER ConQuest, this is described as *quick* errors, see the command estimate.

3.1.3.2.2 Under Quadrature and Monte Carlo Asymptotic standard errors based upon the full observed Fisher's Information are derived in Adams, Wilson, & Wu (1997).

If the observed information matrix is written as

$$I = \begin{bmatrix} I_{\xi\xi'} & I_{\beta\xi'} & I_{\sigma^2\xi'} \\ I_{\xi\beta'} & I_{\beta\beta'} & I_{\sigma^2\beta'} \\ I_{\xi\sigma^2} & I_{\beta\sigma^2} & I_{(\sigma^2)^2} \end{bmatrix}, \tag{3.29}$$

For computational efficiency, two methods are implemented to approximate the approach derived above. If *empirical* errors are requested (see the command estimate), numeric differentiation is used to estimate the complete standard error matrices.

Adams, Wilson, & Wu (1997) show that, for the unidimensional model, the components of the matrix are

$$
\begin{aligned}
I_{\xi\xi'} =& - A' \sum_{n=1}^{N} \left[\int_{\theta_n} E_z\left(zz'|\theta_n\right) h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \right.\\
&- 2 \int_{\theta_n} E_z\left(z|\theta_n\right) E_z\left(z'|\theta_n\right) h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \\
&+ \int_{\theta_n} E_z\left(z|\theta_n\right) h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \\
&\quad \left. \cdot \int_{\theta_n} E_z\left(z'|\theta_n\right) h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \right] A,
\end{aligned}
\tag{3.30}
$$

$$
I_{\beta\beta'} = \sum_{n=1}^{N} \frac{Y_n Y_n'}{\hat{\sigma}^2} \left[\frac{\hat{E}_\theta\left(\theta_n^2\right) - \hat{E}_\theta\left(\theta_n\right)^2}{\hat{\sigma}^2} - 1 \right],
\tag{3.31}
$$

$$
\begin{aligned}
I_{(\sigma^2)^2} = -\frac{N}{2\hat{\sigma}^4} + \frac{1}{4\hat{\sigma}^8} \sum_{n=1}^{N} & \left[\int_{\theta_n} \left(\theta_n - Y_n'\hat{\beta}\right)^4 h_\theta\left(\theta_n; Y_{n'}, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \right.\\
& \left. - \left(\int_{\theta_n} \left(\theta_n - Y_n'\hat{\beta}\right)^2 h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \right)^2 \right],
\end{aligned}
\tag{3.32}
$$

$$
\begin{aligned}
I_{\beta\xi'} = -A' \sum_{n=1}^{N} & \left(\int_{\theta_n} \theta_n \hat{E}_z\left(z|\theta_n\right) h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \right.\\
& \left. - \hat{E}_\theta\left[\hat{E}_z\left(z|\theta_n\right) \hat{E}_\theta\left(\theta_n\right)\right] \right) \frac{Y_n'}{\hat{\sigma}^2},
\end{aligned}
\tag{3.33}
$$

$$
\begin{aligned}
I_{\sigma^2\beta'} = -\frac{A'}{2\sigma^4} \sum_{n=1}^{N} & \left[\int_{\theta_n} \hat{E}_z\left(z|\theta_n\right) \left(\theta_n - Y_n'\hat{\beta}\right)^2 h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \right.\\
& - \int_{\theta_n} \hat{E}_z\left(z|\theta_n\right) h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \\
& \left. \cdot \int_{\theta_n} \left(\theta_n - Y_n'\hat{\beta}\right)^2 h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n\right) d\theta_n \right],
\end{aligned}
\tag{3.34}
$$

$$
\begin{aligned}
I_{\sigma^2\xi'} = -\frac{1}{2\hat{\sigma}^6}\sum_{n=1}^{N}\Bigg[\int_{\theta_n} & Y_n\left(\theta_n - Y_n'\hat{\beta}\right)^3 h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2|x_n\right)d\theta_n \\
& - \int_{\theta_n} Y_n\left(\theta_n - Y_n'\hat{\beta}\right) h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2|x_n\right)d\theta_n \\
& \cdot \int_{\theta_n}\left(\theta_n - Y_n'\hat{\beta}\right)^2 h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2|x_n\right)d\theta_n\Bigg].
\end{aligned} \tag{3.35}
$$

If *quick* errors are requested (see the command estimate), the diagonals of the complete standard error matrices are calculated and the off-diagonals are set to zero.

The estimation of asymptotic standard errors using the observed information can be very time-consuming. The matrix that is computed is of dimension $p + r + 2$ where p is the number of item parameters and r is the number of regression variables; and the computation of each element requires integration over the posterior distribution of each case. The time taken is therefore quadratic in the number of parameters and linear in the number of cases and nodes. Because the estimation of these errors can take considerable time (and memory), ACER ConQuest provides an option to compute *quick* approximations for the error variances, given by:

$$
\begin{aligned}
\operatorname{var}\left(\hat{\xi}_i\right) = \sum_{n=1}^{N}\Bigg\{\operatorname{diag}\Bigg[A'\Bigg(\int_{\theta_n} & E_z\left(zz'|\theta_n\right) h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2|x_n\right)d\theta_n \\
& - \int_{\theta_n} E_z\left(z|\theta_n\right) E_z\left(z'|\theta_n\right) h_\theta\left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2|x_n\right)d\theta_n\Bigg)A\Bigg]\Bigg\}^{-1}
\end{aligned} \tag{3.36}
$$

$$
\operatorname{var}\left(\hat{\beta}_i\right) = \hat{\sigma}^2\left(\sum_{n=1}^{N} Y_n Y_n^T\right)^{-1} \tag{3.37}
$$

$$
\operatorname{var}\left(\hat{\sigma}^2\right) = \frac{2\hat{\sigma}^4}{N} \tag{3.38}
$$

These approximations ignore all of the covariances in the parameter estimates. The approximations of the item parameters (3.36) will generally underestimate the sampling error, particularly for parameters associated with facets that have few levels for the step parameters in multicategory items. The accuracy of (3.37) and (3.38) depends upon the magnitude of the measurement error as it is reflected in the variances of the individual's posterior distributions.

Quick errors may provide a reasonable estimate under some circumstances. For example, when the parameters can be assumed to be independent, e.g., when the location constraint is cases, items are dichotomously scored, the model is unidimensional or multiple dimensions are uncorrelated. The degree to which the parameters of the model are dependant, e.g., step parameters within items in the Partial Credit Model, will reflect the departure from good quality estimates.

3.1.3.2.3 Under Markov Chain Monte Carlo The complete variance covariance matrix is calculated empirically from the retained steps in the chain.

3.1.3.3 Latent Estimation and Prediction

The marginal item response model (3.13) does not include parameters for the latent values θ_n; and hence, the estimation algorithm does not result in estimates of the latent values. ACER ConQuest provides expected a-posteriori (EAP) ability estimates and maximum likelihood ability estimates of the latent quantities.

3.1.3.3.1 Expected a-posteriori ability estimates The EAP ability estimate[1] of the latent quantity for case n is

$$\theta_n^{EAP} = \sum_{r=1}^{Q} \Theta_r h_\Theta \left(\Theta_r; Y_n, \hat{\xi}, \hat{\gamma}, \hat{\Sigma} | \mathrm{x}_n \right). \tag{3.39}$$

Variance estimates for these predictions are estimated using

$$\mathrm{var}\left(\theta_n^{EAP}\right) = \sum_{r=1}^{Q} \left(\Theta_r - \theta_n^{EAP}\right)\left(\Theta_r - \theta_n^{EAP}\right)' h_\Theta \left(\Theta_r; Y_n, \hat{\xi}, \hat{\gamma}, \hat{\Sigma} | x_n \right). \tag{3.40}$$

Where, Q, under Maximum Likelihood methods, refers to the nodes (see (3.22)), and under Markov Chain Monte Carlo methods, referes to retained estimates in the chain.

[1] ACER ConQuest uses Adjusted Monte Carlo approximations when producing PV and EAP ability estimates and variances for those ability estimates.

3.1.3.3.2 Maximum likelihood ability estimates Maximum likelihood ability estimates of the latent quantities are produced by maximising (3.6) with respect to θ_n, that is, solving the likelihood equations

$$\sum_{i\in\Omega}\left[b_{ix_{ni}}-\sum_{j=1}^{K_i}\frac{b_{ij}\exp\left(b_{ij}\theta_n+a'_{ij}\hat{\xi}\right)}{\sum_{k=1}^{K_i}\exp\left(b_{ik}\theta_n+a'_{ik}\hat{\xi}\right)}\right]=0 \tag{3.41}$$

for each case, where $\hat{\xi}$ is the vector of item parameter estimates. These equations are solved using a routine based on the Newton-Raphson method. Solving (3.41) will not produce finite estimates for cases that have responded in the lowest scoring category of each item or for cases that have responded in the highest scoring category of each item. To provide finite estimates for such cases, we add a small constant value to the scores of those cases who have responded in the lowest category, and we subtract a small constant from the scores of those cases who have responded in the highest category.[2]

3.1.3.4 Drawing Plausible Values

Plausible values are random draws from the marginal posterior (3.15) for each student. For details on the uses of plausible values, the reader is referred to Mislevy (1991) and Mislevy et al. (1992).

Unlike previously described methods for drawing plausible values (Beaton, 1987; Mislevy et al., 1992), ACER ConQuest does not assume normality of the Likelihood when approximating the marginal posterior distributions (Adams & Wu, 2007). Recall from (3.15) that the marginal posterior is given by

$$h_\theta(\theta_n;Y_n,\xi,\gamma,\Sigma|x_n)=\frac{f_x(x_n;\xi|\theta_n)f_\theta(\theta_n;Y_n,\gamma,\Sigma)}{\int_\theta f_x(x;\xi|\theta)f_\theta(\theta;Y_n,\gamma,\Sigma)d\theta}. \tag{3.42}$$

The *original* ACER ConQuest procedure for sampling plausible values begins by drawing M vector-valued random deviates[3], $\{\varphi_{mn}\}_{m=1}^{M}$ from the multivariate normal distribution, $f_\theta(\theta_n;Y_n,\gamma,\Sigma)$, for each case n. The probabilities

[2] The value of this constant can be set with the `set` command argument `zero/perfect=r`.

[3] The value M should be large. The default value in ACER ConQuest is 2000. The value of M can be set using the command `set` and the argument `p_nodes=n`, where n is the desired number of nodes.

$$p_{mn} = \frac{f_x\left(x_n;\xi|\varphi_{mn}\right)}{\sum_{j=1}^{M} f_x\left(x_n;\xi|\varphi_{jn}\right)} \tag{3.43}$$

are calculated, so that we obtain the set of pairs , $(\varphi_{mn}, p_{mn})_{m=1}^{M}$, which is used as an approximation of the posterior density (3.42). We then draw plausible values from this sampled density (with replacement).

The above method was employed by previous version of ConQuest. This approach was appealing, as it was mimicking the logical chain of events: we start at the prior distribution of the n-th student, and then evaluate the likelihood that takes the student to their posterior distribution. Although theoretically sound, in practice this approach may lead to some unexpected results. When posterior moments are substantially different from prior moments, this approach samples from a distribution that is too far away from the distribution that we are estimating. As a result, many rotated nodes will be too far from the mode of the likelihood function and will have small relative weights and few extreme nodes will have large weights which will dominate the sampled density, making for a less accurate approximation of the true posterior density than may be desired. These issues are more pronounced in higher dimensions.

The *Adjusted* Monte Carlo method was implement in ACER ConQuest to improve the sampling of plausible values. It begins by drawing M [4] vector-valued random deviates, $\{\varphi_{mn}\}_{m=1}^{M}$ from a *different* multivariate normal distribution, $g_\theta(\theta_n;\mu_n,\Sigma_n)$, for each case n. Here μ_n is an estimate of the true posterior mean of student n and Σ_n is an estimate of the true posterior covariance matrix of student n (for computational speed instead of using individual posterior covariances Σ_n we tend to use average posterior covariances matrix Σ_{post}). Corresponding probabilities for the sampled density are then calculated as

$$p_{mn} = \frac{f_x\left(x_n;\xi|\varphi_{mn}\right)\dfrac{f_\theta(\varphi_{mn};Y_n,\gamma,\Sigma)}{g_\theta(\varphi_{mn};\mu_n,\Sigma_n)}}{\sum_{j=1}^{M} f_x\left(x_n;\xi|\varphi_{jn}\right)\dfrac{f_\theta(\varphi_{jn};Y_n,\gamma,\Sigma)}{g_\theta(\varphi_{jn};\mu_n,\Sigma_n)}} \tag{3.44}$$

so that we obtain the set of pairs, $(\varphi_{mn}, p_{mn})_{m=1}^{M}$, which is used as an approximation of the posterior density (3.42). We then draw plausible values from this sampled density (with replacement).

[4]The value M should be large. The default value in ACER ConQuest is 2000. The value of M can be set using the command `set` and the argument `p_nodes=n`, where n is the desired number of nodes.

The adjusted method is based on the *Importance Sampling* technique and it provides far more accurate sampling densities, particularly in higher dimensions.

3.1.4 Computing Thresholds

One important representation of the difficulty of items is given by the so-called Thurstonian thresholds. ACER ConQuest computes Thurstonian thresholds for items, provided that the items do not contain unused categories and that the items do not use ordered partition scoring.

Suppose an item i has $K_i + 1$ categories and the scores for those categories are $0, 1, ..., K_i$, then that item will have K_i Thurstonian thresholds labelled Γ_k $(k = 1, ..., K_i)$. The threshold Γ_k gives the location on the latent variable at which the probability of achieving a score of k or more is 0.5. The formal definition of Γ_k is the value of θ that satisfies the condition

$$\sum_{j=k}^{K_i} \frac{\exp\left(b_{ij}\theta + a_{ij}^T\hat{\xi}\right)}{\sum_{t=1}^{K_i}\exp\left(b_{it}\theta + a_{it}^T\hat{\xi}\right)} = 0.5. \tag{3.45}$$

ACER ConQuest computes the thresholds to display in tables 5 and 6 of the `show` command using a simple binary-chop searching algorithm.

3.1.5 Separation Reliability

For the set of parameters associated with each term in a model, ACER ConQuest computes a separation reliability index. This reliability is an index of the equality of the parameters. A test of significance is provided by an accompanying chi-squared value.

If $\delta_1, \delta_2, ..., \delta_T$ is the set of parameters associated with a term in the model $\hat{\delta}_1, \hat{\delta}_2, ..., \hat{\delta}_T$ are the estimated values of those parameters, $\hat{\tau}_1, \hat{\tau}_2, ..., \hat{\tau}_T$ are the estimated error variances for the parameter estimates, and $\dot{\delta}$ is the mean of the estimated parameters, then the variance of the parameter estimates for the term is:

$$s = \frac{1}{T-1}\sum_{i=1}^{T}\left(\hat{\delta}_i - \dot{\delta}\right)^2$$

The separation reliability is then defined as

$$R = \frac{s - \frac{1}{T}\sum_{i=1}^{T}\hat{\tau}_i}{s}$$

and the chi-squared value as

$$X = \sum_{i=1}^{T}\frac{\hat{\delta}_i^2}{\hat{\tau}_i}$$

3.1.6 Fit Testing

ACER ConQuest produces a fit statistic for every estimated parameter. The statistics that are used were derived by Wu (1997) and are based on those presented by Wright & Masters (1982). The Wright and Masters statistics were extended by Wu in two ways. First, they were extended for application to a more generalised model, providing the fit at the level of the parameter rather than at the level of the 'item'. Second, the Wright and Masters statistics were developed for use with unconditional maximum likelihood estimates, and so they had to be extended for use with marginal maximum likelihood estimates.

If we let A_p be the p-th column of the design matrix A, the fit statistic is based upon the standardised residual

$$z_{np}(\theta_n) = \frac{\left(A'_p x_n - E_{np}\right)}{\sqrt{V_{np}}},$$

where $A'_p x_n$ is the contribution of person n to the sufficient statistic for parameter p, and E_{np} and V_{np} are, respectively, the conditional expectation and the variance of $A'_p x_n$.

To construct an unweighted fit statistic, the square of this residual is averaged over the cases and then integrated over posterior ability distributions so that we obtain

$$Fit_{out,p} = \int_{\theta_1}\int_{\theta_2}\ldots\int_{\theta_N}\left[\frac{1}{N}\sum_{n=1}^{N}\hat{z}_{np}^2(\theta_n)\right]\prod_{n=1}^{N}h_\theta\left(\theta_n;Y_n,\hat{\xi},\hat{\beta},\hat{\sigma}^2|x_n\right)d\theta_N d\theta_{N-1}\ldots d\theta_1. \tag{3.46}$$

For the weighted fit, a weighted average of the squared residuals is used as follows:

$$Fit_{in,p} = \int_{\theta_1} \int_{\theta_2} \cdots \int_{\theta_N} \left[\frac{\sum_{n=1}^{N} \hat{z}_{np}^2 (\theta_n) V_{np} (\theta_n)}{\sum_{n=1}^{N} V_{np} (\theta_n)} \right] \prod_{n=1}^{N} h_\theta \left(\theta_n; Y_n, \hat{\xi}, \hat{\beta}, \hat{\sigma}^2 | x_n \right) d\theta_N d\theta_{N-1} \ldots d\theta_1. \tag{3.47}$$

In ACER ConQuest, the Monte Carlo method is used to approximate the integrals in equations (3.46) and (3.47). Wu (1997) has shown that the statistics produced by (3.46) and (3.47)[5] have approximate scaled chi-squared distributions. These statistics are transformed to approximate normal deviates using the Wilson-Hilferty transformations:

$$t_{\text{ out },p} = \frac{\left(Fit_{\text{out},p}^{\frac{1}{3}} - 1 + \frac{2}{(9rN)} \right)}{\left(\frac{2}{9rN} \right)^{\frac{1}{2}}}$$

and

$$t_{\text{in},p} = \left[Fit_{\text{in},p}^{\frac{1}{3}} - 1 \right] \times \frac{3}{\sqrt{\operatorname{Var}\left(Fit_{\text{in},p} \right)}} + \frac{\sqrt{\operatorname{Var}\left(Fit_{\text{in},p} \right)}}{3},$$

where r is the number of draws used in the Monte Carlo approximation of (3.46) and

$$\operatorname{Var}\left(Fit_{\text{in},p} \right) = \left[\frac{1}{\sum_n V_{np}} \right]^2 \left[\sum_n \left(E\left(\left(\mathrm{A}_p' \mathrm{X}_n - E_{np} \right)^4 \right) - V_{np}^2 \right) \right].$$

The derivation and justification for these transformations is given in Wu (1997).

3.1.6.1 Weights

During fit analysis, the contribution of the cases, n, to the sufficient statistic for parameter p, may be simple, raw statistics, or they may be weighted. In the descriptions above, simple raw statistics are described for the sake of simplicity only. The caseweight (default is all cases are equally weighted to 1) is used to weight the fit statistics.

[5] In ACER ConQuest, the number of nodes used to approximate the integrals in (3.46) and (3.47) is governed by the `set` command argument `f_nodes=n`, and the number of random draws used by the Monte Carlo integration method is governed by `fitdraws=n`. The default value of `f_nodes` is 2000, and the default value of `fitdraws` is 1.

3.1.7 Design Matrices

The two matrices, A and B, that are used in (3.6) define the specific form of the item response model that is to be fit. We call A the design matrix and B the scoring matrix. Detailed descriptions of how specific forms of these matrices result in various Rasch models is provided in Adams & Wilson (1996) and Adams, Wilson, & Wang (1997).

3.1.7.1 Design Matrices and Different Rasch Models

The number of rows in both the scoring and design matrices is equal to the total number of response categories for all generalised items. For example, to fit the simple logistic model to data collected from a set of 10 dichotomously scored items will require scoring and design matrices with 2 rows for each item, a total of 20. The design matrix will have one column for each item parameter, and the scoring matrix will have one column for each dimension.

Figure 3.1 illustrates the design and scoring matrices for this example. The 20 rows in these matrices are sequenced so that the first row refers to the first category of item 1, the second row refers to the second category of item 1, the third row refers to the first category of item 2, the fourth row refers to the second category of item 2, and so on.

In Figure 3.1, you will note that all of the rows that correspond to the first category in each item contain only zeros. This is because we routinely use the first response category in an item as the reference category. Adams & Wilson (1996) show how the substitution of these particular design and score matrices into (3.3) will result in the simple logistic model[6].

For polytomous data, a model such as Masters' partial credit model (Masters, 1982) can be used. Suppose, for example, that we wish to fit a partial credit model to three items, each with four response categories. This can be achieved with the design and score matrices shown in Figure 3.2.

The models in both Figure 3.1 and Figure 3.2 are unidimensional, that is, they assume that all of the items are indicators for a single latent variable. Both can easily be altered to become multidimensional models through the re-specification of the scoring matrices. In Figure 3.3, we show two scoring matrices that, if used as alternatives to the scoring matrices in Figures 3.1 and 3.2, would result in two- and three-dimensional models respectively.

[6]Note, however, that a model using the matrices in Figure 3.1 will only be identified if the mean of the latent variable, μ, is constrained to be zero.

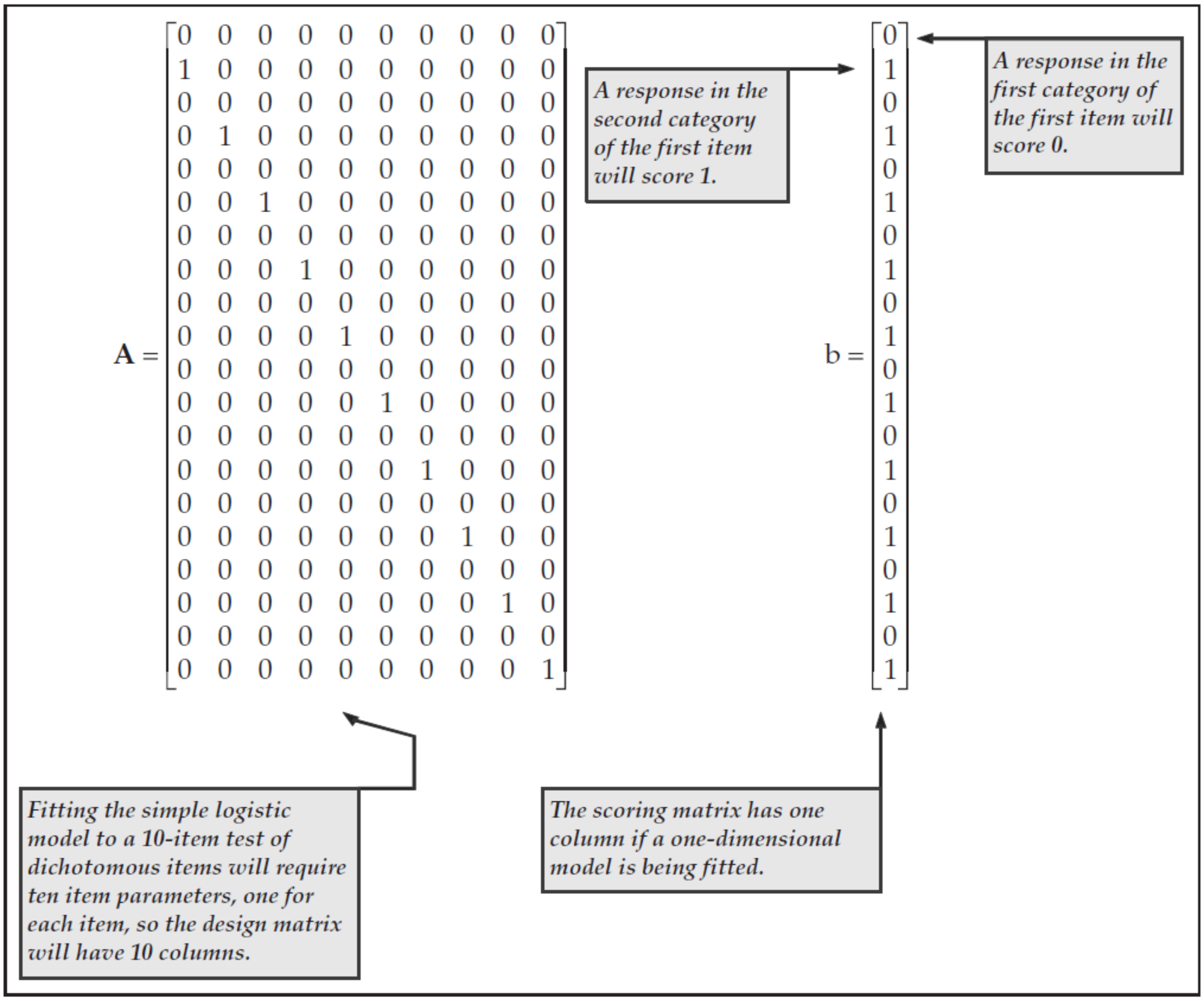


Figure 3.1: Design and Scoring Matrices for a Simple Logistic Model Fitted to Data Collected with 10 Dichotomous Items

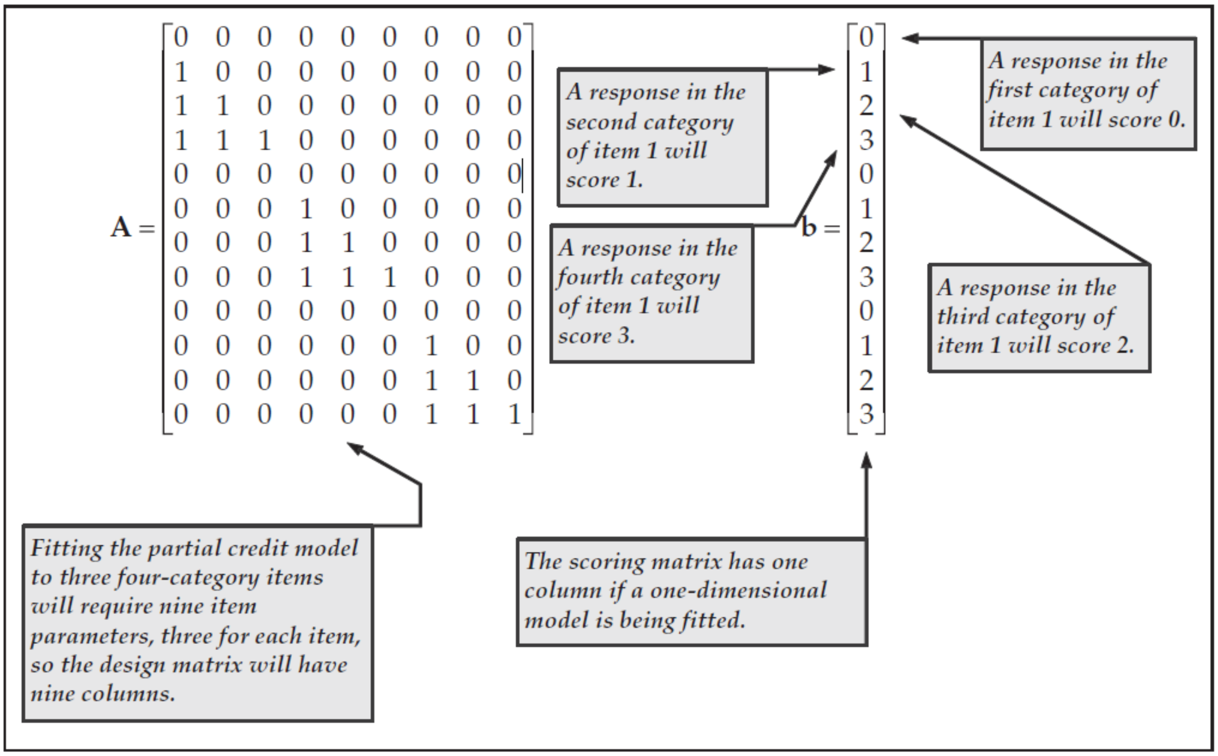


Figure 3.2: Design and Scoring Matrices for a Partial Credit Model Fitted to Data Collected with Three Polytomous Items

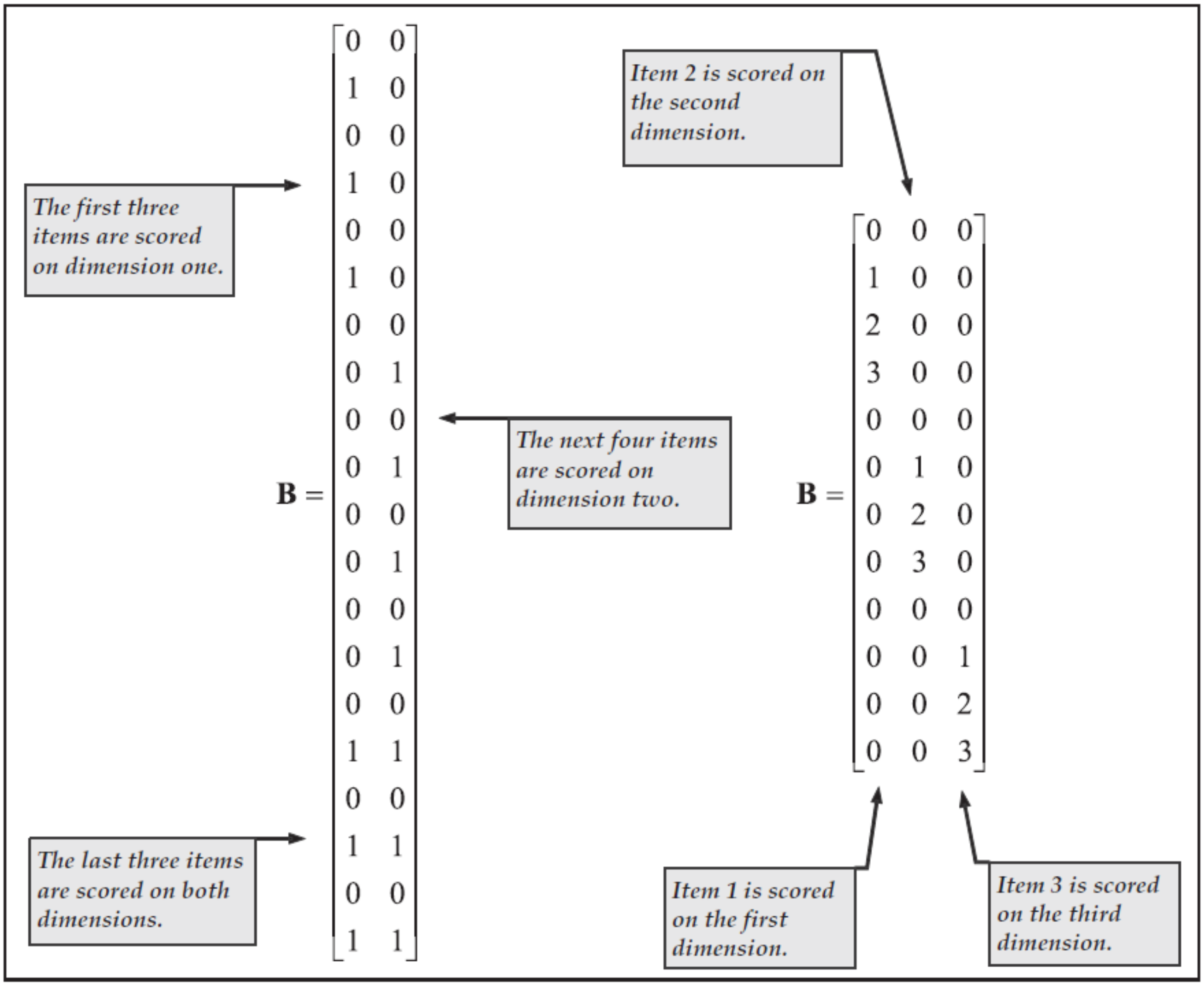


Figure 3.3: Multidimensional Scoring Matrices for Dichotomous and Polytomous Data

As a final example, Figure 3.4 shows the design and scoring matrices that can be used for multifaceted data. Consider an example of a rating context in which students' work is rated against two criteria by two raters and that each rating uses a three-point scale, scored 0, 1, 2. To fit the generalised Rasch model to such data, the combination of the two criteria and the two raters are regarded as four generalised items. Assuming the generalised items are defined in the sequence criterion 1, rater 1; criterion 1, rater 2; criterion 2, rater 1; criterion 2, rater 2; then the matrices in Figure 3.4 fit a two-faceted Rasch model that posits a unique rating structure for each generalised item.

Both matrices have 12 rows because there are four generalised items, each with three response categories. The following sequence for the generalised items has been assumed: criterion 1, rater 1; criterion 1, rater 2; criterion 2, rater 1; criterion 2, rater 2.

$$
\mathbf{A} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
2 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
-2 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
-2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1
\end{bmatrix}
\qquad
\mathbf{b} = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 0 \\ 1 \\ 2 \\ 0 \\ 1 \\ 2 \\ 0 \\ 1 \\ 2 \end{bmatrix}
$$

This parameter is the criterion difficulty parameter.

This parameter is the rater severity parameter.

These columns are for the step parameters.

The scoring matrix has one column because a one-dimensional model is being fitted.

Figure 3.4: Design and Score Matrices for a Multifaceted Polytomous Model

The form of the parameterisation that has been used in Figure 3.4 follows that of Andrich (1978). Ten parameters are used. The first column provides a criterion difficulty parameter, the second provides a rater severity parameter, and columns three to ten are the step parameters.

Even though the data are collected using two criteria, the item response model includes a single criterion difficulty parameter: the design matrix has set the difficulty of the second criterion to be the negative of the difficulty of the first criterion.

