LICENCE AGREEMENT

This Agreement is made

between The Australian Council for Educational Research Limited,
ACN 004 398 145, of 19 Prospect Hill Road, Camberwell,
Victoria, ("ACER")

and You, the licensee

1. DEFINITION

1.1 In this Agreement 'software' means the Quest computer
program and related documentation and materials.

2. PRELIMINARY

2.1 The software is licensed, not sold, to you for use only in
accordance with the terms of this agreement, and ACER
reserves all rights not expressly granted to you. You own
the disks on which the software is originally or subsequently
recorded or fixed, but ACER retains ownership of all copies of
the software itself.

3. LICENCE

3.1 This licence allows:

(a) the software to be used solely by an end user (that is, not
for use for marketing or redistribution, either alone or as a
component of any other product).

(b) the software to be installed on a hard disk for use in
accordance with this Agreement or alternatively, in the
event that you do not have a hard disk system, one copy in
machine readable form to be made for back-up purposes.
You must reproduce on the back-up copy the ACER
copyright notice and any other proprietary writing that was
on the original copy supplied by ACER. The software is
protected by copyright law. You are not authorised to make
any copies of the software, except as permitted by this
paragraph; and

(c) the software (including the back-up copy) and all rights
under this Agreement to be transferred to another person
together with a copy of this Agreement as long as you give
ACER written notice of the transfer and the other party
reads and agrees to the terms of this Agreement.

4. RESTRICTIONS

You may not market, distribute or transfer (other than in
accordance with clause 3.1(c) above) any copy of the software
to others or electronically transfer the software from one
computer to another over a network, either on its own or with
or as part of any other product or equipment, without an
express distribution licence from ACER. You may not modify,
adapt, translate, rent, lease, loan, sell, distribute, network or
create derivative works based on the software or any part of it.

4.2 The software contains trade secrets and to protect them
you may not decompile, reverse engineer, disassemble or
otherwise reduce the software to a humanly perceivable form.
You agree not to divulge, directly or indirectly, any such trade
secrets to any person, unless and until such trade secrets cease
to be confidential for any reason not your own fault.

5 TERMINATION

5.1 This Agreement shall continue until terminated in accordance
with the terms of this Agreement. It will terminate
automatically without notice from ACER if you fail to comply
with any provision of it. Upon termination you must destroy
the software and any copies of it you have made. You may
terminate this Agreement at any time by destroying the
software and any copies you may have made. Upon
termination of this licence for any reason:

(a) you will have no right to refund of the whole or any part of
the licence fees or other amounts paid for this licence or the
software (except as expressly provided in section 6 below) and

(b) you shall continue to be bound by the provisions of section 4
above.

LIMITED WARRANTY

ACER warrants that the disk in which the software is recorded
is free from defects in materials and faulty workmanship under
normal use for a period of 90 days from the date of delivery as
evidenced by a copy of the receipt. During this warranty
period, ACER will, at its option, repair or replace, free of
charge, defective disks upon which the software has been
supplied (and if necessary restore the software on such disks),
provided you return the faulty disk with proof of purchase to
ACER, or an authorised ACER agent, in the country where
the software was first supplied to you. ACER has no responsibility
to repair or replace a disk which in ACER's opinion has been
damaged by accident, abuse or misapplication, or as a result of
service or modification by anyone other than ACER or any
authorised ACER agent. All disks replaced under this
warranty become the property of ACER.

DISCLAIMER OF WARRANTY, LIMITATION OF
REMEDIES

The software is provided 'as is', without any kind of warranty.

To the full extent permitted by law, ACER expressly excludes
all conditions, warranties and liability, whether imposed by
statute or by rule of law or otherwise, which are not expressly
set out in this Agreement.

ACER does not warrant, guarantee, or make any
representations (either expressly or impliedly) regarding the
use, or the results of the use, of the software, with respect to its
correctness, accuracy, reliability, completeness, functionality,
currness or otherwise.

ACER does not warrant that the software's operation will be
uninterrupted or error-free, or that defects in it will be
corrected.

You expressly agree that you assume the entire risk as to the
results and performance of the software. If the software is
incorrect, inaccurate, unreliable, incomplete, out-of-date,
defective or otherwise unsatisfactory, you and (not ACER or
its representatives) assume the entire cost of any servicing,
repair or correction.

No oral or written information or advice given by ACER or its
representatives creates a warranty or in any way increases the
scope of this warranty, and you may not rely on any such
information or advice.

ACER's maximum liability for damages arising under this
Agreement is limited to the licence fees paid by you for the
software.

Nothing in this Agreement is intended or shall be construed as
excluding or modifying any statutory rights, warranties or
conditions which are applicable to this Agreement.

WHOLE OF AGREEMENT

This Agreement constitutes the complete understanding of the
parties and no waiver or modification of any provisions shall
be valid.

GOVERNING LAW

The law applicable to this Agreement is the law in force in the
state of Victoria.
ACER Quest

The Interactive Test Analysis System

Raymond J Adams & Siok-Toon Khoo
HELP DESK

A help desk is provided for support in the use of the software. The type of help provided is essentially with command language and software problems.

Contact can be made by mail, phone, fax or e-mail.

Mail        QUEST Help Desk, ACER
            Private Bag 55, Camberwell Victoria 3124
            AUSTRALIA
Phone       +61 3 9277 5555
Fax          +61 3 9277 5500
E-Mail       Quest@acer.edu.au
OVERVIEW........................................................................................................1
    Special Features..........................................................................................2
    Subgroup and Subscale Analyses...............................................................2
    User-defined Variables.............................................................................2
    Anchoring Parameter Estimates...............................................................2
    Dealing with Missing Data.........................................................................2
    Exporting Files...........................................................................................2
    Scoring and Recoding of Data....................................................................2
    Setting Item Scores....................................................................................3
    Structure of Quest......................................................................................3
    Quest User Interface...................................................................................4
    System Requirements................................................................................5

INSTALLATION ...............................................................................................7
    IBM PC or IBM Compatible Running PC/MS-DOS.......................................7
    System Requirements................................................................................7
    Installation..................................................................................................7
    File Preparation........................................................................................9
    File Access.................................................................................................9
    Printing Output........................................................................................10
    Apple Macintosh........................................................................................10
    System Requirements................................................................................10
    Installation................................................................................................10
    File Preparation........................................................................................11
    File Access.................................................................................................12
    Printing Output..........................................................................................13

COMMAND STRUCTURE ..............................................................................15
    Command Word........................................................................................15
    Arguments..................................................................................................15
    Options.....................................................................................................16
    Redirection................................................................................................16
    Miscellany..................................................................................................17

SAMPLE SESSIONS .....................................................................................19
    Sample 1.....................................................................................................20
    Sample 2.....................................................................................................31
    Sample 3.....................................................................................................36

INDEX OF COMMANDS...............................................................................43
    anchor........................................................................................................44
    bye..............................................................................................................46
    cases..........................................................................................................47
    codes..........................................................................................................48
    compare....................................................................................................49
    correlate...................................................................................................53
CONTENTS

data_file .......................... 55
delete ................................ 56
estimate ................................ 58
format ................................ 60
group ................................ 62
header ................................ 64
iscore ................................ 65
itanal ................................ 67
item_names ............................ 70
key .................................... 71
 kidmap ................................ 72
logit_table ............................ 74
recode ................................ 75
reset .................................. 76
sample ................................ 77
scale .................................. 78
score .................................. 79
set ..................................... 80
show ................................... 82
submit .................................. 86

DISPLAY MANAGER .................................................. 88

TECHNICAL NOTES .................................................. 90
The Quest Model .......................... 90
Item Scoring ................................ 91
Estimation ................................ 92
Fit Indices ................................ 92
Expressions of Item Difficulty ............. 93
Reliability Indices ......................... 94
Discrimination Indices and Distractor Analysis ....... 94
Correlation .............................. 94

BIBLIOGRAPHY ......................................................... 96
<table>
<thead>
<tr>
<th>FIGURES</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inputs to and possible outputs from Quest</td>
<td>3</td>
</tr>
<tr>
<td>Output produced by the estimate command</td>
<td>22</td>
</tr>
<tr>
<td>Summary table for the space subscale item estimates</td>
<td>24</td>
</tr>
<tr>
<td>Summary table for the space subscale case estimates</td>
<td>25</td>
</tr>
<tr>
<td>Settings table for Sample 1</td>
<td>26</td>
</tr>
<tr>
<td>A variable (or item) map for all cases on the space subscale</td>
<td>27</td>
</tr>
<tr>
<td>Item fit map for the space subscale items calibrated with all cases</td>
<td>28</td>
</tr>
<tr>
<td>An extract from the table of item estimates for the space subscale</td>
<td>28</td>
</tr>
<tr>
<td>An extract from the table of case estimates for the number subscale</td>
<td>30</td>
</tr>
<tr>
<td>Tables of intercorrelations</td>
<td>30</td>
</tr>
<tr>
<td>Settings table for Sample 2</td>
<td>33</td>
</tr>
<tr>
<td>Item parameter estimates expressed in three alternative forms</td>
<td>34</td>
</tr>
<tr>
<td>A kidmap</td>
<td>35</td>
</tr>
<tr>
<td>Settings table for Sample 3</td>
<td>39</td>
</tr>
<tr>
<td>The variable (or item) map for Sample 3</td>
<td>40</td>
</tr>
<tr>
<td>Extract from the itanal table for Sample 3</td>
<td>41</td>
</tr>
</tbody>
</table>
Quest offers a comprehensive test and questionnaire analysis environment by providing a data analyst with access to the most recent developments in Rasch measurement theory, as well as a range of traditional analysis procedures. It includes an easy-to-use control language with flexible and informative output. Quest can be used to construct and validate variables based on both dichotomous and polychotomous observations. It scores and analyses such instruments as multiple choice tests, Likert-type rating scales, short answer items, and partial credit items.

The Rasch analysis provides item estimates, case estimates, and fit statistics; and the results from this analysis can be accessed through a variety of informative tables and maps. Additional analyses report counts, percentages, and point-biserials for each possible response to each item. A variety of reliability indices is also available.

Quest runs in batch mode, interactive mode, or a combination of the two. The batch mode conveniently allows a one-step submission of more routine analysis jobs, while the interactive environment facilitates exploration of the properties of test data. In interactive mode, the Quest Display Manager allows analysis results to be viewed on the screen.
SPECIAL FEATURES

Several distinct features in Quest set it apart from other test analysis software.

*Subgroup and Subscale Analyses*

Quest allows you to define subgroups and subscales so that analyses can be performed for any combination of subgroup and subscale.

*User-defined Variables*

User-defined variables not only let you define subgroups, but also let you correlate them with case estimates on any subscale.

*Anchoring Parameter Estimates*

Quest allows you to anchor (i.e. fix) any item or case estimate to known values, typically obtained from previous analyses. This facilitates equating of tests and item banking.

*Dealing with Missing Data*

Quest provides flexible procedures for dealing with missing data. You can make Quest ignore certain codes or ‘holes’ in the data. This means you can calibrate several test forms together as long as links exist between forms with common cases or common items.

*Exporting Files*

Quest lets you export analysis results to text files with a choice of tabs, spaces, or commas as field delimiters to facilitate importation into database, spreadsheet, or other programs.

*Scoring and Recoding of Data*

Quest provides flexible and easy-to-use methods for test scoring and data recoding. Easy recoding of data also facilitates regrouping of items and redefinition of scores.
Setting Item Scores

You can assign different scores to different items in a Quest analysis. This allows you to specify the score that is allocated to each item.

STRUCTURE OF QUEST

Figure 1 outlines the necessary input to Quest and summarises possible outputs that you can extract. Quest acts on the commands issued by you, the person analysing the data. The order in which you issue the commands is based on a logical sequence rather than a fixed one. For example:

- Quest needs to know the name of the data file to find it and needs to know some information about the data to make sense of it before any other commands can be executed.
- Scoring and estimation have to be undertaken before results can be extracted.