This kind of constraint is often applied to identify the item response model and is equivalent to setting the mean of the criteria parameters to zero.[7] Similarly, the model in Figure 3.4 uses a single rater severity parameter, with the severity of the second rater set to be the negative of the severity of the first rater.[8]

The structure of these design and score matrices can, perhaps, be best understood by noting from (3.5) that

$$\log\left(\frac{\Pr\left(X_{ij}=1;A,B,\xi|\theta\right)}{\Pr\left(X_{ij-1}=1;A,B,\xi|\theta\right)}\right)=\left(b_{ij}-b_{ij-1}\right)\theta+\left(a'_{ij}-a'_{ij-1}\right)\xi. \tag{3.48}$$

3.1.7.2 The Structure of ACER ConQuest Design Matrices

ACER ConQuest can import user-defined design matrices, or it can generate its own design matrices by drawing upon the command code that is used to specify a model. Section 2.10, Importing Design Matrices provides two sample analyses that use user-defined design matrices. For many models, however, ACER ConQuest command code can be used so that appropriate design matrices are automatically generated.

The score matrix cannot be imported, but the ACER ConQuest `score` command can be used to generate score matrices. The relationship between the `score` command and the score matrix is direct and need not be described here.

The design matrix is generated from the ACER ConQuest `model` statement, the syntax of which is described in the command reference (Chapter 4, ACER ConQuest Command Reference). The full details of how the design matrix is generated are beyond the scope of this manual; however, it is useful to note how the basic structure of the design matrix is determined.

In the `model` statement, four types of terms can used: terms that involve a single variable, terms that involve the product of two or more variables, the term `step`, and terms that involve the product of `step` and other variables.

[7]As an alternative to setting the mean of a set of parameters to zero, it may be possible to identify the model by setting the mean of the latent variable to zero. This is what would have been required to identify the models in Figures 3.1 and 3.2.

[8]In the multifaceted case, setting the mean of the latent variable to zero removes the requirement of setting the mean of one set of parameters to zero only.

ACER ConQuest must first determine the number of rows in the score and design matrices. It does so by noting all of the different variables used in the `model` statement and then examining the data to identify all possible combinations of the levels of the variables. Each possible combination is called a generalised item. Each valid response category for a generalised item constitutes one row in the score matrix and one row in the design matrix. The valid response categories are all categories between the lowest and highest category found in the data for the generalised item.[9]

A set of parameters (columns of the design matrix) is then generated for each term in the model. If the term involves a single variable, then the number of parameters generated is one less than the number of levels in that variable.[10] If the term involves the product of two or more variables, then the number of parameters generated is $(K_1-1)(K_2-1)(K_3-1)...$, where K_i is the number of levels in the i-th variable used in the term. If the term is `step`, then the number of parameters generated is two less than the maximum of the number of categories in all of the generalised items. If the term involves `step` and other variables, the number of parameters generated is $\sum_t (L_t-2)$, where the summation is over all of the combinations of levels of the variables that are in the term (of course, excluding `step`) and L_t is the maximum of the number of categories in those generalised items that include the t-th combination of variables.

ACER ConQuest then proceeds to construct a design matrix that is based upon the Andrich (1978) parameterisation of polytomous Rasch models. If an imported matrix is used as a replacement for the generated matrix, then each row of the imported matrix must refer to the same category and generalised item as those to which the corresponding row of the generated matrix refers. No constraint is placed on the number of columns (parameters) in the imported matrix.

3.1.8 Item Analysis Statistics

The itanal command displays a variety of traditional and model-based item statistics, most of which are self-explanatory. To assist in their discussion, however, it is important to define the ACER ConQuest concept of *raw score*. The raw score for a case is the sum of the scores achieved by a case divided by the maximum possible score that the case could

[9]The lowest and highest categories are determined as follows. ACER ConQuest identifies all valid codes (after recoding) and then sorts those codes (using an alphanumeric sort) to find the lowest and highest category.

[10]When the `set` command argument `lconstraints=cases` is used and the term is the first term, the number of parameters generated is equal to the number of categories of the variable in the term.

have achieved. More formally, if we let Ξ_n be the set of generalised items to which case n responded, then the raw score s_n for case n is defined as

$$s_n = \frac{\sum_{i \in \Xi} b^*_{ix_{ni}}}{\sum_{i \in \Xi} b^*_{iK_i}}, \tag{3.49}$$

where $b^*_{ix_{ni}}$ is the sum across the dimensions of the score that has been assigned to category x_{ni} of item i.

3.1.8.1 Discrimination

The discrimination index that is printed for each item (see, for example, Figure 2.31) is the product moment correlation between the case scores on this item, $b^*_{ix_{ni}}$, and the corresponding case raw scores, s_n. Only those cases who responded to the item are included in the calculation.

3.1.8.2 Point Biserial

For each response category of an item, a point-biserial correlation and t-statistic are computed. To compute the point biserial for category k on item i, a dummy variable y_{ikn} is constructed so that

$$y_{ikn} = \begin{cases} 1 & \text{if case } n \text{ has responded in category } k \text{ of item } i \\ 0 & \text{otherwise} \end{cases}. \tag{3.50}$$

The point biserial is then the correlation between the set of values y_{ikn} and the corresponding case raw scores s_n. Only those cases who responded to the item are included in the calculation.

If the data set is complete and the items are dichotomously scored, then the discrimination index and the point biserial for the category that is scored `1` will be equal. If the data set is incomplete, this does not hold.

The t-statistic provides a significance test for the point biserial. The degrees of freedom for the statistic are two less than the total number of students who responded to the item. Since this will normally be greater than 30, the t-statistic can be treated as a normal deviate.

3.1.8.3 Summary Statistics

The mean, variance, skewness and kurtosis statistics (see, for example, Figure 2.13) that are reported at the end of an `itanal` run are scaled to a metric that assumes that every case responded once to every item. The Cronbach's alpha coefficient of reliability (which is equal to KR-20 when all items are dichotomously scored) and the standard error of measurement also assume that every case responded once to every item. Further, they are not reported if more than 10% of the response data is missing (compare, for example, Figure 2.13 with Figure 2.32).

3.1.8.4 Item Means

The item analysis reports, for each generalised item, the item observed mean score, item expected mean score, and the item adjusted mean score. The item observed mean score is the simple mean of the scores: that is, the sum of the scored responses to the item divided by the number of responses to the item for the cases in the group for which the report is being provided (the default group is all cases, see command group). The item expected mean score is the average of the expected scores for the cases that responded to the item in the group for which the report is being provided. The observed and expected means scores will be very close when the model is estimated for the group that is being reported. If the report is being provided for a sub-group of cases then the observed and expected values may differ as a function of differential item functioning.

The item *adjusted* mean score is the item expected mean score for all students in the current group. This differs from the item expected mean score which is computed only for those cases that responded to the item.
The two values will typically be different when items are not randomly allocated to students, the canonical example is adaptive testing.

The item expected mean score and adjusted means scores are computed using the model estimated parameter estimates and case latent proficiency estimates.

3.1.8.5 Weights

During item analysis, the scores and counts may be simple, raw statistics, or they may be weighted. In the descriptions above, simple raw statistics are described for the sake of simplicity only. By default, the caseweight (default is all cases are equally weighted to 1) is used to weight statistics within the item analysis. The user may specify the weighting to be used by using the option weight to the command itanal and a full description is given in the documentation for the itanal command.

Chapter 4

ACER ConQuest Command Reference

This chapter contains general information about the syntax of ACER ConQuest command statements followed by an alphabetical reference of ACER ConQuest commands.

All ACER ConQuest commands can be accessed through a command line interface. In addition, the majority of commands with their options can be accessed through the graphical user interface. The graphical user interface is only available for Windows operating systems. This document describes the syntax for the command line interface and the graphical user interface accessibility of each of these commands.

4.1 Command Statement Syntax

An ACER ConQuest statement can consist of between one and five components: a command, arguments, options, an *outdirection* and an *indirection*. The general syntax of an ACER ConQuest statement is as follows:

```
Command Arguments ! Options >> Outdirection << Indirection;
```

The first text in a statement must be a command. The command can be followed by an argument with a space used as a separator. Some commands have optional arguments; others require an argument. An exclamation mark (`!`) separates arguments from options; if there is no argument, the exclamation mark can separate a command from an option.

Where there is more than one legal option they are provided as a comma separated list of options. The value passed into the option (on the right hand side of the equals sign) can be a string literal or a variable. For options that take a boolean value, this can be represented as:

1. the string `true` or `false`
2. the string `yes` or `no`
3. the number `1` or `0`
4. a matrix variable, of which the first element evaluates to `1` or `0`

The characters `<<` or `>>` separate a file redirection (either an *indirection* or an *outdirection*) from the preceding elements of the statement.

ACER ConQuest syntax has the following additional features:

1. A statement must be terminated with a semi-colon (`;`). A statement can continue over many lines or many statements can appear on a single line.
2. A statement can be 16 000 characters in length and can cover any number of lines on the screen or in a command file. No continuation character is required.
3. Comments are placed between `/*` and `*/`. They can appear anywhere in a command file, and their length is unlimited. Comments cannot be nested inside another comment.
4. The command language is not case sensitive. All commands and matrix objects names are folded to lower case. Values in files are case sensitive, eg *arguments* for the commands `codes`, `keepcases` and `dropcases`.
5. The order in which command statements can be entered into ACER ConQuest is not fixed. There are, however, logical constraints on the ordering. For example, `show` statements cannot precede the `estimate` statement, which in turn cannot precede the `model`, `format` or `datafile` statements, all three of which must be provided before ACER ConQuest can analyse a data set.
6. Any text file that you want ACER ConQuest to read must be an UTF-8 ASCII text file.
7. User-provided variable names must begin with an alphabetic character and must be made up of alphabetic characters or digits. Spaces are not allowed in variable names and some characters and names are reserved for ACER ConQuest use (see List of illegal characters and words for variable names at the end of this document).

8. All commands, as well as arguments and options that consist of ACER ConQuest reserved words, can be abbreviated to their shortest unambiguous root. For example, the following are all valid:

```
caseweight, caseweigh, caseweig, casewei, casewe, casew,
case, cas, ca
codes, code, cod, co
converge=, converg=, conver=, conve=, conv=, con=, co=
datafile, datafil, datafi, dataf, data, dat, da, d
estimate, estimat, estima, estim, esti, est, es
export, expor, expo, exp, ex
```

4.1.1 Example Statements

```
codes 0,1,2;
```

`codes` is the *command*, and the *argument* is `0,1,2`.

```
format responses 11-20 ! rater(2),essay(5);
```

`format` is the *command*, `responses 11-20` is the *argument*, and `rater(2)` and `essay(5)` are the *options*.

```
show ! cases=eap >> file.out;
```

`show` is the *command*, there is no argument, `cases=eap` is the *option*, and `>> file.out` is the *redirection*.

4.2 Tokens and the Lexical Preprocessor

4.2.1 Lexical Preprocessor

Before executing a set of commands (e.g., a syntax file) the set of commands is passed through a *lexical preprocessor*. The *lexical preprocessor* handles the commands `let`, `execute`, `dofor`, `doif`, `enddo`, `else` and `endif`. The *lexical preprocessor* also resolves *tokens*.

4.2.2 Tokens

A *token* is an alphanumeric string that is set by a `let` command. For example:

```
let nitems=10;
let path=C:/mywork;
```

After it has been defined, a *token* is referenced by enclosing its name between `%` characters (e.g., `%path%`). When a *token* reference is detected in a set of commands it is replaced by the value it represents. *Tokens* can be used in any context.

A *token* is also set for each iteration of a `dofor` loop. This *token* is referred to as the looping variable and cannot be defined prior to the `dofor` loop.

The *tokens* `version`, `date`, `platform`, `process`, `tempdir` and `interface` are created automatically, and are available (e.g., `%date%`) at any time. The `process` token is a unique integer associated with the current ConQuest session.

The preprocessor will literally process *tokens*, and so the user should ensure they make the distinction between *literal strings* and *strings that should be parsed*. For example, given `let x = 10-1;` see the distinction between:

- a *string that should be parsed:* `print %x%;` output: `9`
- a *literal string:* `print "%x%";` output: `10-1`

4.2.2.1 Example Statements

```
let n=10;
generate ! nitems=%n%;
```

Assigns the string `10` to the *token* `n`, so that when the subsequent `generate` command is executed, the string `%n%` is replaced by the string `10`.

```
dofor x=M,F;
Plot icc ! group=gender; keep=%x%;
enddo;
```

Produces plots for students with gender value `M` and then gender value `F`. `x` is the looping variable.

4.3 Matrix Variables

A matrix variable is a matrix value that is set through a compute command or created, when requested, by an ACER ConQuest procedure. The command `compute` can be omitted, so long as the left hand side is not a protected word.

A variable can be used in a `compute` command, or produced by a `compute` command. A variable can also be used as input in a number of procedures. A variable can be converted to a *token*, for use in the command language, by the `let` command. A variable can also be directly used as a component of the command language. Only the first element `(1,1)` of a matrix variable is used, for example when used as a value for an option to a command.

A number of analysis routines can be directed to save their results as variables – typically sets of matrices. These variables can be subsequently manipulated, saved or plotted.

The matrix variable `version` is automatically created and is an integer expression of the ConQuest version that is running.

4.3.1 Example Statements

```
x=10;
set seed = x;
```

Assigns the value 10 to the variable `x` and then sets the seed to the value in the first element of `x`. Note that the command `compute` was legally omitted.

```
compute n=10;
compute m=n+2;
print m;
```

Assigns the value 10 to the variable `n`, adds 2 to `n` and produces `m` and then prints the value of `m` (i.e., 12).

```
n=fillmatrix(2,2,0);
n[1,1]=10;
n[2,1]=-23;
n[1,2]=0.4;
n[2,2]=1;
compute m=inv(n);
print n,m;
```

n is created as a 2 by 2 matrix which is populated with the four values, the inverse of n is then calculated and saved as m, finally the values of n and m are printed.

```
estimate ! matrixout=r;
compute fit=r_itemfit[,3];
plot r_itemparams fit;
print r_estimatecovariances ! filetype=xlsx >> covariances.xlsx;
```

Estimation is undertaken and a set of matrices containing results is created (see estimate command). The item parameter estimates are plotted against the unweighted mean square fit statistics and then the parameter estimate covariance matrix is saved as an excel file.

4.4 Loops and Conditional Execution

Loops and conditional execution of control code can be implemented through the use of the for, while, if, dofor and doif commands.

dofor (in association with enddo) and doif (in association with endif and else) are dealt with by the preprocessor and are typically used to loop over token values or conditionally execute code based upon tokens.

4.4.1 Example Statements

```
doif %x%==M;
print "Plot for Males";
plot icc ! group=gender; keep=M;
else;
print "Plot for Females";
plot icc ! group=gender; keep=F;
endif;
```

Produces plots for students with gender value M or F depending upon the value of the token %x%.

The for, while, and if commands are not dealt with by the preprocessor. They are ACER ConQuest commands that are typically used to manipulate matrix variables and their contents.

4.5 Explicit and Implicit Variables

When ACER ConQuest reads data from a file identified with the `datafile` command with a structure as described by the `format` command variables of two different types can be generated. Explicit variables are variables that are listed in a `format` statement. Implicit variables are variables that are associated with specific columns in the data file referred to in the format statements as responses. For a full illustration of these two classes of variables see the `format` command.

4.6 Using ACER ConQuest Commands

ACER ConQuest is available with both a graphical user interface (GUI) and a command line, or console, interface. The ACER ConQuest command statement syntax used by the GUI and the console versions is identical. In general, the console version runs faster than the GUI version, but the GUI version is more user friendly. GUI version and console version system files are fully compatible.

4.6.1 Entering Statements via the Console Interface

When the console version of ACER ConQuest is started, the "less than" character (<) is displayed. This is the ACER ConQuest prompt. When the ACER ConQuest prompt is displayed, any appropriate ACER ConQuest statement can be entered.

As with any command line interface, ACER ConQuest attempts to execute the statement when you press the Enter key. If you have not yet entered a semi-colon (`;`) to indicate the end of the statement, the ACER ConQuest prompt changes to a plus sign (`+`) to indicate that the statement is continuing on a new line. On many occasions, a file containing a set of ACER ConQuest statements (i.e., an ACER ConQuest command file) will be prepared with a text editor, and you will want ACER ConQuest to run the set of statements that are in the file. If we suppose the ACER ConQuest command file is called `myfile.cqc`, then the statements in the file can be executed in two ways.

1. In the first method, start ACER ConQuest and then type, at the ACER ConQuest prompt, the statement `submit myfile.cqc;`

2. A second method, which will work when running from a command-line interpreter (cmd on Windows, or Terminal on Mac), is to provide the command file as a command line argument. You launch ACER ConQuest and provide the command file in one step using[1]

   Windows x64: `ConQuestx64console myfile.cqc;`

   Mac: `ConQuest myfile.cqc`

With either method, after you press the Enter key, ACER ConQuest will proceed to execute each statement in the file. As statements are executed, they will be echoed on the screen. If you have requested displays of the analysis results and have not redirected them to a file, they will be displayed on the screen.

4.6.2 Entering Commands via the GUI Interface

Once you have launched the GUI interface (double-click on ConQuest4GUI.exe), you can type command statements or open a command file in the GUI input window and then select

`Run`→`Run All`.

In addition, the GUI interface has menu selections that will build and execute ACER ConQuest command statements. Access to the commands with the GUI is described seperately for each command in the Commands section below.

4.7 Commands

The remainder of this document describes the ConQuest commands. The arguments or options that are listed below the commands are reserved words when used with that command.

4.7.1 about

Reports information about this installation of ACER ConQuest. Includes the version, build, and licencing information.

[1]Note that both of these examples assume you have navigated to the path of your ACER ConQuest install and that your command file is in the same location.

4.7.1.1 Argument

This command does not have an argument.

4.7.1.2 Options

This command does not have a options.

4.7.1.3 Redirection

No redirection.

4.7.1.4 Examples

`about;`

Returns the following:

```
Developed by
    Australian Council for Educational Research
    University of California,Berkeley

Your key: abc-123-1234
Expires: 1 November 2024

Professional Build:  May 25 2025
Version: 5.33.7

Programmers
    Ray Adams, Margaret Wu, Dan Cloney, Greg Macaskill, Alla Berezner, Sam Haldane, Xi
```

4.7.1.5 GUI Access

Help→About this Program

4.7.1.6 Notes

None.

4.7.2 banddefine

Defines the upper and lower bounds, and names of achievement or proficiency bands for latent scales. The proficiency bands are displayed on kidmaps.

4.7.2.1 Argument

This command does not have an argument.

4.7.2.2 Options

`dimension =`*`n`*
n is the number for the dimension of the latent model that that band/s relate to. The default is 1, i.e. the first dimension.

`upper =`*`n`*
n is the upper bound of the band in logits. The default is system missing.

`lower =`*`n`*
n is the lower bound of the band in logits. The default is system missing.

`label =`*`string`*
`string` is the label for the band in quotes. The default is `" "`.

4.7.2.3 Redirection

no redirection.

4.7.2.4 Examples

```
Banddefine ! label = "L0 (critical)", upper = -2.133, lower = -100;
```
Defines a band called "L0 (critical)" for the first dimension. Note the lower bound is set at a large negative value to ensure it encompasses all of the bottom-end of the estimated scale.

4.7.2.5 GUI Access

None.

4.7.2.6 Notes

1. An error will be produced if bands are requested to overlap. Bands are reported on KIDMAPS when the kidmap command is called in conjunction with the option `format=samoa`.
2. Where there is a tie, e.g., a student score is on a band boundary (e.g., the lower bound of Level 5 is 2.1 and so is the upper bound of Level 4) the student is allocated to the lower band (e.g., in this case Level 4).

4.7.3 build

Build design matrices for current model specification without proceeding to estimation.

4.7.3.1 Argument

This command does not have an argument.

4.7.3.2 Options

This command does not have options.

4.7.3.3 GUI Access

Access to this command through the GUI is not available.

4.7.3.4 Example

```
data isa.csv!filetype=csv,
              response=response,
              pid=personid,
              keeps=itemid y4 y5 y6 y7 y8 y9 y10 gender,
```

```
              keepswidth=10;
model itemid;
regression y4 y5 y6 y7 y8 y9 y10 gender;
build;                                /* build a standard design matrix */
export amatrix!filetype=matrix>>x;    /* save design as a matrix object */
```

4.7.4 caseweight

Specifies an explicit variable that is to be used as a case weight.

4.7.4.1 Argument

An explicit variable that is used as a case weight.

4.7.4.2 Options

This command does not have options.

4.7.4.3 Redirection

Redirection is not applicable to this command.

4.7.4.4 Examples

```
caseweight pweight ;
```

The explicit variable pweight contains the weight for each case.

```
caseweight;
```

No case weights are used.

4.7.4.5 GUI Access

`Command→Case Weight`
Select the Case Weight menu item. The radio button allows case weighting to be toggled. If cases are to be weighted then a variable must be selected from the candidate list of explicit variables.

4.7.4.6 Notes

1. The caseweight statement stays in effect until it is replaced with another caseweight statement or until a reset statement is issued. If you have run a model with case weights and then want to remove the case weights from the model, the simplest approach is to issue a caseweight statement with no arguments.
2. A variable that will be a case weight must be listed in the format as an explicit variable.
3. Case weighting is applied to item response model estimation, but not to traditional or descriptive statistics.

4.7.5 categorise

Sets up a dummy code for a categorical regression variable.

4.7.5.1 Argument

var(v1:v2:…:vN) or *var(n)*

When ***var*** is a categorical variable and ***n*** is an integer greater than 1, then the levels of the categorical varible are asumed to be a sequence of integers from 1 to ***n***.

When ***var*** is a categorical variable and ***v1:v2:…:vN*** is a list of values that give levels of the categorical variable.

In both cases, by default a set of N-1 new dichotomously coded variables are created to represent the N categories of the original variable.

If the values that represent levels of the categorical variables contains leading or trailing spaces then the values will need to be enclosed in quotes. If observed levels are omitted from the list they are treated as missing data.

When *var* is specified as a regression variable it will be replaced by the N-1 variables *var_1*, *var_2*, *var_(N-1)*.

The variables *var_1*, *var_2*, *var_(N-1)* cannot be accessed directly by any command.

When matching variable levels with data, two types of matches are possible. EXACT matches occur when a record within the variable is compared to categorise level value using an exact string match including leading and trailing blank characters. The alternative is a TRIM match that first trims leading and trailing spaces from both record within the variable and the categorise level.

4.7.5.2 Options

`Coding`*`method`*
method specifies the type of dummy coding. It can be one of `dummy`, or `effect`. The default is `dummy`. The first category is used as reference category.

4.7.5.3 Redirection

Redirection is not applicable to this command.

4.7.5.4 Examples

```
categorise gender(M:F);
```

Establishes “M” as a reference category so M will be coded “0” and F will be coded “1”.

```
categorise time(3);
```

Establishes “1” as a reference category compared to groups coded “2” and “3”.

```
categorise grade(3:4:5:6:7) ! effect;
```

Establishes four variables to represent the five response categories for `grade`. Effect coding is used and the reference category is “3”.

`grade=3`
corresponds to
grade_1=-1, *grade_2=-1*, *grade_3=-1*, and *grade_4=-1*.

`grade=4`
corresponds to
grade_1=1, *grade_2=0*, *grade_3=0*, and *grade_4=0*.

`grade=5`
corresponds to
grade_1=0, *grade_2=1*, *grade 3=0*, and *grade_4=0*.

`grade=6`
corresponds to
grade_1=0, *grade_2=0*, *grade_3=1*, and *grade_4=0*.

`grade=7`
corresponds to
grade_1=0, *grade_2=0*, *grade_3=0*, and *grade_4=1*.

```
Categorise size (S:M:L);
```

Establishes two variables to represent the three response categories for `size` (Small, Medium and Large). Dummy coding is used and the reference category is "S".

`size=S`
corresponds to
size_1=0; and *size_2=0;*

`size=M`
corresponds to
size_1=1; and *size_2=0;*

`size=L`
corresponds to
size_1=0; and *size_2=1;*

4.7.5.5 GUI Access

Access to this command through the GUI is not available.

4.7.5.6 Notes

1. Any levels of the variable that are omitted from the code list are treated as missing data.
2. To alter the reference level change the order in which the levels are listed.
3. Only one variable can be processed with a categorise command. Use multiple commands to categorise multiple variables.
4. The default match is a trim match, to use exact matching enclose the drop code in quotes (“ “)

4.7.6 chistory

Writes the commands that have been run up to the point where this command is called.

4.7.6.1 Argument

This command has no argument.

4.7.6.2 Options

This command does not have options.

4.7.6.3 Redirection

`>>`*`filename`*
Show results are written to the file named *`filename`*. If redirection is omitted, the results are written to the output window or the console.

4.7.6.4 Examples

```
chistory;
```

Writes the commands that were run up to this command to the output window.

```
chistory >> chistory.txt;
```

Saves all the commands that were run up to this command to a text file called `chistory.txt`, saved in the current working directory.

4.7.6.5 GUI Access

Access to this command through the GUI is not available

4.7.6.6 Notes

1. This command needs the argument `storecommands=yes` in the `set` command. All commands **after** the `set` command up to and including `chistory` will be saved.

4.7.7 clear

Removes variables or tokens from your workspace.

4.7.7.1 Argument

A comma separated list of variables and/or tokens or one of `all`, `tokens` or `variables`. The default is `all`.

4.7.7.2 Options

This command does not have options.

4.7.7.3 Redirection

Redirection is not applicable to this command.

4.7.7.4 Examples

```
clear all;
```

Clears all variables and tokens from your workspace.

```
clear x, date;
```

Deletes the variable (or tokens) `x` and `date` from your workspace.

4.7.7.5 GUI Access

`Workspace→Tokens and Variables`

Results in a dialog box. The box displays the list of available tokens and variables. The Clear All or Clear Selected buttons can be used either to clear all objects listed in the box or clear the selected objects, respectively. The action takes immediate place once the button is clicked.

4.7.7.6 Notes

1. The work space is your temporary working environment and includes any user-defined elements (tokens and matrices). Elements can be saved using the print command, otherwise they will be deleted when closing ACER ConQuest.

4.7.8 codes

Lists the characters that are to be regarded as valid data for the responses.

4.7.8.1 Argument

A comma-delimited or space-delimited list of response codes.

4.7.8.2 Options

This command does not have options.

4.7.8.3 Redirection

Redirection is not applicable to this command.

4.7.8.4 Examples

```
codes 0,1,2,3;
```

The valid response codes are 0, 1, 2 and 3.

```
codes a b c d;
```

The valid response codes are a, b, c and d.

```
codes 1, 2, 3, 4, 5, " ";
```

The valid response codes are 1, 2, 3, 4, 5, and a blank.

```
codes " 1", " 2", " 3", "10";
```

Each response code takes two columns. The first three that are listed have leading spaces, which must be included.

4.7.8.5 GUI Access

`Command`→`Codes`

The list of codes must be entered using the same syntax guidelines as described above for the `codelist`.

4.7.8.6 Notes

1. If a blank is to be used as a valid response code or if a blank is part of a valid response code, double quotation marks (`" "`) must surround the response code that includes the blank.
2. `Codelist` specifies the response codes that will be valid after any recoding has been performed by the `recode` statement.
3. If a `codes` statement is provided, then any character found in the response block of the data file (as defined by the format statement) and not found in `codelist` will be treated as missing-response data.

4. Any missing-response codes (as defined by the `set` command argument `missing`) in `codelist` will be ignored. In other words, `missing` overrides the `codes` statement.
5. If a `codes` statement is not provided, then all characters found in the response block of the data file, other than those specified as missing-response codes by the `set` command argument `missing`, will be considered valid.
6. The number of response categories modelled by ACER ConQuest is equal to the number of unique response codes (after recoding).
7. Response categories and item scores are *not* the same thing.

4.7.9 colnames

overwrites the names of an ACER ConQuest matrix object.

4.7.9.1 Argument

A name of a current matrix object.

4.7.9.2 Options

A comma separated list of new column names in order and of the same length as the matrix object passed in as an argument.

4.7.9.3 Redirection

Redirection is not applicable to this command.

4.7.9.4 Examples

```
mymatrix = fillmatrix(2,2,0);
write mymatrix ! filetype = csv >> mymatrix_defaultcolumnlabels.csv;
colnames mymatrix ! column1, column2;
write mymatrix ! filetype = csv >> mymatrix_customcolumnlabels.csv;
```

Creates a 2x2 matrix, filled with zeros. Writes the matrix, `mymatrix` to the file `mymatrix_defaultcolumnlabels.csv` with default column labels (“col_1” and “col_2”).

Overwrites the column names of `mymatrix` with "column1", "column2" and saves them to the file `mymatrix_customcolumnlabels.csv`. Note files are saved in the current working directory, which is printed to the screen using the command `dir;`

4.7.9.5 Notes

1. The list of column labels must be the same length as the number of columns in the matrix.
2. To find the number of columns in a matrix object, use the command print, or alternatively you can assign the value to an object:

```
a = cols(mymatrix);
print a
```

4.7.10 compute

Undertakes mathematical computations and creates an ACER ConQuest data object to store the result. The data object can be a real number or a matrix. The command word `compute` is optional and is assumed if a command word is omitted and the argument is a legal mathematical expression.

4.7.10.1 Argument

lvalue = *expression*

lvalue is one of:

- The name of a matrix that will be created when *expression* is evaluated, or
- A single sub-element of an existing matrix, or

expression is a legal mathematical expression in ACER ConQuest syntax. See Compute Command Operators and Functions.

If the matrix named, `lvalue` already exists, it will be overwritten when *expression* is evaluated. If the user is referencing a sub-element of the matrix named, `lvalue`, it must referenced by row and column and the *expression* must resolve to a 1x1 matrix.

4.7.10.2 Options

This command does not have options.

4.7.10.3 Redirection

Redirection is not applicable to this command.

4.7.10.4 Examples

```
compute x=10;
x=10;
```

Alternatives for creating a 1-by-1 matrix with x[1,1]=10.

```
compute x={1,2,3,4};
x={1,2,3,4};
```

Either form populates a pre-existing matrix with these values. Columns cycle fastest, so the result is x[1,1]=1, x[1,2]=2, x[2,1]=3, and x[2,2]=4. A matrix with as many elements as given numbers must have been pre-defined via a `compute` command.

```
compute x=a+b;
x=a+b;
```

Alternatives for creating the matrix x as matrix sum of matrices a and b.

```
compute m[10,3]=5;
m[10,3]=5;
```

Either form sets the row=10, column=3 element of the matrix m to 5.

```
t1=4;
```

Produces a 1x1 matrix named, `t1`, where the value `t[1,1]` is equal to 4.

```
t2=fillmatrix(2,2,0);
t2={1,2,3,4};
```

Produces a 2x2 matrix named, `t2`, filled with zeros. Then populates the elements of `t2` with the values between the braces. Values in braces must be a comma separated list of real numbers with the same length as the number of elements in the matrix. Note that columns cycle fastest so that the matrix is populated row-wise.

```
t3=identity(2);
x=2;
y=1;
t3[1,1]=20;
t3[x,y]=10;
```

Produces a 2x2 identity matrix named, `t3` (i.e. filled with 1 on the diagonal, and 0 on the off-diagonal elements). Then creates a 1x1 matrix named, `x`, where the value `x[1,1]` is equal to 2. Then creates a 1x1 matrix named, `y`, where the value `y[1,1]` is equal to 1. Then replaces the value `t3[1,1]` with the value `20`. Then replaces the value `t3[2,1]` (note: `x` and `y` are resolved as indices of `t3`) with the value `10`. Note that the row and column references must be explicit integers or passed-in 1x1 matrices.

4.7.10.5 GUI Access

Access to this command through the GUI is not available.

4.7.10.6 Notes

1. The available functions and operators are listed and described in section 4.8, Compute Command Operators and Functions.
2. Parentheses can be nested to 10 deep.
3. To populate a matrix with a set of values that matrix must be previously defined using the `let` command. If the right hand side of the assignment (‘=’) is a matrix or mathematical expression, then the output matrix need not be defined in advance.
4. Sub matrices can be extracted from matrices by appending [*`rowstart`*: *`rowend`*, *`colstart`*: *`colend`*] to the name of a matrix variable. If all rows are required *`rowstart`* and *`rowend`* can be omitted. If all columns are required *`colstart`* and *`colend`* can be omitted. If a single row is required *`rowend`* and the colon `":"` can be omitted. If a single column is required *`colend`* and the colon `":"` can be omitted.

5. Single elements of a matrix can be specified to the left of the equal operator '=' by appending [*row* ,*col*] to the name of a matrix variable. Sub matrices cannot be specified to the left of the equal operator '='.

4.7.11 datafile

Specifies the name, location and type of file containing the data that will be analysed.

4.7.11.1 Argument

filename
filename is the name or pathname (in the format used by the host operating system) that contains the data to be analysed. The file type can be ASCII text file (fixed format), a csv file or SPSS system file.

4.7.11.2 Options

`filetype =`*type*
type indicates the format of the datafile. Available options are `spss`, `csv` and `text`. The default is `text`. If an input file has csv or SPSS format then a format command is automatically generated by ACER ConQuest.

`responses =`*varlist*
A space delimited list of variables from the csv or SPSS file that are to be used as the (generalised) item responses in the model. The keyword '`to`' can be used to specify a list of sequentially placed variables in the csv or SPSS file. This option is not applicable for fixed format input files.

`facets =`*string*
Describes the implicit variables that underlie the responses (see `format` command). This option is not applicable for fixed format input files.

`columnlabels =`*yes/no*
If the filetype is `spss`, or `csv` and one (default) facet is used (usually "item"), the column names from the datafile are read in as labels. When a csv file has no header, the default names become labels ("v1, v2, … , vn"). The default is `no`.

`echo =`*yes/no*
If the filetype is `spss`, or `csv` format and weight commands are auto generated. If echo is '*yes*' these commands are displayed. The default is `no`.

`keeps =`*varlist*
A space delimited list of additional variables read from the SPSS file and retained as explicit variables. The keyword '`to`' can be used to specify a list of sequentially placed variables in the SPSS file. This option is not applicable for fixed format input files.

`weight =`*var*
A variable from the SPSS file to be used as a caseweight variable. The default is no caseweight variable. This option is not applicable for fixed format input files.

`pid =`*var*
A variable from the SPSS file to be used as a case identifier. The default is no `pid`. See `format` for a description of the `pid` variable. This option is not applicable for fixed format input files.

`width =`*n*
A value to use as the width of the response variables. The *n* left most characters of the SPSS response variables are retained and used as the (generalised) item responses. The default width is *1*. This option is not applicable for fixed format input files.

`pidwidth =`*n*
A value to use as the width of the person ID (PID) variable. The *n* left most characters (**including** the decimal point) of the keeps variables are retained. For SPSS file the default width is the "width" value specified for the variable in SPSS. This value is shown in the Variable View in SPSS. See note 5. For CSV files there is no default width and `pidwidth` must be declared. This option is not applicable for fixed width format input files.

`keepswidth =`*n*
A value to use as the width of the keeps variables. The *n* left most characters (**including** the decimal point) of the keeps variables are retained. For SPSS file the default width is the "width" value specified for the variable in SPSS. This value is shown in the Variable View in SPSS. See note 5. For CSV files there is no default width and `keepswidth` must be declared. Note for PID variables, the default width for CSV files is 15 unless `keepswidth` is declared. This option is not applicable for fixed width format input files.

`header =`*yes/no*
Used when filetype is csv to indicate whether the file contains a header row or not.
The default is *yes*. If the value is *no*, then variable names are constructed as `v1…vn`, where n is the number of fields on first record.

`display =`*n*
Echo first *n* records read from csv or SPSS file on screen.

4.7.11.3 Redirection

`<<`*filename*
The name or pathname (in the format used by the host operating system) of the ASCII text file, csv or SPSS system file that contains the data to be analysed. The specification of the filename as an argument or as a redirection are alternatives.

`>>`*outfilenames*
An optional list of file names. If a single file name is given, a text version of the data file is provided. If a comma separated list of two file names are given, a text version of the data file is provided (first file name) and a text version of the labels file is provided (second file name).

The outfile is used in conjunction with the file type spss and csv option and results in a text copy of the analysed data being retained.

4.7.11.4 Examples

```
datafile mydata.txt;
```

The data file to be analysed is called `mydata.txt`, and it is in the same directory as the ACER ConQuest application.

```
datafile /math/test1.dat;
```

The data file to be analysed is called test1.dat, and it is located in the directory math.

```
datafile << c:/math/test1.dat;
```

The data file to be analysed is called test1.dat, and it is located in the directory math on the C: drive.

```
datafile test2.sav
    ! filetype=spss, responses=item1 to item16, keeps=country,
    weight= pwgt, facets=tasks(16), pid=id
        >> test.dat;
```

The data file to be analysed is called test2.sav, and it is an SPSS file. The set of SPSS variables beginning with item1 and concluding with item16 are retained as responses, country is retained as an explicit variable, pwgt will be used as a caseweight and id as a pid. The responses will be referred to as tasks. The requested data will be written to the file test.dat and it will be retained after the analysis. Use of this datafile command is equivalent to the following three commands:

```
datafile << test2.dat;
format pid 1-15 responses 16-31(a1) pwgt32-42 country 42-51 ! tasks(16);
caseweight pwgt;
```

```
datafile test2.sav
    ! filetype=spss, responses=item1 to item16, keeps=country,
    weight=pwgt, facets=tasks(16), pid=id;
```

This example is equivalent to the previous example except that the requested data will be written to a scratch file that will not be retained after the analysis.

```
datafile test2.sav
    ! filetype=spss, responses=item1 to item16,
    keeps=GINI_index, keepswidth=5;
```

This example shows that the variable GINI_index is retained as an explicit variable. The values are speficified to be 5 characters wide, regardless of the width specification in the original SPSS file. For example, if in the original SPSS file the variable width is 7, a case with GINI_index of 2.564227 will be truncated to 2.564.

4.7.11.5 GUI Access

`Command`→`Data File.`
Note that GUI access does not yet support SPSS file imports.

4.7.11.6 Notes

(1) The actual format of *`filename`* will depend upon the host operating system.

(2) When inputting the response data in a data file, remember that ACER ConQuest treats blanks and periods found in the responses as missing-response data unless you either use a `codes` statement to specify that one or both are to be treated as valid response codes, or use the `set` command argument missing to change the missing-response code.
(3) The layout of your data file lines and records must conform to the rules of the `format` command.
(4) A file of simulated data can be created with the `generate` command.
(5) When using SPSS files, both character and numeric variables can be used. The conversion for use by ACER ConQuest of numeric variables is governed by the "width" property of the variables in the SPSS file. For numeric variables, "width" refers to how many digits should be *displayed* (including decimal digits, but *excluding* the decimal point) in SPSS. However, if ACER ConQuest uses the converted variables as strings, a leading blank will be added. This needs to be accounted for when specifying particular values for example in the `keep` and `drop` options of various command statements.
(6) The maximum width of a variable read from an SPSS files is 256 characters
(7) System missing numeric values in SPSS are converted to a period character (.) in a field of width set by the width property in the SPSS file.
(8) If using variables that are treated as string, for example in `group` statement, is recommended to convert the type to String within SPSS before running in ACER ConQuest.
(9) The option `columnlabels` is only useful in the case of simple models (a single or default facet). In other cases, read in a labels file using the command `labels`. If the `labels` command is used and it provides names for the single facet they will over-write the labels from the column names.

4.7.12 delete

Omit data for selected implicit variables from analyses.

4.7.12.1 Argument

This command has no argument.

4.7.12.2 Options

A list of implicit variables and the levels that are to be omitted from the analysis for each variable.

4.7.12.3 Redirection

Redirection is not applicable to this command.

4.7.12.4 Examples

```
delete ! item (1-10);
```

Omits items 1 through 10 from the analysis.

```
delete ! rater (2, 3, 5-8);
```

The above example omits data from raters 2, 3, 5, 6, 7, and 8 from the analysis.

4.7.12.5 GUI Access

`Command`→`Delete`. The list of candidate implicit variables is listed in the list box. Multiple selections can be made by shift-clicking.

4.7.12.6 Notes

1. `delete` statement definitions stay in effect until a `reset` statement is issued.
2. `delete` preserves the original numbering of items (as determined by the `format` and the `model` statements) for the purposes of data display and for labels. Note however that it does change parameter numbering. This means that anchor and initial values files may need to be modified to reflect the parameter numbering that is altered with the any deletions.
3. `delete` statements are processed after `score` statements. All items declared in a `format` or `datafile` statement must be included in the `score` before they can then be deleted.

4. To omit data for specified values of explicit variables the missing data command can be used.
5. See the `dropcases` and `keepcases` commands which are used to limit analysis to a subset of the data based on explicit variables.

4.7.13 descriptives

Calculates a range of descriptive statistics for the estimated latent variables.

4.7.13.1 Argument

This command does not have an argument.

4.7.13.2 Options

`estimates =`*`type`*
`type` can be `eap`, `latent`, `mle` or `wle`. If `estimates=eap`, the descriptive statistics will be constructed from expected a-posteriori values for each case; if `estimates=latent`, the descriptive will be constructed from plausible values for each case; if `estimates=mle`, the descriptive statistics will be constructed from maximum likelihood cases estimates and if `estimates=wle`, the descriptive statistics will be constructed from weighted likelihood cases estimates.

`group =`*`v1`*`[by`*`v2`*`by …]`
An explicit variable to be used as grouping variable or a list of group variables separated using the word "by". Results will be reported for each value of the group variable, or in the case of multiple group variables, each observed combination of the specified group variables. The variables must have been listed in a previous `group` command. The limit for the number of categories in each group is 1000.

`percentiles =`*`n1:n2:…:ni`*
`ni` is a requested percentile to be computed.

`cuts =`*`n1:n2:…:ni`* Requests calculation of the proportion of students that lie within a set of intervals on the latent scale. *`ni`* is a requested cut point. The specification of *`i`* cut points results in *`i+1`* intervals.

`bench =`*`n1:n2:n3`*
Requests calculation of the proportion of students that lie either side of a benchmark

location on the latent scale. *n1* is the benchmark location, *n2* is the uncertainty in the location, expressed as standard deviation and *n3* is the number of replications to use to estimate the standard error of the proportion of students above and below the benchmark location.

`filetype =`*`type`*
type can take the value `xls`, `xlsx`, `excel` or `text`. It sets the format of the results file. Both `xls` and `excel` create files readable by all versions of Excel. The `xlsx` format is for Excel 2007 and higher. The default is `text`. If no redirection file is provided this option is ignored.

`matrixout =`*`name`*
name is a matrix (or set of matrices) that will be created and will hold the results in your workspace. Any existing matrices with matching names will be overwritten without warning. The content of the matrices is described in section 4.9, Matrix Objects Created by Analysis Commands.

`display =`*`reply`*
By default *reply* is `long`. If *reply* is `short`, results will not be displayed for individual plausible values.

4.7.13.3 Redirection

`>>`*`filename`*
filename is the name of a file to which results can be written.

4.7.13.4 Examples

```
descriptives ! estimates=latent;
```

Using plausible values produces the mean, standard deviation and variance (and the associated error variance) for each of the latent dimensions.

```
descriptives ! estimates=latent, group=gender;
```

Using plausible values produces the mean, standard deviation and variance (and the associated error variance) for each of the latent dimensions for each value of gender.

```
descriptives ! estimates=mle, percentiles=10:50:90;
```

Using maximum likelihood estimates produces the mean, standard deviation and variance (and the associated error variance) for each of the latent dimensions. The 10th, 50th and 90th percentiles are also estimated for each dimension.

```
descriptives ! estimates=latent, cuts=-0.5:0.0:0.5;
```

Using plausible values estimates produces the mean, standard deviation and variance (and the associated error variance) for each of the latent dimensions. The proportion of students in the four intervals: less than –0.5; between –0.5 and 0.0; between 0.0 and 0.5; and greater than 0.5 are also estimated for each dimension.

```
descriptives ! estimates=latent, bench=-1.0:0.1:1000;
```

Using plausible values estimates produces the mean, standard deviation and variance (and the associated error variance) for each of the latent dimensions. The proportion of students above and below a benchmark of –1.0 is also estimated for each dimension. The error in these proportions is based upon an uncertainty of 0.1 in the benchmark location. The error was estimated using 1000 replications.

4.7.13.5 GUI Access

`Analysis`→`Descriptives`→`Latent Variables`.

4.7.13.6 Notes

1. The ability estimates requested (`wle`, `mle`, `eap` and `latent`) must have been previously estimated (see `show` command).

4.7.14 directory

Displays the name of the current working directory. The working directory is where ACER ConQuest looks for files and writes files when a full directory path is not provided as part of a file specification.

4.7.14.1 Argument

This command does not have an argument.

4.7.14.2 Options

This command does not have options.

4.7.14.3 Redirection

Redirection is not applicable to this command.

4.7.14.4 Example

```
directory;
```

4.7.14.5 GUI Access

Access to this command through the GUI is not available.

4.7.14.6 Notes

1. For the GUI version of ACER ConQuest the result of this command is shown in the status bar.
2. To change the working directory see the `set` argument option `directory`.

4.7.15 dofor

Allows looping of syntax.

4.7.15.1 Arguments

- *`list of comma-separated arguments`*
  Takes the form of the definition of a loop control variable followed by an equals sign folowed by the list of elements that will be iterated over. For example `dofor x=M,F;` defines the loop control variable, `x`, and the list of `M` and `F` will be iterated over. Optionally, elements in the *`list of comma-separated arguments`* can take the form *i1* - *i2* (where *i1* and *i2* are integers and $i1 < i2$) and the element will be expanded to be a list of all integers from *i1* to *i2* (inclusive).

4.7.15.2 Options

This command does not have options.

4.7.15.3 Redirection

Redirection is not applicable to this command.

4.7.15.4 Example

```
dofor x=M,F;
    Plot icc ! group=gender; keep=%x%;
enddo;
```

Produces plots for students with gender value M and then gender value F.

4.7.15.5 GUI Access

Access to this command through the GUI is not available.

4.7.15.6 Notes

1. `dofor` must be used in conjunction with `enddo`.
2. `dofor` loops cannot be nested.
3. See the command `for` for alternative looping options.

Table 4.1: Comparison operators

Operator	Meaning
==	equality
=>	greater than or equal to
>=	greater than or equal to
=<	less than or equal to
<=	less than or equal to
!=	not equal to
>	greater than
<	less than

4.7.16 doif

Allows conditional execution of syntax.

4.7.16.1 Argument

`logical condition`
If *`logical condition`* evaluates to `true`, the set of ACER ConQuest commands is executed. The commands are not executed if the *`logical condition`* does not evaluate to `true`.

The *`logical condition`* can be `true`, `false` or of the form *`s1 operator s2`*, where *`s1`* and *`s2`* are strings and *`operator`* is one of the following

For each of *`s1`* and *`s2`* ACER ConQuest first attempts to convert it to a numeric value. If *`s1`* is a numeric value the operator is applied numerically. If not, a string comparison occurs between *`s1`* and *`s2`*.

4.7.16.2 Options

This command does not have options.

4.7.16.3 Redirection

Redirection is not applicable to this command.

4.7.16.4 Example

```
doif %x%==M;
    Plot icc ! group=gender; keep=%x%;
endif;
```

Produces plots for students with gender value M.

4.7.16.5 GUI Access

Access to this command through the GUI is not available.

4.7.16.6 Notes

1. `doif` must be used in conjunction with `endif`.
2. `doif` conditions cannot be nested.
3. See the command `if` for alternative conditional execution options.

4.7.17 dropcases

List of values for explicit variables that if matched will cause a record to be omitted from the analysis.

4.7.17.1 Argument

`list of drop codes`

The *`list of drop codes`* is a comma separated list of values that will be treated as drop values for the subsequently listed explicit variable(s).

When checking for drop codes two types of matches are possible. EXACT matches occur when a code in the data is compared to a drop code value using an exact string match. A code will be regarded as a drop value if the code string matches the drop string exactly, including leading and trailing blank characters. The alternative is a TRIM match that first trims leading and trailing spaces from both the drop string and the code string and then compares the results.

The key words `blank` and `dot`, can be used in the *`list of drop codes`* to ensure TRIM matching of a blank character and a period. Values in the *`list of drop codes`* that are placed in double quotes are matched with an EXACT match. Values not in quotes are matched with a TRIM match.

4.7.17.2 Options

A comma separated list of explicit variables.

4.7.17.3 Redirection

Redirection is not applicable to this command.

4.7.17.4 Examples

```
dropcases blank, dot, 99 ! age;
```

Sets `blank`, `dot` and `99` (all using a trim match) as drop codes for the explicit variable `age`.

```
dropcases blank, dot, " 99" ! age;
```

Sets `blank`, and `dot` (using a trim match) and `99` with leading spaces (using an exact match) as drop codes for the explicit variable `age`.

```
dropcases M ! gender;
```

Sets `M` as a drop code for the explicit variable `gender`.

4.7.17.5 GUI Access

`Command`→`Drop Cases`.
Select explicit variables from the list (shift-click for multiple selections) and choose the matching drop value codes. The syntax of the drop code list must match that described above for *`list of drop codes`*.

4.7.17.6 Notes

1. Drop values can only be specified for explicit variables.
2. Complete data records that match drop values are excluded from all analyses.
3. If multiple records per case are used in conjunction with a `pid`, then the `dropcases` applies at the record level not the case level.
4. See the `missing` command which can be used to omit specified levels of explicit variables from an analysis and the `delete` command which can be used to omit specified levels of implicit variables from an analysis.
5. See the `keepcases` command which can be used to keep specified levels of explicit variables in the analysis.
6. When used in conjunction with SPSS or csv input, note that character strings may include trailing or leading spaces and this may have implications for appropriate selection of a match method.
7. The default match is a trim match, to use exact matching enclose the drop code in quotes (“ “)

4.7.18 else

Used as part of a `doif` condition.

4.7.18.1 Argument

This command does not have an argument.

4.7.18.2 Options

This command does not have options.

4.7.18.3 Redirection

Redirection is not applicable to this command.

4.7.18.4 Example

```
doif %x%==M;
    print "Plot for Males";
    plot icc ! group=gender; keep=M;
else;
    print "Plot for Females";
    plot icc ! group=gender; keep=F;
endif;
```

Produces plots for students with gender value M or F depending upon the value of the token %x%.

4.7.18.5 GUI Access

Access to this command through the GUI is not available.

4.7.18.6 Notes

1. `else` must be used in conjunction with `doif` and `endif`.

4.7.19 enddo

Terminates a `dofor` loop.

4.7.19.1 Argument

This command does not have an argument.

4.7.19.2 Options

This command does not have options.

4.7.19.3 Redirection

Redirection is not applicable to this command.

4.7.19.4 Example

```
dofor x=M,F;
    plot icc ! group=gender; keep=%x%;
enddo;
```

Produces plots for students with gender value M and then gender value F.

4.7.19.5 GUI Access

Access to this command through the GUI is not available.

4.7.19.6 Notes

1. `enddo` must be used in conjunction with `dofor`.

4.7.20 endif

Terminates a `doif` condition.

4.7.20.1 Argument

This command does not have an argument.

4.7.20.2 Options

This command does not have options.

4.7.20.3 Redirection

Redirection is not applicable to this command.

4.7.20.4 Example

```
doif %x%=M;
    plot icc ! group=gender; keep=%x%;
endif;
```

Produces plots for students with gender value M.

4.7.20.5 GUI Access

Access to this command through the GUI is not available.

4.7.20.6 Notes

1. `endif` must be used in conjunction with `doif`.

4.7.21 equivalence

Produce a raw score to ability estimate equivalence.

4.7.21.1 Argument

estimate type

estimate type must be either `wle` or `mle`

4.7.21.2 Options

`matrixin =`*name*
name is an existing matrix than can be used as source for the item parameter values.

`matrixout =`*name*
name is a matrix that will be created and will hold the results. It will be matrix with three columns and as many rows as there are score point. Column 1, contains the score value, column 2 the matching maximum likelihood estimate, and 3 contains the standard error. More detail on the content of the matrices is described in section 4.9, Matrix Objects Created by Analysis Commands.

`display =`*`reply`*
If *`reply`* is `no`, results will not be displayed. The default is `yes`.

4.7.21.3 Redirection

`>>`*`filename`*
A file name for output.

`<<`*`filename`*
A file name of item parameters.

4.7.21.4 Examples

```
equivalence wle;
```

Produces a raw score to weighted likelihood estimate equivalence table.

```
equivalence mle  >> mle.txt;
```

Produces a raw score to maximum likelihood estimate equivalence table and save it in the file mle.txt.

4.7.21.5 GUI Access

- `Tables→Raw Score↔Logit Equivalence→MLE`
- `Tables→Raw Score↔Logit Equivalence→WLE`
- `Tables→Raw Score↔Logit Equivalence File→MLE`
- `Tables→Raw Score↔Logit Equivalence File→WLE`

4.7.21.6 Notes

1. The equivalence table assumes a complete response vector and integer scoring.
2. Maximum and minimum values for maximum likelihood values are set using the `perfect/zero=` option of the `set` command.

3. If an input file is not specified then a model must have been estimated and the table is provided for the current model.

4. If an input file is specified then equivalence table can be requested at any time.

5. An input file must be an ASCII file containing a list of item parameter estimates. Each line of the file should consist of the information for a single parameter with the item parameters being supplied in the Andrich delta plus tau format. Each line of the file should contain three values: item number, category number and the parameter value. The item difficulty parameter is signified by a category number of zero. For example to indicate 3 dichotomous items the file could look as follows:

   ```
   1 0 0.6
   2 0 -1.5
   3 0 2.3
   ```

   To indicate 3 items each with three response categories the file could look as follows:

   ```
   1 0 0.6
   1 1 -0.2
   2 0 -1.5
   2 1 -0.5
   3 0 2.3
   3 1 1.1
   ```

   Note that the order of the parameters does not matter and there is one fewer category parameter than there is categories.
   The last category parameter is assumed to be the negative sum of those provided.

6. An input matrix must contain three columns. Each row of the matrix should consist of the information for a single parameter with the item parameters being supplied in the Andrich delta plus tau format. Column 1 is the item number, column 2, the category number and column 3 the parameter value. The item difficulty parameter is signified by a category number of zero.

7. An input matrix cannot be used at the same time as an input file.

4.7.22 estimate

Begin estimation.

4.7.22.1 Argument

This command does not have an argument.

4.7.22.2 Options

`method =`*`type`*
Indicates the type of numerical integration that is to be used. *`type`* can take the value `gauss`, `montecarlo`, `adjmc`, `quadrature`, `JML` or `patz`. The default is `gauss` when there are no regressors in the model (intercept only) and `quadrature` when regressors are included in the model. Adjusted Monte Carlo (`adjmc`) is used to draw plausible values after estimation, and is available for estimation is all item parameters are anchored and there are no regressors in the model (intercept only).

`switchtoadjmc =`*NUMBER*
When using Adjusted Monte Carlo, the first `NUMBER` iterations are estimated using Monte Carlo before switching to Adjusted Monte Carlo. The default is 2.

`distribution =`*`type`*
Specifies the (conditional) distribution that is used for the latent variable. *`type`* can take the value `normal`, or `discrete`. The default is `normal`. If a `discrete` population distribution is chosen, `eaps` and `pvs` cannot be computed, and fit statistics cannot be computed. A `discrete` distribution is not available with regressors. If `patz` or `jml` method is used the option `distribution` is not relevant and is ignored.

`nodes =`*n*
Specifies the number of nodes that will be used in the numerical integration. If the `quadrature` or `gauss` method has been requested, this value is the number of nodes to be used for each dimension. If the `montecarlo` method has been selected, it is the total number of nodes. The default value is 15 per dimension if the method is `gauss` or `quadrature`, and 1000 nodes in total if the method is `montecarlo`. The nodes option is ignored if `method` is `JML` or `patz`.

`minnode =`*f*
Sets the minimum node value when using the `quadrature` method. The default is -6.0. All other methods ignore this option.

`maxnode =`*f*
Sets the maximum node value when using the `quadrature` method. The default is 6.0. All other methods ignore this option.

`iterations =`*n*
if `method = gauss`, `montecarlo`, `quadrature`, or `JML` specifies the maximum number of iterations for the maximum likelihood algorithm. Estimation will terminate when either the iteration criterion or the convergence criterion is met. If `method = patz` specifies the number of MCMC steps. Note that this number will be divided by the value in `skip` to give the final saved chain length. The default value is 2000.

`convergence =`*f*
Instructs estimation to terminate when the largest change in any parameter estimate between successive iterations is less than *f*. The default value is 0.0001.

`deviancechange =`*f*
Instructs estimation to terminate when the change in the deviance between successive iterations of the EM algorithm is less than *f*. The default value is 0.0001.

`abilities =`*reply*
reply takes the format of a colon separated list of strings, or the string `yes` or `no`. When a ist of strings, of length one or more, is given, each string represents the ability estimate type that will be calculated. Valid strings include `eap`, `pvs`, `pv`, `wle`, or `mle`. If *reply* is yes, all ability estimates (WLE, MLE, EAP and plausible values) will be generated after the model has converged. This may accelerate later commands that require the use or display of these estimates. The default is `no`.

`stderr =`*type*
Specifies how or whether standard errors are to be calculated. *type* can take the value `quick`, `empirical` or `none`. `empirical` is the default and uses empirical differentiation of the likelihood. While this method provides the most accurate estimates of the asymptotic error variances that ACER ConQuest can compute, it may take a considerable amount of computing time, even on very fast machines. `quick` standard errors are suitable when dichotomous items are used with a single facet and with `lconstraint=cases`. If JML estimation is used then `quick` is the default and `empirical` is not available. If `patz` method is used the option `stderr` is not relevant and is ignored. For pairwise models, the option `stderr` is not relevant and is ignored.

`fit =`*reply*
Generates item fit statistics that will be included in the tables created by the show statement. If *reply* is `no`, fit statistics will be omitted from the show statement tables. The default is `yes` (see also the estimates option of the show command).

`ifit =`*reply*
Same as the fit option.

`pfit =`*reply*

Computes case fit estimates following estimation. The default is `no`. If ***reply*** is `yes`, person fit is accessible in conjunction with the matrixout option or in the output of `show cases` when abilities are `MLE` or `WLE`.

`matrixout =`*`name`*
`name` is a matrix (or set of matrices) that will be created and will hold the results. These results are stored in the temporary workspace. Any existing matrices with matching names will be overwritten without warning. The contents of the matrices is described in the section, Matrix Objects Created by Analysis Commands.

`xsiincmax =`*`f`*
Sets the maximum allowed increment for item response model location parameters in the M-Step. The default value is `1`.

`facoldxsi =`*`f`*
f is a value between 0 and 1, which defines the weight of location parameter values in the previous iteration. If ξ_t denotes a parameter update in iteration t, and ξ_{t-1} is the parameter value of iteration $t-1$, then the modified parameter value is defined as $\xi_t^* = (1-f)\xi_t + f\xi_{t-1}$. Especially in cases where the deviance increases, setting the parameter larger than 0 (maybe .4 or .5) is helpful in stabilizing the algorithm. The default value is `0`.

`tauincmax =`*`f`*
Sets the maximum allowed increment for item response model scoring parameters in the M-Step. The default value is `0.3`.

`facoldtau =`*`f`*
f is a value between 0 and 1, which defines the weight of scoring parameter values in the previous iteration. If τ_t denotes a parameter update in iteration t, and τ_{t-1} is the parameter value of iteration $t-1$, then the modified parameter value is defined as $\tau_t^* = (1-f)\tau_t + f\tau_{t-1}$. Especially in cases where there are convergence issues, setting the parameter larger than 0 (maybe .4 or .5) is helpful in stabilizing the algorithm. The default value is `0.3`.

`tauskip =`*`i`*
Sets the number of iterations skipped during estimation. The default value is `1` (the scoring parameters are updated on every other iteration). ***i*** must be an integer greater than or equal to 0.

`cqs =`*`name`*
Instructs ConQuest to write a system file to disk at the end of each iteration. This can be useful to save the state of the software during estimation where a very long run or

calculation may result in premature termination (e.g., a system update forcing a restart). `name` can be any valid filename and or path.

`compress =`*`bool`*
Should the system file written at the end of each step in the iteration be compressed? The value must be `false` to work with the R library `conquestr`. The default is `false`.

These options are only available with `method = patz` and are otherwise ignored:

`burn =`*`n`*
Sets the number of MCMC iterations discarded before starting to save.

`skip =`*`n`*
Specifies the number of MCMC iterations discarded between saved iterations. For example, if *n* `= 10` then the 10th, 20th, 30th, … , up to the value provided in `iterations` is saved to the chain.

`xsipropvar =`*`n`*
fixed item param proposal variance - variance of distribution sampled from for proposed value, defaults to 0.02. If not provided the proposal variance is dynamically set to result in approximately 44% of draws being accepted.

`taupropvar =`*`n`*
fixed tau param proposal variance - variance of distribution sampled from for proposed value, defaults to 0.002. If not provided the proposal variance is dynamically set to result in approximately 44% of draws being accepted.

`thetapropvar =`*`n`*
fixed theta param proposal variance - variance of distribution sampled from for proposed value, defaults to 0.5. If not provided the proposal variance is dynamically set to result in approximately 44% of draws being accepted.

`blockbeta =`*`reply`*
Should the regression parameters estimates be updated simultaneously (treated as a single block, drawn from a random MVN distribution) or simultaneously by dimension (treated as a block, per dimension, drawn from d random MVN distributions) or one at a time (drawn from a random normal distribution)? `all` - update all regression parameters estimates at the same time for all dimensions. `bydim` - update all regression parameters estimates at the same time for each dimension. `no` - update regression parameters estimates one at a time. the default is `no`.

`adaptiveacceptance =`*`bool`*
Automatically adjust `xsipropvar`, `taupropvar`, and `thetapropvar` to target an acceptance rate of 44%. The default is `yes`.

`retainchain =`*`bool`*
When additional call to the command estimate are made, should the estimation history and case abilities be retained? By implication this means that the sampled and retained parameter estimates in the chain are kept in the next estimation run and contribute to the final estimates of the model parameters. Each subsequent call to estimate will be reflected by incrementing the value “RunNo” in the history file. See command export and argument “history”. The default is `yes`.

`keepestimates =`*`bool`*
When additional call to the command estimate are made, should any point estimates (JML, MLE, WLE, EAP) and case fit that have previously been computed be retained? This option is ignored unless `retainchain =`*`true`*. The default is `yes`.

`retaininits =`*`n`*
If there are initial values, for what proportion of the burn are these initial values held constant. Valid values are between 0 and 1. The default is `0.5`.

4.7.22.3 Redirection

Redirection is not applicable to this command.

4.7.22.4 Examples

```
estimate;
```

Estimates the currently specified model using the default value for all options.

```
estimate ! method=jml;
```

Estimates the currently specified model using joint maximum likelihood.

```
estimate ! converge=0.0001, method=quadrature, nodes=15;
```

Estimates the currently defined model using the `quadrature` method of integration. It uses 15 nodes for each dimension and terminates when the change in parameter estimates is less than 0.0001 or after 200 iterations (the default for the `iterations` option), whichever comes first.

```
estimate ! method=montecarlo, nodes=200, converge=.01;
```

In this estimation, we are using the Monte Carlo integration method with 200 nodes and a convergence criterion of 0.01. This analysis (in conjunction with export statements for the estimated parameters) is undertaken to provide initial parameter estimates for a more accurate analysis that will follow.

```
estimate ! method=montecarlo, nodes=2000;
show cases ! estimates=latent >> mdim.pls;
```

Estimate the currently defined model using the Monte Carlo integration method with 2000 nodes. After the estimation, write plausible values, EAP estimates, residual variance and reliability to the file `mdim.pls`.

```
score (0,1,2,3,4) (0,1,2,3,4) ( ) ! tasks(1-9);
score (0,1,2,3,4) ( ) (0,1,2,3,4) ! tasks(10-18);
model tasks + tasks*step;
estimate ! fit=no, method=montecarlo, nodes=400, converge=.01;
```

Initiates the estimation of a partial credit model using the Monte Carlo integration method to approximate multidimensional integrals. This estimation is done with 400 nodes, a value that will probably lead to good estimates of the item parameters, but the latent variance-covariance matrix may not be well estimated. Simulation studies suggest that 1000 to 2000 nodes may be needed for accurate estimation of the variance-covariance matrix. We are using 400 nodes here to obtain initial values for input into a second analysis that uses 2000 nodes. We have specified `fit=no` because we will not be generating any displays and thus have no need for this data at this time. We are also using a convergence criterion of just 0.01, which is appropriate for the first stage of a two-stage estimation.

4.7.22.5 GUI Access

`Analysis` → `Estimate`.

4.7.22.6 Notes

1. ACER ConQuest offers three approximation methods for computing the integrals that must be computed in marginal maximum likelihood estimation (MML): `quadrature` (Bock/Aitken quadrature), `gauss` (Gauss-Hermite quadrature) and `montecarlo` (Monte Carlo). The `gauss` method is generally the preferred approach for problems of three or fewer dimensions, while the `montecarlo` method is preferred in problems with higher dimensions. `gauss` cannot, however, be used when there are regressors or if the distribution is `discrete`.
2. In the absence of regression variables, the `gauss` method is the default method. In the presence of regression variables `quadrature` is the default.
3. Joint maximum likelihood (`JML`) cannot be used if any cases have missing data for all of the items on a dimension.
4. The order in which `command` statements can be entered into ACER ConQuest is not fixed. There are, however, logical constraints on the ordering. For example, `show` statements cannot precede the `estimate` statement, which in turn cannot precede the `model`, `format` or `datafile` statements, all three of which must be provided before estimation can take place.
5. The iterations will terminate at the first satisfaction of any of the `converge`, `deviancechange` and `iterations` options. Except for `method = patz` when all iterations are always completed.
6. Fit statistics can be used to suggest alternative models that might be fit to the data. Omitting fit statistics will reduce computing time.
7. Simulation results illustrate that 10 nodes per dimension will normally be sufficient for accurate estimation with the `quadrature` method.
8. The `stderr=quick` is much faster than `stderr=empirical` and can be used for single faceted models with `lconstraint=cases`. In general, however, to obtain accurate estimates of the errors (for example, to judge whether DIF is observed by comparing the estimates of some parameters to their standard errors, or when you have a large number of facets, each of which has only a couple of levels) `stderr=quick` is not advised.
9. It is possible to recover the ACER ConQuest estimate of the latent ability correlation from the output of a multidimensional analysis by using plausible values. Plausible values can be produced through the `estimate` command or through the `show` command with argument `cases` in conjunction with the option `estimates=latent`.
10. The default settings of the `estimate` command will result in a Gauss-Hermite method that uses 15 nodes for each latent dimension when performing the integrations that are necessary in the estimation algorithm. For a two-dimensional

model, this means a total of $15^2 = 225$ nodes. The total number of nodes that will be used increases exponentially with the number of dimensions, and the amount of time taken per iteration increases linearly with the number of nodes. In practice, we have found that a total of 4000 nodes is a reasonable upper limit on the number of nodes that can be used.

11. If the estimation method chosen is `JML`, then it is not possible to estimate item scores.
12. In the case of MML estimation, ability estimate matrices are only available if `abilities=yes`, is used.
13. To create a file containing plausible values and EAP estimates for all cases use the `show` command with the argument `request_type = cases` and the option `estimates=latent`. (As in the fifth example above.)
14. The estimation history is accessible via the command export and argument "history".

4.7.23 execute

Runs all commands up to the execute command.

4.7.23.1 Argument

This command does not have an argument.

4.7.23.2 Options

This command does not have options.

4.7.23.3 Redirection

Redirection is not applicable to this command.

4.7.23.4 Example

```
let length=50;
execute;
dofor i=1-1000;
```

```
data file_%i%.dat;
format responses 1-%length%;
model item;
estimate;
show >> results_%i%.shw;
enddo;
```

If this code is submitted as a batch, the `execute` command ensures the length token is substituted prior to the execution of the loop. Without the `execute` the substitution of the token would occur after the loop is executed, which would result in much slower command parsing.

4.7.23.5 GUI Access

Access to this command through the GUI is not available.