![Diagram of Quest structure](image)

**Figure 1**  Inputs to and possible outputs from Quest

Quest presents analysis results in easy-to-read tables and maps that you can view on screen or write to files. Various options and combinations of options are available to extract the results from the analyses. An example of the range of possible output is illustrated in the Sample Sessions section of this manual.
You can obtain Rasch estimates and fit statistics for items (refer to the Technical Notes section for the types of item estimates available) and for cases. They can be obtained for different subgroup and subscale combinations and also in different orders: entry order, estimate order, fit order, and name (alphabetical) order. You can export these results to file in:

- anchor format, which is used for anchoring in later analyses
- export format, which allows you to import the results into other software packages

Quest presents some results in map form: e.g. the item maps, fit maps, and kidmaps. Item maps present the items along a logit scale according to their estimates, and case distribution maps are displayed along the same continuum. Fit maps present the fit statistics pictorially. Fit statistics for items are calculated over all cases, but Quest also gives you the option to display item fits for a particular case. Similarly, you can calculate case fits either over all items on the scale or for just a selected item. Kidmaps provide a pictorial profile and diagnostic information for the performance of selected individuals on each item, as well as on the whole test.

Summary tables give mean and standard deviation statistics, reliability indices, and current program settings.

**QUEST USER INTERFACE**

The Quest user interface is designed primarily to encourage you to explore test data. The interactive environment and the Display Manager allow you to browse through the results of commands. On the basis of these results, you can decide upon the next sequence of actions and commands, carry them out, and study the new results.

Quest command statements are logical, are easy to learn, and have minimal fixed syntax. The Display Manager lets you scroll up or down an
output one screen at a time, request the next output, and save the output to file.

A batch processing mode allows you to submit a sequence of commands stored in an external submit file. Quest also allows you to stack jobs, which means that any number of jobs can be lined up and run without attendance.

Quest can read information about anchoring, item names, and so forth from external files and can redirect output to external files. This capability helps you to easily organise and store Quest input and output.

SYSTEM REQUIREMENTS

Quest has been implemented on several platforms, including Vax/VMS, IBM PCs or compatibles running PC-DOS or MS-DOS, and Apple Macintosh.

The PC/MS-DOS version runs on any IBM compatible but requires at least 3 megabytes of free RAM, a hard disk, a math coprocessor and a 80386 processor. This version can make use of extended memory.

Macintosh versions of Quest require at least a 68020 processor with 6 megabytes of free RAM and a math coprocessor (FPU).

For information on installing and running Quest on an IBM PC or compatible or an Apple Macintosh, refer to the Installation section.

For information on system requirements or installation for other platforms, please contact ACER.
IBM PC OR IBM COMPATIBLE RUNNING PC/MS-DOS

System Requirements

The minimum requirements for PC/MS-DOS versions of Quest are 3 megabytes of free RAM, a hard disk, a math coprocessor and a 80386 processor.

Installation

The following instructions assume that your hard disk is drive C and that you have placed the Quest master disk in floppy drive A. Substitute the actual name of your hard or floppy drive in the commands if required.

1 Type
   dir A:
   at the C:\> prompt to check that theQuest master disk contains two files:
   PKARC.COM
   QUEST.ARC

2 Type
   mkdir quest
   at the C:\> prompt to create a subdirectory named QUEST on your hard disk.

3 Type
   a:pkxarc a:quest c:\quest
   at the C:\> prompt. This command unarchives the files in the QUEST.ARC file and copies them to your QUEST subdirectory.
4 Type
   dir \quest

at the C:/> prompt to check that the QUEST subdirectory on your
hard disk contains ten files:

   QUEST.EXE    the Quest application
   SAMPLE1.CTL  the submit file for Sample 1
   SAMPLE1.DAT  the data file for Sample 1
   SAMPLE2.CTL  the submit file for Sample 2
   SAMPLE2.DAT  the data file for Sample 2
   SAMPLE3.CTL  the submit file for Sample 3
   SAMPLE3.DAT  the data file for Sample 3
   SAMPLE3.NAM  the item name file used in Sample 3
   SAMPLE3.DEL  the item delete file used in Sample 3
   SAMPLE3.ANC  the item anchor file used in Sample 3

5 Eject the Quest master disk and put it away for safekeeping.

6 To run Quest, you must either:
   — set a path to the QUEST subdirectory in your AUTOEXEC.BAT
     file
   — make sure you are in the QUEST subdirectory

Then type
   quest

at the DOS prompt.

7 When you are presented with the Quest > prompt, you can start
doing data analysis by typing in Quest command statements at the
prompt. (See the Command Structure and the Index of Commands
sections.)

Alternatively, you can run one of the sample jobs by typing
   submit sample\n.ctl

where n is a number from 1 to 3. (The Sample Sessions section of the
manual describes the command statements in the sample submit
files and some of their output.)
**File Preparation**

1. The input files read by Quest are standard ASCII (or text) files. This means you can create or edit Quest input files using a text editor or a word processor that can save files in ASCII (often called Text Only) format. For example, if you are using Microsoft Word to edit input files, make sure that you save the file in the Text Only format that is provided by Word. Quest input files include data, submit, item or case anchor, item or case delete, item name, and score files. Formats for the latter five file types are given in the descriptions of the submit, anchor, delete, item_names, and iscore commands (see Index of Commands section).

2. The output files created by Quest are text files. This means you can view, edit, or print Quest output files using a text editor or a word processor that can read ASCII (or text) files. Just launch your word processor and then open the Quest output file that you would like to view, edit, or print. Remember to save the output file in ASCII (Text Only) format if you edit it with a word processor.

**File Access**

1. To ensure that Quest can access input files, you must do one of the following:
   - place the input files in the QUEST subdirectory
   - use the complete DOS pathname when you specify the input file in a command line (for example, submit C:\qinput\namelist tells Quest to read commands from the file NAMELIST in the QINPUT subdirectory on the C drive)
   - use the directory= argument of the set command to direct Quest to the subdirectory that contains the input files (for example, set directory=\qinput tells Quest to look for input files in the QINPUT directory on the same drive on which Quest is installed).
Quest writes output files either to:

- the subdirectory you have specified in the `directory=` argument of the `set` command
- the QUEST subdirectory if you have not issued a `set directory=` command statement.

Printing Output

1. The Quest output files can be printed by a text editor or word processor. To open a Quest output file with your word processor, launch the word processor first and then use the appropriate command to open the file.

2. Before printing an output file, you may need to do some reformatting. For example, you should use a nonproportional (monospaced) font to ensure that output columns remain correctly aligned.

3. Some Quest output uses a wide format, and you may need to experiment with the `width=` argument of Quest’s `set command` and the scaling facilities of your printer or word processor to produce an attractive printout.

APPLE MACINTOSH

System Requirements

The minimum requirements for the standard Apple Macintosh version of Quest are six megabytes of free RAM, a 68020 processor, and a math coprocessor.

Installation

1. Insert the Quest master disk into the floppy drive.

2. Create a new folder (command+N) on your hard disk and name it `Quest`. 
3 Drag (copy) all files from the Quest master disk into the Quest folder.

4 Double-click the Quest folder to open it. Make sure the folder contains eleven files:

```plaintext
quest.exe  the Quest application
errmsg.sub a run time library of error messages
sample1.ctl the submit file for Sample 1
sample1.dat the data file for Sample 1
sample2.ctl the submit file for Sample 2
sample2.dat the data file for Sample 2
sample3.ctl the submit file for Sample 3
sample3.dat the data file for Sample 3
sample3.nam the item name file used in Sample 3
sample3.del the item delete file used in Sample 3
sample3.anc the item anchor file used in Sample 3
```

5 Eject the Quest master disk and put it away for safekeeping.

6 Launch Quest by double-clicking on the Quest program icon. Note that you cannot launch Quest by double-clicking on a Quest input or output file.

7 When you are presented with the Quest > prompt, you can start doing data analysis by typing in Quest command statements at the prompt. (See the Command Structure and the Index of Commands sections.)

Alternatively, you can run one of the sample jobs by typing

```
submit samplen.ctl
```

where `n` is a number from 1 to 3. (The Sample Sessions section of the manual describes the command statements in the sample submit files and some of their output.)

**File Preparation**

1 The input files read by Quest are standard ASCII (or text) files. This means you can create or edit Quest input files using a text editor or a word processor that can save files in ASCII (often called Text Only) format. For example, if you are using Microsoft Word to edit input
files, make sure that you save the file in the Text Only format that is provided by Word. Quest input files include data, submit, item or case anchor, item or case delete, item name, and score files. Formats for the latter five file types are given in the descriptions of the submit, anchor, delete, item_names, and iscore commands (see Index of Commands section).

The output files created by Quest are text files. This means you can view, edit, or print Quest output files using a text editor or a word processor that can read ASCII (or text) files. Just launch your word processor and then open the Quest output file that you would like to view, edit, or print. Remember to save the output file in ASCII (Text Only) format if you edit it with a word processor.

File Access

To ensure that Quest can access input files, you must do one of the following:

- place the input files in the Quest folder

- use the complete Apple pathname when you specify the input file in a command line (for example, submit dataframe20qinputnamelist tells Quest to read commands from the namelist file in the qinput folder on the dataframe20 hard disk)

- use the directory= argument of the set command to direct Quest to the folder that contains the input files (for example, set directory=Dataframe 20qinput tells Quest to look for input files in the qinput folder on the Dataframe 20 hard disk).

Quest writes output files either to:

- the hard disk and folder you have specified in the directory= argument of the set command

- the Quest folder if you have not issued a set directory= command statement.
Printing Output

1. The Quest output files can be printed by a text editor or word processor. To open a Quest output file with your word processor, double-click the word processor icon and then use the Open command to open the file.

2. Before printing an output file, you may need to do some reformatting. For example, you should use a nonproportional (monospaced) font to ensure that output columns remain correctly aligned.

3. Some Quest output uses a wide format, and you may need to experiment with the width= argument of Quest's set command and the scaling facilities of your printer or word processor to produce an attractive printout.
COMMAND STRUCTURE

Quest is a fully interactive system that you control through the use of typed commands entered in response to the Quest > prompt.

A Quest command can consist of up to four components:
- a command word
- arguments
- options
- redirection

A complete command is called a command statement and can comprise one or more command lines.

COMMAND WORD

All command statements begin with a command word (indicated by bold type in this manual). Example command words include:
- quit, which ends a session
- estimate, which starts an estimation of item and/or case parameters

ARGUMENTS

Most (but not all) command words require an argument. Any argument required is entered following the command word. It is separated from the command word by one or more spaces. In many instances, Quest provides default arguments so that you do not have to enter the most commonly used arguments. Some arguments are user-defined variables (indicated by bold italic type), while others are Quest-provided terms
(indicated by bold type). Example command word and argument combinations are:

- **codes** `abcdef`, which specifies those codes in the item response data that are considered valid
- **key** `abcdefaacddfeefabb`, which enters a user-defined scoring key
- **show items**, which displays the Rasch model item estimates

**OPTIONS**

Command words occasionally require options (indicated by bold type). Some options use user-defined variables (indicated by bold italic type). Options are entered following the command word or following the command word and argument (if an argument is entered). The beginning of an option list is indicated by an exclamation mark (!). Multiple options are separated by semicolons (;). Quest always provides the most common options as defaults. Example command statements that include options are:

- **key** `abcdefaacddfeefabb` ! **score**:2, which enters a scoring key and has it scored as 2
- **scale** `21-37` ! **number**, which defines a subscale that consists of items 21 through 37 and gives it the name **number**
- **estimate** ! **iter**:5; **scale**: **number**, which estimates parameters for the item subscale named number, pausing every five iterations of the estimation process to check on convergence

**REDIRECTION**

Quest can read input from or write output to external files. To cause output to be redirected to a file, the output redirection symbol (`>>`) is entered followed by a file name. To cause input to be taken from a file, the input redirection symbol (`<<`) is entered followed by a file name. Redirection is the final part of a command statement; it follows the command word and
all arguments and options. Quest displays a warning message if you attempt to redirect data to a file that already exists. The additional redirection symbols -> and >>= have been added to the version 1 symbols (>>) and (<<). The append redirection symbol (>>) causes output to be appended to the file that is given after the symbol. The overwrite redirection symbol (->) causes the output to be written over the file that is given after the symbol. If the file given to the right of the redirection symbol does not already exist, then both new redirecton symbols behave in the same manner as the output redirection symbol (>>). Example command statements that include redirection are:

- delete << items.del, which deletes the items listed in the item delete file items.del
- score << myData.dat, which scores the data in the data file myData.dat
- show item ! form=map >> map.out, which produces a display of Rasch model item estimates, in the map form, and writes them to an output file named map.out

STARTUP COMMAND FILE

Quest is now able to automatically execute a set of commands at start up. If a file called quest.ini is located in the same directory as Quest, then Quest will attempt to process this file when it is executed.