4.7.23.6 Notes

1. The `execute` command is the only ACER ConQuest command that cannot be abbreviated and must be in lower case and with no space prior to the semi-colon.
2. An `execute` statement cannot be contained in a loop.

4.7.24 export

Creates files that contain estimated values for any of the parameters, a file that contains the design matrix used in the estimation, a scored data set, an iteration history, or a log file containing information about the estimation.

4.7.24.1 Argument

`info type`
`info type` takes one of the values in the following list and indicates the type of information that is to be exported. The format of the file that is being exported will depend upon the *`info type`*.

- *parameters* or *xsi*
  The file will contain the estimates of the item response model parameters. If text output is requested the format of the file is identical to that described for the `import` command argument `init_parameters`.

- *reg_coefficients* or *beta*
  The file will contain the estimates of the regression coefficients for the population model. If text output is requested the format of the file is identical to that described for the `import` command argument `init_reg_coefficients`.

- *covariance* or *sigma*
  The file will contain the estimate of the variance-covariance matrix for the population model. If text output is requested the format of the file is identical to that described for the `import` command argument `init_covariance`.

- *tau*
  The file will contain the estimates of the item scoring parameters. If text output is requested the format of the file is identical to that described for the `import` command argument `init_tau`.

- *itemscores*
  The file will contain the estimated scores for each category of each item on each dimension. Please note that itemscores are NOT model parameters, they are the interaction/product of taus and the scoring matrix. If only initial or anchor taus want to be specified it is therefore important to export and read in TAUS rather than item scores.

- *designmatrix* or *amatrix*
  The file will contain the design matrix that was used in the item location parameter estimation. The format of the file will be the same as the format required for importing a design matrix.

- *cmatrix*
  The file will contain the design matrix that was used in the scoring parameter estimation. The format of the file will be the same as the format required for importing a design matrix.

- *logfile*
  The file will contain a record of all statements that are issued after it is requested, and it will contain results on the progress of the estimation.

- *scoreddata*
  The file will contain scored item response vectors for each case. The file contains one record per case. It includes a sequence number, then a pid (if provided, otherwise the sequence number) followed by scored responses to each (generalised) item.

- *history*
  The file will contain a record for each estimation iteration showing the deviance and parameter estimates at that time.

- *labels*
  The file will contain currently assigned labels (see labels command for format).

- *gamma* or *threshold*
  The file will contain the Thurstonian thresholds (gammas). Note that the gammas are not parameters of the model, but rather the location on the measure at which the cumulative probability of being in category k, or any higher category equals some probability (usually 0.5, the default). There are, therefore k-1 gammas. The response probability defaults to 0.5, and can be set with the option *rp*.

4.7.24.2 Options

`filetype =`*type*
type can take the value `matrix`, `spss`, `excel`, `csv` or `text`. It sets the format of the output file. This option does not apply to the argument `logfile`, `labels`, or `history`. The default is `text`.

`estimatedscores =`*boolean*
boolean can take the value `yes`, `true`, `1`, `no`, `false` or `0`. Used in conjunction with the argument scoreddata. Sets the type of scoreddata exported. When true, the scored responses (e.g., estimated scores under 2PL models) are exported. When false the raw scores are exported. For example, if the user wants the data file that is used after keys and recodes are applied, then raw scores may be preferred. If the user is interested in analysing the scores relative to the residuals, the estimated scores may be preferred. Raw scores and estimated scores are identical under 1PL models. The default is `false` (raw scores are exported).

`rp =`*number*
rp can be a number greater than 0 and less than 1. *rp* is used in conjunction with the argument *gamma* (or *threshold*) to specify the response probability. The default is `0.5`.

4.7.24.3 Redirection

`>>`*`filename`*
For ***type*** `spss`, `excel`, `csv` or `text` an export file name must be specified. For ***type*** `matrix` redirection is to a matrix variable.

4.7.24.4 Examples

```
export parameters >> p.dat;
```

Item response model parameters are to be written to the file `p.dat`.

```
export amatrix!filetype=matrix>>x;
```

Saves the location design matrix to the matrix object x.

4.7.24.5 GUI Access

`File`→`Export`
Export of each of the file types is accessible as a file menu item.

4.7.24.6 Notes

1. If using text output the format of the export files created by the `xsi`, `beta`, `sigma` and `tau` arguments matches the format of ACER ConQuest import files so that export files can be re-read as either anchor files or initial value files. See the `import` command for the formats of the files.
2. The `logfile` and `labels` arguments can be used at any time. The `scoreddata`, `itemscores`, `history` arguments are only available after a model has been estimated. The `amatrix` and `cmatrix` arguments are available after a build command or after model estimation. The other arguments are only possible after a model has been estimated. The `xsi`, `tau`, `beta`, `sigma`, and `theta` arguments can be used prior to estimation (a file will be written after each iteration) or after estimation (a single file will be written). In this case, the files are updated after each iteration.
3. The export file names remain specified until the export occurs.

4. The best strategy for manually building a design matrix (either item location or scoring) usually involves running ACER ConQuest, using a `model` statement and a `build` statement to generate a design matrix, and then exporting the automatically generated matrix, using the `amatrix` and `cmatrix` arguments. The exported matrix can then be edited as needed and then imported.

4.7.25 filter

Allows specification of a set of item-case combinations that can be omitted from the analysis. Filtering can be based on data in a file or in a matrix variable. The file (or matrix variable) can contain '0' or '1' filter indicators or real values tested against a specified value.

4.7.25.1 Argument

This command does not have an argument.

4.7.25.2 Options

`method =`*`reply`*
`reply` takes the value `binary`, `value` or `range`. If `binary` is used then it is assumed that input data consists of zeros and ones, and item case combinations with a value of '1' are retained. Those with the value '0' will be filtered out of subsequent analyses. If `reply=value` then the value is tested against the `match` option. If `reply=range` the value is tested against the `min` and `max` options. The default is `value`.

`matrixin =`*`name`*
`name` is a matrix variable used as the data source. The dimensions must be number of cases by number of items. This option cannot be used in conjunction with an infile redirection.

`matrixout =`*`name`*
`name` is a matrix variable that is created and with dimensions number of cases by number of items. It will contain a value of '1' for case item combinations retained and a value of '0' for those case item combinations that are filtered out of subsequent analyses.

`filetypein =`*`type`*
`type` can take the value `spss` or `text`. This option describes the format of infile. If an SPSS file is used, it must have the same number of cases as the data set that is being

analysed and it must have number of items plus 2 variables. The first two variables are ignored and the remaining variables provide data for each item. When used with `method` or with `min` and `max` options, the variables must be numeric. The default is `text`.

`filetypeout =`*`type`*
`type` can take the value `spss`, `excel`, `xls`, `xlsx` or `text`. This option sets the format of the results file. The default is `text`.

`match =`*`value`*
Case/item combinations for which the input data matches *`value`* are omitted from analysis, whilst those that do not match are retained. Requires the `method=value` option.

`min =`*`n`*
Case/item combinations for which the input data are less than *`n`* are omitted from analysis. Requires the `method=range` option. The default is `0`.

`max =`*`n`*
Case/item combinations for which the input data are greater than *`n`* are omitted from analysis. Requires the `method=range` option. The default is `1`.

4.7.25.3 Redirection

`<<`*`infilename`*
Read or filter data from file named *`infilename`*.

`>>`*`outfilename`*
`outfilename` is the name of a file of ones and zeros showing which cases/item combinations are retained or omitted.

4.7.25.4 Examples

```
filter ! filetypein=spss, method=value, match=T
  << filter.sav;
```

Filters data when a value of `T` is provided for the case item combinations in the SPSS system file `filter.sav`.

```
filter ! matrixin=f, method=range, min=0.25;
```

Filters data when the value in `f` associated with a case/item combination is less than or equal to `0.25`, or greater than or equal to 1.0

4.7.25.5 GUI access

Access to this command through the GUI is not available.

4.7.25.6 Notes

1. The most common utilisation of filter is to remove outlying observations from the analysis.
2. The format of the SPSS system file produced by `show expected` matches that required by filter as SPSS input file.
3. Filtering is turned on by the filter command and stays in place until a reset command is issued.

4.7.26 fit

Produces residual-based fit statistics.

4.7.26.1 Argument

`L1:L2:…:LN`
The **optional** argument takes the form *`L1:L2:…:LN`* Where *`Lk`* is a list of column numbers in the default fit design matrix. This results in *N* fit tests. In the fit tests the columns in each list are summed to produce a new fit design matrix.

Either an argument or an input file can be specified, but not both.

4.7.26.2 Options

`group =`*`v1`*`[by`*`v2`*`by …]`
An explicit variable to be used as grouping variable or a list of group variables separated using the word `by`. Results will be reported for each value of the group variable, or in the case of multiple group variables, each observed combination of the specified group variables. The variables must have been listed in a previous `group` command. The limit for the number of categories in each group is 1000.

`matrixout =`*`name`*
`name` is a matrix (or a set of matrices) that will be created and will hold the fit results.

The matrix will be added to the workspace. Any existing matrices with matching names will be overwritten without warning. The contents of the matrices is described in section 4.9, Matrix Objects Created by Analysis Commands.

`filetype =`*`type`*
`type` can take the value `spss`, `excel`, `xls`, `xlsx` or `text`. This option sets the format of the output file. The default is `text`.

4.7.26.3 Redirection

`<<`*`infilename`*
A file name for the fit design matrix can be specified. The fit design matrix has the same format as a model design matrix (see `import designmatrix`).

`>>`*`outfilename`*
A file name for the output of results.

4.7.26.4 Examples

```
fit >> fit.res;
```

Uses the default fit design matrix and writes results to the file `fit.res`.

```
fit 1-3:4,5,7 >> fit.res;
```

Performs two fit tests. The first test is based upon the sum of the first three columns of the default fit design matrix and the second is based upon the sum of columns, 4, 5 and 7 of the default fit design matrix. Results are written to the file `fit.res`.

```
fit << fit.des >> fit.res;
```

Uses the fit design matrix in the file `fit.des` and write results to the file `fit.res`.

4.7.26.5 GUI Access

`Analysis`→`Fit`.
Selecting the fit menu item displays the open file dialog box for selection of a file that contains the fit design matrix.

4.7.26.6 Notes

1. An argument and `infile` cannot be combined.
2. At the moment, `filetype=excel` is the same as `filetype=xls` or `filetype=xlsx`.

4.7.27 for

Allows looping of syntax and loop control for the purposes of computation.

4.7.27.1 Argument

```
(range) {
set of ACER ConQuest commands
};
```

range is an expression that must take the form *var*`in`*low*`:`*high* where *var* is a variable and *low* and *high* evaluate to integer numeric values. The numeric values can be a scalar value, a reference to an existing 1x1 matrix variable or a 1x1 submatrix of an existing matrix variable. The numeric values cannot involve computation.

The set of commands is executed with *var* taking the value *low* through to *high* in increments of one.

4.7.27.2 Options

This command does not have options.

4.7.27.3 Redirection

Redirection is not applicable to this command.

4.7.27.4 Example

```
let x=matrix(6:6);
compute k=1;
for (i in 1:6)
{
```

```
  for (j in 1:i)
  {
   compute x[i,j]=k;
   compute k=k+1;
  };
};
print x; print ! filetype=xlsx >> x.xlsx;
print ! filetype=spss >> x.sav;
```

Creates a 6 by 6 matrix of zero values and then fills the lower triangle of the matrix with the numbers 1 to 21. The matrix is then printed to the screen and saved as both an Excel and an SPSS file.

4.7.27.5 GUI Access

Access to this command through the GUI is not available.

4.7.27.6 Notes

1. There are no limits of the nesting of loops.

4.7.28 format

A format command is required if the datafile is a *fixed width text file*. Datafiles that are *csv*, or *spss* format do not require a format command: the format of these filetypes is defined within the datafile command itself.

Describes the layout of the data in a data file by specifying variable names and their locations (either explicitly by column number or implicitly by the column locations that underlie the `responses` variable) within the data file.

4.7.28.1 Argument

A list of space-delimited variables that are to be analysed. Each variable is followed by a column specification.

Every `format` statement argument must include the reserved variable `responses`. The `responses` variable specifies the location of the 'item' responses. The column specifications for `responses` are referred to as the *response block*.

A response-width indicator can be given after the final response block. The width indicator, (a*n*), indicates that the width of each response is *n* columns. All responses must be of the same width.

The reserved variable `pid` links data that are from a single case but are located in different records in the data file. It provides a case identification variable that will be included in case outputs. By default `pid` links data that are from a single case but are located in different records in the input data file. See notes (2), (3) and (14), and the `set` option `uniquepid`.

Additional user-defined variables that are listed in the argument of a format statement are called *explicit variables*.

The reserved word `to` can be used to indicate a range of variables.

A slash (/) in the `format` statement argument means move to the next line of the datafile (see note (5)).

4.7.28.2 Options

A list of user-provided, comma-separated variables that are implicitly defined through the column locations that underlie the `responses` variable. The default implicit variable is `item` or `items`, and you may use either in ACER ConQuest statements.

4.7.28.3 Redirection

Redirection is not applicable to this command.

4.7.28.4 Examples

```
format class 2 responses 10-30 rater 43-45;
```

The user-defined explicit variable `class` is in column 2. Item 1 of the response data is in column 10, item 2 in column 11, etc. The user-defined explicit variable `rater` is in columns 43 through 45.

```
format responses 1-10,15-25;
```

The response data are not stored in a contiguous block, so we have used a comma `,` to separate the two column ranges that form the response block. The above example states that response data are in columns 1 through 10 and columns 15 through 25. Commas are not allowed between explicit variables or within the column specifications for other variables.

```
format responses 1-10 / 1-10;
```

Each record consists of two lines. Columns 1 through 10 on the first line of each record contain the first 10 responses. Columns 1 through 10 on the second line of each record contain responses 11 through 20.

```
format responses 21-30 (a2);
```

If each response takes more than one column, use (`a`*n*) (where *n* is an integer) to specify the width of each response. In the above example, there are five items. Item 1 is in columns 21 and 22, item 2 is in columns 23 and 24, etc. All responses must have the same width.

```
format class 3-6 rater 10-11 responses 21-30 rater 45-46 responses 51-60;
```

Note that `rater` occurs twice and that `responses` also occurs twice. In this data file, two raters gave ratings to 10 items. The first rater's identifier is in columns 10 and 11, and the corresponding ratings are in columns 21 through 30. The second rater's identifier is in columns 45 and 46, and the corresponding ratings are in columns 51 through 60. There is only one occurrence of the variable `class` (in columns 3 through 6). This variable is therefore associated with both occurrences of `responses`. If explicit variables are repeated in a `format` statement, the *n*-th occurrence of `responses` will be associated with the *n*-th occurrence of the other variable(s); or if *n* is greater than the number of occurrences of the other variable(s), the *n*-th occurrence of `responses` will be associated with the highest occurrence of the other variable(s).

```
format responses 11-20 ! task(10);
```

The option `task(10)` indicates that we want to refer to the implicit variable that underlies `responses` as 10 tasks. When no option is provided, the default name for the implicit variable is `item`.

```
format responses 11-20 ! item(5), rater(2);
```

The above example has two user-defined implicit variables: `item` and `rater`. There are five items and two raters. Columns 11 through 15 contain the ratings for items 1 through 5 by rater 1. Columns 16 through 20 contain the ratings for items 1 through 5 by rater 2. In general, the combinations of implicit variables are ordered with the elements of the leftmost variables cycling fastest.

```
format responses 1-48 ! criterion(8), essay(3), rater(2);
```

Columns 1 through 8 contain the eight ratings on essay 1 by rater 1, columns 9 through 16 contain the eight ratings on essay 2 by rater 1, and columns 17 through 24 contain the eight ratings on essay 3 by rater 1. Columns 25 through 48 contain the ratings by rater 2 in a similar way.

```
format pid 1-5 class 12-14 responses 31-50 rater 52-53;
```

The identification variable `pid` is in columns 1 through 5. The variable `class` is in columns 12 through 14. Item response data are in columns 31 through 50. The `rater` identifier is in columns 52 and 53. Here we have assumed that a number of raters have rated the work of each student and that the ratings of each rater have been entered in separate records in the data file. The specification of the `pid` will ensure that all of the records of a particular case are located and identified as belonging together.

```
format pid 1-5 var001 to var100 100-199;
```

The identification variable `pid` is in columns 1 through 5. A set of explicit variables labelled var01 through var100 are defined and read from columns 100-199.

4.7.28.5 GUI Access

`Command`→`Format`.
This dialog box can be used to build a format command. Selecting each of the radio buttons in turn allows the specification of explicit variables, responses and implicit variables. Each specification needs to be added to the format statement.

4.7.28.6 Notes

1. User-provided variable names must begin with an alphabetic character and must be made up of alphabetic characters or digits. Spaces are not allowed in variable names. A number of reserved words that cannot be used as variable names are provided in the List of illegal characters and words for variable names, at the end of this document.

2. The reserved explicit variable `pid` means person identifier or case identifier. If `pid` is not specified in the `format` statement, then ACER ConQuest generates identifier values for each record on the assumption that the data file is 'by case'. If `pid` is specified, ACER ConQuest sorts the records in order of the `pid` field first before processing. While this means that the data for each case need not be all together and thus allows for flexibility in input format, the cost is longer processing time for doing the sort.

3. If `pid` is specified, output to person estimates files include the `pid` and will be in `pid` order. Otherwise output to the files will be in sequential order.

4. The `format` statement is limited to reading 50 lines of data at a time. In other words, the maximum number of slash characters you can use in a `format` statement is 49. See note (8) for the length of a line.

5. The total number of lines in the data set must be exactly divisible by the number of lines that are specified by the use of the slash character (`/`) in the `format` statement. In other words, each record must have the same number of lines.

6. Commas can only be used in the column specifications of the `responses` variable. Column specifications for all other explicit variables must be contiguous blocks.

7. The width (number of columns) specified for each `responses` variable must be the same. For example, the following is **not** permitted:
   `format responses 1-4 (a2) responses 5-8 (a1);`

8. The maximum number of columns in a data file must be less than 3072.

9. If the `format` statement does not contain a `responses` variable in its argument, ACER ConQuest will display an error message.

10. In Rasch modelling, it is usual to identify the model by setting the mean of the item difficulty parameters to zero. This is also the default behaviour for ACER

ConQuest, which automatically sets the value of the 'last' item parameter to ensure an average of zero. If you want to use a different item as the constraining item, then you can read the items in a different order. For example:

```
format id 1-5 responses 12-15, 17-23, 16;
```

would result in the constraint being applied to the item in column 16. But be aware, it will now be called item 12, not item 5, as it is the twelfth item in the response block.

11. The level numbers of the `item` variable (that is, item 1, item 2, etc.) are determined by the order in which the column locations are set out in the response block. If you use

    ```
    format responses 12-23;
    ```

    item 1 will be read from column 12.

    If you use

    ```
      format responses 23,12-22;
    ```

    item 1 will be read from column 23.

12. In some testing contexts, it may be more informative to refer to the `responses` variable as something other than `item`. Specifying a user-defined variable name, such as `task` or `question`, may lead to output that is better documented. However, the new variable name for `responses` must then be used in the `model`, `labels`, `recode`, and `score` statements and any label file to indicate the `responses` variable.

13. If each case has a unique `pid` and the data file contains a single record for each case then use of the `set` option `uniquepid=yes` will result in the `pid` being included in case output files, but processing speed will be increased. This is particularly useful for large data sets (e.g., greater than 10 000 cases) with unique student identifiers. This option should not be used without prior confirmation that the identifiers are unique.

14. The format command is not used when the input file specified in ***`datafile`*** is of type `spss` or `csv`. For these file types the format is automatically generated and can be viewed in the log fle.

4.7.29 generate

Generates data files according to specified options. This can be used to generate a single data set.

4.7.29.1 Argument

This command does not have an argument.

4.7.29.2 Options

`nitems =`*`n1:n2:...:nd`*
ni is the number of items on *i*-th dimension and *d* is the number of dimensions. The default is one dimension of 50 items.

`npersons =`*`p`*
p is the number of people in the test. The default is `500`.

`maxscat =`*`k`*
k is the maximum number of scoring categories for each item. For example, if the items are dichotomous, *k* should be 2. Note that *k* applies to all items, so you can't generate items with different numbers of categories. The default value is `2`.

`itemdist =`*`type`*
type is one of the following to specify the item difficulties distribution: `normal(`*`m:b`*`)`, `uniform(`*`c:d`*`)`, or *`filename`*. `normal(`*`m:b`*`)` draws item difficulties from a normal distribution with mean *m* and variance *b*. `uniform(`*`c:d`*`)` draws item difficulties from a uniform distribution with range *c* to *d*. Supplying the *`filename`* of a file containing item difficulties is the third option. The file should be a standard text file with one line per item parameter. Each line should indicate, in the order given, the item number, the step number and the item parameter value.

For example, the file might look like:

```
1 0 -2.0
1 1  0.2
1 2  0.4
2 0 -1.5
..................
```

Note that the lines with a step number equal to 0 give the item difficulty and that the lines with a step number greater than 0 give the step parameters.

The default value is `uniform(-2:2)`.

`centre =`*`reply`*
Sets the location of the origin for the generated data. If *reply* is `cases`, the items parameters are left as randomly generated, and the cases are adjusted to have a mean of zero. If *reply* is `items`, the item location parameters are set to a mean of zero and the cases are left as generated. If *reply* is `no`, both cases and items are left as generated. The default is `items`.

`scoredist =`*`type`*
type is one of the following to specify the item scores (ie discrimination) distribution: `normal(`*m:b*`)`, `uniform(`*c:d*`)`, or *filename*. `normal(`*m:b*`)` draws item scores from a normal distribution with mean *m* and variance *b*. `uniform(`*c:d*`)` draws item scores from a uniform distribution with range *c* to *d*. Supplying the *filename* of a file containing item scores is the third option. The file should be a standard text file with one line per item parameter. Each line should indicate, in the order given, the item number, the step number and the item score value.

For example, the file might look like:

```
1 1 1.0
1 2 1.5
2 1 0.8
..................
```

The default value is for the scores to be set equal the category label. That is for the Rasch model to apply.

`abilitydist =`*`type`*
type is one of the following to specify the distribution of the latent abilities:

`normal(`*m:b*`)`
`normal2(`*m1:b1:m2:b2:k*`)`
`normalmix(`*m1:b1:m2:b2:p*`)`
`uniform(`*c:d*`)`
`u(`*c:d*`)`
`t(`*d*`)`
`chisq(`*d*`)`

`mvnormal(`*`m1:b1:m2:b2:...md:bd:r12:...:r1d:r23:...:r(d-1)(d)`*`)`
`file_name`

`normal(`*`m:b`*`)` draws abilities from a normal distribution with mean *`m`* and variance *`b`*.

`normal2(`*`m1:b1:m2:b2:k`*`)` draws abilities from a two-level normal distribution. Students are clustered in groups of size *`k`*. The within group mean and variance are *`m1`* and *`b1`* respectively, while the between group mean and variance are *`m2`* and *`b2`* respectively. If a two-level distribution is specified the group-level means of the generated values are written to the generated data file for use in subsequent analysis.

`normalmix(`*`m1:b1:m2:b2:p`*`)` draws abilities from a mixture of two normal distributions with group one mean and variance *`m1`* and *`b1`*, and group two mean and variance *`m2`* and *`b2`*. *`p`* is the proportion of the mixture that is sampled from group one.

`uniform(`*`c:d`*`)` draws abilities from a uniform distribution with range *`c`* to *`d`*.

`u(`*`c:d`*`)` draws abilities from a u-shaped distribution with range *`c`* to *`d`*.

`t(`*`d`*`)` draws abilities from a t distribution with *`d`* degrees of freedom.

`chisq(`*`d`*`)` draws abilities from a standardised (ie scaled to mean zero and standard deviation one) chi squared distribution with *`d`* degrees of freedom.

`mvnormal(`*`m1:b1:m2:b2:...md:bd:r12:...:r1d:r23:...:r(d-1)(d)`*`)` draws abilities from a d-dimensional multivariate normal distribution. *`m1`* to *`md`* are the means for each of the dimensions, *`b1`* to *`bd`* are the variances and *`r12`* to *`r(d-1)(d-1)`* are the correlations between the dimensions. For example, a 3-dimensional multivariate distribution with the following mean vector and variance matrix:

$$\begin{bmatrix} 0.5 \\ 1.0 \\ 0.0 \end{bmatrix} \begin{bmatrix} 1.0 & 0 & -0.2 \\ 0 & 1.0 & 0.8 \\ -0.2 & 0.8 & 1.0 \end{bmatrix}$$

is specified as `mvnormal(0.5:1:1:1:0:1:0:-0.2:0.8)`

Lastly, also the *`filename`* of a file containing abilities can be supplied. If the option `importnpvs` is NOT being used the file should be a standard text file with one line per case. Each line should indicate, in the order given, the case number, and a number of ability values, one per dimension.

For example, in the case of a three-dimensional model the file might look like:

```
1   -1.0   1.45  2.45
```

```
2    0.23  0.01  -0.55
3   -0.45  -2.12  0.33
4   -1.5   0.01  3.05
```

If the option `importnpvs` is being used then the file format should match that of a file produced by `show cases ! estimates=latent`. The number of plausible values and dimensions in the file must match the numbers specified by `importnpvs` and `importndims`. The default value is `normal(0:1)`.

`regfile =`*`filename(v1:v2:v3:...:vn)`*

`filename` is a file from which a set of regression variables can be read. The names of the regression variables are given in parenthesis after the file name, and separated by colons (:) *`v1:...:vn`*.

The values of the regression variables are written into the generated data file for use in subsequent analysis.

The first line of the file must give *`n`* regression coefficients. This is followed by one line per person. Each line should indicate, in the given order, the case number and then the value or regression variable *`v1`*, then *`v2`*, and so on, until *`vn`*.

For example, the file might look like:

```
     3.0 2.1 -0.5
1    0.230 0.400 -3.000
2   -0.450 0.500  2.000
3   -1.500 3.222 -4.000
```

`model =`*`model name`*
Set the type of model. The only valid *`model name`* is `pairwise` which results in the generation of data that follows the Bradley-Terry-Luce (BTL) model.

`matrixout =`*`name`*
`name` is a matrix (or set of matrices) that will be created and will hold the results. Any existing matrices with matching names will be overwritten without warning. The content of each of the matrices is described in section 4.9, Matrix Objects Created by Analysis Commands.

`importnpvs =`*`n`*
`n` is the number of plausible values in an import file that will be used to produce multiple output data sets, one for each plausible value set.

`importndims =`*`n`*
`n` is the number of dimensions in an import file that will be used to produce multiple output data sets, one for each plausible value set.

`group =`*`variable`*
An explicit variable to be used as grouping variable. Used only when importing plausible value and undertaking a posterior predictive model checking. If a group is specified then summary statistics are saved as matrix variables for each group. Groups can only be used if they have been previously defined by a group command and a model has been estimated.

`missingmatrix =`*`matrix variable name`*
`matrix variable name` is a matrix variable (in the workspace) of dimension number of persons by number of items. If the value in the matrix for a person item combination is '0' then that combination is set to missing data. For any value other than '0' data is generated.

The name `incidence` is reserved and if used will result in a missing data pattern that matches that of the most previously estimated data set.

`missingfile =`*`spss file name`*
`spss file name` is an SPSS file with number of persons records by number of items variables. If the value in the data file for a person item combination is '0' then that combination is set to missing data. For any value other than '0' data is generated.

4.7.29.3 Redirection

`>>`*`filename1, filename2, filename3`*
`filename1` is the name of the generated data file. *`filename2`* and *`filename3`* are optional. *`filename2`* is the name of the generated item difficulties file, and *`filename3`* is the name of the generated abilities file. When `abilitydist=normal2` is used the mean of each groups abilities is also written to `filename1`. The mean is for all students in the group with the current student excluded. When `regfile=filename` is used the regression variables are also written to *`filename1`*. If the `scoredist` argument is used and a *`filename2`* is requested then an additional file with the name `filename2_scr` is created and it contains the generated score parameters.

When the option `importnpvs` is used then a set of data files with names `filename1_pvn.dat` will be produced, where `n` runs from one to the number of plausible values.

4.7.29.4 Examples

```
generate ! nitems=30, npersons=300, maxscat=2,
itemdist=item1.dat, abilitydist=normal(0:1) >> sim1.dat;
```

A data set called sim1.dat is created. It contains the responses of 300 students to 30 dichotomously scored items. The generating values of the item difficulty parameters are read from the file item1.dat, and the latent abilities for each person are randomly drawn from a unit normal distribution with zero mean and a variance of 1.

```
generate ! nitems=20, npersons=500, maxscat=3,
itemdist=uniform(-2:2), abilitydist=normal(0:1.5)
>> sim1.dat, sim1.itm, sim1.abl;
```

A data set called sim1.dat is created along with a file containing the generating values of the item parameters (sim1.itm) and another containing the generating values of the latent abilities (sim1.abl). The data set will contain the generated responses of 500 persons to 20 partial credit items with three response categories that are scored 0, 1 and 2 respectively. All of the item parameters were randomly drawn from a uniform distribution with minimum -2 and maximum 2, and the abilities are drawn from a normal distribution with zero mean and a variance of 1.5.

```
generate ! nitems=20, npersons=500, maxscat=3,scoredist=uniform(0.5:2),
itemdist=uniform(-2:2), abilitydist=normal(0:1.5)
>> sim1.dat, sim1.itm, sim1.abl;
```

As for the previous example but with scoring parameters generated and written to the file sim1_scr.itm.

```
generate ! nitems=20, npersons=500, maxscat=3,
abilitydist=normal2(0:0.7:0:0.3:20)
>> sim1.dat, sim1.itm, sim1.abl;
```

A data set called sim1.dat is created along with a file containing the generating values of the item parameters (sim1.itm) and another containing the generating values of the latent abilities (sim1.abl). The data set will contain the generated responses of 500 persons to 20 partial credit items with three response categories that are scored 0, 1 and 2 respectively. All of the item parameters were randomly drawn from a uniform distribution with minimum -2 and maximum 2 (default). The abilities are drawn from a two-level normal distribution with within group zero mean and a variance of 0.7, and between group zero mean and variance of 0.3. The group size is 20. The means of the generated abilities for each group will also be written to the data set (`sim1.dat`). Note that the group mean excludes the current student.

```
generate ! nitems=30, npersons=300, maxscat=2,
itemdist=item1.dat, abilitydist=normal(0:1),
regfile=reg1.dat(gender:ses)>> sim1.dat;
```

A data set called `sim1.dat` is created. It contains the responses of 300 students to 30 dichotomously scored items. The generating values of the item difficulty parameters are read from the file `item1.dat`, and the latent abilities for each person are randomly drawn from the regression model $\theta = \alpha_1 gender + \alpha_2 ses + \epsilon$ where $\alpha_1 gender + \alpha_2 ses$ is computed based upon the information given in `reg1.dat` and ϵ is randomly generated as a unit normal deviate with zero mean and a variance of 1.

```
generate ! nitems=30, npersons=3000, maxscat=2,
scoredist=uniform(0.5:2), abilitydist=normal(0:1), matrixout=2pl >> sim1.dat;
```

A data set called `sim1.dat` is created. It contains the responses of 3000 students to 30 dichotomously scored items with scoring parameters randomly drawn from a uniform distribution with minim 0.5 and maximum 2. The generating values of the item difficulty parameters use the default of a uniform distribution with minimum -2 and maximum 2, and the latent abilities for each person are randomly drawn from a unit normal distribution. The matrixout results in the production of four matrix variables 2pl_items, 2pl_cases, 2pl_scores and 2pl_responses.

```
generate ! nitems=15:15, npersons=3000, maxscat=2,
scoredist=uniform(0.5:2), abilitydist=mvnormal(0:1:0:1:0.5), matrixout=2d2pl >> sim1.d
```

A data set called `sim1.dat` is created. It contains the responses of 3000 students to 30 dichotomously scored items, 15 for each of two dimensions. Scoring parameters are randomly drawn from a uniform distribution with minim 0.5 and maximum 2. The generating values of the item difficulty parameters use the default of a uniform distribution with minimum -2 and maximum 2. The latent abilities for each person are randomly drawn from a bivariate standard normal distribution with correlation 0.5. The `matrixout` option results in the production of four matrix variables 2d2pl_items, 2d2pl_cases, 2d2pl_scores and 2d2pl_responses.

```
generate ! nitems=15:15,importnpvs=50,importndims=2,
npersons=3000,scoredist=uniform(0.5:2),
abilitydist=ex1.pv, matrixout=ex1 >> sim1.dat;
```

A set of data sets called `sim1_pv1.dat` to `sim1_pv50.dat` are created. The data sets contain the responses of 3000 students to 30 dichotomously scored items, 15 for each of two dimensions based upon the plausible values provided in `ex1.pv`. Scoring parameters are randomly drawn from a uniform distribution with minim 0.5 and maximum 2. The generating values of the item difficulty parameters use the default of a standard normal distribution The `matrixout` option results in the production of four matrix variables ex1 _items, ex1 _scores and ex1 _statistics.

4.7.29.5 GUI Access

Access to this command through the GUI is not available.

4.7.29.6 Notes

1. The `generate` command is provided so that users interested in simulation studies can easily create data sets with known characteristics.
2. If `abilitydist=normal2(m1:b1:m1:b1:k)` is used, the total number of persons must be divisible by k.
3. The random number generation is seeded with a default value of '2'. This default can be changed with the `seed` option in the `set` command. Multiple runs of `generate` within one session use a single random number sequence, so any change to the default seed should be made before the first generate command is issued.
4. The `pairwise` model is undimensional and does not use discrimination or ability parameters.

4.7.30 get

Reads a previously saved system file.

4.7.30.1 Argument

This command does not have an argument.

4.7.30.2 Options

This command does not have any options.

4.7.30.3 Redirection

`<<`*`mysysfile.sys;`*
`mysysfile.sys` is the name of an ACER ConQuest system file saved during a previous ACER ConQuest session or earlier in the current session.

4.7.30.4 Example

```
get << mysysfile.sys;
```

Loads the system file `mysysfile.sys`.

4.7.30.5 GUI Access

`File`→`Get System File`.

4.7.30.6 Notes

1. Loading a system file replaces all previously entered commands.

4.7.31 group

Specifies the grouping variables that can be used to subset the data for certain analyses and displays.

4.7.31.1 Argument

A list of explicit variables to be used as grouping variables. The list can be comma-delimited or space-delimited.

4.7.31.2 Options

This command does not have options.

4.7.31.3 Redirection

Redirection is not applicable to this command.

4.7.31.4 Example

```
group age grade gender;
```

Specifies `age`, `grade` and `gender` as grouping variables.

4.7.31.5 GUI Access

`Command`→`Grouping Variables`.

The available grouping variables are shown in the list. Multiple groups can be selected by shift- or control-clicking.

4.7.31.6 Notes

1. Each of the grouping variables that are specified in a `group` statement must take only one value for each measured object (typically a person), as these are 'attribute' variables for each person. For example, it would be fine to use `age` as a group variable, but it would not make sense to use `item` as a regression variable.
2. Group variables are read as strings. If using group variables read from SPSS files that are Numeric in type, they will be converted to strings. See Note 5 in `datafile`.
3. The `group` statement stays in effect until it is replaced with another `group` statement or until a `reset` statement is issued.
4. The `group` statement must be specified prior to estimation of the model.

Table 4.2: Comparison operators

Operator	Meaning
==	equality
=>	greater than or equal to
>=	greater than or equal to
=<	less than or equal to
<=	less than or equal to
!=	not equal to
>	greater than
<	less than

4.7.32 if

Allows conditional execution of commands.

4.7.32.1 Argument

```
(logical condition) {
set of ACER ConQuest commands
};
```

If *logical condition* evaluates to true, the *set of ACER ConQuest commands* is executed. The commands are not executed if the *logical condition* does not evaluate to true.

The *logical condition* can be `true`, `false` or of the form *s1 operator s2*, where *s1* and *s2* are strings and *operator* is one of the following:

For each of *s1* and *s2* ACER ConQuest first attempts to convert it to a numeric value. The numeric value can be a scalar value, a reference to an existing 1x1 matrix variable or a 1x1 submatrix of an existing matrix variable. A numeric value cannot involve computation.

If *s1* is a numeric value the operator is applied numerically. If not a string comparison occurs between *s1* and *s2*.

4.7.32.2 Options

This command does not have options.

4.7.32.3 Redirection

Redirection is not applicable to this command.

4.7.32.4 Example

```
x=fillmatrix(20, 20, 0);
compute k=1;
for (i in 1:20)
{
  for (j in 1:i)
  {
   if (j<i)
   {
     compute x[i,j]=k;
     compute x[j,i]=-k;
     compute k=k+1;
   };

   if (j==i)
   {
    compute x[i,j]=j;
   };
  };
};

print x;
```

Creates a 20 by 20 matrix of zero values and then fills the lower triangle of the matrix with the numbers 1 to 190, the upper triangle with -1 to -190 and the diagonal with the numbers 1 to 20. The matrix is then printed to the screen.

4.7.32.5 GUI Access

Access to this command through the GUI is not available.

4.7.32.6 Notes

1. There are no limits on the nesting of conditions.

4.7.33 import

Identifies files that contain initial values for any of the parameter estimates, files that contain anchor values for any of the parameters, or a file that contains a design matrix.

4.7.33.1 Argument

info type
info type takes one of the values in the following list and indicates the type of information that is to be imported. The format of the file that is being imported will depend upon the *info type*.

- `init_parameters` or `init_xsi`
  Indicates initial values for the response model parameters. The file will contain two pieces of information for each response model parameter that has an initial value specified: the parameter number and the value to use as the initial value. The file must contain a sequence of values with the following pattern, in the order given: parameter number, initial value, parameter number, initial value, and so forth.

  For example, the following may be the contents of an `init_parameters` file:

  ```
  1   0.567
  2   1.293
  3   -2.44
  8   1.459
  ```

- `init_tau`
  Indicates initial values for the tau scoring parameters used with the `scoresfree` option. The file will contain two pieces of information for each tau parameter that

has an initial value specified: the parameter number and the value to use as the initial value. The file must contain a sequence of values with the following pattern, in the order given: parameter number, initial value, parameter number, initial value, and so forth. Details of the tau parameterisation can be found in ACER ConQuest note "Score Estimation and Generalised Partial Credit Models (revised)".

For example, the following may be the contents of an `init_tau` file:

```
1   0.5
2   1.293
3   2.44
4   1.459
```

- `init_reg_coefficients` or `init_beta`
  Indicates initial values for the regression coefficients in the population model. The file will contain three pieces of information for each regression coefficient that has an initial value specified: the dimension number, the regression coefficient number, and the value to use as the initial value. Dimension numbers are integers that run from 1 to the number of dimensions, and regression coefficient numbers are integers that run from 0 to the number of regressors. The zero is used for the constant term. When there are no regressors, 0 is the mean. The file must contain a sequence of values with the following pattern: dimension number, regressor number, initial value, dimension number, regressor number, initial value, and so forth.

  For example, the following may be the contents of an `init_reg_coefficients` file:

  ```
  1   0  0.233
  2   0  1.114
  1   1 -0.44
  2   1 -2.591
  ```

  If you are fitting a one-dimensional model, you must still enter the dimension number. It will, of course, be 1.

- `init_covariance` or `init_sigma`
  Indicates initial values for the elements of the population model's variance-covariance matrix. The file will contain three pieces of information for each element of the covariance matrix that has an initial value specified: the two dimension specifiers and the value to use as the initial value. Dimension specifiers are integers that run

from 1 to the number of dimensions. As the covariance matrix is symmetric, you only have to input elements from the upper half of the matrix. In fact, ACER ConQuest will only accept initial values in which the second dimension specifier is greater than or equal to the first. The file must contain a sequence of values with the following pattern: dimension specifier one, dimension specifier two, initial value, dimension specifier one, dimension specifier two, initial value, and so forth.

For example, the following may be the contents of an `init_covariance` file

```
1   1  1.33
1   2 -0.11
2   2  0.67
```

If you are fitting a one-dimensional model, the variance-covariance matrix will have only one element: the variance. In this case, you must still enter the dimension specifiers in the file to be imported. They will, of course, both be 1.

- `init_theta`
  Indicates initial values for the cases under JML or MCMC. Ignored under MML. The file must contain three values: the case number (case sequence ID - note that if you use a PID, this may result in the data being reordered), the dimension number, and the initial value.

  For example, the following may be the contents of an `init_theta` file:

```
1   1 0.567
2   1 1.293
3   1 -2.44
8   1 1.459
```

- `anchor_parameters` or `anchor_xsi`
  The specification of this file is identical to the specification of the `init_parameters` file. The values, however, will be taken as fixed; and they will not be altered during the estimation.

- `anchor_tau`
  The specification of this file is identical to the specification of the `init_tau` file. The values, however, will be taken as fixed; and they will not be altered during the estimation.

- `anchor_reg_coefficients` or `anchor_beta`
  The specification of this file is identical to the specification of the `init_reg_coefficients` file. The values, however, will be taken as fixed; and they will not be altered during the estimation.

- `anchor_covariance` or `anchor_sigma`
  The specification of this file is identical to the specification of the `init_covariance` file. The values, however, will be taken as fixed; and they will not be altered during the estimation.

- `anchor_theta`
  The specification of this file is identical to the specification of the `init_theta`. The values, however, will be taken as fixed; and they will not be altered during the estimation. Indicates initial values for the cases under JML or MCMC. Ignored under MML.

- `designmatrix` or `amatrix`
  Specifies an arbitrary item response model. For most ACER ConQuest runs, the model will be specified through the combination of the `score` and `model` statements. However, if more flexibility is required than these statements can offer, then an arbitrary design matrix can be imported and estimated. For full details on the relations between the `model` statement and the design matrix and for rules for defining design matrices, see Design Matrices (section 3.1.7) and Volodin and Adams (Volodin & Adams, 1995).

- `cmatrix`
  Specifies an arbitrary model for the estimation of the tau scoring parameters used with the `scoresfree` option of the model command. A default scoring design is provided for ACER ConQuest runs using the `scoresfree` option, but explicit specification of the Cdesign matrix allows more flexibility. For full details on the relations between the model statement and the C-design matrix and for rules for defining Cdesign matrices, see ACER ConQuest note "Score Estimation and Generalised Partial Credit Models (revised)".

4.7.33.2 Options

`filetype =`*`type`*
`type` can take the value `matrix` or `text` when importing A and C Matrix designs. For all other parameter types, *`type`* must be `text`. The default is `text`.

`all =`*`NUMBER`* or *`off`*
`NUMBER` sets a value for all parameters of this type. *`off`* turns off all anchors for this parameter type.

4.7.33.3 Redirection

`<<`*`filename`*
An import file name must be specified.

4.7.33.4 Examples

```
import init_parameters << initp.dat;
```

Initial values for item response model parameters are to be read from the file `initp.dat`.

```
import init_parameters << initp.dat;
import anchor_parameters << anch.dat;
```

Initial values for some item response parameters are to be read from the file `initp.dat`, and anchor values for other item response parameters are to be read from `anch.dat`.

```
import designmatrix << design.mat;
```

Imports a design matrix from the file `design.mat`.

```
import designmatrix ! filetype = matrix << m;
```

Imports a design matrix from an internal matrix object names *m*. Using matrix objects can be helpful if import information is stored in other file formats, including `spss` and `csv` files.

4.7.33.5 GUI Access

`File`→`Import`.

Import of each of the file types is accessible as a file menu item.

4.7.33.6 Notes

1. After being specified, all file imports remain until a `reset` statement is issued.
2. If any parameter occurs in both an anchor file and an initial value file, then the anchor value will take precedence.
3. If any parameter occurs more than once in an initial or anchor value file (or files), then the most last read value is used.
4. Initial value files and anchor values files can contain any subset of the full parameter set.
5. Importing and exporting cannot occur until the `estimate` statement is executed. If a model has been estimated then an `export` statement writes the current estimates to a file. If a model has not been estimated then an export of results will occur immediately after estimation. Also see note 8.
6. Importing does not result in a change to the internally held estimates until a subsequent estimation command is issued.
7. You can use the same file names for the import and export files in an analysis: initial values will be read from the files by the `import` statement, and then the `export` statement will overwrite the values in those files with the current parameter estimates as the estimation proceeds or at the end of the estimation.
8. The number of rows in the imported design matrix must correspond to the number of rows that ACER ConQuest is expecting. ACER ConQuest determines this using a combination of the `model` statement and an examination of the data. The `model` statement indicates which combinations of facets will be used to define generalised items. ACER ConQuest then examines the data to find all of the different combinations; and for each combination, it finds the number of categories. The best strategy for manually building a design matrix usually involves running ACER ConQuest, using a `model` statement to generate a design matrix, and then exporting the automatically generated matrix, using the `designmatrix` argument of the `export` statement. The exported matrix can then be edited as needed before importing it with the `designmatrix` argument of the `import` statement.
9. Comments can be included in any initial value or anchor value files. Comments are useful for documentation purpose, they are included between the comment delimiters "/" *and "/"*
10. If a parameter is not identified, ACER ConQuest drops this parameter from the parameter list. This has implications for the parameter sequence numbering in anchor and initial value files. The values in these files must correspond to the parameters numbers **after** removal of non-identified parameters from the parameter list.

11. When using `anchor_theta` under JML, values must be provided for all dimensions for each case that is anchored.

4.7.34 itanal

Performs a traditional item analysis for all of the generalised items.

4.7.34.1 Argument

This command does not have an argument.

4.7.34.2 Options

`format =`*`type`*
`type` can take the value `long`, `summary`, `export`, or `none`. If the type is `summary` then a compact output that includes a subset of information for each item on a single line is provided. If the type is `export` then a complete output is provided but with some omitted formatting. Both `summary` and `export` formats may facilitate reading of the results into other software. If the type is `none` then the output is suppressed. The default is `long`.

The export format is as follows.

- The first 5 lines are headers.
- There is then one line per response category for each item. Each line contains
  - the response label,
  - the score for the response,
  - the number of students who gave the response,
  - this percentage of the total number of respondents to the item who gave the response,
  - the point-biserial for the category,
  - a t-test for the point-biserial
  - the p-value of the t-test
  - the mean ability for students giving this response (based upon plausible values),
  - the standard deviation of ability for students giving this response (based upon plausible values).

If the model is multidimensional additional columns showing mean and standard deviations of abilities for each extra dimension will be shown.

The `summary` format provides a line of information for each generalised item. The information given is restricted to the item label, facility, discrimination, fit and item parameter estimates.

`group =`*v1*`[by`*v2*`by …]`
An explicit variable to be used as grouping variable or a list of group variables separated using the word "by" . Results will be reported for each value of the group variable, or in the case of multiple group variables, each observed combination of the specified group variables. The variables must have been listed in a previous `group` command. The limit for the number of categories in each group is 1000.

`estimates =`*type*
type can take the value `latent`, `wle`, `mle` or `eap`. This option controls the estimator used for the mean and standard deviation of the students that respond in each reported category. The default is `latent`.

`filetype =`*type*
type can take the value `excel`, `xls`, `xlsx` or `text`. This option sets the format of the results file. The default is `text`.

`matrixout =`*name*
name is a matrix (or set of matrices) that will be created and will hold the results. These results are stored in the temporary work space. Any existing matrices with matching names will be overwritten without warning. The contents of the matrices is described in section 4.9, Matrix Objects Created by Analysis Commands. If the the argument `conquestr` to the command 4.7.54, Set is "yes" or "true" then matrix objects are automatically created with the prefix "itan_".

`weight =`*type*
Which caseweight should be applied to the values calculated in itanal? Affects all values, including counts within response categories, classical item statistics, and averages of ability estimates within response categories. *type* can take the value `none`, `raw`, `pvwt` or `mlewt`. The default value for *type* depends on the choice made in the option `estimates`. For example, when `estimates = latent`, `weight` will default to `pvwt`.

4.7.34.3 Redirection

`>>`*filename*

If redirection to a file is specified, the results will be written to that file. If redirection is omitted, the results will be written to the output window or to the console.

4.7.34.4 Examples

```
itanal;
```

Performs a traditional item analysis for all of the generalised items and displays the results in the output window or on the console.

```
itanal >> itanal.out;
```

Performs a traditional item analysis for all of the generalised items and writes the results to the file `itanal.out`.

```
itanal ! estimate=wle, format=export >> itanal.out;
```

Performs a traditional item analysis for all of the generalised items and writes the results to the file `itanal.out` in `export` format. WLE values are used to estimate category means and standard deviations.

4.7.34.5 GUI Access

`Tables`→`Export Traditional Item Statistics`.
Can be used to produce an export format file of traditional statistics.

`Tables`→`Traditional Item Statistics`.
Results in a dialog box. This dialog box is used to select the estimate type, the format and set any redirection.

4.7.34.6 Notes

1. The analysis is undertaken for the categories as they exist after applying `recode` statements but before any recoding that is implied by the `key` statement.

2. Traditional methods are not well-suited to multifaceted measurement. If more than 10% of the response data is missing—either at random or by design (as will often be the case in multifaceted designs)—the test reliability and standard error of measurement will not be computed.
3. Whenever a `key` statement is used, the `itanal` statement will display results for all valid data codes. If the `key` statement is not used, the `itanal` statement will display the results of an analysis done after recoding has been applied.
4. If the `export` format is used the results must be redirected to a file.
5. The `caseweight` command does not influence `itanal` results.

4.7.35 keepcases

List of values for explicit variables that if not matched will cause a record to be dropped from the analysis.

4.7.35.1 Argument

`list of keep codes`
The *`list of keep codes`* is a comma separated list of values that will be treated as keep values for the subsequently listed explicit variable(s).

When checking for keep codes two types of matches are possible. EXACT matches occur when a code in the data is compared to a keep code value using an exact string match. A code will be regarded as a keep value if the code string matches the keep string exactly, including leading and trailing blank characters. The alternative is a TRIM match that first trims leading and trailing spaces from both the keep string and the code string and then compares the results.

The key words `blank` and `dot`, can be used in the keep code list to ensure TRIM matching of a blank character and a period. Values in the list of codes that are placed in double quotes are matched with an EXACT match. Values not in quotes are matched with a TRIM match.

4.7.35.2 Options

A comma separated list of explicit variables.

4.7.35.3 Redirection

Redirection is not applicable to this command.

4.7.35.4 Examples

```
keepcases 7, 8, 9 ! grade;
```

Retains cases where grade is one of 7, 8 or 9.

```
keepcases M ! gender;
```

Sets M as a keep code the explicit variable gender.

4.7.35.5 GUI Access

`Command`→`Keep Cases`.
Displays a dialog box. Select explicit variables from the list (shift-click for multiple selections) and choose the matching keep value codes. The syntax of the keep code list must match that described above for ***`list of keep codes`***.

4.7.35.6 Notes

1. Keep values can only be specified for explicit variables.
2. Complete data records that do not match keep values are excluded from all analyses.
3. If multiple records per case are used in conjunction with a `pid`, then the `keepcases` applies at the record level not the case level.
4. See the `missing` command which can be used to omit specified levels of explicit variables from an analysis and the `delete` command which can be used to omit specified levels of implicit variables from an analysis.
5. See also `dropcases`.
6. When used in conjunction with SPSS input, note that character strings may include trailing or leading spaces and this may have implications for appropriate selection of a match method.

4.7.36 key

Provides an alternative to the `recode` command that may be more convenient when analysing data from a simple multiple choice or perhaps a partial credit test.

4.7.36.1 Argument

codelist
The ***codelist*** is a string that has the same length as the response blocks given in the `format` statement. When a response block is read, the value of the first response in the block will be compared to the first value in the ***codelist*** argument of any `key` statements. Then the value of the second response in the response block will be compared to the second value in the ***codelist***, and so forth. If a match occurs, then that response will be recoded to the value given in the `tocode` option of the corresponding `key` statement, after all the `key` statements have been read.

If leading or trailing blank characters are required, then the argument can be enclosed in double quotation symbols (`" "`).

When one or more `key` statements are supplied, any response that does not match the corresponding value in one of the codelists will be recoded to the value of `key_default`, which is normally `0`. The value of `key_default` can be changed with the `set` command.

If the argument is omitted, then all existing key definitions are cleared.

4.7.36.2 Options

tocode
The value to which matches between the response block and the ***codelist*** are recoded. The column width of the ***tocode*** must be equal to the width of each response as specified in the `format` statement. The ***tocode*** cannot contain trailing blank characters, although embedded or leading blanks are permitted. If a leading blank is required, then the ***tocode*** must be enclosed within double quotation symbols (`" "`).

4.7.36.3 Redirection

Redirection is not applicable to this command.

4.7.36.4 Examples

```
format responses 1-14;
key abcdeaabccabde ! 1;
```

The `format` statement indicates that there are 14 items, with each response taking one column. Any time the first response is coded a, it will be recoded to 1; any time the second response is coded b, it will be recoded to 1; and so on.

```
format responses 1-14 ! rater(2), items(7);
key abcdeaabccabde ! 1;
```

The `format` statement indicates that there are seven items and two raters, with each response taking one column. The recoding will be applied exactly as it is in the first example. Note that this means a different set of recodes will be applied for the items for each rater.

```
format responses 1-14 (a2);
key " a b c d e a a" ! " 1";
```

The `format` statement indicates that there are seven items, with each response taking two columns. Any time the first response is coded a with a leading blank, it will be recoded to 1 with a leading blank. Any time the second response is coded b with a leading blank, it will be recoded to 1 with a leading blank, and so on.

```
format responses 1-14;
key abcdeaabccabde ! 1;
key caacacdeeabccd ! 2;
```

The `format` statement indicates that there are 14 items, with each response taking one column. Any time the first response is coded a, it will be recoded to 1; if it is coded c, it will be recoded to 2. Any time the second response is coded b, it will be recoded to 1; if it is coded a, it will be recoded to 2; and so on.

```
format responses 1-14;
key abcd1111111111 ! 1;
key XXXX2222222222 ! 2;
```

The `format` statement indicates that there are 14 items, with each response taking one column. The item set is actually a combination of four multiple choice and ten partial credit items, and we want to recode the correct answers to the multiple choice items to 1 and the incorrect answers to 0, but for the partial credit items we wish to keep the codes 1 as 1 and 2 as 2. The Xs are inserted in the ***codelist*** argument of the second `key` statement because the response data in this file has no Xs in it, so none of the four multiple choice items will be recoded to 2. While the second `key` statement doesn't actually do any recoding, it prevents the 2 codes in the partial credit items from being recoded to 0, as would have occurred if only one `key` statement had been given.

4.7.36.5 GUI Access

`Command`→`Scoring`→`Key`.
Selecting the key menu item displays a dialog box. This dialog box can be used to build a key command. The syntax requirements for the string to be entered as the Key String are as described above for the ***codelist***.

4.7.36.6 Notes

(1) The recoding that is generated by the `key` statement is applied after any recodes specified in a `recode` statement.
(2) Incorrect responses are not recoded to the `key_default` value (0 unless changed by the `set` command) until all `key` statements have been read and all correct-response recoding has been done.
(3) The `key_default` value can only be one character in width. If the responses have a width that is greater than one column, then ACER ConQuest will pad the `key_default` value with leading spaces to give the correct width.
(4) Whenever a `key` statement is used, the `itanal` command will display results for all valid data codes. If the `key` statement is not used, the `itanal` command will display the results of an analysis done after recoding has been applied.
(5) Any missing-response values (as defined by the `set` command argument `missing`) in ***codelist*** will be ignored. In other words, `missing` overrides the `key` statement.
(6) `tocode` can be a missing-response value (as defined by the `set` command argument `missing`). This will result in any matches between the responses and ***codelist*** being treated as missing-response data.

4.7.37 kidmap

Produces kidmaps.

4.7.37.1 Argument

This command does not have an argument.

4.7.37.2 Options

`cases =`*`caselist`*
`caselist` is a list of case numbers to display. The default is `all`.

`group =`*`v1`*`[by`*`v2`*`by …]`
An explicit variable to be used as grouping variable or a list of group variables separated using the word "by" . Results will be reported for each value of the group variable, or in the case of multiple group variables, each observed combination of the specified group variables. The variables must have been listed in a previous `group` command. The limit for the number of categories in each group is 1000.

`estimates =`*`type`*
`type` can take the value `latent`, `wle`, `mle` or `eap`. This option controls the estimator that is used for the case location indicator on the map. The default is `wle`.

`pagelength =`*`n`*
Sets the length, in lines, of the kidmaps for each case to *`n`*. The default is `60`.

`pagewidth =`*`n`*
Sets the width, in lines, of the kidmaps for each case to *`n`*. The default is `80`.

`orientation =`*`response`*
`response` can be `left` or `right`. This sets the side that the achieved items are placed on. The default is `right`.

`format =`*`response`*
`response` can only be `samoa`. This provides custom headers for kidmaps as developed by The Ministry of Education and Sports and Culture (MESC) in Samoa. There is no default.

4.7.37.3 Redirection

`>>`*`filename`*
If redirection to a file named *`filename`* is specified, the results will be written to that file. If redirection is omitted, the results will be written to the output window or to the console.

4.7.37.4 Examples

```
kidmap;
```

Displays kidmaps for every case in the output window or on the console.

```
kidmap >> kidmap.out;
```

Writes kidmaps for every case to the file `kidmap.out`.

```
kidmap ! cases=1-50, estimate=eap, pagelength=80
>> kidmap.out;
```

Writes kidmaps for cases 1 to 50 to `kidmap.out`. EAP values are used for case locations and the page length for each map is set to 80 lines.

```
kidmap ! group=schid, estimate=eap >> kidmap.out;
```

Writes kidmaps for all cases grouped by schid (in ascending order). The number of groups should not be more than 1000. If grouped output is requested, the `case` option cannot be used and subsets of cases cannot be produced.

4.7.37.5 GUI Access

`Tables`→`Kidmap`.

4.7.37.6 Notes

1. Case fit statistics are only reported if they are available (see `estimate`)
2. If the model is multidimensional, a map is prepared for each dimension. Within-item multidimensional items are omitted from the displays

4.7.38 labels

Specifies labels for any or all of the implicit, variables, explicit variables, dimensions and parameters.

4.7.38.1 Argument

The `labels` statement has two alternative syntaxes. One reads the labels from a file; and one directly specifies the labels.

If the `labels` statement is provided without an argument, then ACER ConQuest assumes that the labels are to be read from a file and that redirection is be provided.

If an argument is provided, it must contain two elements separated by one or more spaces. The first element is the level of the variable (e.g., 1 for item 1), and the second element is the label that is to be attached to that level. If the label contains blank characters, then it must be enclosed in double quotation marks (`" "`).

4.7.38.2 Options

The option is only used when the labels are being specified directly.

`variable name`
The *`variable name`* to which the label applies. The *`variable name`* can be one of the implicit variables or one of the explicit variables or it can be one of `dimensions`, `xsi`, `tau`, or `fitstatistics`.

`dimensions` is used to enter labels for the dimensions in a multidimensional analysis.

`xsi` is used to enter labels for the location parameters in an imported design matrix (amatrix).

`tau` is used to enter labels for the scoring parameters in an imported design matrix (cmatrix).

`fitstatistics` is used to enter labels for the tests in an imported fit matrix.

4.7.38.3 Redirection

`<<`*`filename`*
Specifies the name of a file that contains labels. Redirection is not used when you are directly specifying labels.

The label file must begin with the special symbol `===>` (a string of three equal signs and a greater than sign) followed by a variable name. The following lines must each contain two elements separated by one or more spaces. The first element is the level, and the second element is the label for that level. If a label includes blanks, then that label must be enclosed in double quotation marks (`" "`). The following is an example:

```
===> item
1   BSMMA01
2   BSMMA02
3   BSMMA03
4   BSMMA04
5   BSMMA05

===> rater
1 Frank
2 Nikolai
3 "Ann Marie"
4 Wendy
```

4.7.38.4 Examples

```
labels << example1.nam;
```

A set of labels is contained in the file example1.nam.

```
labels 1 "This is item one" ! item
```

This gives the label 'This is item one' to level 1 for the variable item.

4.7.38.5 GUI Access

`Command`→`Labels`
Direct label specification is only available using the command line interface.

4.7.38.6 Notes

1. The `reset` statement removes all label definitions.
2. Assigning a label to a level for a variable that already has a label assigned will cause the original label to be replaced with the new label.
3. There is no limit on the length of labels, but most ACER ConQuest displays are limited in the amount of the label that can be reported. For example, the tables of parameter estimates produced by the `show` statement will display only the first 11 characters of a label.
4. Labels are not required by ACER ConQuest, but they are of great assistance in improving the readability of any ACER ConQuest printout, so their use is strongly recommended.
5. Labels can also be set by using the option `columnlabels` to the command `datafile`. Note this is only available when the datafile type is `csv` or `spss` and the model contains a single facet (usually "item").

4.7.39 let

Creates an ACER ConQuest token.

4.7.39.1 Argument

`t =`*`string`*
Sets the value of the token `t` to the *`string`*.
or `t = string(`*`value`*`)`
Sets `t` to a string version of the contents of *`value`*. *`value`* must be a 1 by 1 matrix object.

4.7.39.2 Options

This command does not have options.

4.7.39.3 Redirection

Redirection is not applicable to this command.

4.7.39.4 Examples

```
let x=10;
```

Sets the token `x` to the value `10`.

```
let path=/w:cycle2/data/;
```

Sets the token `path` to the value `/w:cycle2/data/`

```
let x=10;
let path=/w:cycle2/data/;
datafile %path%run1.dat;
format responses 1-%x%;
model item;
estimate;
show >> %path%run1.shw;
```

Sets the token `x` to the value `10` and the token `path` to the value `/w:cycle2/data/`. In the subsequent code, the tokens contained between the `%` characters are replaced with the corresponding strings.

```
m=fillmatrix(1,1,2);
let x=string(m);
```

Set token `x` to the value stored in `m`, in this case `2`.

4.7.39.5 GUI access

Access to this command through the GUI is not available.

4.7.39.6 Notes

1. If a token is defined more than once then the last definition takes precedence.
2. A `reset all` command clears all tokens.
3. Tokens implement a simple string substitution; as such they cannot be used until after the `let` command is executed.
4. If a batch of submitted code includes both `let` commands and `dofor` commands, then the `dofor` commands are executed prior to the `let` commands. If large loops (e.g. greater than 100) contain tokens command parsing may be slow. The `execute` command can be used to force execution of the `let` commands prior to loop execution. This will accelerate command parsing.
5. The character `;` can be used in the `let` statement by enclosing the argument in quotes. Eg `let x="print x;";` .
6. The `print` command can be used to display all currently defined variables and tokens.

4.7.40 matrixsampler

Draws a sample of matrices that has a set of marginal frequencies (sufficient statistics) that are fixed and defined by the current data set. The `matrixsampler` implements a Markov Chain Monte Carlo algorithm.

4.7.40.1 Argument

This command does not have an argument.

4.7.40.2 Options

`sets =`*n*
n is the number of matrices to sample. The default is `1000`.

`burn =`*n*
n is the number of matrices to sample and then discard before the first retained matrix. The default is `1000`.

`step =`*n*
n is the number of matrices to sample and then discard before each retained matrix. The default is `64`.

`filetype =`*`type`*
`type` can take the value `spss`, `excel`, `xls`, `xlsx`, `csv` or `text`. This option sets the format of the results file. The default is `text`.

`manyout =`*`reply`*
`reply` can be `yes` or `no`. If `yes`, an output file is created for each sampled matrix. If `manyout=no`, a single file containing all matrices is produced. The default value is `no`.

`matrixout =`*`name`*
`name` is a matrix that will be created and will hold selected summary statistics for the sampled matrices. The content of the matrices is described in section 4.9, Matrix Objects Created by Analysis Commands. This option can also be specified by `results`, which is now deprecated.

`fit =`*`reply`*
`reply` can be `yes` or `no`. If `yes`, a matrix that will be containing estimated item fit statistics for the sampled matrices. The content of the matrices is described in section 4.9, Matrix Objects Created by Analysis Commands.

4.7.40.3 Redirection

`>>`*`filename`*
If redirection to a file named *`filename`* is specified, the results will be written to that file in the format specified by the `filetype` option. If `manyout` is specified then multiple files using the name provided with a file number addition will be produced. If redirection is omitted, then no results will be written.

4.7.40.4 Examples

```
matrixsampler ! filetype=spss >> sampler.sav;
```

Samples 1000 matrices and writes the results to the SPSS system file `sampler.sav`.

```
matrixsampler ! filetype=spss, manyout=yes, results=correlations >> sampler.sav;
```

Samples 1000 matrices and writes them to 1000 separate SPSSsystem files (`sampler_1.sav` to `sampler_1000.sav`). A matrix variable with the name correlations is added to the workspace.

4.7.40.5 GUI access

Access to this command through the GUI is not available.

4.7.40.6 Notes

1. The matrixsampler can take a considerable amount of time, especially with large numbers of items and/or cases.

4.7.41 mh

Reports Mantel-Haenszel statistics.

4.7.41.1 Argument

This command does not have an argument.

4.7.41.2 Options

`gins =`*`ginlist`*
`ginlist` is a list of generalised item numbers. The default is `all`.

`bins =`*`n`*
`n` is the number of groups that the cases are divided into for analysis. See also the option `bintype`. The default is 10 bins.

`bintype =`*`size/width`*
Specifies that the bins are either of equal `size` (in terms of number of cases) or of equal `width` (in terms of logits). The default is `size`. If `bintype=size`, then the `mincut` and `maxcut` options are ignored.

`estimates =`*`type`*
`type` is one of `wle`, `mle`, `eap` and `latent`. This option sets the type of case estimate that is used. The default is `latent`.

`group =`*`variable`*
`variable` is an explicit variable to be used as a grouping variable. Raw data plots will be reported for each value of the group variable. *`variable`* must have been listed in a previous `group` command.

`reference =`*`variable`*
The specification of the reference group used to report Mantel-Haenszel. *`variable`* must have been the value from the group variable.

`mincut =`*`k`*
`k` is the logit cut between the first and second groups of cases. T he default is `-5`.

`maxcut =`*`k`*
`k` is the logit cut between the last and second last groups of cases. The default is `5`.

`keep =`*`keeplist`*
`keeplist` is a list of group identification labels separated by colons. Only those values in the *`keeplist`* will be retained.

`drop =`*`droplist`*
`droplist` is a list of group identification labels separated by colons. Those values in the *`droplist`* will be omitted.

`filetype =`*`type`*
`type` can take the value `excel`, `xls`, `xlsx` or `text`. This option sets the format of the results file. The default is `text`.

`matrixout =`*`name`*
`name` is a matrix that will be created and will hold the results. These results are stored in the temporary work space (see command Print to view the contents of the temporary work space). Any existing matrices with matching names will be overwritten without warning. The contents of the matrix is `r` * 8, where r is the number of rows, equal to the number of generalised items analysed (see option `gins`) * the number of bins (see option `bins`) * the number of groups (which may vary the items). The eight columns are:

- The lower bound of the ability distribution for this bin.
- The upper bound of the ability distribution for this bin.
- The mean ability for this bin.
- The number of cases in this bin.
- The average scored response for cases in this bin (the observed value for the bin)
- The average model expectation for cases in this bin (the expected value for the ability range within the bin)
- The Chi Square statistic for the observed - expected for this bin.
- The *p* value for the hypothesis test that the observed - expected is equal to zero.

4.7.41.3 Redirection

`>>`*`filename`*
If redirection to a file is specified, the results will be written to that file in the format specified by the `filetype` option.

4.7.41.4 Examples

```
mh ! group=gender, reference=M;
```

Performance Mantel-Haenszel analysis based upon gender with `M` as the reference category.

```
mh ! group=gender, reference=M, bins=5, estimates=wle;
```

Performance Mantel-Haenszel analysis based upon gender with `M` as the reference category, and using five groups based upon case WLE estimates.