MISCELLANY

- Each command line is ended by pressing the Enter or Return key.
- All command words can be abbreviated to their shortest unambiguous root.

Example:
estimate, estimat, estima, estim, esti, est, and es can all be used for estimate, but e cannot because the command end also begins with the letter e
In most contexts, Quest ignores case. See the group command in the Index of Commands section for an exception to this.

Example:

QUIT, Quit and quit are all treated identically.

Extraneous spaces are ignored. You can use file, directory, or drive names that contain spaces if your operating systems allows this. At least one space must separate a command word and an argument.

Example:

score  \(<\<\ MyData.dat and score<\<MyData.dat are treated identically.

Command lines must not exceed 70 characters. If a command statement (e.g., a key command statement) requires more than 70 characters of input, then enter a hyphen (-), the continuation character, to indicate that the command statement continues on the next line. If the last character entered on a line is a hyphen, then the next line is treated as a continuation of the current command statement. When Quest detects a continuation, it changes its prompt from > to +. There is no limit on the number of continuation lines, but a command statement cannot exceed 360 characters. The calculation of the command line and command statement line lengths includes spaces but excludes continuation characters and comments (discussed below).

Comments can be included in command lines and in all input files, other than data files, by preceding them with an asterisk (*). If you need to use the continuation character (-), it must precede the asterisk.

Examples:

* All of this is a comment line

codes abcd * Everything after the asterisk is commenting

key abdbcdbd - * The continuation symbol must precede

babcbdbd * the asterisk.
This section describes the three sample jobs provided with Quest, as well as the output that they produce. These jobs do not show all of Quest's potential, but they introduce the major commands and provide an opportunity to describe the Quest output. The data files and submit files used in these sample jobs are provided on your Quest master disk.

The first sample job analyses a multiple choice mathematics test with two subscales. The second sample job is a partial credit analysis of ten performance assessments for a subgroup of the sample. The final sample job is a partial credit analysis of a set of open-ended science items that uses previously estimated item estimates.

Each of the sample job descriptions includes a listing of the associated submit file in bold type. Each Quest command statement has been numbered so that it can be discussed along with the action that results from issuing the command.
The first sample job, an analysis of a multiple choice mathematics test with two subscales, is run by typing

```
submit sample1.ctl
```

at the Quest > prompt. Below is a listing of the sample1.ctl file:

```c
1 title Sample Run One: Multiple Choice Test
2 data_file sample1.dat
3 codes 0123459
4 format name 1-5 sex 6 items 10-46
5 key 4132144232431223212232314231244241433 !score=1
6 scale 1-20 ! space
7 scale 21-37 ! number
8 estimate ! scale=space,number
9 show ! scale=space,number
10 show items ! scale=space,number
11 show cases ! scale=number
12 correlate sex,space,number
13 quit
```

**Line 1:** The `title` command is used to specify a heading that will be printed at the top of all Quest output. The `title` command can be issued at any time and applies to all output until the job is concluded or a subsequent `title` command is issued. The `header` command (described in the Index of Commands section) is an alternative for the `title` command.

**Line 2:** The `data_file` command provides the name of the data file from which response data are to be read. The file names that are allowed will depend upon the operating system of the machine that is being used. The `estimate` and `score` commands (described in the Index of Commands section) provide alternative mechanisms for specifying a data file name.

**Line 3:** The `codes` command specifies those codes in the item response data that are considered valid. In this sample, the data come from a multiple choice test, and 0, 1, 2, 3, 4, 5, and 9 are regarded as legitimate data. Any other code in the item response columns will be treated as indicating missing data.
Line 4: The format command specifies the format of the data. In this case, a student identification is in columns 1 to 5, column 6 contains a sex code, and the item response data is in columns 10 to 46 (a total of thirty-seven items has been specified).

Line 5: Multiple choice tests commonly include a scoring key. The key command gives the scoring key for the thirty-seven multiple choice items. The option score=1 indicates that item responses that match this key should be assigned a score of one. Because a score of one is the default when the key command is used, this option is unnecessary in this instance; but if multiple score keys were used to give different scores to different alternatives, then the score= option might be necessary.

Lines 6 and 7: The scale command is used to name subscales. This item set contains two subscales. The first twenty items are from a subscale named space, while the remaining seventeen items belong to a subscale named number. Identifying subscales of items allows the separate calibration of any item subset within the data file.

Line 8: The estimate command initiates a Rasch model estimation of the data. In this case, estimation has been requested for each of the two subscales number and space. Quest begins by scoring the data for each of the subscales and then performs separate calibrations. Figure 2 shows an extract of the information that Quest reports as it performs this Rasch estimation.

Line 9: The show command is the principal mechanism for viewing the results of estimation. The show command is issued here without an argument so it will produce the three default tables and two default maps for each of the subscales named in the scale= option (that is, a total of ten displays). Quest output can be viewed on the screen or redirected to a file. This sample show command does not include redirection, so the tables and maps it produces will be displayed on the screen. Control of screen displays is managed by a Quest subsystem called the Display Manager.
Scoring all -- space

Scoring all -- number

Starting Estimation for all -- space
iteration 1:
  item estimates -->
  maximum change (item 3) = 1.2013
  average change = 0.3926
  person estimates -->
  maximum change (case 483) = 0.2174
  average change = 0.1336

iteration 2:
  item estimates -->
  maximum change (item 3) = 0.0056
  average change = 0.0025
  person estimates -->
  maximum change (case 483) = 0.0264
  average change = 0.0169
  estimation converged with:
  maximum change in item estimates = 0.0005
  maximum change in person estimates = 0.00

Performing fit analysis

Starting Estimation for all -- number
iteration 1:
  item estimates -->
  maximum change (item 37) = 0.7701
  average change = 0.3361
  person estimates -->
  maximum change (case 496) = 0.1418
  average change = 0.0811

iteration 4:
  item estimates -->
  maximum change (item 37) = 0.0028
  average change = 0.0007
  person estimates -->
  maximum change (case 496) = 0.0054
  average change = 0.0028
  estimation converged with:
  maximum change in item estimates = 0.0002
  maximum change in person estimates = 0.00

Performing fit analysis

Figure 2  Output produced by the estimate command
which allows you to inspect large output tables (see the Display Manager section).

Extracts from the first three tables and two maps produced by this show command statement are illustrated and described in Figures 3, 4, 5, 6, and 7. These displays all provide results for the space subscale. The second set of five displays (not shown in the manual) contains the results for the number subscale.

Figures 3 and 4 are the summary tables for items and cases respectively. They provide test and sample level summary statistics that describe the estimation.

Figure 5 contains the settings table. The settings table indicates the following: the name and format of the data file, the name of the log file (if any), the widths and lengths that have been specified for both page (file) and screen output, the codes that have been identified as valid, the subgroups and subscales that have been defined, the cases and items that have deleted or anchored, and, when used, any recodes or scoring keys.

Figure 6 is a variable map (sometimes referred to as an item map). The variable map shows the distribution of item difficulties and the distribution of case estimates over the variable.

Figure 7 is an item fit map. It shows the infit mean square for each item on the requested subscale. The right-hand vertical dotted line indicates a mean square that is 30 per cent above its expected value, and the left-hand vertical dotted line indicates a mean square that is 30 per cent below its expected value. We suggest these as basic rules of thumb for determining the adequacy of item fit to the model. Additional discussion of the fit indices is provided in the Technical Notes section. Quest can also produce a case fit map.
### Sample Run One: Multiple Choice Test

**Item Estimates (Thresholds)**
All on space \((N = 500 \, L = 20)\)

<table>
<thead>
<tr>
<th>Summary of Item Estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mean</strong></td>
</tr>
<tr>
<td><strong>SD</strong></td>
</tr>
<tr>
<td><strong>SD (adjusted)</strong></td>
</tr>
<tr>
<td><strong>Reliability of estimate</strong></td>
</tr>
</tbody>
</table>

**Fit Statistics**

<table>
<thead>
<tr>
<th>Infit Mean Square</th>
<th>Outfit Mean Square</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mean</strong></td>
<td>1.00</td>
</tr>
<tr>
<td><strong>SD</strong></td>
<td>0.07</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Infit t</th>
<th>Outfit t</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mean</strong></td>
<td>0.07</td>
</tr>
<tr>
<td><strong>SD</strong></td>
<td>0.94</td>
</tr>
</tbody>
</table>

0 items with zero scores
0 items with perfect scores

These results are for the threshold form of the item estimates provided by the Rasch estimation (see Technical Notes section). They are for all cases on the space subscale. The sample size was 500 and the subscale contains 20 items.

Mean and SD are the mean and standard deviation of the item thresholds for this subscale. SD (adjusted) is the standard deviation corrected for measurement error. Reliability of estimate is the proportion of the observed estimate variance that is considered true. This is the index called item separation reliability by Wright and Masters (1982).

These are the means and standard deviations of the infit (weighted) and outfit (unweighted) fit statistics in their mean square and transformed \(t\) forms. When the data are compatible with the model, the expected value of the mean squares is approximately one and the expected value of the \(t\)-values is approximately zero.

These indicate the number of items that could not be estimated because no correct response was found or because all responses to the item were correct.

---

**Figure 3** Summary table for the space subscale item estimates
Mean and SD are the mean and standard deviation of the case estimates for this subscale. SD (adjusted) is the standard deviation corrected for measurement error. Reliability of estimate is the proportion of the observed estimate variance that is considered true. This is the index called person separation reliability by Wright and Masters (1982).

These are the means and standard deviations of the infit (weighted) and outfit (unweighted) fit statistics in their mean square and transformed (t) forms. When the data are compatible with the model, the expected value of the mean squares is approximately one and the expected value of the t-values is approximately zero.

These indicate the number of cases that could not be estimated because they scored all items correctly or because they scored all items incorrectly.