4.7.41.5 GUI access

`Analysis`→`Mantel-Haenszel`
Results in a dialog box. You can select to report Mantel-Haenszel statistics for all or a subset of items. The command options can then be entered following the syntax guidelines given above.

4.7.41.6 Notes

1. The Mantel-Haenszel statistic can also be accessed in conjunction with the `plot` command.

4.7.42 missing

Sets missing values for each of the explicit variables.

4.7.42.1 Argument

A list of comma separated values that will be treated as missing values for the subsequently listed explicit variable(s).

When checking for missing codes two types of matches are possible. EXACT matches occur when a code in the data is compared to a missing code value using an exact string match. A code will be regarded as missing if the code string matches the missing string exactly, including leading and trailing blank characters. The alternative is a TRIM match that first trims leading and trailing spaces from both the missing string and the code string and then compares the results.

The key words, `blank` and `dot`, can be used in the missing code list to ensure TRIM matching of a blank character and a period. Values in the list of codes that are placed in double quotes are matched with an EXACT match. Values not in quotes are matched with a TRIM match.

4.7.42.2 Option

A list of explicit variables. The list can be comma-delimited or space-delimited. A range of variables can be indicated using the reserved word `to`.

4.7.42.3 Redirection

Redirection is not applicable to this command.

4.7.42.4 Examples

```
missing blank, dot, 99 ! age;
```

Sets blank, dot and 99 (all using a trim match) as missing data for the explicit variable `age`.

```
missing blank, dot, " 99" ! age;
```

Sets blank, and dot (using a trim match) and 99 with leading spaces (using an exact match) as missing data for the explicit variable `age`.

4.7.42.5 GUI Access

`Command→Missing Values`.

Select explicit variables from the list (shift-click for multiple selections) and choose the matching missing value codes. The syntax of the missing code list must match that described above for `list of missing codes`.

4.7.42.6 Notes

1. This command control setting missing values for explicit variables only. For setting the missing values for response data see the `respmiss` option of the `set` command and the `recode` command.

4.7.43 model

Specifies the item response model that is to be used in the estimation. A `model` statement must be provided before any estimation can be undertaken.

4.7.43.1 Argument

The `model` statement argument is a list of additive terms containing implicit and explicit variables. It provides an expression of the effects that describe the difficulty of each of the responses. The argument `rater+item+item*step`, for example, consists of three terms: `rater`, `item` and `item*step`. The `rater` and `item` terms indicate that we are modelling the response probabilities with a main effect for the rater (their harshness, perhaps) and a main effect for the item (its difficulty). The third term, an interaction between `item` and `step`, assumes that the items we are modelling are polytomous and that the step transition probabilities vary with `item` (See note (1)).

Terms can be separated by either a plus sign (`+`) or a minus sign (`-`) (a hyphen or the minus sign on the numeric keypad), and interactions between more than two variables are permitted.

4.7.43.2 Options

`type =`*`model`*

model can be one of `rasch`, `pairwise`, `scoresfree` or `bock`. `rasch` yields a model where scores are fixed. `pairwise` results in the use of a BLT pairwise comparison mode. `scoresfree` results in a generalised model in which item scores are estimated for each item. `bock` results in a generalised model in which scores are estimated for each response category.
The default is `rasch`.

`random =`*facet*
facet needs to be the name of a facet in the current model statement. For this random facet, ConQuest will estimate the variance and report it in the show statement. This option must be used with `method = patz` in the model statement and the facet must be named in the current model statement (e.g., `model = item + rater ! random = rater;`). Currently only one random facet is supported.

`positivescores =`*boolean*
boolean can be `true` or `false`, or equivalently `yes` or `no`. If set to `true` estimated scores (taus) in 2PL models, are forced to be positive. If an estimated value becomes negative, it is set to 0 for the next iteration in the estimation. The default is `false`.

4.7.43.3 Redirection

Redirection is not applicable to this command.

4.7.43.4 Examples

```
model item;
```

The `model` statement here contains only the term `item` because we are dealing with single-faceted dichotomous data. This is the simple logistic model.

```
model item + item * step;
```

This is the form of the `model` statement used to specify the partial credit model. In the previous example, all of the items were dichotomous, so a model statement without the `item*step` term was used. Here we are specifying the partial credit model because we want to analyse polytomous items or perhaps a mixture of dichotomous and polytomous items.

```
model item + step;
```

In this example, we assume that `step` doesn't interact with `item`. That is, the step parameters are the same for all items. Thus we have the rating scale model.

```
model rater + item + rater * item * step;
```

Here we are estimating a simple multifaceted model. We estimate `rater` and `item` main effect and then estimate separate step-parameters for each combination of `rater` and `item`.

```
model item - gender + item * gender;
```

The `model` statement that we are using has three terms (`item`, `gender`, and `item*gender`). These three terms involve two facets, `item` and `gender`. As ACER ConQuest passes over the data, it will identify all possible combinations of the `item` and `gender` variables and construct generalised items for each unique combination. The `model` statement asks ACER ConQuest to describe the probability of correct responses to these generalised items using an item main effect, a gender main effect and an interaction between item and gender.

The first term will yield a set of item difficulty estimates, the second term will give the mean abilities of the male and female students respectively, and the third term will give an estimate of the difference in the difficulty of the items for the two gender groups. This term can be used in examining DIF. Note that we have used a minus sign in front of the `gender` term. This ensures that the gender parameters will have the more natural orientation of a higher number corresponding to a higher mean ability (See note (2)).

```
model rater + criteria + step;
```

This `model` statement contains three terms (`rater`, `criteria` and `step`) and includes main effects only. An interaction term `rater*criteria` could be added to model variation in the difficulty of the criteria across the raters. Similarly, we have applied a single step-structure for all rater and criteria combinations. Step structures that were common across the criteria but varied with raters could be modelled by using the term `rater*step`, step structures that were common across the raters but varied with criteria could be modelled by using the term `criteria*step`, and step structures that varied with rater and criteria combinations could be modelled by using the term `rater*criteria*step`.

```
model essay1 - essay2 ! pairwise;
```

Results in a pairwise comparison model where it is assumed that the explicit variables `essay1` and `essay2` provide information on what has been compared.

```
score (0,1,2,3) (0,1,2,3) ( ) ( ) ( ) ( ) ! item (1-6);
score (0,1,2,3) ( ) (0,1,2,3) ( ) ( ) ( ) ! item (7-13);
score (0,1,2,3) ( ) ( ) (0,1,2,3) ( ) ( ) ! item (14-17);
score (0,1,2,3) ( ) ( ) ( ) (0,1,2,3) ( ) ! item (18-25);
score (0,1,2,3) ( ) ( ) ( ) ( ) (0,1,2,3) ! item (26-28);
model item + item * step;
```

The `score` statement indicates the number of dimensions in the model. The model that we are fitting here is a partial credit model with five dimensions, as indicated by the five score lists in the `score` statements. For further information, see the `score` command.

4.7.43.5 GUI Access

`Command`→`Model`.
This dialog box can be used to build a model command. Select an item from the list and add it to the model statement.

4.7.43.6 Notes

1. The `model` statement specifies the formula of the log odds ratio of consecutive categories for an item. For example, we supply the model statement

   ```
   model rater + item + rater*item*step;
   ```

   If we then use P_{nrik} to denote the probability of the response of person n to item i being rated by rater r as belonging in category k, then the model above corresponds to

   $log(P_{nrik}/P_{nrik-1}) = \theta_n - (\rho_r + \delta_i + \tau_{irk})$

   where θ_n is person ability; ρ_r is rater harshness; δ_i is item difficulty; and τ_{irk} is the step parameter for item i, rater r, and category k.
   Similarly, if we use the `model` statement

   ```
   model -rater + item + rater*item*step;
   ```

then the corresponding model will be

$log(P_{nrik}/P_{nrik-1}) = \theta_n - (-\rho_r + \delta_i + \tau_{irk})$.

2. The signs indicate the orientation of the parameters. A plus sign indicates that the term is modelled with difficulty parameters, whereas a minus sign indicates that the term is modelled with easiness parameters.

3. In section 3.1.7.2, The Structure of ACER ConQuest Design Matrices, we describe how the terms in the `model` statement argument result in different versions of the item response model.

4. The `model` statement can be used to fit different models to the same data. The fitting of a multidimensional model as an alternative to a unidimensional model can be used as an explicit test of the fit of data to a unidimensional item response model. The deviance statistic can be used to choose between models. Fit statistics can be used to suggest alternative models that might be fit to the data.

5. When a partial credit model is being fitted, all score categories between the highest and lowest categories must contain data. (This is not the case for the rating scale model.) See section 2.8, Multidimensional Models for an example and further information.

6. If ACER ConQuest is being used to estimate a model that has within-item multidimensionality, then the set command argument `lconstraints=cases` must be provided. ACER ConQuest can be used to estimate a within-item multidimensional model without `constraints=cases`. This will, however, require the user to define and import a design matrix. The comprehensive description of how to construct design matrices for multidimensional models is beyond the scope of this manual.

7. A `model` statement must be supplied even when a model is being imported. The imported design matrix replaces the ACER ConQuest generated matrix. The number of rows in the imported design matrix must correspond to the number of rows in the ACER ConQuest-generated design matrix. In addition, each row of the imported matrix must refer to the same category and generalised item as those to which the corresponding row of the ACER ConQuest-generated design matrix refers. ACER ConQuest determines this using a combination of the model statement and an examination of the data. The `model` statement indicates which combinations of facets will be used to define generalised items. ACER ConQuest then examines the data to find all of the different combinations; and for each combination, it finds the number of categories.

8. Pairwise models are restricted in their data layout. The format must include at least two explicit variables in addition to the responses. The two explicit variables given in the model describe the objects that are being compared through the matching set of responses. If the first listed variable in the model statement is judged "better" than the second then a response of one is expected, if the second listed variable in the `model` statement is judged "better" than a response of zero is expected.

9. If a model that estimates scores is selected then `lconstraints` must be set to `cases` or `none`. In the case `none` the user will need to ensure other constraints are provided to ensure identification

4.7.44 plot

Produces a variety of graphical displays.

4.7.44.1 Argument

`plot type`
`plot type` takes one of the values in the following list and indicates the type of plot that is to be produced.

- `icc`
  Item characteristic curves (by score category).
- `mcc`
  Item characteristic curves (by response category).
- `ccc`
  Cumulative item characteristic curves.
- `conditional`
  Conditional item characteristic curves.
- `expected`
  Item expected score curves.
- `tcc`
  Test characteristic curve.
- `iinfo`
  Item information function.
- `tinfo`
  Test information function.

- `wrightmap`
  Wright map.
- `ppwrightmap`
  Predicted probability Wright map.
- `infomap`
  Test information function plotted against latent distribution.
- `loglike`
  Log of the likelihood function.

4.7.44.2 Options

`filetype =`*`type`*
`type` can take the values:

- `png`, `bmp`,
- `csv`, `excel`, `xls`, `xlsx` `text`, or
- `rout`

This option sets the format of the output file. When the filetype is `png` or `bmp` an image is implied and this is only available in the GUI version (on Windows). When the format is `csv`,`excel`, `xls`, `xlsx` or `text` a `table` is implied. When the format is `rout` a binary file that can be read by the R library conquestr (Cloney & Adams, 2022) is implied.

`showplot =`*`reply`*
If *`reply`* is `no` the rendering of plots to the display is suppressed. The default is `yes`. Note, plots are only rendered on Windows GUI.

`showtable =`*`reply`* Where *`reply`* is either `yes`, `no`. If *`reply`* is `yes` a data table accompanying each plot is written to the output window. The data table includes a test of fit of the empirical and modelled data. If *`filetype`* is used in conjunction with this option, the data tables accompanying each plot are written file. All files written by plot are specified using outfile redirection `>>`. The default is `no`.

`gins =`*`ginlist`*
`ginlist` is a list of generalised item numbers. For the arguments; `icc`, `ccc`, `expected`, and `iinfo` one plot is provided for each listed generalised item. For the arguments `tcc` and `tinfo` a single plot is provided with the set of listed items treated as a test. The default is `all`.

`bins =`***n***
n is the number of groups of cases that are used for the raw data. The default is `60` for the Wright Maps and `10` for all other plots. For `loglike` it is the number of points to plot.

`bintype =`***reply***
reply can take the value `size` or `width`. `bintype=size` specifies that the bins are of equal size (in terms of number of cases), and `bintype=width` that they are of equal width (in terms of logits). The default is `size`. If `bintype=size`, then the `mincut` and `maxcut` options are ignored. `bintype=width` is not available for Wright Maps.

`binspec =`***reply***
reply can take the value `fixed`, `group`, `gin` or `groupgin`. This option applies when `bintype=size` and is for plots of characteristic curves. `binspec=fixed` specifies that the cuts for the bins are constructed based on the ability distribution for all cases. `binspec=group` specifies that the cuts for the bins are constructed based on the ability distribution for the group being plotted. `binspec=gin` specifies that the cuts for the bins are constructed based on the ability distribution for cases with responses on the generalised item being plotted. `binspec=groupgin` specifies that the cuts for the bins are constructed based on the ability distribution for cases in the group being plotted with responses on the generalised item being plotted . This option may be useful when the ability distribution for groups are very different from the overall sample, or for when an incomplete design (e.g., and adaptive design) means that the ability distribution for the subsample who respond to the item is very different from the overall sample. The default is `fixed`.

`mincut =`***k***
For the arguments; `icc`, `ccc`, `expected`, and `iinfo` ***k*** is the logit cut between the first and second groups of cases. For the arguments `tcc` and `tinfo` ***k*** is the minimum value for which the plot is drawn. The default is `-5`.

`maxcut =`***k***
For the arguments; `icc`, `ccc`, `expected`, and `iinfo` ***k*** is the logit cut between the last and second last groups. For the arguments `tcc` and `tinfo` ***k*** is the maximum value for which the plot is drawn. The default is `5`.

`minscale =`***k***
Specifies the minimum value (***k***) for which the plot is drawn. If this command is not used, the minimum value will be calculated automatically. In `infomap`, this option specifies the minimum value for the vertical axis of the latent distribution.

`maxscale =`***k***
Specifies the maximum value (***k***) for which the plot is drawn. If this command is not used,

the maximum value will be calculated automatically. In `infomap`, this option specifies the maximum value for the vertical axis of the latent distribution.

`raw =`*`reply`*
Controls display of raw data. If *`reply`* is `no` the raw data is not shown in the plot. If *`reply`* is `yes` the raw data is shown in the plot. The default is `yes`.

`legend =`*`reply`*
If *`reply`* is `yes` legend is supplied. The default is `yes` for Wright Maps and `no` for all other plots.

`overlay =`*`reply`*
For the arguments: `icc`, `mcc`, `ccc`, `expected`, `conditional` and `iinfo` if *`reply`* is `yes` the set of requested plots are shown in a single window. If *`reply`* is `no` the set of requested plots are each shown in a separate window.

For the argument `infomap`, in conjunction with `group`, `keep`, and `drop` options, if *`reply`* is `yes` the requested plots for the specified groups are plotted against the information function on the same plot.

For the arguments `tcc` and `tinfo` if *`reply`* is `yes` the requested plots are displayed in the current active plot window. If no window is currently active a new one is created. If *`reply`* is `no` the requested plot is shown in a new separate window. The default is `no`.

This option is not available for Wright Maps.

`estimates =`*`type`*
`type` is one of `wle`, `mle`, `eap` and `latent`. This option sets the type of case estimate that is used for constructing the raw data. The default is `latent`. This option is ignored for the arguments `tcc`, `iinfo` and `tinfo`.

`group =`*`variable`*
`variable` is an explicit variable to be used as grouping variable. Raw data plots will be reported for each value of the group variable. The *`variable`* must have been listed in a previous `group` command.

`mh =`*`variable`*
The specification of the reference group used to report Mantel-Haenszel. The *`variable`* must have been listed as a group variable. The `table` option under `plot` command must be set to `yes` or a *`filename`* specified in order to show the Mantel-Haenszel statistics and it can only be used in conjunction with the arguments `icc`, `mcc`, `ccc`, `conditional` and `expected`. The default is `no`.

`keep =`*`keeplist`*
`keeplist` is a list of group identification labels separated by colons. Only those values in

the *keeplist* will be retained in plots. This option can only be used in conjunction with a `group` option and cannot be used with `drop`.

`drop =`*droplist*
droplist is a list of group identification labels separated by colons. Those values in the `droplist` will be omitted from plots. This option can only be used in conjunction with a `group` option and cannot be used with `keep`.

`bydimension =`*reply*
Only applicable to Wright Maps. If *reply* is `yes` a plot is supplied for each dimension. If *reply* is `no`, all dimensions are printed on a single plot.

`ginlabels =`*reply*
Only applicable to Wright Maps. If *reply* is `yes` each generalised item is labelled. If *reply* is `no` the labels are suppressed. The default is `yes`.

`order =`*reply*
Only applicable to Wright Maps. If *reply* is `value` generalised items are ordered by estimate value. If *reply* is `entry` generalised items are ordered by sequence number. The default is `entry`.

`series =`*reply*
Only applicable to Wright Maps. The default is `all`.

- If *reply* is `all`, a single series is used for display of item parameter estimates.
- If *reply* is `gin`, a series is provided for each generalised item.
- If *reply* is `gingroup`, a series is provided for each defined gingroup.
- If *reply* is `level`, a series is provided for each level of response and
- if *reply* is `dimension`, a series is provided for the generalised items allocated to each dimension. Generalised items are ordered by sequence number.

`xsi =`*n*
n is the item location parameter number for which the likelihood is to be plotted. This option is only applicable for the `loglike` argument.

`tau =`*n*
n is the scoring parameter number for which the likelihood is to be plotted. This option is only applicable for the `loglike` argument.

`beta =`*n1:n2*
n1 is the dimension number and *n2* is the variable number for the regression parameter

for which the likelihood is to be plotted. This option is only applicable for the `loglike` argument.

`sigma =`*`n1:n2`*
`n1` and *`n2`* are the dimensions references for the (co)variance parameter for which the likelihood is to be plotted. This option is only applicable for the `loglike` argument.

`weight =`*`type`*
Which case weight should be applied to the values calculated in itanal? Affects all values, including counts within response categories, classical item statistics, and averages of ability estimates within response categories. *`type`* can take the value `none`, `raw`, `pvwt` or `mlewt`. The default value for *`type`* depends on the choice made in the option `estimates`. For example, when `estimates = latent`, `weight` will default to `pvwt`.

4.7.44.3 Redirection

`>>`*`filename`*
The name or pathname and (optionally) name (in the format used by the host operating system) is appended to the front of the file name of a table, image, or *rout* file.

4.7.44.4 Examples

```
plot icc;
```

Plots item characteristics curves for all generalised items in separate windows.

```
plot icc ! gins=1-4:7;
```

Plots item characteristics curves for generalised items 1, 2, 3, 4 and 7 in separate windows.

```
plot icc ! gins=1-4:7, raw=no, overlay=yes;
```

Overlays item characteristics curve plots for generalised items 1, 2, 3, 4 and 7 in a single window and does not show raw data.

```
plot tcc ! gins=1-4:7, mincut=-10, maxcut=10;
```

Plots a test characteristic curve, assuming a test made up of items 1, 2, 3, 4 and 7 and uses ability range from –10 to 10.

```
plot tcc ! gins=1-6, mincut=-10,maxcut=10;
plot tcc ! gins=7-12, mincut=-10, maxcut=10, overlay=yes;
```

Displays two test characteristic curves in the same plot. One for the first six items and one for items 7 to 12.

```
plot infomap ! minscale=-4, maxscale=4;

plot infomap
    ! minscale=-4, maxscale=4, overlay=yes,
    group=country, keep="country2"
;
```

Displays two latent distributions against the test information function on the same plot. The first latent distribution is for all students. The second distribution is for students in country2. The plot uses latent ability range from -4.0 to +4.0 which is the vertical scale for the latent distribution.

```
plot icc! gins=1:2,showplot=no,showtable=yes,estimates=latent;
```

Displays tables of data relating to generalised items 1 and 2. No image is produced. PVs are used to generate the tables.

```
plot icc! gins=1:2,filetype=png,showplot=yes>>png3_;
```

Displays plots for generalised items 1 and 2 to the screen (Windows GUI only). Saves PNG to the working directory with the prefix, "png3_"

4.7.44.5 GUI Access

The various plot types are accessed through the items in the Plot menu.

4.7.44.6 Notes

1. For dichotomous items the first category is not plotted in the item characteristic curve plot.
2. The last category is not plotted for cumulative item characteristic curves.
3. The item thresholds and item parameters estimates are displayed for the plotted generalised item.
4. If a `pairwise` model has been estimated the only plot available is `wrightmap`.
5. Fit statistics are provided if (a) they have been estimated and (b) if the model is of the form x+x*step.
6. The horizontal axis in `infomap` does not have the same scale in either side of the vertical axis, which is why it is not labelled. The total area under the latent distribution is 1.0. The horizontal scale for the latent distribution side of the horizontal axis is set so that the bin with the largest frequency just fits. The test information function is then scaled to have the same maximum. The total area under the test information function is equal to the number of score points.
7. When `filetype` is *`text`*, *`csv`*, *`excel`*, or *`xlsx`* it is not possible to also set `showtable` is *`yes`*. That is, it is not possible to both display table output to the console *and* to save it as a file.

4.7.45 print

Displays the contents of defined variables and tokens.

4.7.45.1 Argument

`List of variables`, *`tokens`*, a *`quoted string`*, or a valid *`compute expression`*
The *`List of variables`*, *`tokens`* or the *`quoted string`* is printed to the screen, or if requested to a file. If the *`List of variables`* is omitted, then the names of all available variables and the amount of memory they are using is listed.

4.7.45.2 Option

`filetype =`*`type`*
`type` can take the value `csv`, `spss`, `excel`, `xls`, `xlsx` or `text`. This option sets the format of the results file. The default is for the display to be directed to the screen. If `filetype`

is specified, a name for the output file should be given using **redirection**. If `filetype` is specified and no redirection is provided, it will result in an error message.

`decimals =`*`n`*
n is an integer value that sets the number of decimal places to display when printing to the screen. The decimal option is ignored for outputs to files.

`labels =`*`bool`*
If *`yes`* the the row and column labels are displayed if available.

`rows =`*`n`*
n is an integer value that sets describes how many rows should be displayed. The string "all" can be provided to print all of the rows. The defaults is 10 or all of the rows, whichever is smaller.

`columns =`*`n`*
n is an integer value that sets describes how many columns should be displayed. The string "all" can be provided to print all of the columns. The defaults is 10 or all of the columns, whichever is smaller.

4.7.45.3 Redirection

`>>`*`filename`*
If redirection into a file named *`filename`* is specified, the results will be written to that file. If redirection is omitted the results will be written to the output window or to the console. If no redirection is provided and `filetype` has been specified, it will result in an error.

4.7.45.4 Examples

```
print item;
```

Prints the contents of the variable or token, `item`.

```
print "Hello World"
```

Prints the text: Hello World.

```
print;
```

Prints the names of all variables and the memory they consume and all tokens.

```
print counter(10)*y;
```

Prints the content of the result of the computation counter(10)*y.

4.7.45.5 GUI Access

`Workspace→Tokens and Variables`.
Displays a dialog box with the available tokens and variables. The dialog box can be used to print the values of the selected token/variable. A "Columns label" window displays the names for each column of the printed/saved output.

If the selected variable is a matrix you can save the values to a file. Available formats for saving files are text, Excel (.xls or .xlsx), csv or SPSS.

4.7.45.6 Notes

1. The `filetype` option and redirection are only available for matrix variables.

4.7.46 put

Saves a system file.

4.7.46.1 Argument

This command does not have an argument.

4.7.46.2 Options

`compress =`*`response`*
`response` can take the value `yes`, or `no`. To use system files with the conquestr library for R, the system file must be uncompressed (*`response`* equals `no`). The default is *`response`* equals `yes`.

4.7.46.3 Redirection

`>>`*`mysysfile.sys`*
`mysysfile.sys` is the name of an ACER ConQuest system file that will be created. Reading this file with the get command allows the current session to be continued at a later time.

4.7.46.4 Example

```
put >> mysysfile.sys;
```

Saves the system file `mysysfile.sys`.

4.7.46.5 GUI Access

`File`→`Save System file`.

4.7.46.6 Notes

No notes.

4.7.47 quit

Terminates the program. `exit` has the same effect.

4.7.47.1 Argument

This command does not have an argument.

4.7.47.2 Options

This command does not have options.

4.7.47.3 Redirection

Redirection is not applicable to this command.

4.7.47.4 Example

```
quit;
```

ACER ConQuest terminates. All ACER ConQuest system values will be set to their default values when you next run the application.

4.7.47.5 GUI Access

`File`→`Exit`.

4.7.47.6 Notes

1. If you execute a command file that includes a `quit` statement, the `quit` statement will terminate the ACER ConQuest program. If you do not wish to terminate the ACER ConQuest program at that point, omit the `quit` statement from the command file.

4.7.48 read

Read a file into a matrix object.

4.7.48.1 Argument

Name of a matrix object to be created (or replaced if it already exists).

4.7.48.2 Options

`filetype =`*`type`*
`type` can take the value `spss`, `csv` or `text`. The default is `text`.

`header =`*`reply`*
`reply` can be `yes` or `no`. Used for `csv` and `text` files. The default value is `no`.

`nrows =`*`n`*
`n` The number of rows in the matrix object. Required if the file is `text`.

`ncols =`*`n`*
`n` The number of columns in the matrix object. Required if the file is `text`.

4.7.48.3 Redirection

`<<`*`filename`* the name of the file to read.

4.7.48.4 Example

```
read items!filetype=csv,head=yes<<itemsalloriginal.csv;
```

Reads the csv file `itemsalloriginal.csv` into the matrix object *items*

4.7.48.5 GUI Access

Access to this command through the GUI is not available.

4.7.48.6 Notes

1. For text input the stream of values is processed with column cycling fastest.
2. Matrix variables can only contain numeric values. Non-numeric values will be treated as missing values.
3. Column labels will be assigned to the matrix variable based on the header or spss variable names.

4.7.49 recode

Changes raw response data to a new set of values for implicit variables.

4.7.49.1 Argument

`(from1 from2 from3…) (to1 to2 to3…)`
The argument consists of two code lists, the *from* codes list and the *to* codes list. When ACER ConQuest finds a response that matches a *from* code, it will change (or recode) it to the corresponding *to* code. The codes in either list can be comma-delimited or space-delimited.

4.7.49.2 Options

`list of variables and their levels`
Specifies the items to which the recoding in the to codes list should be applied. The default is to apply the recoding to all responses.

4.7.49.3 Redirection

Redirection is not applicable to this command.

4.7.49.4 Examples

```
recode (a b c d) (0 1 2 3);
```

Recode `a` to `0`, `b` to `1`, `c` to `2` and `d` to `3`. The `recode` is applied to all responses.

```
recode (a,b,c,d) (0,1,2,3) ! item (1-10);
```

Recode `a` to `0`, `b` to `1`, `c` to `2` and `d` to `3`. The `recode` is applied to the responses to items 1 through 10.

```
recode (" d" " e") (3 4);
```

Recode `d` with a leading blank to `3`, and recode `e` with a leading blank to `4`. If you want to use leading, trailing or embedded blanks in either code list, they must be enclosed in double quotation marks (`" "`).

```
recode (1 2 3) (0 0 1) ! rater (2, 3, 5-8);
```

The above example states that for raters 2, 3, 5, 6, 7, and 8, recode response data `1` to `0`, `2` to `0`, and `3` to `1`.

```
recode (e,f) (d,d) ! essay (A,B), school("  1001", "  1002", "  1003");
```

Recode responses `e` and `f` to `d` when the essays are `A` and `B` and the school code is `1001`, `1002` or `1003` preceded by two blanks. The options here indicate an **AND** criteria.

```
recode (e,f) (d,d) ! essay (A,B);
recode (e,f) (d,d) ! school("  1001","  1002", "  1003");
```

Recode responses `e` and `f` to `d` when the essays are `A` or `B` or when the school code is `1001`, `1002` or `1003` preceded by two blanks or when both criteria apply. The use of the two `recode` statements allows the specification of an **OR** criteria.

4.7.49.5 GUI Access

`Command`→`Recode`.
The list will show all currently defined implicit variables. To recode for specific variables select them from the list (shift-click for multiple selections) and select Specify Recodes. A recode dialog box will then be displayed. A *from* codes list and a *to* codes list can then be entered following the syntax guidelines given above.

4.7.49.6 Notes

1. The length of the *to* codes list must match the length of the *from* codes list.
2. `recode` statement definitions stay in effect until a `reset` statement is issued.
3. If a `key` statement is used in conjunction with a `recode` statement, then any `key` statement recoding is applied *after* the `recode` statement recoding. The `recode` statement is only applied to the raw response data as it appears in the response block of the data file.
4. Any missing-response value (as defined by the `set` command argument `missing`) in the *from* code list will be ignored.
5. Missing-response values (as defined by the `set` command argument `missing`) can be used in the *to* code list. This will result in any matches being recoded to missing-response data.

6. Any codes in the response block of the data file that do not match a code in the *from* list will be left untouched.
7. When ACER ConQuest models the data, the number of response categories that will be assumed for each item will be deter¬mined from the number of distinct codes after recoding. If item 1 has three distinct codes, then three categories will be modelled for item 1; if item 2 has four distinct codes, then four categories will be modelled for item 2.
8. When a partial credit model is being fitted, all score categories between the highest and lowest categories must contain data. (This is not the case for the rating scale model.) The `recode` statement is used to do this. See section 2.8, Multidimensional Models for an example and further information.
9. A `score` statement is used to assign scores to response codes. If no `score` statement is provided, ACER ConQuest will attempt to convert the response codes to scores. If this cannot be done, an error will be reported.

4.7.50 regression

Specifies the independent variables that are to be used in the population model.

4.7.50.1 Argument

A list of explicit variables to be used as predictors of the latent variable. The list can be comma-delimited or space-delimited. A range of variables can be indicated using the reserved word `to`. The variables can be restricted to particular latent dimensions by replacing dimension numbers in parenthesis after the variable name.

4.7.50.2 Options

This command does not have options.

4.7.50.3 Redirection

Redirection is not applicable to this command.

4.7.50.4 Examples

```
regression age grade gender;
```

Specifies `age`, `grade` and `gender` as the independent variables in the population model; that is, we are instructing ACER ConQuest to regress latent ability on age, grade and gender.

```
regression ses, y1, y2;
```

Specifies `ses`, `y1` and `y2` as the independent variables in the population model.

```
regression ses to y2;
```

Specifies all variables from `ses` to `y2` as independent variables. The variables included from `ses` to `y2` depend on the order given by the user in a previous `format` command (ie if `y1` is listed after `y2` in the `format` command it will not be included in this specification).

```
regression age(2);
```

Regresses dimension two (2) on `age`, but does not regress any other dimensions on age.

```
regression;
```

Specifies a population model that includes a mean only.

4.7.50.5 GUI Access

`Command`→`Regression Model`.
Select regression model variables from the currently defined list of explicit variables (shift-click to make multiple selections).

4.7.50.6 Notes

1. Each of the independent variables that are specified in a `regression` statement must take only one value for each measured object (typically a person), as these are 'attribute' variables for each person. For example, it would be fine to use `age` as a regression variable, but it would not make sense to use `item` as a regression variable.
2. If no `regression` statement is supplied or if no variable is supplied in the `regression` statement, a constant is assumed, and the regression coefficient that is estimated is the population mean.
3. A `constant` term is always added to the supplied list of regression variables.
4. If you want to regress the latent variable onto a categorical variable, then the categorical variable must first be appropriately recoded. For example, dummy coding or contrast coding can be used. A variable used in regression must be a numerical value, not merely a label. For example, gender would normally be coded as 0 and 1 so that the estimated regression is the estimated difference between the group means. Remember that the specific interpretation of the latent regression parameters depends upon the coding scheme that you have chosen for the categorical variable. See the `categorise` command.
5. The `regression` statement stays in effect until it is replaced with another `regression` statement or until a `reset` statement is issued. If you have run a model with regression variables and then want to remove the regression variables from the model, the simplest approach is to issue a `regression` statement with no argument.
6. If any of the independent variables that are specified in a `regression` statement have missing data, the records are deleted listwise. Because of the cumulative effect of listwise deletion, the overall number of records deleted may increase substantially more than the proportion of missing data in each independent variable as more independent variables are added. This has important consequences in terms of parameter bias, especially if the overall missing data rate substantially exceeds the suggested cut-off values in the literature (ranges from 5–20%, see for example Schafer, 1999 and Peng et al., 2006). The point at which the amount of missing data becomes detrimental will depend on a number of factors including the pattern of missingness, and is beyond the scope of this manual. However, it is recommended in these situations that the user *not* use `regression` or alternatively seek other external methods to handle the missing data (e.g., through multiple imputation, FIML, etc).

4.7.51 reset

Resets ACER ConQuest system values to their default values. It should be used when you wish to erase the effects of all previously issued commands.

4.7.51.1 Argument

Can be the word `all` or `blank`. When used without `all`, tokens and variables are not cleared.

4.7.51.2 Options

This command does not have options.

4.7.51.3 Redirection

Redirection is not applicable to this command.

4.7.51.4 Examples

```
reset;
```

Reset all values except tokens and variables.

```
reset all;
```

Reset all values including tokens and variables.

4.7.51.5 GUI Access

`Workspace`→`Reset`.
`Workspace`→`Reset All`.

4.7.51.6 Notes

1. The `reset` statement can be used to separate jobs that are put into a single command file. The `reset` statement returns all values to their defaults. Even though many values may be the same for the analyses in the command file, we advise resetting, as you may be unaware of some values that have been set by the previous statements.
2. When a `reset` statement is issued, the output buffer is cleared automatically, with no prior warning.