Sample Run One: Multiple Choice Test
Case Estimates
All on space (N = 500 L = 20)

Summary of Case Estimates

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>1.75</td>
</tr>
<tr>
<td>SD</td>
<td>1.09</td>
</tr>
<tr>
<td>SD (adjusted)</td>
<td>0.84</td>
</tr>
<tr>
<td>Reliability of estimate</td>
<td>0.59</td>
</tr>
</tbody>
</table>

Fit Statistics

<table>
<thead>
<tr>
<th></th>
<th>Infit Mean Square</th>
<th>Outfit Mean Square</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>1.00</td>
<td>Mean</td>
</tr>
<tr>
<td>SD</td>
<td>0.21</td>
<td>Mean</td>
</tr>
<tr>
<td>SD</td>
<td>0.10</td>
<td>Mean</td>
</tr>
<tr>
<td>SD</td>
<td>0.75</td>
<td>Mean</td>
</tr>
<tr>
<td>SD</td>
<td>0.68</td>
<td>Mean</td>
</tr>
</tbody>
</table>

0 cases with zero scores
42 cases with perfect scores

Figure 4 Summary table for the space subscale case estimates
Sample Run One: Multiple Choice Test

Current System Settings 13-AUG-92 11:40:40
all on all (N = 7 L = 37)

Data File = sample1.dat
Data Format = name 1-5 sex 6 items 10-46
Log file = LOG not on
Page Width = 80
Page Length = 65
Screen Width = 78
Screen Length = 24
VALID DATA CODES 0 1 2 3 4 5 6

GROUPS
1 all ( 7 cases ) : All cases

SCALES
1 all ( 20 items ) : All items
2 space ( 20 items ) : 1-20
3 number ( 17 items ) : 21-37

DELETED AND ANCHORED CASES:
No case deletes or anchors

DELETED AND ANCHORED ITEMS:
No item deletes or anchors

RECODES

SCORING KEYS
Score = 1 4132142324312321222321231242241433

The settings table provides a summary of many of the current program settings.

Figure 5  Settings table for Sample 1
The figures on the extreme left of the map represent the logit scale on which both items and cases are calibrated.

The XXXs on the left-hand side of the map represent the distribution of case estimates over the logit scale.

The figures on the right-hand side of the map represent items plotted according to their difficulty.

Figure 6  A variable (or item) map for all cases on the space subscale
### Sample Sessions

**Figure 7** Item fit map for the space subscale items calibrated with all cases

**Figure 8** An extract from the table of item estimates for the space subscale

**SCOR** is the item's raw test score or the number of points scored on this item by all cases in the calibrating group.

**MAAS** is the maximum possible score that the item could have received. For a dichotomous item, where correct responses to each item have been given a score of one, it is simply how many cases took the item.

**THRS** is the parameter estimate that describes the difficulty of the item. For dichotomous data, the threshold is the usual item difficulty estimate. For a discussion of alternative expressions for the difficulty of polychotomously scored items, see the Technical Notes section. The number below each item threshold is the associated standard error.

The **INFT MNSQ** and **OUTFT MNSQ** and the **INFT t** and **OUTFT t** values are expressions of the fit of the item to the model. If the item fits the model, both the infit and outfit mean squares have expected values of one and the infit t and outfit t values have expected values of zero. The differences between these statistics are discussed in more detail in the Technical Notes section.
Line 10: The `show` command is used here with the `items` argument and the `scale=` option, resulting in two tables of item parameter estimates: one for the space subscale and one for the number subscale. Figure 8 is an extract from the first table produced by this command. The second table (not shown in the manual) has the same format, but it will contain results for the number subscale.

Line 11: The `show` command is used here with the `case` argument and the `scale=` option to produce a table that contains the ability estimates of all 500 cases for the number subscale. An extract from the resulting table, showing estimates for the first twenty-three cases, is given in Figure 9.

Line 12: The `correlate` command requests correlations between the scale scores for the subscales space and number and the input variable `sex`. The resulting output is shown in Figure 10.

Line 13: `Quit` can be used to exit from Quest. The commands `bye`, `end`, `exit`, and `stop` are all alternatives to `quit`. If you want to run another set of analyses without exiting from Quest, the `reset` command can be used to return all program settings to their default values.
Sample Run One: Multiple Choice Test

Inter-correlation matrices
All on All ($N = 500$ $L = 17$)

Correlations using Raw Scores

<table>
<thead>
<tr>
<th>sex</th>
<th>space</th>
<th>number</th>
</tr>
</thead>
<tbody>
<tr>
<td>sex</td>
<td>0.14</td>
<td>0.15</td>
</tr>
<tr>
<td>space</td>
<td>500</td>
<td>0.69</td>
</tr>
<tr>
<td>number</td>
<td>500</td>
<td></td>
</tr>
</tbody>
</table>

Correlations using Rasch Estimates

<table>
<thead>
<tr>
<th>sex</th>
<th>space</th>
<th>number</th>
</tr>
</thead>
<tbody>
<tr>
<td>sex</td>
<td>0.09</td>
<td>0.15</td>
</tr>
<tr>
<td>space</td>
<td>458</td>
<td>0.62</td>
</tr>
<tr>
<td>number</td>
<td>436</td>
<td>409</td>
</tr>
</tbody>
</table>

The first table provides the intercorrelations between the raw test scores of the cases on the selected subscales.

The second table provides the intercorrelations between the Rasch estimates of the cases.

The value above the diagonal is the correlation, and the value below the diagonal is the number of cases that have been used in the calculation of that correlation.

Figure 10 Tables of intercorrelations
Sample 2 illustrates a partial credit analysis, introduces the recode and kidmap commands, and applies the group command to define a subgroup of the cases in the data set.

To run the second sample job, type

submit sample2.ctl

at the Quest > prompt.

This file does not end with a bye or quit command, so the Quest > prompt will still be displayed after Quest has executed the commands in the submit file. You may want to experiment with some commands at that point or just type

bye

if you want to end the Quest session.

Below is a listing of the sample2.ctl file:

```
1 title Sample Run Two: Partial Credit Test
2 data_file sample2.dat
3 codes 1234
4 recode (1234) (0123)
5 format name 1-4 sex 5 age 7-8 items(11-20
6 group ((age>09) .and. (sex=f)) ! f>9
7 show settings
8 estimate ! iter=20;group=f>9
9 show items ! stat=delta;group=f>9
10 show items ! stat=tau;group=f>9
11 show items ! group=f>9
12 kidmap ! group=f>9
```

Lines 1 to 3: The first three command statements are similar to those described in Sample 1 (page 20). They specify a title, the name of the data file, and the valid data codes.

Line 4: The recode command can be used to redefine the values found in the data file for any selection of items. In this sample, we have chosen to use the recode command to score the data by mapping the observed
data codes 1, 2, 3, and 4 to the scores 0, 1, 2, and 3. This recoding is applied
to all items, which is the default when no item is specified.

Line 5:  This format command statement specifies the columns for the
three identification variables (name, sex, and age) and the ten items.

Line 6:  The data set contains a subgroup of individuals that are of
special interest: the girls who are older than 9. The group command
identifies the cases that satisfy the given conditions and gives the sub-
group the name f>9.

Line 7:  The settings table that results from the show command with the
settings argument is given in Figure 11. In Sample 1, the settings table was
one of the default displays produced by the show command when it was
given without an argument.

Line 8:  This estimate command statement initiates a Rasch estimation
for all items and those cases in subgroup f>9. The iter= option requests
Quest to pause every 20 iterations to ask whether estimation should
continue or whether it should be terminated. The default value for iter=
is 5.

Lines 9 to 11:  These three show commands are similar in that they each
produce item parameter estimates for all items calibrated using the f>9
subgroup. The stat= option, however, is used to request different represen-
tations of these item parameter estimates. Line 11 does not include a
stat= option, so it produces the default representation, which is
stat=threshold. Figure 12 shows extracts from the tables produced by
each of these show commands. A description of the three representations
for the item parameter estimates is given in the Technical Notes section.

Line 12:  Kidmaps illustrate the response patterns of individual cases.
They can be produced for any selection of cases in any subgroup on any
subscale. Here kidmaps are requested for all cases in the f>9 subgroup
(Figure 13).
This is the logical condition that defines the b>9 subgroup. It contains 657 cases.

The recode command resulted in the following set of recodes for each item.

<table>
<thead>
<tr>
<th>RECODES</th>
<th>SCORING KEYS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 item 1</td>
<td>(1234) TO (1023)</td>
</tr>
<tr>
<td>2 item 2</td>
<td>(1234) TO (1023)</td>
</tr>
<tr>
<td>3 item 3</td>
<td>(1234) TO (1023)</td>
</tr>
<tr>
<td>4 item 4</td>
<td>(1234) TO (1023)</td>
</tr>
<tr>
<td>5 item 5</td>
<td>(1234) TO (1023)</td>
</tr>
<tr>
<td>6 item 6</td>
<td>(1234) TO (1023)</td>
</tr>
<tr>
<td>7 item 7</td>
<td>(1234) TO (1023)</td>
</tr>
<tr>
<td>8 item 8</td>
<td>(1234) TO (1023)</td>
</tr>
<tr>
<td>9 item 9</td>
<td>(1234) TO (1023)</td>
</tr>
<tr>
<td>10 item 10</td>
<td>(1234) TO (1023)</td>
</tr>
</tbody>
</table>

Figure 11  Settings table for Sample 2
Sample Run Two: Partial Credit Test

Item Estimates (Category Deltas) in Input Order f=0 on All (N = 657 L = 10)

<table>
<thead>
<tr>
<th>ITEM NAME</th>
<th>SCORE MAXSCR</th>
<th>DELTA/S</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1  2   3</td>
</tr>
<tr>
<td>1 item 1</td>
<td>337 1956</td>
<td>-.30   1.96 1.98</td>
</tr>
<tr>
<td></td>
<td></td>
<td>.21   .29   .36</td>
</tr>
<tr>
<td>2 item 2</td>
<td>347 1962</td>
<td>-.50   2.12 3.38</td>
</tr>
<tr>
<td></td>
<td></td>
<td>.11   .22   1.01</td>
</tr>
</tbody>
</table>

This table shows the item parameter estimates in their delta form. It is produced by using the option stats=delta.

Sample Run Two: Partial Credit Test

Item Estimates (Difficulty and Tau) in Input Order f=0 on All (N = 657 L = 10)

<table>
<thead>
<tr>
<th>ITEM NAME</th>
<th>SCORE MAXSCR</th>
<th>DIFFICULTY</th>
<th>TAU/S</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1  2   3</td>
<td></td>
</tr>
<tr>
<td>1 item 1</td>
<td>337 1956</td>
<td>1.03  -1.32 0.84 0.36</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>.07   .09   .18   .3</td>
<td></td>
</tr>
<tr>
<td>2 item 2</td>
<td>347 1962</td>
<td>1.66  -2.16 0.45 1.71</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>.07   .09   .20   1.0</td>
<td></td>
</tr>
</tbody>
</table>

This table shows the item parameter estimates in their difficulty and tau forms. It is produced by using the option stats=tau.

Sample Run Two: Partial Credit Test

Item Estimates (Thresholds) in Input Order f=0 on All (N = 657 L = 10)

<table>
<thead>
<tr>
<th>ITEM NAME</th>
<th>SCORE MAXSCR</th>
<th>THRESHOLD/S</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1  2   3</td>
</tr>
<tr>
<td>1 item 1</td>
<td>337 1956</td>
<td>-.41  1.42 2.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td>.18   .28   .36</td>
</tr>
<tr>
<td>2 item 2</td>
<td>347 1962</td>
<td>-.59  1.98 3.58</td>
</tr>
<tr>
<td></td>
<td></td>
<td>.20   .37   .98</td>
</tr>
</tbody>
</table>

This table shows the item parameter estimates in their threshold form. It is produced by using the option stats=threshold or by omitting the stats-option (i.e., it is the default form).

Figure 12 Item parameter estimates expressed in three alternative forms
The heading gives the name of the case, the subgroup and subscale combination that has been used in the estimation, the logit ability estimate of the case, the mean square fit statistic, and the percentage correct score that was achieved by the case.

The vertical scale of the kidmap is the logit scale, and items are plotted at their difficulty level.

Items are indicated in the form $x\_y$, where $x$ is the item number and $y$ is the item level. For example, the entry 2.3 is plotted at the difficulty of obtaining level 3 on item 2.

Items are plotted on the left side of the kidmap if they were passed by the case and on the right if they were not passed.

The row of three Xs (XXX) indicates the ability estimate of the case, and the horizontal dotted lines are drawn at plus and minus one standard error.

When a case's pattern of responses conforms to the model, we expect the majority of items below the ability estimate of the case (the three Xs) to be plotted on the left of the kidmap and the majority of items above the ability estimate of the case to be plotted on the right of the kidmap.

Figure 13  A kidmap
Sample 3 is a partial credit analysis of a set of open-ended science items that uses previously estimated item estimates. Sample 3 uses the recode command to score a set of items where each item has a different set of scoring keys, and it introduces the item_names, delete, anchor, and itanal commands.

To run the third sample job, type

```plaintext
submit sample3.ctl
```

at the Quest > prompt.

This file does not end with a bye or quit command, so the Quest > prompt will still be displayed after Quest has executed the commands in the submit file. You may want to experiment with some commands at that point or just type

```plaintext
bye
```

if you want to end the Quest session.

Below is a listing of the sample3.ctl file:

```plaintext
1   title Sample Run Three: What happened last night
2   data sample3.dat
3   codes ABCDEFWX
4   recode (ABCWX) (21100) !1
5   recode (ABCWX) (32100) !2
6   recode (ABCDEWFX) (32211000) !3
7   recode (ABCWX) (21000) !4
8   recode (ABCDEWFX) (3211100) !5
9   recode (ABWFX) (2100) !6
10  recode (ABCWX) (32100) !7
11  recode (ABCDEWX) (321100) !8
12  format rater 1 stratum 2-3 name 2-7 items 10-17
13  item_names <<sample3.nam
14  delete <<sample3.del
15  anchor <<sample3.anc
16  group (stratum<41) !YR5
17  group (stratum>=41) !YR9
18  show settings
19  estimate !conv=.05
20  show items !form=map
21  itanal
```
Lines 1 to 3: The first three lines are similar to those described in Sample 1 (page 20). They specify a title, the name of the data file, and the legitimate data codes. Note the use of an abbreviated form of the command word data_file and the use of alphabetic data codes.

Lines 4 to 11: This set of recode command statements scores the data. In this case, eight recodes are required because each item uses a unique mapping of response codes onto scores.

Line 12: This format command statement gives the columns for three identification variables for each case (rater, stratum, and name) and eight items.

Line 13: Under most circumstances, Quest displays will be more informative if you use an item name to describe each item rather than accepting the default, which is item1, item2 etc. This item_names command causes a set of item names to be read from the item name file sample3.nam. If redirection is omitted from the item_names command statement, Quest will prompt you to enter the item names.

Line 14: Items can be omitted from any consideration by deleting them. Here one item (item 2) is deleted because it is listed in the item delete file sample3.del. If redirection is omitted from the delete command statement, Quest will prompt you to enter the items to delete.

Line 15: The items in this data set have been previously estimated by Quest, and their difficulty estimates were exported to the item anchor file sample3.anc. The anchor command requests that these previously estimated values be read back into the program and used as anchor values. Any item or case that is anchored is not estimated by Quest; instead, it is fixed at the previously estimated value.

Lines 16 and 17: These lines use the group command to split the data into two subgroups, YR5 and YR9.
Line 18: The show command produces the settings table displayed in Figure 14.

Line 19: The estimate command initiates the estimation of all items for all cases (since no subgroup or subscale is specified). The option conv=.05 instructs estimation to terminate when the largest change in parameters estimates is less than 0.05. The default value of conv= is 0.005, so this choice will result in faster, less accurate estimation. Note also that, because all undeleted items have been anchored, this estimation will be completed in just one iteration and only results on case convergence will be output.

Line 20: This show command produces a variable (or item) map for all items and all cases. The resulting map is shown and described in Figure 15.

Line 21: The final command in this sample produces a range of traditional and Rasch item statistics. An extract from the table produced by the itanal command is shown in Figure 16. Itanal results are particularly useful for undertaking distractor analyses for multiple choice tests.
Sample Run Three: What happened last night

Current System Settings  13-AUG-92 12:40:18
all on all (N = 7 L = 7)

Data File = sample3.dat
Data Format = rater 1 stratum 2-3 name 2-7 items 10-17
Log file = LOG not on
Page Width = 80
Page Length = 65
Screen Width = 78
Screen Length = 24

VALID DATA CODES  A B C D E F W X

GROUPS
1 all    ( ? cases ) : All cases
2 yr5    ( 576 cases ) : (stratum<41)
3 yr9    ( 455 cases ) : (stratum>41)

SCALES
1 all    ( ? items ) : All items

DELETED AND ANCHORED CASES:
No case deletes or anchors

DELETED AND ANCHORED ITEMS:

1  Earth shape  anchored at -1.03 0.22
2  Earth pic     deleted
3  Falling off   anchored at -1.10 1.99 1.86
4  What is Sun   anchored at -0.96 1.55
5  Moonshine    anchored at -0.12 -0.14 0.04
6  Moon and night anchored at -0.87 1.14
7  Night and day anchored at -0.03 -0.01 -0.44
8  Breathe on moon anchored at 0.52 0.50 -2.74

RECODES
1  Earth shape (ABCDEFWX) TO (211DEF000)
2  Earth pic   (ABCDEFWX) TO (211DEF000)
3  Falling off (ABCDEFWX) TO (32211000)
4  What is Sun (ABCDEFWX) TO (210DEF000)
5  Moonshine  (ABCDEFWX) TO (32111F00)
6  Moon and night (ABCDEFWX) TO (321CEF00)
7  Night and day (ABCDEFWX) TO (321DEF000)
8  Breathe on moon (ABCDEFWX) TO (32111EF00)

SCORING KEYS

Figure 14  Settings table for Sample 3
This variable map has the same form as the one shown in Figure 6.

Items are plotted on the left side of the map according to their estimated thresholds. The notation $x_y$ is used to indicate the threshold for category $y$ in item $x$.

Figure 15 The variable (or item) map for Sample 3
### Sample Run Three: What happened last night

Item Analysis Results for Observed Responses  09/25/81  09:31:48
all on all (N = 1031 L = 7)

<table>
<thead>
<tr>
<th>Item</th>
<th>Earth shape</th>
<th>Infinit MMLQ = 1.52</th>
</tr>
</thead>
<tbody>
<tr>
<td>Categories</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>Count</td>
<td>448</td>
<td>156</td>
</tr>
<tr>
<td>Percent (%)</td>
<td>43.5</td>
<td>15.1</td>
</tr>
<tr>
<td>Pr-Biserial</td>
<td>.52</td>
<td>-.17</td>
</tr>
<tr>
<td>p-value</td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>Mean Ability</td>
<td>.41</td>
<td>-.25</td>
</tr>
<tr>
<td>Step Labels</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Thresholds</td>
<td>-1.33</td>
<td>.07</td>
</tr>
<tr>
<td>Error</td>
<td>.00</td>
<td>.00</td>
</tr>
</tbody>
</table>

Mean test score: 5.33
Standard deviation: 4.21
Internal Consistency: .70

The individual item statistics are calculated using all available data.

The overall mean, standard deviation and internal consistency indices assume that missing responses are incorrect. They should only be considered useful when there is only a limited amount of missing data.

---

**Figure 16  Extract from the item table for Sample 3**

This extract from a response alternative analysis item table provides a summary of traditional and Rasch statistics for each item.

- **Count** is the number of cases providing this response.
- **Percent** is the count as a percentage of the total number of valid responses.
- **Pr-Biserial** is the product moment correlation between the response type and total test raw score, and **p-value** is the significance level of that correlation.

If a Rasch estimation has been undertaken, the mean logit ability of the cases who responded in each category is reported as **Mean Ability** and the item thresholds are reported with their standard errors. In this case, the errors are zero because the item was anchored.

The item table is followed by the mean test raw score, standard deviation, and Cronbach's alpha as a measure of internal consistency.
INDEX OF COMMANDS

This section describes each of the Quest commands and the arguments, options, and redirection that are available for each command. Examples are provided to illustrate the use of the commands, and a set of notes and warnings is provided to clarify the use of the commands.

In the formal description of the commands, default arguments and options are marked with an asterisk. The argument marked with an asterisk is the one that will be used if you do not specify any other argument. Similarly, the option marked with an asterisk is the one that will be used if you do not specify any other option.

Words that form part of the command syntax (command words, arguments, and options) are set in bold type. Syntax placeholders for characters or words to be provided by you (such as file names; lists of codes, items, or cases; and subscale or subgroup names) are set in italic type. User-provided characters or words in the examples are set in bold italic type.

Each argument in a command statement must be preceded by at least one space.

The options portion of the command statement begins with an exclamation mark (!). If you use more than one option in a command statement, you must separate each option with a semicolon (;).

See the Command Structure section for additional information about command syntax.
anchor

turns item and case anchoring on or off, allowing lists of items or cases to be anchored or to be freed

Arguments

on* is the default argument. It indicates that some items or cases will be anchored

off is used when some or all previously anchored items or cases are to be freed for estimation

Options

items* anchor or free items
cases anchor or free cases

Redirection

<< infile either reads the cases or items and their anchor values from infile or reads the cases or items to be freed from infile. If redirection is omitted, Quest will prompt you for input

Examples

anchor

item anchoring is switched on, and Quest prompts you for the anchor items and their anchor values

anchor off !cases
case anchoring is switched off, and Quest prompts you for the cases to be freed for estimation

anchor ! cases << anchor.lis
case anchoring is switched on, and the cases to be anchored and their anchor values are read from the file anchor.lis
anchor off cases << anchor lens

Case anchoring is switched off, and the cases to be freed are read from the file anchor lens

Notes and Warnings

1. Case anchor files consist of one line per case to be anchored or freed. Each line contains a case number followed by an anchor value (free field input so one or more spaces are used to separate fields). The case number is the sequence number of the case in the complete data set, and the anchor value is a logit ability value. The same file can be used to anchor or free cases.

2. Item anchor files consist of one line per item to be anchored. Each line contains an item number followed by one or more anchor values (free field input so one or more spaces are used to separate fields). The item number is defined by the order in which the items are specified in the format statement (see note 2 of the format command), and the anchor values are the item delta values. The same file can be used to anchor or free items.

3. If a large number of items or cases is to be anchored and the anchor values come from a previous Quest analysis, then the form=anchor export format described under the show command can be used, with redirection, to create item or case anchor files.

4. When an item or a case is anchored, the anchor value or values apply to all subgroup and subscale combinations to which the item or case belongs.

5. The anchor command does not check to ensure that the number of anchor values for an item corresponds to the number of estimable parameters. A mismatch between the number of anchor values and the number of estimable parameters has unpredictable consequences.
INDEX OF COMMANDS

bye

terminates a Quest analysis session. This command does not have any arguments.

*Notes and Warnings*

1. The commands `end`, `exit`, `quit`, and `stop` are alternatives for this command.
cases

selects cases by order of entry into the data set

Examples

cases n select the first n case in order for use in the data set for analysis

cases 250 selects the first 250 cases in the data file
INDEX OF COMMANDS

codes

allows specification of valid data codes; any code in the data that is not to be treated as a missing data indicator must be listed

Argument

codelist

the argument consists of a list of valid data codes. If codelist is enclosed in apostrophes, then every character between the apostrophes, including spaces, is treated as a valid data code. If the apostrophes are omitted, every non-space character is treated as a valid data code

Examples

codes 0123

valid codes are set to 0, 1, 2, and 3

codes a b c d *

valid codes are set to a, b, c, d, and *

codes '019 '

valid codes are set to 0, 1, 9, and space

Notes and Warnings

1 When the codes command is used, the currently valid codes are replaced by those given in the argument of the codes command. The codes command also nullifies the effect of any preceding recode command, so if recodes are to be used, they must follow the codes command statement.