4.7.52 scatter

Produces a scatter plot of two variables.

4.7.52.1 Argument

x, *y*
x and *y* must be two existing matrix variables, or a valid compute expression. The matrix variables must each have one column and an equal number of rows. In the case where the compute expression is used, the result must have one column and an equal number of rows to the other variable or expression.

4.7.52.2 Options

`title =`*text*
text to be used as a graph title. The default is `scatter`.

`subtitle =`*text*
text to be used as a graph subtitle. The default is *x*`against`*y*.

`seriesname =`*text*
text to be used as a series name. The default is *x*`against`*y*.

`xlab =`*text*
text to be used as a series name subtitle. The default is the *x*-variable name.

`ylab =`*text*
text to be used as a series name subtitle. The default is the *y*-variable name.

`xmin =`*`k`*
Specifies the minimum value (*k*) for the horizontal axis. If this option is not used, the minimum value will be calculated automatically.

`ymin =`*`k`*
Specifies the minimum value (*k*) for the vertical axis. If this option is not used, the minimum value will be calculated automatically.

`xmax =`*`k`*
Specifies the maximum value (*k*) for the horizontal axis. If this option is not used, the maximum value will be calculated automatically.

`ymax =`*`k`*
Specifies the maximum value (*k*) for the vertical axis. If this option is not used, the maximum value will be calculated automatically.

`legend =`*`reply`*
If *`reply`* is `yes` a legend is supplied. The default is `no`.

`overlay =`*`reply`*
If *`reply`* is `yes` the scatter plot is overlayed on the existing active plot (if there is one). The default is `no`.

`join =`*`type`*
If *`type`* is `yes` the points in the plot are joined by a line. The default is `no`.

4.7.52.3 Redirection

`>>`*`filename`*
The name or pathname (in the format used by the host operating system) is appended to the front of the plot window name and plots are written to a file in PNG graphics file format. If no redirection is provided and `filesave=yes`, plots will be saved to the working directory with the plot window name.

4.7.52.4 Example

```
a=fillmatrix(14,1,0);
b=fillmatrix(14,1,0);
compute a={-18,16,-7,3,8,-4,6,-5,-9,-4,6,5,-12,-15};
compute b={5,4,9,3,7,-6,-5,1,0,-16,2,-13,-17,5};
scatter a,b ! legend=yes, seriesname=A vs B, title=Comparison of A and B;
```

Creates two matrices (a and b) of 14 rows and one column each. Displays a scatter plot of a against b, including a legend with the series name and a title.

4.7.52.5 GUI Access

Access to this command through the GUI is not available.

4.7.52.6 Notes

1. If plots are overlayed the options for the last plot are used for labels and axes ranges.

4.7.53 score

Describes the scoring of the response data.

4.7.53.1 Argument

`(code1 code2 code3…) (score1dim1 score2dim1 score3dim1…) (score1dim2 score2dim2 score3dim2…) …`

The first set of parentheses contains a set of codes (the *codes* list). The second set of parentheses contains a set of scores on dimension one for each of those codes (a *score* list). The third set contains a set of scores on dimension two (a second *score* list) and so on. The number of separate codes in the *codes* list indicates the number of response categories that will be modelled for each item. The number of *score* lists indicates the number of dimensions in the model. The codes and scores in the lists can be comma-delimited or space-delimited.

4.7.53.2 Options

`list of variables and levels`

Specifies the responses to which the scoring should be applied. The default is to apply the scoring to all responses.

4.7.53.3 Redirection

Redirection is not applicable to this command.

4.7.53.4 Examples

```
score (1 2 3) (0 1 2);
```

The code 1 is scored as 0, code 2 as 1, and code 3 as 2 for all responses.

```
score (1 2 3) (0 0.5 1.0);
```

The code 1 is scored as 0, code 2 as 0.5, and code 3 as 1.0 for all responses.

```
score (a b c) (0 0 1);
```

The code a is scored as 0, code b as 0 and code c as 1 for all responses. As there are three separate codes in the *codes* list, the model that will be fitted if this score statement is used will have three response categories for each item. The actual model will be an ordered partition model because both the a and b codes have been assigned the same score.

```
score (a b c) (0 1 2) ! items (1-10);
score (a b c) (0 0 1) ! items (11- 20);
```

The code a is scored as 0, b as 1, and c as 2 for items 1 through 10, while a is scored 0, b is scored 0, and c is scored 1 for items 11 through 20.

```
score ( a , <b,c>, d) (0,1,2) ! items (1-30);
```

The angle brackets in the code list indicate that the codes b and c are to be combined and treated as one response category, with a score of 1. Compare this with the next example.

```
score (a, b, c, d) (0, 1, 1, 2) ! items (1-30);
```

In contrast to the previous example, this score statement says that b and c are to be retained as two separate response categories, although both have the same score of 1.

```
score (a+," a",b+," b",c+," c") (5,4,3,2,1,0) ! essay(1,2), rater(A102,B223);
```

The option list can contain more than one variable. This example scores the responses in this fashion for essays 1 and 2 and raters A102 and B223. Double quotation marks are required when a code has a leading blank.

```
score (1 2 3) (0 1 2) (0 0 0) (0 0 0) ! items (1-8,12);
score (1 2 3) (0 0 0) (0 1 2) (0 0 0) ! items (9,13-16,18);
score (1 2 3) (0 0 0) (0 0 0) (0 1 2) ! items (10,11,17);
```

To fit multidimensional models, multiple score lists are provided. Here, the `score` statement has three score lists after the codes list, so the model that is fitted will be three-dimensional. Items 1 through 8 and item 12 are on dimension one; items 9, 13 through 16 and 18 are on dimension two; and items 10, 11 and 17 are on dimension three. Because each item is assigned to one dimension only (as indicated by the zeros in all but one of the score lists for each `score` statement), we call the model that will be fitted when the above `score` statements are used is a between-item multidimensional model.

```
score (1 2 3) (0 1 2) (     ) ! items (1-8,12);
score (1 2 3) (     ) (0 1 2) ! items (9,13-16,18);
score (1 2 3) (0 1 2) (0 1 2) ! items (10,11,17);
```

If nothing is specified in a set of parentheses in the score list, ACER ConQuest assumes that all scores on that dimension are zero. This sequence of `score` statements will result in a two-dimensional model. Items 1 through 8 and item 12 are on dimension one; items 9, 13 through 16 and 18 are on dimension two; and items 10, 11 and 17 are on both dimension one and dimension two. We call models of this type within-item multidimensional. See note (4).

4.7.53.5 GUI Access

`Command`→`Scoring`→`Non-Key`.
To score for specific variables select them from the list (shift-click for multiple selections) and select Specify Scores. A score dialog box will then be displayed. A *from* codes list to *codes* list can then be entered following the syntax guidelines given above. Scoring needs to be specified for each dimension.

4.7.53.6 Notes

1. When estimation is requested, ACER ConQuest applies all recodes and then scores the data. This sequence is independent of the order in which the `recode` and `score` statements are entered.

2. `Score` statements stay in effect until a `reset` statement is issued.

3. A `score` statement that includes angle brackets results in the automatic generation of a `recode` statement. For example:

   ```
   score ( a , <b,c>, d) (0,1,2);
   ```

   becomes the equivalent of

   ```
   recode (b,c) (b,b);
   score (a,b,d) (0,1,2);
   ```

   and stays in effect until a `reset` statement is issued.

4. A `score` and `model` statement combination can automatically generate within-item multidimensional models only when the `set` command argument `constraints=cases` is specified. To estimate within-item multidimensional models without setting `constraints=cases`, pecify the desired `score` and `model` statements, ignore the warnings that are issued and then supply an imported design matrix.

5. ACER ConQuest makes an important distinction between response categories and response levels (or scores). The number of response categories that will be modelled by ACER ConQuest for an item is determined by the number of unique codes that exist for that item, after performing all recodes. ACER ConQuest requires a score for each response category. This can be provided via the `score` statement. Alternatively, if the `score` statement is omitted, ACER ConQuest will treat the recoded responses as numerical values and use them as scores. If the recoded responses are not numerical values, an error will be reported.

6. In a unidimensional analysis, a `recode` statement can be used as an alternative to a `score` statement. See note (5).

7. The `score` statement can be used to indicate that a multidimensional item response model should be fitted to the data. T he fitting of a multidimensional model as an alternative to a unidimensional model can be used as an explicit test of the fit of the data to a unidimensional item response model.

8. If non-integer scoring is used ACER ConQuest can fit two-parameter models and generalised partial credit.

4.7.54 set

Specifies new values for a range of ACER ConQuest system variables or returns all system values definable through the `set` command to their default values.

4.7.54.1 Arguments

- `addextension =`*`reply`*
  `reply` can be `yes` or `no`. `addextension=no` leaves output file names as specified by user, `addextension=yes` appends an appropriate file extension if the user-specified output filename does not include a valid file extension for the `filetype`. The default value is `yes`. The extensions for accepted file types are the following: `text` → .txt, `excel` → .xls, `xls` → .xls, `xlsx` → .xlsx, `spss` → .sav, `rout` → .rout, `csv` → .csv. See note 7.

- `buffersize =`*`n`*
  Number of character that can be accumulated in the output window. The default is `32676`.

- `conquestr =`*`reply`*
  `reply` can be `yes` or `no`. Using `yes` sets a collection of options that facilitate interface with R. `conquest =`*`reply`* is equivalent to `progress =`*`reply`*, `exit_on_error =`*`reply`* and `warnings =`*`reply`*.

- `directory =`*`directory`*
  Sets the name of the directory that will be assumed as home directory.

- `echo =`*`reply`*
  `reply` can be `yes` or `no`. Using `no` turns off command echoing and suppresses the display of estimation progress. The default value is `yes`.

- `exit_on_error =`*`reply`*
  `reply` can be `yes` or `no`. Using `yes` terminates ACER ConQuest when an error is reported. The default value is `no`. This functionality is designed for use cases where ACER ConQuest is called from another application and an appropriate exit status is required.

- `f_nodes =`*`n`*
  Sets the number of nodes that are used in the approximation of the posterior distributions in the calculation of fit statistics. The default is `2000`.

- `fieldmax =`*`n`*
  `n` can be any positive integer less than `1 048 576`. This is the maximum allowed fields declared in a format statement. The default value is `1 000`.

- `fitdraws =`*`n`*
  Sets the number of draws from the posterior distributions that are used in estimating fit statistics. The default is `5`.

- `innerloops =`*`n`*
  Sets the maximum number of Newton steps that will be undertaken for each item response model parameter in the M-Step. The default value is `10`.

- `iterlimit =`*`n`*
  Sets the maximum number of iterations for which estimation will proceed without improvement in the deviance. The minimum value permitted is `5`. The default value is `100`.

- `lconstraints =`*`type`*
  Sets the way in which item parameter identification ("location") constraints are applied. *`type`* can take the values `smart`, `items`, `cases` or `none`.
  If `lconstraints` is set to `items`, then identification constraints will be applied that make the mean of the parameter estimates for each term in the `model` statement (excluding those terms that include `step` zero). For example, the model `item+rater` would be identified by making the average item difficulty zero and the average rater harshness zero. This is achieved by setting the difficulty of the last item on each dimension to be equal to the negative sum of the difficulties of the other items on the dimension.
  If `lconstraints` is set to `cases`, then:

  - constraints will be applied through the population model by forcing the means of the latent variables (intercept term in population/regression model) to be set to zero and allowing all item parameters to be free.
  - If regressors are included in the model, the conditional mean (intercept term) will be set to zero and other regression parameters freely estimated. If anchors are supplied, then the regression parameters will be fixed at the values provided (including the intercept term, if included in the anchors).
  - The first term in the `model` statement will not have a location constraint imposed, but any additional terms will generate sets of parameter estimates that are constrained to have a mean of zero.

  If the location constraint (`lconstraints`) is set to `smart`, then `lconstraints=cases` will be applied if all regression parameters are found to be anchored; otherwise, `lconstraints=items` will be used.
  The default value is `items` if no `lconstraints` argument is provided.

- `keeplastests =`*`reply`*
  `reply` can be `yes` or `no`. If iterations terminate at a non-best solution then setting `keeplastests` to `yes` will result in current (non-best) parameter estimates being written retained. The default value is `no`.

- `key_default =`*`n`*
  The value to which any response that does not match its corresponding value in a `key` statement (and is not a missing-response code) will be recoded. The default is `0`.

- `logestimates =`*`reply`*
  `reply` can be `yes` or `no`. If a log file is requested, setting `logestimates` to `yes` will result in parameter estimates being written to the log file after every iteration. The default value is `yes`.

- `memorymodel =`*`i`*
  Indicates whether the case records file (responses) will be created on disk or stored in memory. *`i`* can be an integer in 0,1,2,3. `0` is the slowest setting, but uses least memory. `3` is the fastest setting but uses the most memory.

- `mhmax =`*`n`*
  Number of gins that can be included in a call to the command `mh`. The default is `100`. This argument has an alias, `plotwindows =`*`n`*.

- `mle_criteria =`*n*
  The convergence criterion that is used in the Newton-Raphson routine that provides maximum likelihood case estimates. The default is `0.005`.

- `mle_max =`*n*
  The upper limit for an MLE estimate. The default is `15`.

- `mvarmax =`*n*
  n can be any positive integer less than `1 048 576`. This is the maximum number of variables allowed to be declared in the model, including implicit variables, explicit variables, regressors, groups, case weight, and PID. The default value is `1 000`.

- `n_plausible =`*n*
  Sets the number of vectors of plausible values to be drawn for each case when a plausible value file is requested in estimation. The default is `5`. When the number of vectors of plausible values to be drawn for each case is set to a new value, any plausible values that have been drawn will be discarded and new plausible values will need to be re-drawn.

- `nodefilter =`*p*
  Is used when `method=gauss` is chosen for estimation. The nodes with the smallest weight are omitted from the quadrature approximation in the estimation. The set of nodes with least weight which add to the proportion *p* of the density are omitted. This option can dramatically increase the speed for multidimensional models. The default is `p=0`.

- `outerloops =`*n*
  Sets the maximum number of passes through item response model parameters in the M-Step after population parameters have converged. The default value is `5`.

- `p_nodes =`*n*
  Sets the number of nodes that are used in the approximation of the posterior distributions, which are used in the drawing of plausible values and in the calculation of EAP estimates. The default is `2000`.

- `plotwindows =`*n*
  Number of plot windows that can be displayed at one time. The default is `100`. This is also relevant to users exporting tables or *rout* files using the plot command. This argument has an alias, `mhmax =`*n*.

- `progress =`***`reply`***
  `reply` can be `yes` or `no`. Using `no` turns off status messages. The default value is `yes`.

- `respmiss =`***`reply`***
  Controls the values that will be regarded as missing-response data. ***`reply`*** can be `none`, `blank`, `dot` or `both`. If `none` is used, no missing-response values are used. If `blank` is used, then blank response fields are treated as missing-response data. If `dot` is used, then any response field in which the only non-blank character is a single period (.) is treated as missing-response data. If `both` is used, then both the blank and the period are treated as missing-response data. The default is `both`.

- `sconstraint =`***`type`***
  Sets the scale constraints. ***`type`*** can take the values `cases` or `none`.
  If `sconstraint` is specified to be `cases`, the latent variance for all dimensions is set to 1. In multidimensional models the covariance matrix is therefore the correlation matrix. If `sconstraints` takes the value `none`, the latent variance can be freely estimated. Note that anchored scores (taus) may be required in order for a model to be identified when `sconstraints` is `none`. See the command `import`.
  The default value is `cases`.

- `scoresmax =`***`n`***
  `n` can be any positive integer. This is the maximum allowed value for a score (tau) parameter in a 2PL model. Estimated values greater than ***`n`*** will be set to ***`n`***. The default value is `5`.

- `seed =`***`n`***
  Sets the seed that is used in drawing random nodes for use in Monte Carlo estimation method and in simulations runs. ***`n`*** can be any integer value or the word `date`. If `date` is chosen the seed is the time in seconds since January 1 1970. The default seed is `2`. When the seed is set to a new value, any plausible values that have been drawn will be discarded and new plausible values will need to be re-drawn.

- `skipwtzero =`***`reply`*** Indicates whether cases with weights of zero should be included in analysis and tabulations of summary statistics (e.g., the count of cases in the data file). See the command caseweight. The default is `yes`.

- `softkey =`***`key`***
  Activates a license key. Where ***`key`*** is a valid key provided by ACER. Requires restart. See License key instructions

- `storecommands =`*`reply`*
  `reply` can be `yes` or `no`. Using `yes` stores in memory the commands that were run. These commands can be outputted to a file for recording purposes via the `chistory` command. The default value is `yes`.

- `uniquepid =`*`reply`*
  `reply` can be `yes` or `no`. Use `yes` for datasets with unique PIDs (i.e., each record corresponds to only one case and only one PID; see `format` command) to drastically reduce the processing time especially for large datasets. The default value is `no`.

- `warnings =`*`reply`*
  `reply` can be `yes` or `no`. If `warnings` are set to `no`, then messages that do not describe fatal or fundamental errors are suppressed. The default value is `yes`.

- `zero/perfect =`*`r`*
  If maximum likelihood estimates of the cases are requested, then this value is used to compute finite latent ability estimates for those cases with zero or perfect scores. The default value is `0.3`.

4.7.54.2 Options

This command does not have options.

4.7.54.3 Redirection

Redirection is not applicable to this command.

4.7.54.4 Examples

```
set lconstraints=cases, seed=20;
```

Sets the identification constraints to `cases` and the seed for the Monte Carlo estimation method to 20.

```
set;
```

Returns all of the `set` arguments to their default values.

4.7.54.5 GUI Access

`Workspace`→`Set`.

4.7.54.6 Notes

1. All of the `set` arguments are returned to their default values when a `set` statement without an argument is issued. If a model has been estimated, then issuing this statement will require that the model be re-estimated before `show` or `itanal` statements are issued.
2. If the `set` statement has an argument, then only those system variables in the argument will be changed.
3. The `key_default` value can only be one character in width. If the responses have a width that is greater than one column, then ACER ConQuest will pad the `key_default` value with leading spaces to give the correct width.
4. If `warnings` is set to `no`, then the output buffer will be automatically cleared, without warning, whenever it becomes full. This avoids having to respond to the 'screen buffer is full' messages that will be displayed if you are running an analysis using the GUI interface.
5. ACER ConQuest uses the Monte Carlo method to estimate the mean and standard deviation of the marginal posterior distributions for each case. The system value `p_nodes` governs the number of random draws in the Monte Carlo approximations of the integrals that must be computed.
6. `lconstraints=cases` must be used if you want ACER ConQuest to automatically estimate models that have within-item multidimensionality. If you want ACER ConQuest to estimate within-item multidimensional models without the use of `lconstraints=cases`, you will have to define and import your own design matrices. The comprehensive description of how to construct design matrices for multidimensional models is beyond the scope of this manual.
7. Note that the `filetype` option, in conjunction with the default `addextension=yes`, will append the default file extension if it is not a valid file extension for that filetype. For example, in the command `show`, if the file type was specified as text and the user chooses an .xls extension for the output filename, the resulting file will have ".txt" appended and will still be text and cannot be opened as an Excel workbook. Where the user specifies `addextension=no`, the `filetype` option will still be honoured, and in the example above a text file will be written with an ".xls" file extension. This file may not behave as expected depending on the file assocations set by the user in their OS.

4.7.55 show

Produces a sequence of displays to summarise the results of the estimation.

4.7.55.1 Argument

`request_type`
Where *`request_type`* takes one of the four values in the following list:

`parameters`
Requests displays of the parameter estimates in tabular and graphical form. These results can be written to a file or displayed in the output window or on the console. This is the default, if no argument is provided.

`cases`
Requests parameter estimates for the cases. These results must be written to a file using redirection.

`residuals`
Requests residuals for each case-generalised item combination. These results must be written to a file and are available for: MLEs, WLEs, EAPs, and PVs. The type of ability estimate is controlled by the option *estimate*.

For pairwise models, the `residuals` statement requests residuals for each fixed pair-outcome combination. The residuals can be interpreted as prediction errors (i.e., the difference between the observed and the predicted outcomes).

`expected`
Requests expected scores for each case/generalised item combination. These results must be written to a file and are only available for weighted likelihood ability estimates.

4.7.55.2 Options

`estimates =`*`type`*
`type` can be `eap`, `latent`, `mle`, `wle` or `none`.

When the argument is `parameters` or no argument is provided, this option specifies what to plot for the case distributions.

- If `estimates=eap`, the distribution will be constructed from expected a-posteriori values for each case.

- If `estimates=latent`, the distribution will be constructed from plausible values so as to represent the latent distribution.
- If `estimates=mle` or `wle`, the distribution will be constructed from maximum likelihood or weighted likelihood cases estimates. This provides a representation of the latent population distribution.
- If `estimates=none`, then the case distributions are omitted from the `show` output.
- If no `estimates` option is provided and the `estimate` statement includes `fit=yes` (explicitly or by default), the default is to use plausible values. If the `estimate` statement includes `fit=no`, the default is to omit the distributions from the `show` output.

When the argument is `cases`, this option gives the type of estimate that will be written to an output file. (See 'Redirection' below for the file formats.) `estimates=none` cannot be used, and there is no default value. Therefore, you must specify `eap`, `latent`, `wle` or `mle` when the argument is `cases`. In this context, `eap` and `latent` produce the same output.

`tables =`*`value list`*
If `parameters` output is requested, a total of eleven different tables can be produced. If a specific set of tables is required, then the `tables` option can be used to indicate which tables should be provided. `value list` consists of one or more of the integers 1 through 11, separated by colons (`:`) if more than one table is requested.

The contents of the tables are:

1. A summary showing the model estimated, the number of parameters, the name of the data file, the deviance and the reason that iterations terminated.
2. The estimates, errors and fit statistics for each of the parameters in the item response model.
3. Estimates for each of the parameters in the population model and reliability estimates.
4. A map of the latent distribution and the parameter estimates for each term in the item response model.
5. A vertical map of the latent distribution and threshold estimates for each generalised item. The response probability used in the thresholds defaults to 0.5, but can be set with the option *rp*.
6. A horizontal map of the latent distribution and threshold estimates for each generalised item.
7. A table of threshold estimates for each generalised item. The response probability used in the thresholds defaults to 0.5, but can be set with the option *rp*.

8. A table of item parameters estimates for each generalised item.
9. A map of the latent distribution and the parameter estimates for each term in the item response model with items broken out by dimension.
10. A table of the asymptotic error variance/covariance matrix for all parameters
11. A table of the score estimates for each category of each generalised item ("scores").
12. A table of the scoring parameter estimates ("taus").

The default tables are as follows, depending on the model (see `model` command) that is estimated – Rasch models: tables=1:2:3:4; 2PL models: tables=1:2:3:4:11; other models with scores: tables=1:2:3:11; pairwise models: tables=1:2. For multidimensional models (see `score` command), table 9 is also produced as default. For partial credit models, it is useful to include table 5 (which is not produced as default) in the requested tables.

`labelled =`*`reply`*
`reply` can be `yes` or `no`. `labelled=no` gives a simple form of the output that only includes a list of parameter numbers and their estimates. `labelled=yes` gives an output that includes parameter names and levels for each term in the `model` statement. `labelled=yes` is the default, except when a design matrix is imported, in which case `labelled=yes` is not available.

`expanded =`*`reply`*
`reply` can be `yes` or `no`. This option used in conjunction to table 5 to control the display of the item thresholds. `expanded=yes` separates the thresholds horizontally so that a new column is given for each item. `expanded=no` is the default.

`itemlabels =`*`reply`*
`reply` can be `yes` or `no`. This option is used in conjunction to table 5 to control the display of the item thresholds. `itemlabels=yes` uses item labels for each generalised item. `itemlabels=no` is the default.

`pfit =`*`reply`*
`reply` can be `yes` or `no`. This option is used in conjunction to the argument `cases` and the option `estimates=wle` and adds person fit statistics to the estimates file. `pfit=no` is the default.

`filetype =`*`type`*
`type` can take the value `spss`, `excel`, `csv`, `xls`, `xlsx` or `text`. This option sets the format of the results file. The default is `text`. The `spss` option is available if the argument is `cases`, `residuals` or `expected`.

`xscale =`*`n`*
Sets the number of cases to be represented by each 'X' in Wright maps. The default

value is a value that ensures that the largest bin uses all available bin space. The value is replaced by the default if the display would not otherwise fit in the available space.

`plotmax =`*n*
Sets the maximum logit value for the range of Wright maps.

`plotmin =`*n*
Sets the minimum logit value for the range of Wright maps.

`plotbins =`*n*
Sets the number of bins used for the range of Wright maps. The default value is `60`.

`itemwidth =`*n*
Sets the width in characters of the region available for item (facet) display in Wright maps. The default value is `40`.

`regressors =`*reply*
reply can be `yes` or `no`. This option used when the argument is`cases` and adds the case regression variables to the output file.

`rp =`*number*
rp can be a number greater than 0 and less than 1. *rp* is used in conjunction with the argument *gamma* (or *threshold*) to specify the response probability. The default is `0.5`.

4.7.55.3 Redirection

`>>`*filename*
Specifies a file into which the show results are written. If redirection is omitted and the argument is `parameters` or no argument is given, the results are written to the output window or the console. If the argument is `cases`, `residuals` or `expected`, then an output file must be given.

When the argument is `cases`, the format of the file of case estimates is as follows. In describing the format of the files we use *nd* to indicate the number of dimensions in the model.

For plausible values (`estimates=latent`) and expected a-posteriori estimates (`estimates=eap`):

The file will contain one row for each case. Each row will contain (in order):

- Sequence ID

- PID (if PID is not specified in `datafile` or `format` than this is equal to the Sequence ID)
- Plausible values. Note there will be *np* plausible values (default is 5) for each of *nd* dimensions. PVs cycle faster than dimensions, such that for $nd = 2$, and $np = 3$, the columns are in the order `PV1_D1, PV2_D1, PV3_D1, PV1_D2, PV2_D2, PV3_D2`. This is the same order as in the matrixout object for the command estimate.
- the posterior mean (EAP), posterior standard deviation, and the reliability for the case, for each dimension. Note that these columns cycle faster than dimensions such that for $nd = 2$, and $np = 3$, the columns are in the order `EAP_1, PosteriorSD_1, Reliability_1, EAP_2, PosteriorSD_2, Reliability_2`.

For maximum likelihood estimates and weighted likelihood estimates (`estimates=mle` or `estimates=wle`):

The file will contain one row for each case that provided a valid response to at least one of the items analysed (one item per dimension is required for multidimensional models). The row will contain the case number (the sequence number of the case in the data file being analysed), the raw score and maximum possible score on each dimension, followed by the maximum likelihood estimate and error variance for each dimension. The format is (i5, *nd*(2(f10.5, 1x)), *nd*(2(f10.5, 1x))). If the `pfit` option is set then an additional column is added containing the case fit statistics. The format is then (i5, *nd*(2(f10.5, 1x)), *nd*(2(f10.5, 1x)), f10.5)

4.7.55.4 Examples

```
show;
```

Produces displays with default settings and writes them to the output window.

```
show ! estimates=latent >> show.out;
```

Produces displays and writes them to the file `show.out`. Representations of the latent distributions are built from plausible values.

```
show parameters ! tables=1:4, estimates=eap;
```

Produces displays 1 and 4, represents the cases with expected a-posteriori estimates, and writes the results to the output window.

```
show cases ! estimates=mle >> example.mle;
```

Produces the file `example.mle` of case estimates, using maximum likelihood estimation.

```
show cases ! estimates=latent >> example.pls;
```

Produces the file `example.pls` of plausible values.

```
show cases ! estimates=wle, pfit=yes >> example.wle;
```

Produces the file `example.wle` of weighted likelihood estimates and person fit statisics.

```
show residuals ! estimates=wle, pfit=yes >> example.res;
```

Produces the file `example.res` of residuals for each case.

4.7.55.5 GUI Access

The various displays are accessed through the items in the Tables menu.

4.7.55.6 Notes

1. The tables of parameter estimates produced by the `show` command will display only the first 11 characters of the labels.
2. The method used to construct the ability distribution is determined by the `estimates` option used in the `show` statement. The `latent` distribution is constructed by drawing a set of plausible values for the cases and constructing a histogram from the plausible values. Other options for the distribution are `eap` and `mle`, which result in histograms of expected a-posteriori and maximum likelihood estimates, respectively.
3. It is possible to recover the ACER ConQuest estimate of the latent ability correlation from the output of a multidimensional analysis by using plausible values. Plausible values can be produced through the use of the `show` command argument `cases` in conjunction with the option `estimates=latent`.
4. The `show` statement cannot produce individual tables when an imported design matrix is used.

5. Neither `wle` nor `mle` case estimates can be produced for cases that had no valid responses for any items on one or more dimension. Plausible values are produced for all cases with complete background data.
6. Table 10, as described under the `tables` option above, is only available if empirical standard errors have been estimated. Table 10 is not applicable for pairwise models.
7. Plausible values and EAP estimates contain stochastic elements and may differ marginally from run to run with identical data.
8. Showing cases is not applicable for pairwise models.

4.7.56 structural

Fits a structural path model using two-stage least squares.

4.7.56.1 Argument

The structural statement argument is a list of regression models that are separated by the character `/` (slash). Each regression model takes the form

dependent`on`*independent_1, independent_2, …, independent_n*.

4.7.56.2 Options

`export =`*reply*
reply can be `yes` or `no`. This option controls the format of the output. The export format does not use labelling and is supplied so that results can be read into other software easily. `export=no` is the default.

`filetype =`*type*
type can take the value `xls`, `xlsx`, `excel` or `text` and it sets the format of the results file. The default is `text` when used in conjunction with a file redirection. If no file redirection is given the results are written to the output window.

`matrixout =`*name*
name is a matrix (or set of matrices) that will be created and will hold the results. These results are stored in the temporary workspace. Any existing matrices with matching names will be overwritten without warning. The contents of the matrices is described in section 4.9 Matrix Objects Created by Analysis Commands.

4.7.56.3 Redirection

`>>`*`filename`*
Specifies a file into which the `show` results are written.

4.7.56.4 Example

```
structural /dimension_1 on dimension_2 dimension_3 grade
/dimension_2 on dimension_3 grade sex
/dimension_3 on grade sex ! export=yes;
```

Fits the path model shown in Figure 4.1

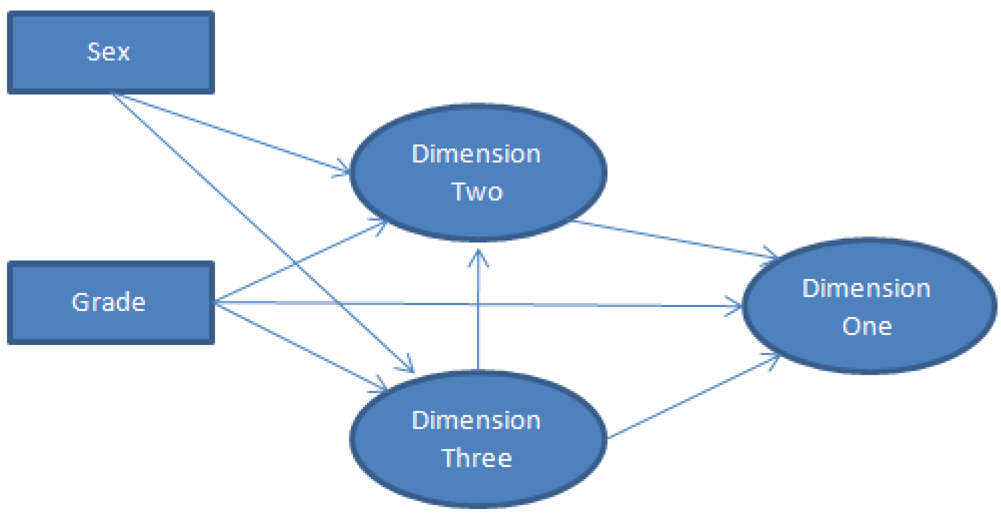


Figure 4.1: Structural Path Model Diagram

4.7.56.5 GUI Access

`Analysis`→`Structural`.
Select regression variables from the currently defined list of regression variables and latent variables.

4.7.56.6 Notes

No notes.

4.7.57 submit

Executes the ACER ConQuest command statements in the file named in its argument.

4.7.57.1 Argument

filename
The name of the text file containing the statements.

4.7.57.2 Options

This command does not have options.

4.7.57.3 Redirection

Redirection is not applicable to this command.

4.7.57.4 Example

```
submit example1.cqc
```

Executes the statements in the file `example1.cqc`.

4.7.57.5 GUI Access

`File`→`Submit Commands`.

4.7.57.6 Notes

1. `submit` commands can be nested. That means a file of submitted commands can contain a `submit` command.

4.7.58 system

Allows a DOS command to be executed.

4.7.58.1 Argument

DOS Command
The command to be executed.

4.7.58.2 Options

This command does not have options.

4.7.58.3 Redirection

Redirection is not applicable to this command.

4.7.58.4 Example

```
system dir;
```

Shows the contents of the current working directory.

4.7.58.5 GUI Access

Access to this command through the GUI is not available.

4.7.58.6 Notes

1. In the GUI version the results of some operating system commands will be written to a command window that will not stay open after the command has executed.

4.7.59 title

Specifies the title that is to appear at the top of any printed ACER ConQuest output.

4.7.59.1 Argument

This command does not have an argument.

4.7.59.2 Options

This command does not have options.

4.7.59.3 Redirection

Redirection is not applicable to this command.

4.7.59.4 Example

```
title This is a great analysis!;
```

The words `This is a great analysis!` will appear on the top of each ACER ConQuest printout from this analysis.

4.7.59.5 GUI Access

`Command`→`Title`.