2 Codes must be one character in width, and any printable ASCII character is allowed.

3 Codes within codelist cannot be delimited (separated) by any character other than a space; any non-space character will be interpreted as a code.
compare

a general comparative routine that calculates and reports a range of item bias indices including Mantel-Haenszel tests of differential item functioning (DIF) for dichotomous items and tests of parameter invariance for both case and item parameter estimates (Holland and Wainer, 1993)

**Arguments**

item_est
* compares the two sets of item parameter estimates obtained for a set of items when they have been calibrated with two different subgroups

case_est
 compares the two sets of case parameter estimates obtained for a set of cases when they have been measured with two different subscales

**Options**

scale=
a subscale or a pair of subscales. When the argument is item_est, then the scale= option gives the scale for which the comparison should be undertaken. Multiple scales cannot be specified, and the default is all. When the argument is case_est, then the scale= option must give the two different scales for which the comparison of cases is to be undertaken

group=
a subgroup or pair of subgroups. When the argument is case_est, then the group= option gives the group for which the comparison should be undertaken. Multiple groups cannot be specified, and the default is all. When the argument is item_est, then the group= option must give the two different groups for which the comparison of items is to be undertaken
INDEX OF COMMANDS

Form=
displays comparison results in one or more of the following forms:

**table**
a table listing the parameter estimates, their differences and statistical tests of the magnitude of those differences

**diffmap**
a plot showing the magnitude and the direction of the difference between parameter estimates

**plot**
a scatter plot comparing the two sets of parameter estimates

**mh**
a table containing Mantel-Haenszel DIF information

**order=**
Many of the results can be displayed in a variety of orders. A set of different orders can be requested in the same command by separating them using commas

**input**
items are displayed in item number order and cases are displayed in sequence order

**estimate**
items or cases are sorted according to their difficulty or ability estimates respectively

**chisq**
items or cases are sorted according to the chi-square test of their difference

**stddiff**
items or cases are sorted according to the standardised difference

**MHalpaha**
items are sorted according to the Mantel-Haenszel alpha statistic (available only for form=mh)

The following set of options is only available when a Mantel-Haenszel analysis is requested:
nsets= number of match groups to be used in the Mantel-Haenszel calculations. A default value of 6 is used if nsets is not specified

minfreq= minimum number of cases to be allocated to each match group

cutoffs= a set of raw score cutoffs for defining groups. They are listed as a set of comma-delimited integers

method= if cutoffs are not specified, three methods for assigning cases to match groups are available

  eqpercent* equal proportions of students are allocated to each of the required number of match groups

  eqinterval the required number of match groups is created with uniformly separated cutoffs

  thin match groups that correspond to one raw score point each are used

Redirection

>> outfile the output is written to outfile. If redirection is omitted, output will appear on the screen

Examples

compare item_est group=m>9,f>9;scale=odd;form=plot,table,diffmap,mh

compares the item parameter estimates for those items on the scale odd that are derived from the m>9 group with the estimates on the same item derived from the f>9 group. The results of the comparison are presented in the four forms plot, table, diffmap and mh. In determining match groups for the Mantel-Haenszel test, the defaults of nsets=6 and method=eqpercent are used

compare case_est group=m>9;sc=odd,even;form=table,plot,diffmap

compares the case parameter estimates for those cases in the group m>9 that are derived from the scale odd with those for the same cases that are derived from the scale even. The results of the comparison are presented in the three forms table, plot and diffmap
INDEX OF COMMANDS

`compare !group=male,female;scale=space;form=d,mh,t;order=chisq,cut-offs=6,10,13,16>>compare.out`

compares the item parameter estimates for those items on the scale space that are derived from the male group with the estimates on the same item derived from the female group. The results of the comparison are presented in the three forms diffmap, mh and table. (Abbreviated forms of these form requests have been used.) Each of the reports will be presented in chi-square order. In determining match groups for the Mantel-Haenszel test, the cutoff command will result in 5 match groups with the following raw score ranges; 0-6, 7-10, 11-13, 14-16, 17 and above. Output is directed to the file compare.out.

`compare !group=male,female;scale=space;from=mh,t,nset=10;order=mhal>>compare.out`

performs a Mantel-Haenszel DIF test for the items on the space scale. The DIF is tested between males and females using 10 match groups with `method=eqpercent` as default. The results are ordered by the Mantel-Haenszel alpha statistics and appended to the file compare.out.
correlate

calculates product moment correlations between the scores on subscales, between identification variables, and between the scores on subscales and identification variables

**Argument**

`list`  
a list of subscales and/or identification variables separated by commas

**Option**

`group=`  
specifies the subgroup or subgroups to be used. Multiple subgroups are separated by commas. If the `group=` option is omitted, then all cases are used

**Redirection**

`>> outfile`  
the output is written to `outfile`. If redirection is omitted, output will appear on the screen

**Examples**

```
scale 1-10 ! space
scale 11-20 ! number
score
correlate space,number,all >> cor.out
```

calculates the product moment correlation between the scores on the subscales `space`, `number`, and `all` (which is the total test) and writes the results to the file `cor.out`

```
scale 1,3,5,7,9 ! odds
scale 2,4,6,8,10 ! evens
score
correlate odds,evens
```

calculates the product moment correlation between the scores on the odd-numbered items and the scores on the even-numbered items. This results in a split-half reliability index
INDEX OF COMMANDS

format name 1-5 hmwk 6-8 year 9 items 10-59
scale 1-20 !space
group (year=5) !Year5
score
correlate hmwk, space, all !group=Year5

calculates the product moment correlation between the identification
variable hmwk and the scores on the space and all subscales for the Year
5 student subgroup

scale 1-40 ! formA
scale 41-80 ! formB
score
correlate formA, formB

calculates the product moment correlation between the scores on forms A
and B. This results in a parallel forms reliability index.

Notes and Warnings

1 Correlations can be computed only after the score, estimate, or
   itanal command has scored the data.

2 Pairwise deletion of missing data is used.

3 When correlating identification variables, you must ensure that
   they are in an appropriate form. For example, the correlation
   between a sex indicator coded F and M and any other variable will
   be zero, whereas if sex were coded as 0 and 1, then the required
   point-biserial correlation would be computed.

4 If a Rasch estimation has been performed, then correlations will be
   reported for both the raw test scores and the ability estimates.
data_file

indicates the data file to be used in the analysis

Argument

datafile \quad \text{read the raw data from the file } \text{datafile}

Redirection

<< datafile \quad \text{read the raw data from the file } \text{datafile}

Examples

data_file << myData.dat

indicates that the raw data is to be read from the file myData.dat

data_file myData.dat

indicates that the raw data is to be read from the file myData.dat

Notes and Warnings

1. As an alternative to using the \textbf{data_file} command, you can specify the data file by redirection with either the \textbf{score} or the \textbf{estimate} command.

2. After a data file has been indicated, it remains as the specified data file until it is superseded by the definition of another data file through the \textbf{data_file}, \textbf{score}, or \textbf{estimate} command.
delete

turns item and case deletion on or off

*Arguments*

*on*  
is the default argument. It indicates that some items or cases will be deleted

*off*  
is used when some or all previously deleted items or cases are to be reincluded for estimation

*Options*

*items*  
delete or reininclude items

*cases*  
delete or reininclude cases

*Redirection*

<< *inile*  
either reads the items or cases to be deleted from *inile* or reads the items or cases to be reincluded from *inile*. If redirection is omitted, Quest will prompt you for input

*Examples*

*delete*

item deletion is switched on, and you are prompted for the items to delete

*delete ! cases << delete.lis*

case deletion is switched on, and the cases to be deleted are read from the file delete.lis

*delete off! items << delete.lis*

item deletion is switched off for the items listed in the file delete.lis

*Notes and Warnings*

1 Case and item delete files consist of one line per case or item. Cases are identified by their sequence number in the data set, and items are
identified by their item number, which is defined by the order in which the items are specified in the format statement (see note 2 of the format command).

2. When items or cases are deleted, they are deleted from all subgroups and subscales.

3. To delete items from an individual subscale, use the `scale` command to redefine the subscale with the item omitted.
INDEX OF COMMANDS

estimate

starts a Rasch analysis

Arguments

credit* use the partial credit model

rate use the rating scale model. This model can be used only when every item has the same number of response categories

Options

iter= an integer value that indicates when Quest should pause to give you the opportunity to terminate estimation manually. The default is 5

scale= a list of subscales to estimate. Multiple subscales are separated by commas. If the scale= option is omitted, all items are estimated

group= a list of subgroups to estimate. Multiple subgroups are separated by commas. If the group= option is omitted, all cases are estimated

conv= the convergence criteria for the estimation loop. The default value is 0.005

The estimate command now has the additional option centre=.

centre= indicates whether the model is to be identified by centring items or cases

items* set the mean of the item difficulty estimates to zero

cases set the mean of the case estimates to zero

If any items or cases are anchored, the centre= option will be ignored.
Redirection
<< datafile
the data file can be specified as an input to the estimate
command. If redirection is omitted, then the data in the
most recently specified data file will be used or, if no data
file has yet been specified, Quest will display an error
message.

Examples

estimate
starts a partial credit analysis of the currently specified data file

estimate rate ! iter=10<<myData.dat
starts a rating scale analysis of the file myData.dat, pausing every 10
iterations to prompt for termination.

estimate ! scale=space group=males
starts a partial credit analysis of the space subscale for the males in the
currently specified data file.

Notes and Warnings

1 When estimation is requested, any unscored data is automatically
scored, a model is estimated, and fit statistics are computed.

2 Estimation proceeds until the largest change in any parameter
estimate is less than 0.005 or the value you have specified with the
conv= option. The iter= option can also be used to get estimation to
pause and let you stop the estimation.

3 If iteration is to proceed to termination without a pause, then set the
iter= option to a very large number, such as 1000. A large iter= value
is useful for running unattended batch jobs or a stack of jobs.

4 The estimate command automatically activates the score com-
mand.
INDEX OF COMMANDS

format

specifies the format of the data file

Argument

uname colnos  the format information is specified by indicating variable
names and their associated columns in the data file. One
variable that must be entered is items, which is used to
refer to the items that will be used in the analysis. In
addition to the items variable, five other variables can be
specified. These are typically identification variables for
the cases and can be used in subgroup definitions and
output labelling. Column specifications can be made by
using a column range, e.g. 10-18, or by using a Fortran-
type format, e.g. (t20,2(a1,1x)).

Examples

format items 1-10

ten items are to be analysed, they are in columns 1 to 10 of the data file, and
no case identification information is provided

format name 1-10 items 11-20

ten items are to be analysed, they are in columns 11 to 20 of the data file,
and a case name is to be read from columns 1 to 10

format name 1-10 year 11 sex 12 items 21-30 / items 1-10

twenty items are to be analysed. There are two lines for each case, the first
ten items are in columns 21 to 30 of line one, and the next ten are in
columns 1 to 10 of line two. Case identification is provided through name,
year, and sex information in columns 1 to 10, 11, and 12 respectively of line
one

format name (t1,a10) year (t11,a1) sex (t12,a1) items (t21,10a1) / -
items (t11,10a1)

as above, except the longer Fortran-type formats require two command lines
format items (t1,10(a1,1x)) name 30-40

ten items are to be analysed. The items are in columns 1,3,5,7,9,11,13,15,17, and 19. A name field is to be read from columns 30 to 40.

Notes and Warnings

1 Item input is restricted to one column per item.

2 It is sometimes necessary to identify items by their item number (e.g., in deletion or anchoring). The item number is defined by the order in which the item is specified in the format statement. For example:
   format items 1-10 (here, item 1 is the item in the first column of the data file, item 2 is the item in the second column of the data file, etc.);
   format items 10 items 1-9 (here, item 1 is the item in the tenth column of the data file, item 2 is the item in the first column of the data file, etc.).