4.7.59.6 Notes

1. If a title is not provided, the default, `ConQuest: Generalised Item Response Modelling Software`, will be used.

4.7.60 while

Allows conditional execution of commands

Table 4.3: Comparison operators

Operator	Meaning
==	equality
=>	greater than or equal to
>=	greater than or equal to
=<	less than or equal to
<=	less than or equal to
!=	not equal to
>	greater than
<	less than

4.7.60.1 Argument

```
(logical condition) {
set of ACER ConQuest commands
};
```

While *`logical condition`* evaluates to `true` the *`set of ACER ConQuest commands`* is executed. The commands are not executed if the logical condition does not evaluate to `true`.

The logical condition can be `true`, `false` or of the form *`s1 operator s2`*, where *`s1`* and *`s2`* are strings and *`operator`* is one of the following:

For each of *`s1`* and *`s2`* ACER ConQuest first attempts to convert it to a numeric value. The numeric value can be a scalar value, a reference to an existing 1x1 matrix variable or a 1x1 submatrix of an existing matrix variable. A numeric value cannot involve computation.

If s1 is a numeric value the operator is applied numerically. If not a string comparison occurs between *`s1`* and *`s2`*.

4.7.60.2 Options

This command does not have options.

4.7.60.3 Redirection

Redirection is not applicable to this command.

4.7.60.4 Example

```
x=fillmatrix(20,20,0);
compute k=1;
compute i=1;
while (i<=20)
{
for (j in 1:i)
 {
  if (j<i)
   {
    compute x[i,j]=k;
    compute x[j,i]=-k;
    compute k=k+1;
   };

  if (j==i)
   {
    compute x[i,j]=j;
   };
 };
compute i=i+1;
};
print x;
```

Creates a 20 by 20 matrix of zero values and then fills the lower triangle of the matrix with the numbers 1 to 190, the upper triangle with -1 to -190 and the diagonal with the numbers 1 to 20. The matrix is then printed to the screen.

4.7.60.5 GUI Access

Access to this command through the GUI is not available.

4.7.60.6 Notes

1. There are no limits on the nesting of conditions.

4.7.61 write

Write a matrix object to file.

4.7.61.1 Argument

Name of a matrix object to be writen to file.

4.7.61.2 Options

`filetype =`*`type`* *`type`* can take the value `spss`, `csv` or `text`. The default is `text`.

4.7.61.3 Redirection

`>>`*`filename`* the name of the file to write

4.7.61.4 Example

```
write my_resp ! filetype = csv >> output/data/myResponses.csv;
```

Writes the matrix object *my_resp* to a csv file `myResponses.csv` in the path `output/data/` below the current working directory.

4.7.61.5 GUI Access

Access to this command through the GUI is not available.

4.7.61.6 Notes

1. Column labels will be used to create a header in csv files and column names in spss files.

Table 4.4: Comparison operators

Operator	Meaning
==	equality
=>	greater than or equal to
>=	greater than or equal to
=<	less than or equal to
<=	less than or equal to
!=	not equal to
>	greater than
<	less than

4.8 Compute Command Operators and Functions

4.8.1 Operators

The standard binary mathematical operators are: addition (`+`), subtraction (`-`), multiplication (`*`), and division (`/`). All are available and operate according to their standard matrix definition when applied to conformable matrices. Division by a matrix is treated as multiplication by the matrix inverse. If the operators are applied to non-conformable matrices then the operators return a null matrix excepting when one of the arguments is a double (or 1 by 1 matrix), then the operator is applied element-wise.

The unary negation operator (`-`) is available and is applied element wise to a matrix.

The exponentiation operator (`^`) is available but cannot be applied to matrices.

Two special binary mathematical operators are provided for element-wise matrix multiplication (`**`) and division (`//`). The `**` operator multiplies each of the matching elements of two identically dimensioned matrices. The `//` operator divides each element of the first matrix by the matching element of the second matrix.

The following 6 logical operators are available:

These operators are applied element-wise to a pair of matrices and return matrices of '1' and '0' with 1 if an element-wise comparison is true and 0 if it is false.

4.8.2 Standard Computation Functions for Matrices

The following standard functions are available. Each of the functions takes a single matrix argument and is applied element-wise to the matrix. Each function, therefore, returns a matrix of the same dimensions as the input.

`abs(`*`matrix`*`)` Each element of the input `matrix` is a double. Returns a matrix of the same dimensions as the input, `matrix` where each element is the absolute value of the corresponding elements in the input matrix.

`ceil(`*`matrix`*`)` Each element of the input `matrix` is a double. Returns a matrix of the same dimensions as the input, `matrix` where each element is the ceiling of the corresponding elements in the input matrix.

`exp(`*`matrix`*`)` Each element of the input `matrix` is a double between 0 and 1. Returns a matrix of the same dimensions as the input, `matrix` where each element is exponent base e of the the corresponding element in the input matrix.

`floor(`*`matrix`*`)` Each element of the input, `matrix` is a double. Returns a matrix of the same dimensions as the input, `matrix` where each element is the floor of the corresponding elements in the input matrix.

`int(`*`matrix`*`)` Each element of the input `matrix` is a double. Returns a matrix of the same dimensions as the input, `matrix` where each element is the integer component of the corresponding elements in the input matrix.

`invgcdf(`*`p`*`)` Each element of the input `matrix` is a probability (a double between 0 and 1). Returns a matrix of the same dimensions as the input, `matrix` where each element is the inverse Gaussian distribution function for `p`.

`log(`*`matrix`*`)` Each element of the input `matrix` is a double greater than 0. Returns a matrix of the same dimensions as the input, `matrix` where each element is the natural log of the corresponding element in the input matrix.

`log10(`*`matrix`*`)` Returns a matrix of the same dimensions as the input, `matrix` where each element is the log base 10 of the corresponding element in the input matrix. Each element of `matrix` must be greater than 0.

`logit(`*`matrix`*`)` Returns a matrix of the same dimensions as the input, `matrix` where each element is the logit of the corresponding element in the input matrix. Each element of `matrix` must be between 0 and 1.

`rnd(`*`matrix`*`)` Returns a matrix of the same dimensions as the input, `matrix` where each element is the integer-rounded value of the corresponding elements in the input matrix.

`sqrt(`*`matrix`*`)` Returns a matrix of the same dimensions as the input, `matrix` where each element is the square root of the corresponding element in the input matrix. Each element of `matrix` must be greater than or equal to 0.

4.8.3 Accessing Matrix Information

Sub matrices can be extracted from matrices by appending `[`*`rowstart`*`:`*`rowend`*`,`*`colstart`*`:`*`colend`*`]` to the name of a matrix variable, for example `m[2:5,5:10]`. If all rows are required, *rowstart* and *rowend* can be omitted. If all columns are required, *colstart* and *colend* can be omitted. If a single row is required, *rowend* and the *colon* "`:`" can be omitted. If a single column is required, *colend* and the *colon* "`:`" can be omitted.

Column and row indexing commence at one. So that, for example, `m[10,3]` refers to the element in the 10-th row and 3-th column.

Single elements of a matrix can be specified to the left of the equal operator '`=`' by appending `[`*`row`*`,`*`col`*`]` to the name of a matrix variable. Sub matrices cannot be specified to the left of the equal operator '`=`'.

4.8.4 Matrix Manipulation Functions

Two binary operators are available for concatenating matrices. Column concatenation of two matrices, *`m1`* and *`m2`* is performed using *`m1|^m2`*. In this case, *`m1`* and *`m2`* must have column conformability and the matrix *`m1`* is added under matrix *`m2`*. Row concatenation of two matrices, *`m1`* and *`m2`* is performed using *`m1->m2`*. In this case *`m1`* and *`m2`* must have row conformability and the matrix *`m1`* is added to the left of matrix *`m2`*.

Function arguments can themselves be functions or computed values but those functions or computed value must be enclosed in parentheses.

The following functions are available for manipulating the content of matrices:

`counter(`*`int`*`)`
Returns a matrix with dimensions *`int`* x 1 filled with integers running from 1 to *`int`*.

`diag(`*`matrix`*`)`
Returns the diagonal elements of the input, `matrix`: a square matrix.

`fillmatrix(`*`nrow`*`,`*`ncol`*`,`*`val`*`)`
Returns a matrix with dimensions *`nrow`* x *`ncol`* filled with the value *`val`*.

`identity(`*arg*`)`
Returns a matrix of dimension *arg*.

`iif(`*arg1*`,`*arg2*`,`*arg3*`)`
All three arguments must be matrices of the same dimensions.
The result is a matrix where an element takes its value from *arg2* if the matching *arg1* element is '1', otherwise it takes its value from *arg3*.

`selectifcolumn(`*matrix*`,`*colref*`,`*val*`)` *matrix* is a matrix, *colref* is a column reference and, *val* is a value. The result is a matrix that contains only those rows of *matrix* where column *colref* takes value *val*.

`transpose(`*matrix*`)`
Transpose matrix of *matrix*.

`vec(`*arg*`)`
Returns a vector, which is the vec of *arg*.

`vech(`*arg*`)`
Returns a vector, which is the vech of *arg*.

`inv(`*arg*`)`
Inverse of matrix *arg*.

`det(`*arg*`)`
Determinant of matrix *arg*.

`trace(`*arg*`)`
Trace of matrix *arg*.

`rows(`*arg*`)`
Number of rows of matrix *arg*.

`cols(`*arg*`)`
Number of columns of matrix *arg*.

`min(`*arg*`)`
Number minimum of all elements in matrix *arg*.

`max(`*arg*`)`
Number maximum of all elements in matrix *arg*.

`sum(`*arg*`)`
Sum of all elements in matrix *arg*.

`sum2(`*arg*`)`
Sum of squares of all elements in matrix *arg*.

`colcp(`*`arg`*`)`
Column cross-products, returns a row × row matrix equal to *arg**transpose(*arg*).

`rowcp(`*`arg`*`)`
Row cross-products, returns a column × column matrix equal to transpose(*arg*)**arg*.

`rowcov(`*`arg`*`)`
Row covariance, returns a column × column matrix which is the covariance matrix of the columns.

`rowcor(`*`arg`*`)`
Row correlations, returns a column × column matrix which is the correlation matrix of the columns.

`colsum(`*`arg`*`)`
Returns a row which contains the sum over each of the columns of the *arg*.

`rowsum(`*`arg`*`)`
Returns a vector which contains the sum over each of the rows of the *arg*.

`sort(`*`arg`*`)`
Returns a vector which is contains the rows of *arg* sorted in ascending order. The argument must be a vector.

`long(`*`matrix`*`)`
Takes in a matrix and returns a new matrix with dimensions:

- number of rows equal to (rows × cols of *matrix*)
- 3 columns

Each row in the returned matrix represents an element of the original matrix. Column 1 of the returned matrix is the row number of the element of the original matrix. Column 2 of the returned matrix is the column number of the element of the original matrix. Column 3 of the returned matrix is the element (value) of the original matrix.

`mod(`*`m1,m2`*`)`
Takes 2 matrices as input and returns the element wise modulus.

`percentiles(`*`data,centiles`*`)`
Takes a matrix of data and returns a matrix of size rows of `centiles` x columns of `data`. The columns of the returned matrix are the percentiles of the columns of `data`.

4.8.5 Random Number Generators

The following random number generators are used. To control the seed use `set seed =`*n*.

`rbernoulli(`*probs*`)`
Matrix of Bernoulli variables where *probs* is a matrix of p values.

`rchisq(`*df*`)`
Chi square deviate with *df* degrees of freedom.

`rint(`*min,max*`)`
The integer component of a random uniform deviate from the range *min* and *max*. Both *min* and *max* must be integers.

`rinvshisq(`*df*`)`
Inverse chi-square deviate with *df* degrees of freedom.

`rlefttnormal(`*q*`)`
Deviate from a standard normal left truncated at *q*.

`rmvnormal(`*mu,var*`)`
A random multivariate normal deviate with mean vector *mu* and covariance matrix *covar*.

`rmvnmatrix(`*mu,sigma,nrow*`)`
Returns a matrix of dimensions *nrow* by length of *mu*. Rows are independent multivariate normal deviates with mean vector *mu* and covariance matrix *sigma*.

`rnormal(`*mu,sd*`)`
A random normal deviate with mean *mu* and standard deviation *sd*.

`rnormalmatrix(`*mu,sd,nrow,ncol*`)`
An *nrow* x *ncol* matrix of random deviates, with mean *mu* and standard deviation *sd*.

`rpg(`*b,c*`)`
A random deviate from a Pólya-gamma distribution with parameters "b" (int $>$ 0) and "c" (double).

`rpgmatrix(`*b,c,nrow,ncol*`)`
An *nrow* x *ncol* matrix of random deviates random deviates from a Pólya-gamma distribution with parameters "b" (int $>$ 0) and "c" (double).

`rrighttnormal(`*q*`)`
Deviate from a standard normal right truncated at *q*.

`runiform(`*min,max*`)`
A random uniform deviate from the range *min* and *max*.

`runiformmatrix(`*`min,max,nrow,ncol`*`)`
An *`nrow`* x *`ncol`* matrix of random unifomr deviates from the range *`min`* and *`max`*.

4.9 Matrix Objects Created by Analysis Commands

A number of analysis returns can save their results in a family of matrix objects that are added to the ACER ConQuest variable list (see the command *print*). These variables then become available for manipulation or other use. Note that the matrix objects created cannot be directly modified by the user. The matrix objects can be, however, copied and then manipulated.

The commands that can produce matrix variables are: `descriptives`, `estimate`, `fit` `generate`, and `matrixsampler`. For each of these commands the option `matrixout=stem` is used to request the variables and to set a prefix for their name. The variables produced by each command and their format is provided below.

All matrix objects created have a user-specified prefix, followed by an underscore ("_"), followed by a suffix as defined for each command below.

4.9.1 Descriptives Command

The following four matrices are produced regardless of the estimator option:

- descriptives
  Number of dimensions by eight, providing for each dimension the dimension number, number of cases, mean, standard deviation, variance, standard error of the mean, standard error of the standard deviation, and standard error of the variance.
- percentiles
  Number of dimensions by the number of requested percentiles plus two, providing for each dimension the dimension number, number of cases, and then each of the percentiles.
- bands
  Number of dimensions by twice the number of requested bands plus two, providing for each dimension the dimension number, number of cases, and then proportion in each of the bands follow by standard errors for each of the band proportions.

- bench
  Number of dimensions by four, providing for each dimension the dimension number, number of cases, proportion below the benchmark and standard error of that proportion.

If `latent` is chosen as the estimator then in addition to the above the following matrices are available:

- pv_descriptives
  Number of dimensions times number of plausible values by six, providing for each dimension and plausible value, the dimension number, the plausible value number, number of cases, mean, standard deviation, and variance.
- pv_percentiles
  Number of dimensions times number of plausible values by the number of percentiles plus three, providing for each dimension and plausible value, the dimension number, the plausible value number, number of cases, and then each of the percentiles.
- pv_bands
  Number of dimensions times number of plausible values by the number of requested bands plus three, providing for each dimension and plausible value, the dimension number, the plausible value number, number of cases, and then proportion in each of the bands.

4.9.2 Estimate Command

Regardless of the options to the command estimate used, the following two matrices are produced:

- xsi
  A single column of the estimated item location parameters.
- history
  Number of iterations by total number of estimated parameters plus three. The first column is the run number. The second column is the iteration number within the run. The third column is the deviance. The remaining columns are for the parameter estimates.

If the model includes estimated scoring parameters, then the following matrix is also produced:

- tau
  A single column of the estimated item scoring parameters.

Depending on the options specified the following matrices are also available: If the `method=jml` option is chosen or `abilities=yes` in conjunction with an MML method then the following two matrices of case estimates are produced.

- mle
  Number of cases by number of dimensions providing for each case the MLE latent estimate for each case.
- wle
  Number of cases by number of dimensions providing for each case the WLE latent estimate for each case.

If `abilities=yes` is used in conjunction with an MML method, or MCMC estimation is used, then a matrix of case plausible values and a matrix of case EAPs is produced.

- pvs
  Number of cases by number of dimensions times number of plausible values. For the columns the plausible values cycle fastest. For example, if there are three dimensions and two plausible values, column one would contain plausible value one for dimension one, column two would contain plausible value two for dimension one, column three would contain plausible value one for dimension two and so on.

- eap
  Number of cases by number of dimensions.

If `ifit=yes` is used (the default) a matrix of item fit values is produced.

- itemfit
  Number of fit test by four. The four columns are the unweighted T, weighted T, unweighted MNSQ, and weighted MNSQ.

If `pfit=yes` is used a matrix of case fit values is produced.

- casefit
  Number of cases by one. Providing for each case the unweighted mean square.

If `stderr=empirical` is used (the default for MML) then the estimate error covariance matrix is produced.

- estimatecovariances
  A number of parameter by number of parameter matrix of estimate error covariances.

If `stderr=quick` is used (the default for JML) then the following estimate error variances matrix objects are produced.

- xsierrors
  Number of item location parameter estimates by one, providing for each item location parameter the associated estimate variance.
- regressionerrors
  Number of regression parameters by one, providing for each regression parameter the estimate variance.
- covarianceerrors
  Number of covariance parameters by one, providing for each regression parameter the estimate variance.

And, if item scoring parameters are estimated,

- tauerrors
  Number of item scoring parameters by one, providing for each item scoring parameter the associated estimate variance.

4.9.3 Fit Command

Produces a set of matrices, one for each level of the group used in the `group=option`.

- userfit
  Each matrix with the suffix userfit will be preceded by the group name as well as the user defined prefix. Each matrix (one per group) has dimension number of fit tests by four, providing for each test the un-weighted t-fit, weighted t-fit, un-weighted mean square and weighted mean square.

4.9.4 Generate Command

The matrices that are produced by generate depend upon the options chosen. Regardless of the options chosen the following matrix is produced:

- items
  Number of items by three, providing for each item the item number, category number, and the generated parameter value.

If the option `scoresdist` is used then a matrix of scoring parameters is produced.

- scores
  Number of total item scoring categories by number of dimensions plus two, providing for each item category, the item number, category number and score for each dimension.

If the option `importnpvs` is NOT used then the following two matrices are produced:

- responses
  Number of cases by number of items, providing for each case a response to each item.
- cases
  Number of cases by number of dimensions plus one, providing for each case a case number and a generated ability for each of the dimensions.

If the option `importnpvs` is used then the following matrices of summary statistics are produced for each dimension and group:

- statistics
  Number of plausible values by three times the number of items plus three. It contains mean raw scores, raw score variances, Cronbach's alpha and then for each item, mean item score and point biserial statistics (biased and unbiased).

4.9.5 Itanal Command

Produces a set of matrices, one for each level of the group used in the `group=option`. The name of the matrix is provided by the `matrixout=option`. The matrices produced are as follows.

- counts Number of items by number of response categories, providing for each item, the frequency of responses in each category.
- itemstats Number of items by six, providing for each item: item-total correlations, item-rest correlations, observed mean score, expected mean score, adjusted mean score, and item delta dot. For details see command itanal.
- ptbis Number of items by three times the number of response categories, providing for each item and category the:

  (1) point-biserial correlation with the total score,
  (2) the t-test of the point-biserial, and
  (3) the associated p value.

- abilitymeansd Number of items by number of response categories by dimension by two, providing for each item, category, and dimension the mean and standard deviation of the ability estimate (when using PVs the first plausible value is used) of the cases who responded in that category.
- summarystats Descriptive statistics for the raw scores. Matrix is one by ten. Percent Missing, N, Mean, SD, Variance, Skew, Kurtosis, Standard error of mean, Standard error of measurement, Alpha.

4.9.6 Matrixsampler Command

Produces matrices with the name provided in the `matrixout=option`. This produces all of the matrices produced under the itanal command, plus one that contains descriptive statistics for simulated data, one that contains fit statistics for the user's data, and one that contains fit statistics for simulated data. The two matricies (fit and userfit) containing fit statistics are only provided if `fit=yes` is specified in the command.

- raw
  contains a row for each sampled matrix and columns providing the inter-item and item-total correlations.

- inputfit
  contains a row for each parameter and sampled matrix combination and columns Unweighted_t, Weighted_t, Unweighted_MNSQ, Weighted_MNSQ, Parameter and replication Set.
- samplerfit
  contains a row for each parameter and columns: Unweighted_t, Weighted_t, Unweighted_MNSQ, Weighted_MNSQ, Parameter number. These are the estimates from the analysis of the user's dataset.

4.9.7 Structural Command

Produces a set of matrices, one for each regression model and four matrices of sums of squares and cross-products. The name of the matrix is provided by the `matrixout=option`. The matrices produced are as follows.

- fullsscp
  Square matrix with dimension equal to the total number of variables in the structural model providing the sums of squares and cross-products.
- osscp
  Square matrix with dimension equal to the number of observed variables (non latent variables) in the structural model providing the sums of squares and cross-products.
- losscp
  Number of latent by number of observed variables providing the cross-products.
- lsscp
  Square matrix with dimension equal to the total number of latent variables in the structural model providing the sums of squares and cross-products.
- results_eqn$n$
  Where the matrix object contains the results of the estimation of each of the n regression equations in the structural model. The cell 1,1 contains the R-squared, and there are additional rows for each independent variable, column one of each of those additional rows is the estimated regression parameter and the second column is its standard error estimate.

4.10 List of Illegal Characters and Words for Variable Names

Table 4.5: Illegal characters to use in tokens names.

<table>
<tr><th>Character</th></tr>
<tr><td>|</td></tr>
<tr><td>/</td></tr>
<tr><td>$</td></tr>
<tr><td>~</td></tr>
</table>

Table 4.6: Words that cannot be used as names of matrices, implicit variables, or explicit variables

Term	Type
all	Word
category	Word
dimensions	Word
fitstatistics	Word
on	Word
parameters	Word
step	Word
steps	Word
to	Word
tokens	Word
variables	Word
abs	Function
ceil	Function
colcp	Function
cols	Function
colsum	Function
counter	Function
det	Function
exp	Function
fillmatrix	Function
floor	Function
identity	Function

iif	Function
int	Function
inv	Function
log	Function
log10	Function
logit	Function
max	Function
min	Function
percentiles	Function
rbernoulli	Function
rchisq	Function
rinvchisq	Function
rleftnormal	Function
rmvnormal	Function
rnd	Function
rnormal	Function
rnormalmatrix	Function
rowcor	Function
rowcov	Function
rowcp	Function
rows	Function
rowsum	Function
rpg	Function
rpgmatrix	Function
rrightnormal	Function
runiform	Function
runiform	Function
selectifcolumn	Function
sort	Function
sqrt	Function
sum	Function
sum2	Function
trace	Function
transpose	Function
banddefine	Command name

build	Command name
caseweight	Command name
categorise	Command name
chistory	Command name
clear	Command name
codes	Command name
compute	Command name
datafile	Command name
delete	Command name
descriptives	Command name
directory	Command name
display	Command name
dofor	Command name
doif	Command name
dropcases	Command name
else	Command name
enddo	Command name
endif	Command name
equivalence	Command name
estimates	Command name
execute	Command name
exit	Command name
export	Command name
facets	Command name
filter	Command name
fit	Command name
for	Command name
format	Command name
function	Command name
generate	Command name
get	Command name
gingroup	Command name
group	Command name
if	Command name
import	Command name

itanal	Command name
keepcases	Command name
key	Command name
kidmap	Command name
labels	Command name
let	Command name
matrixsampler	Command name
mh	Command name
missing	Command name
model	Command name
plot	Command name
print	Command name
put	Command name
quit	Command name
read	Command name
recodes	Command name
regression	Command name
reset	Command name
scatter	Command name
scores	Command name
set	Command name
show	Command name
structural	Command name
submit	Command name
system	Command name
systemclean	Command name
timerstart	Command name
timerstop	Command name
title	Command name
while	Command name
write	Command name

The suffixes added to matrix objects created using the option “matrixout” are also protected words. These suffices cannot be within declarations (e.g., a sub string of the declaration).

Table 4.7: Words that cannot be used within names (including as sub strings) of matrices, implicit variables, or explicit variables

Term	Type
_bands	Extension
_bench	Extension
_casefit	Extension
_cases	Extension
_counts	Extension
_covarianceerrors	Extension
_descriptives	Extension
_estimatecovariances	Extension
_fullsscp	Extension
_history	Extension
_inputfit	Extension
_itemerrors	Extension
_itemfit	Extension
_itemparams	Extension
_items	Extension
_itemtotrestcor	Extension
_losscp	Extension
_lsscp	Extension
_mle	Extension
_osscp	Extension
_percentiles	Extension
_ptbis	Extension
_pv_bands	Extension
_pv_descriptives	Extension
_pv_percentiles	Extension
_pvmeansd	Extension
_pvs	Extension
_raw	Extension
_regressionerrors	Extension
_responses	Extension

_results_eqn	Extension
_samplerfit	Extension
_scores	Extension
_statistics	Extension
_userfit	Extension
_wle	Extension

Chapter 5

References

Adams, R. J. (2005). Reliability as a Measurement Design Effect. *Studies in Educational Evaluation*, *31*(2-3), 162–172. https://doi.org/10.1016/j.stueduc.2005.05.008

Adams, R. J., Doig, B. A., & Rosier, M. (1991). *Science Learning in Victorian Schools*. Australian Council for Educational Research.

Adams, R. J., & Gonzales, E. J. (1996). *Third International Mathematics and Science Study. Technical Report Volume 1: Design and Development* (M. O. Martin & D. L. Kelly, Eds.; Vol. 1). Center for the Study of Testing, Evaluation and Educational Policy. Boston, Massachusetts: Boston College.

Adams, R. J., & Wilson, M. R. (1996). A Random Coefficients Multinomial Logit: A Generalized Approach to Fitting Rasch Models. In G. Engelhard & M. R. Wilson (Eds.), *Objective Measurement III: Theory into Practice* (pp. 143–166). Ablex.

Adams, R. J., Wilson, M. R., & Wang, W. (1997). The Multidimensional Random Coefficients Multinomial Logit Model. *Applied Psychological Measurement*, *21*, 1–24. https://doi.org/10.1177%2F0146621697211001

Adams, R. J., Wilson, M. R., & Wu, M. L. (1997). Multilevel Item Response Models: An Approach to Errors in Variables Regression. *Journal of Educational and Behavioural Statistics*, *22*, 46–75.

Adams, R. J., & Wu, M. L. (2007). *The mixed-coefficients multinomial logit model: A generalized form of the rasch model* (M. von Davier & C. H. Carstensen, Eds.; pp. 57–75). Springer New York. https://doi.org/10.1007/978-0-387-49839-3_4

Andersen, E. B. (1985). Estimating Latent Correlations Between-Repeated Testings. *Psychometrika*, *50*, 3–16. https://doi.org/10.1007/BF02294143

Andrich, D. (1978). A Rating Formulation for Ordered Response Categories. *Psychometrika*, *43*, 561–573. https://doi.org/10.1007/BF02293814

Beaton, A. E. (1987). *Implementing the New Design: The NAEP 1983–84 Technical Report* (15-TR-20). Educational Testing Service.

Beaton, A. E., Mullis, I. V. S., Martin, M. O., Gonzales, E. J., Kelly, D. L., & Smith, T. A. (1996). Mathematics Achievement in the Middle School Years. In *IEA's Third International Mathematics and Science Study*. Boston College.

Birnbaum, A. (1968). Some Latent Trait Models and Their Use in Inferring an Examinee's Ability. *Statistical Theories of Mental Test Scores.*

Bock, D. R. (1972). Estimating Item Parameters and Latent Ability When Responses Are Scored in Two or More Nominal Categories. *Psychometrika*, *37*, 29–51. https://link.springer.com/article/10.1007/BF02291411

Bock, D. R., & Aitkin, M. (1981). Marginal Maximum Likelihood Estimation of Item Parameters: An Application of the EM Algorithm. *Psychometrika*, *46*, 443–459.

Bradley, R. A., & Terry, M. E. (1952). Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika*, *39*(3/4), 324–345. https://doi.org/10.2307/2334029

Cloney, D., & Adams, R. J. (2022). *Conquestr*. https://cran.r-project.org/package=conquestr

Congdon, P., & McQueen, J. (1997, March 24–28). The Stability of Rater Severity Estimates in Large Scale Performance Assessment Programmes. *Annual Meeting of the American Educational Research Association.*

Crocker, L. M., & Algina, J. (1986). *Introduction to Classical and Modern Test Theory.* Holt Rinehart and Winston.

De Boeck, P., & Partchev, I. (2012). Irtrees: Tree-based item response models of the GLMM family. *Journal of Statistical Software*, *48*.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society*, *39*, 1–38.

DiTrapani, J., Jeon, M., De Boeck, P., & Partchev, I. (2016). Attempting to differentiate fast and slow intelligence: Using generalized item response trees to examine the role of speed on intelligence tests. *Intelligence*, *56*, 82–92. https://doi.org/https://doi.org/10.1016/j.intell.2016.02.012

Embretson, S. E. (1991). A Multidimensional Latent Trait Model for Measuring Learning and Change. *Psychometrika*, *56*, 495–515.

Engle, R. F. (1984). Wald, Likelihood Ratio, and Lagrange Multiplier Tests in Econometrics. In undefined, *Handbook of Econometrics II* (pp. 775–826). Elsevier. http://www.stern.nyu.edu/rengle/LagrangeMultipliersHandbook_of_Econ___II____Engle.pdf

Fischer, G. H. (1973). The Linear Logistic Model as an Instrument in Educational Research. *Acta Psychologica*, *37*, 359–374.

Fischer, G. H. (1983). Logistic Latent Trait Models with Linear Constraints. *Psychometrika*, *48*, 3–26.

Glas, C. A. W. (1989). *Contributions to Estimating and Testing Rasch Models* [Doctoral dissertation]. University of Twente.

Glickman, M. E. (1999). Parameter Estimation in Large Dynamic Paired Comparison Experiments. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *48*(3), 377–394. https://doi.org/10.1111/1467-9876.00159

Jeon, M., & De Boeck, P. (2015). A generalized item response tree model for psychological assessments. *Behavior Research Methods*, *48*. https://doi.org/10.3758/s13428-015-0631-y

Kelderman, H., & Rijkes, C. P. M. (1994). Loglinear Multidimensional IRT Models for Polytomously Scored Items. *Psychometrika*, *59*, 149–176.

Linacre, J. M. (1994). *Many-Facet Rasch Measurement.* MESA Press.

Lokan, J., Ford, P., & Greenwood, L. (1996). *Maths and Science On the Line: Australian Junior Secondary Students' Performance in the Third International Mathematics and Science Study.* Australian Council for Educational Research.

Lord, F. M. (1983). Unbiased Estimators of Ability Parameters, of Their Variance, and Parallelforms Reliability. *Psychometrika*, *48*, 233–245.

Lord, F. M. (1984). *Maximum Likelihood and Bayesian Parameter Estimation in Item Response Theory* (Research Report RR-84-30-ONR). Educational Testing Service.

Luce, R. D. (2005). *Individual Choice Behavior: A Theoretical Analysis.* Dover Publications.

Masters, G. N. (1982). A Rasch Model for Partial Credit Scoring. *Psychometrika*, *47*, 149–174.

Mislevy, R. J. (1984). Estimating Latent Distributions. *Psychometrika*, *49*, 359–381.

Mislevy, R. J. (1985). Estimation of Latent Group Effects. *Journal of the American Statistical Association*, *80*, 993–997.

Mislevy, R. J. (1991). Randomization-Based Inference about Latent Variables from Complex Samples. *Psychometrika*, *56*, 177–196.

Mislevy, R. J., Beaton, A. E., Kaplan, B., & Sheehan, K. M. (1992). Estimating Population Characteristics from Sparse Matrix Samples of Item Responses. *Journal of Educational Measurement*, *29*, 133–161.

Mislevy, R. J., & Sheehan, K. M. (1989). The Role of Collateral Information about Examinees in Item Parameter Estimation. *Psychometrika*, *54*, 661–679.

Muraki, E. (1992). A Generalized Partial Credit Model. *Applied Psychological Measurement*, *16*, 159–176. https://conservancy.umn.edu/handle/11299/115645

Rasch, G. (1960). *Probabilistic models for some intelligence and attainment tests.* Danish Institute for Educational Research.

Rasch, G. (1980). *Probabilistic Models for Some Intelligence and Attainment Test*. University of Chicago Press.

Roberts, L., Wilson, M. R., & Draney, K. (1997). *The SEPUP Assessment System: An Overview*. University of California.

Volodin, N., & Adams, R. J. (1995). Identifying and Estimating a D-Dimensional Item Response Model. *International Objective Measurement Workshop, University of California*.

Wang, W. (1995). *Implementation and Application of the Multidimensional Random Coefficients Multinomial Logit* [Unpublished doctoral dissertation]. University of California.

Whitely, S. E. (1980). Multicomponent Latent Trait Models for Ability Tests. *Psychometrika*, *45*, 479–494.

Wilson, M. R. (1992). The Ordered Partition Model: An Extension of the Partial Credit Model. *Applied Psychological Measurement*, *16*, 309–325.

Wilson, M. R., & Adams, R. J. (1995). Rasch Models for Item Bundles. *Psychometrika*, *60*, 181–198.

Wilson, M. R., & Masters, G. N. (1993). The Partial Credit Model and Null Categories. *Psychometrika*, *58*, 87–99.

Wright, B. D., & Masters, G. N. (1982). *Rating Scale Analysis: Rasch Measurement*. MESA Press.

Wright, B. D., & Panchapakesan, N. (1969). A Procedure for Sample-Free Item Analysis. *Educational and Psychological Measurement*, *29*, 23–48.

Wright, B. D., & Stone, M. H. (1979). *Best Test Design: Rasch Measurement*. MESA Press.

Wu, M. L. (1997). *The Development and Application of a Fit Test for Use with Marginal Maximum Likelihood Estimation and Generalised Item Response Models* [Unpublished masters dissertation]. University of Melbourne.

Wu, M. L., & Adams, R. J. (1993). Simulating Parameter Recovery for the Random Coefficients Multinomial Logit. *Fifth International Objective Measurement Workshop*.

Zammit, S. A. (1997). English and Home Background Languages in Australian Primary Schools. In P. McKay (Ed.), *The Bilingual Interface Project Report* (pp. 111–146). Department of Employment, Education and Training.