3 Fortran-type formats and column ranges can be mixed in the same format command statement.

4 Fortran-type formats must begin with br, where n indicates the starting column. Fortran-type formats cannot include a slash (/) within the format itself, because the slash character is used to indicate that the data file has multiple lines for each case.

5 If items are entered using a Fortran-type format, then items must be specified using a1. If other identification variables are indicated with Fortran-type formats, then the columns must be specified with an an where n is an integer indicating the variable's width.

6 Any data assigned to the variable name is used by Quest in the Name or Candidate field provided in some output formats (see, for example, the first column of Figure 9 or the heading of Figure 13 in the Sample Sessions section). If name is one of the specified variables, the first four characters of name will be used if you request alphabetical sorting of cases in the show command. If name is omitted, the first identification variable specified is used for alphabetical sorting.
group

specifies cases to be included in a subgroup

Argument

(logic exp) a logical expression that specifies the criteria for selection of a subgroup by stating values for variable names defined in the format command. The expression can be compounded or nested using the logical operators =, >, <, <=, >=, <=, .and., and .or.

Option

groupname a name that is assigned to the subgroup that satisfies the condition in the argument and that is used for subsequent references to the subgroup. The default names are group01, group02, ... group05

Example

format sex 2 year 3 age 5 school 6-7 items 10-29
group (sex=m) ! males
group (year=5).and.(sex=f) !year5_girls
group (age=7).or.(age=8).and.(year=3) !young_year3
group (age=9).and.(year=3) !old_year3
!young_year3
group (school=' 6')!school6
!young_year3
!old_year3

Notes and Warnings

1 Groupname can be up to 20 characters long. It cannot contain spaces.
2 The group command cannot be invoked until a data file and a format have been specified.
3 Values for variables that contain spaces must be enclosed between apostrophes.
The values in the logical expressions are case sensitive. For example, `group (sex=m) ! males` and `group (sex=M) ! males` would identify two different groups of cases.

Cases can appear in more than one subgroup.

The subgroup name `all` is reserved for the complete data set, and this subgroup is automatically created by Quest.

A maximum of five user-defined subgroups can be specified.

Parentheses are used in a logical expression to override a default left to right evaluation. For example, `group (age=6 .and. sex=f .or. sex=F) ! f6` would produce a group that contained all cases with F in the sex field along with all cases with 6 in the age field and f in the sex field; whereas `group (age=6 .and. (sex=f or. sex=F)) ! f6` would define a group of cases with 6 in the age field and f or F in the sex field.
header

specifies a heading that will be printed on all outputs. Title is an alias for this command.

Argument

string  any string of up to sixty characters, including spaces, can be entered here as a title for subsequent analyses

Example

header YOU TYPE YOUR OWN HEADER HERE
iscore

specifies the maximum score that is to be given to a test item

**Arguments**

*off* makes the item score equal to the default value (see the Technical Notes section)

*on* the default argument. It indicates that some items will have scores set by the user

**Redirection**

*<< infile* either reads the items and their maximum scores from *infile* or reads the items whose scores are to be returned to the default value from *infile*. If redirection is omitted, Quest will prompt you for input

**Examples**

`iscore`

you are prompted for item numbers and scores

`iscore << scores.lis`

item numbers and their scores are read from the file *scores.lis*

`iscore off << scores.lis`

item scores for the items in the file *scores.lis* are returned to the default values

**Notes and Warnings**

1. Score files consist of one line per item. Each line contains an item number followed by a maximum score. Items are identified via their item number, which is defined by the order in which the items are specified in the `format` command (see note 2 of the `format` command). The same file can be used with the on or off argument.
INDEX OF COMMANDS

2. The default score for an item is one less than the number of available response categories. For example, a dichotomous item has a score of 1, and a five-category item has a score of 4.

3. See the Technical Notes section for a more complete discussion of item scoring.
performs a response alternative analysis

*Arguments*

- `raw*` performs the analysis on the raw response categories; that is, before scoring and recoding
- `scored` performs the analysis on scored data

*Options*

- `scale=` a list of subscales to analyse. Multiple subscales are separated by commas. If the `scale=` option is omitted, all items are analysed
- `group=` a list of subgroups to analyse. Multiple groups are separated by commas. If the `group=` option is omitted, all cases are analysed
- `form=` specifies the contents of the analysis
  - `long*` provides complete information
  - `short` for multiple choice tests only; provides information in a summary form
  - `export` a tabular display of the results from `form=long`, printed in a form that makes them suitable for easy interface with other software. The table produced by the export form contains the following information recorded on four records (lines) for each item if a Rasch model has been estimated and recorded on two records (lines) for each item if a Rasch model has not been estimated
INDEX OF COMMANDS

Line 1: item number and name, number of cases responding to each response category

Line 2: item number and name, item discrimination, point biserial for each response category

Line 3: item number and name, the mean ability of the cases in each response category

Line 4: item number and name, the item difficulty and its standard error, the step difficulties and their standard errors, infit and outfit mean squares and the infit and outfit statistics

delimiter= when export files are produced (form=export), a character can be specified to separate fields in the export file. This facilitates reading the file into other software:

    space*      fields are separated by a space
    tab         fields are separated by a tab
    comma       fields are separated by a comma

Redirection

>> outfile   the output is written to outfile. If redirection is omitted, output will appear on the screen

Examples

itanal

performs a response alternative analysis on raw data for all items and cases and displays output on the screen

itanal scored >> itanal.out

performs a response alternative analysis on the scored data and writes output to the file itanal.out
`itatnl!group=boys;scale=literacy`

performs a response alternative analysis on raw data for the subgroup boys on the subscale literacy and displays output on the screen.

*Notes and Warnings*

1. The `itatnl` command automatically activates the `score` command.
item_names

used to give names to the items

**Arguments**

* on* the default argument. It indicates that names are to be provided for some or all items

* off* resets the names of the specified items to their default values (item1, item2, etc.)

**Redirection**

* $<$ < infile* either reads the item numbers and their names from infile or reads the item numbers that are to be reset to their default values from infile. If redirection is omitted, Quest will prompt you for input

**Examples**

* item_names you are prompted for item numbers and names

* item_names $<$ < names.lis* item numbers and their names are read from the file names.lis

* item_names off $<$ < names.lis* item names are reset to the default values for the items in the file name.lis

**Notes and Warnings**

1. Item name files consist of one line per item. Each line contains an item number followed by the name given to the item. The same file can be used with the on or off argument.

2. Item names can be up to 20 characters long, but they will be truncated to a shorter length for some outputs. They can contain spaces.

3. If you request alphabetical order in the show command, it will be based on the first four characters of the item name.
key

specifies scoring keys

_argument_

_string_ a string of characters with the scoring key of the ith item as the ith character

_option_

_score_ an integer score that corresponds to the given keys. A default score of 1 is used if the _score_ option is omitted

_examples_

_key_abbabdebcdbec_ the items that match the keys are given a score of 1

_key_abbabdebcdbec!score=2_ the items that match the keys are given a score of 2

_notes and warnings_

1. Up to ten different sets of scoring keys can be provided, each with its own score; and multiple sets can be used in a single analysis.
2. More than one key can be mapped to the same score.
3. The length of the key is not checked against the number of items indicated in the _format_ command.
kidsmap

produces kidmaps for individual cases

*Arguments*

**all**  produce kidmaps for all of the cases in the specified subgroup(s)

**caselist**  produce a kidmap for each case in the list or range specified by caselist

*Options*

**scale**  a list of subscales for which to produce kidmaps. Multiple subscales are separated by commas. If the scale option is omitted, all items are used

**group**  a list of subgroups for which to produce kidmaps. Multiple subgroups are separated by commas. If the group option is omitted, all cases are used.

*Redirection*

**>> outfile**  the output is written to outfile. If redirection is omitted, output will appear on the screen

*Examples*

**kidsmap**

produces a kidmap for each case, using all items and writing results to the screen

**kidsmap 1-10 >> kidmap.out**

produces kidmaps for the first ten cases in the file, using all items and writing results to the file kidmap.out
kidmap !group=boys;scale=literacy

produces a kidmap for each case in the subgroup boys, using the subscale literacy. Output is written to the screen.

Notes and Warnings

1. Kidmaps cannot be produced until after estimation of the requested subscale and subgroup combination.
logit_table

produces tables of equivalences between raw scores and Rasch estimates. The Rasch estimates can be reported in the usual logit metric or transformed onto scale scores with a specified scaling factor and origin.

Options

scale= a list of subscales for which the display is required. Multiple subscales are separated by commas. If the scale= option is omitted, all items are used.

group= a list of subgroups for which the display is required. Multiple subgroups are separated by commas. If the group= option is omitted, all cases are used.

unit= specifies the scaling factor for the transformation of the logit estimates to scale scores (the default is unit=1).

origin= specifies the logit value that is to be transformed to zero (the default is origin=0).

Redirection

>>outfile the output is written to outfile. If redirection is omitted, output will appear on the screen.

Example

logit_table !scale=numeracy;unit=6;origin=3

produces tables of equivalences between raw scores and Rasch estimates on the numeracy subscale, and transforms the logit values to scale scores with a unit of 6 and an origin of 3.
recode

allows data to be recoded

Argument

(\textit{old1 old2 old3 ...}) \textit{(new1 new2 new3 ...)}

the argument consists of two code lists. Codes in the first list are mapped to their corresponding element in the second list.

Options

\textit{all*} \hspace{1cm} \textit{all items are recoded}

\textit{itemlist} \hspace{1cm} \textit{the items in the list or range are recoded}

Examples

recode (0 1 2) (2 1 0)

recodes a set of data so that the 0 becomes 2 and 2 becomes 0 for all items.

recode (02) (20)

the same as above.

recode (\textit{abc}) (012)1,3,5,9

recodes \textit{a} to 0, \textit{b} to 1, and \textit{c} to 2 for items 1, 3, 5, and 9 only.

Notes and Warnings

1. Recode statements must be given after codes statements.
2. Multiple recode statements can be used. In data scoring, they are applied in the order in which they are entered.
reset

returns all program settings to their default values

Notes and Warnings

1. When you issue a reset command the following occurs:
   - codes are set to 0 and 1
   - header is set to Quest: The Interactive Test Analysis System
   - page length is set to 65 lines
   - page width is set to 80 columns
   - screen length is set to 24 lines for IBM PCs or compatibles and to 39 lines for Apple Macintosh
   - screen width is set to 78 columns for IBM PCs or compatibles and to 110 columns for Apple Macintosh
   - the logging of output is turned off
   - the working directory (the one in which Quest automatically looks for input and output files) is set to the directory from which Quest was executed (launched)
   - all user-defined item names are erased
   - all cases and items are undeleted
   - all cases and items are unanchored
   - all item scores are deleted
   - all subgroup and subscale definitions are cleared
   - the data format is cleared
   - the data file name is cleared

2. You can stack jobs to run unattended by placing a reset command between jobs in a submit file. See also note 3 of the estimate command.
scale

specifies items to be included in a subscale

**Argument**

*itemlist*: a list or range of items to be included in the subscale named in the *scalename* option

**Option**

*scalename*: a name that is assigned to the subscale and used for subsequent references to it. The default names are scale01, scale02, ..., scale05

**Example**

```
scale 1,2-5,9,10!space
```

creates a subscale named space that comprises items 1, 2, 3, 4, 5, 9, and 10

**Notes and Warnings**

1. Items can appear in more than one subscale.
2. The subscale name all is reserved for the complete data set, and this subscale is automatically created by Quest.
3. A maximum of five user-defined subscales can be specified.
4. Scalename can be up to 20 characters in length. It cannot contain spaces.
score

scores the data

Redirection

<< datafile reads and scores the data in datafile. If redirection is omitted, then the data in the most recently specified data file will be scored or, if no data file has yet been specified, Quest will display an error message.

Examples

score

scores the data in the most recently specified data file

score << mydata

scores the data in the file mydata

Notes and Warnings

1. When a score command is issued, all currently unscored combinations of subgroups and subscales are scored.

2. Scoring is activated automatically by the itanal and estimate commands. Score is a command that is rarely needed.
INDEX OF COMMANDS

set

sets some system settings

Arguments

width= sets the width in columns of any output

length= sets the length in lines of any output

directory= sets the path for the working directory for input and output files

probability= sets the probability level to be used to obtain estimates of item thresholds. The default is 0.5

tolerance= an option for the fit maps. Defines the critical regions for the upper and lower bounds for item and case fit maps

logon begin writing all screen messages and commands to a log file

logoff turn logging off

Options

page length and width options are being set for output to a file

screen length and width options are being set for output to the screen

Redirection

>> outfile output is written to outfile. This is used to indicate a log file name

Examples

set width=132!page

sets the width for output redirected to a file to 132 columns
set length=25!screen

sets the length for output displayed on the screen to 25 lines

set p=.7

sets the probability level at 0.7

show param_type|form=map;stat=fit;tol=n

produces a fit map with critical bounds of 1/(1+n), 1+n

set logon >>quest.log

creates a log file named quest.log to which all subsequent Quest screen messages and command statements will be written

set directory=\quest\mydata

tells Quest to look for input files in and write output files to the MYDATA subdirectory of the QUEST subdirectory. With PC/MS-DOS versions of Quest, the default drive name (the drive on which Quest is installed) cannot be changed

set directory=Hard disk:Quest:My data

tells Quest to look for input files in and write output files to the My data folder in the Quest folder on the disk drive named Hard disk. With Apple Macintosh versions of Quest, you can change the default drive name, as well as the default folder name

Notes and Warnings:

1. The maximum width for any output is 132 columns. There is no limit on output length.

2. A log file remains active until (a) logging is turned off by a set logoff command, (b) a new log file is specified by a set logon command, (c) a reset command is issued, or (d) a Quest session is terminated with a bye, end, exit, quit, or stop command.

3. The value entered for the tolerance must be between 0 and 1.
INDEX OF COMMANDS

show

produces a range of displays

Arguments

table* produces one or more of a set of predefined tables if the
table= option is used

map produces one or more of a set of predefined maps when
used with the map= option

items produces a display of item-based statistics

cases produces a display of case-based statistics

settings produces a display of the current program settings

counts produces a display of the counts of each response type for
each item

Options

scale= a list of subscales for which the display is required. Multi-
ple subscales are separated by commas. If the scale=
option is omitted, all items are used

group= a list of subgroups for which the display is required.
Multiple subgroups are separated by commas. If the group=
option is omitted, all cases are used

form= displays the results in one or more of the following forms:
table* a tabular display of the results

map a graphical display of the results

summary a summary of key results

anchor a format compatible with the requirements
of the anchor command
export  a tabular display of results in a format that makes them suitable for easy interface with other software

order= Many of the results can be displayed in a variety of orders. A set of different orders can be requested in the same command by separating them using commas (e.g., order=name,fit):

entry* items are displayed in item number order, and cases are displayed in sequence order

name items or cases are sorted alphabetically according to the first four characters of their name. Item names are defined using the item_names command, and casenames are defined by the identification variable name in the format command. If the name variable is not specified, then the first identification variable in the format command is used. If no identification variable is specified in the format command, then the cases are displayed in sequence order

estimate items or cases are sorted according to their difficulty or ability estimates

fit items or cases are sorted according to their infit mean square values

stat= a variety of output statistics is available:

threshold* the item parameter estimates reported are Thurstonian-type thresholds

delta the item parameter estimates reported are Masters-type deltas
tau
the item parameter estimates reported are
Andrich-type location and threshold
parameters

fit
used in combination with the form=map option to pro-
duce graphical displays of item and/or case fit. Fit statis-
tics are supplied in every table, so this option is only useful
for requesting fit maps for items and cases

delimiter=
when export files are being produced (form=export), a
character can be specified to separate fields in the export
file. This facilitates reading the file into other software:
  space* fields are separated by a space
  tab    fields are separated by a tab
  comma  fields are separated by a comma

table=
followed by the number of one or more of the predefined
tables. The table numbers are given under Notes and
Warnings below

map=
followed by the number of one or more of the predefined
maps. The map numbers are given under Notes and
Warnings below

min=
the lowest logit value to be displayed in a variable map

max=
the highest logit value to be displayed in a variable map

length=
the length (number of lines) for a variable map (de-
default=110)

Redirection

>> outfile
the output is written to outfile. If redirection is omitted,
output will appear on the screen
Examples

show

produces tables 1 and 2, maps 1 and 2, and the settings table for all cases
and all items (see the numbered list of tables and maps under Notes and
Warnings below)

show !table=3;order=fit

produces table 3 in fit order for all cases and all items and writes them on
the screen (see the numbered list of tables and maps under Notes and
Warnings below)

show cases order=estimate;scale=numeracy >> case.out

writes a table of case parameter estimates on the numeracy subscale,
sorted by estimate, to the file case.out

show item !form=export;stat=delta;delimiter=tab >> item.out

writes an export file of item delta parameter estimates to the file item.out
with fields delimited by tab characters

show items !form=map;min=-1,max=4; length=120

produces a variable map 120 lines long but displays only those cases and
items that lie between -1 and +4 logits

Notes and Warnings

The following tables and maps are predefined and can be requested
by using the show command without an argument and with the
\texttt{table=} or \texttt{map=} options. The \texttt{order=} , \texttt{scale=} , \texttt{group=} , and \texttt{stat=}
options can be used in conjunction with predefined tables and maps

<table>
<thead>
<tr>
<th>Tables</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>item summary table</td>
</tr>
<tr>
<td>2</td>
<td>case summary table</td>
</tr>
<tr>
<td>3</td>
<td>table of item statistics</td>
</tr>
<tr>
<td>4</td>
<td>table of case statistics</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Maps</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>variable map (a map of items with histogram showing case distribution)</td>
</tr>
<tr>
<td>2</td>
<td>item fit map</td>
</tr>
<tr>
<td>3</td>
<td>case fit map</td>
</tr>
</tbody>
</table>
submit

submits a file of Quest commands to be processed as a batch before returning control to the user.

**Argument**

*inile*  
execute the Quest commands contained in *inile*

**Redirection**

<<*inile*  
execute the Quest commands contained in *inile*

**Examples**

`submit << myJob.lis`

executes the commands in the file *myJob.lis*

`submit myJob.lis`

executes the commands in the file *myJob.lis*

**Notes and Warnings**

1. Submit files consist of one line per Quest command line. A command line cannot exceed 70 characters, and a command statement cannot exceed 360 characters. The calculation of the command line or command statement length includes spaces but excludes continuation characters and comments.

2. Submit statements cannot be nested; in other words, a submit file cannot contain a submit statement.

3. Multiple submit files can be entered sequentially.

4. You can intersperse submit commands with manually entered commands during a Quest session.
When you enter a `compare`, `kidmap`, `itanal`, or `show` command without redirection, Quest displays the output on the screen. The Display Manager allows you to scroll up, down, left, and right through these output displays. When a single command produces a number of different displays, Display Manager also allows you to scroll through those displays one at a time.

You control the Display Manager by entering the appropriate command at the Display Manager prompt (Enter Display Command->):

- `down n` scroll down $n$ lines. If $n$ is omitted, the default is one screen
- `up n` scroll up $n$ lines. If $n$ is omitted, the default is one screen
- `right n` scroll right $n$ lines. If $n$ is omitted, the default is one screen
- `left n` scroll left $n$ lines. If $n$ is omitted, the default is one screen
- `more` scroll down one page
- `half` scroll down half a page
- `end` go to the end of the display
- `top` go to the top of the display
- `next` proceed to the next display
- `save filename` save the current display to a file
- `quit` exit from the Display Manager. The Quest > prompt will be displayed

Each of the Display Manager commands can be abbreviated to its shortest unambiguous root.
The width and length of the Display Manager display are specified with the set command, using the `length=` and `width=` arguments with the `screen` option.

If you enter a `save` command, Quest writes the display to the specified output file using the length and width settings that have been specified for the screen rather than the settings specified for output to a file. To ensure the correct layout of a table or map in an output file, use one of the Quest commands that allows redirection.
This section provides basic information on the models and statistical methods implemented by Quest. The details of the references in this section are given in the Bibliography section for those who wish to follow up on any of the methods.

**THE QUEST MODEL**

The central element of Quest is a Rasch IRT model that can be used with ordered category item response data. If the response of individual \( n \) to item \( i \) is indicated by the item score \( X_{ni} \), which can take any of the integer values 0, 1, ..., \( m \), then the Quest model describes the probability of observing a specific score \( x_n \) as

\[
P(X_{ni} = x_n) = \frac{\exp \sum_{j=0}^{x_n} w_{ij}(\beta_n - \delta_i - \tau_{ij})}{\sum_{k=0}^{m} \exp \sum_{j=0}^{k} w_{ij}(\beta_n - \delta_i - \tau_{ij})},
\]

where \( \beta_n \) is the ability of individual \( n \), \( w_{ij} \) is the score assigned to step \( j \) in item \( i \), and \( \delta_i \) and \( \tau_{ij} \) characterise the difficulty of item \( i \). So that (1) can be written as a single expression, we define

\[
\exp \sum_{j=0}^{0} w_{ij}(\beta_n - \delta_i - \tau_{ij}) = 1;
\]

and to ensure identification, we apply the constraints

\[
\sum_{j=0}^{m} \tau_{ij} = 0 \text{ and } \sum_{i=1}^{I} \delta_i = 0.
\]
As such the model is a slightly generalised form of Masters' (1982) partial credit model (see Wright and Masters, 1982). The generalisation is through the introduction of the step scores \( w_i \). The step scores were introduced so that Quest could estimate data that included null categories (see Wilson and Masters, in press) but they can also be used as item scores or item discrimination indices (see Andersen, 1983).

Two important special cases of (1) that are included in Quest are Andrich's (1978) rating scale model and the one parameter logistic model (Rasch, 1980; Wright and Stone, 1979).

If model (1) is applied to dichotomous (or binary) data, then it reduces to

\[
P(X_i = x_i) = \frac{\exp(x_iw_i(\beta_i - \delta_i))}{1 + \exp(x_iw_i(\beta_i - \delta_i))},
\]

which is the usual Rasch (or one parameter logistic) model.

If model (1) is applied to ordered category data and the \( \tau \) parameters are constrained so that \( \tau_i = \tau_i = \tau_i = ... \), then (1) becomes the Andrich rating scale model.

**ITEM SCORING**

Under most conditions, the \( w_i \) values are set to one for all \( i \) and \( j \). This is the Quest default, and it is the standard practice in Rasch measurement. It is possible, however, to have Quest estimate a model with other specified values of \( w_i \) by using the iscore command. The item scores that are entered with the iscore command correspond to

\[
w_i = \sum w_{ij}.
\]
Individual $w_{ij}$ are then generated by Quest by distributing $w_i$ uniformly over the number of possible steps for item and adjusted following Wilson and Masters (in press) to allow for null categories.

The $w_{ij}$ describe the discrimination of items; they can be fixed but not estimated by Quest.

**ESTIMATION**

Quest uses the joint (or UCON) maximum likelihood procedure to estimate both the item and the case parameters. It has been shown that UCON provides biased estimates of parameter estimates (Andersen, 1973; Wright and Panchapakesan, 1969), but the majority of the bias is removed by the correction factor $1-1/L$, which is applied by Quest after the UCON procedure has converged.

When all of the parameters are free, the estimated model is identified with the constraints given above. If any item or case is anchored (see the anchor command), then the anchor value serves as identification constraint.

**FIT INDICES**

The item and case infit and outfit statistics reported by Quest are the weighted and unweighted residual-based statistics described by Wright and Masters (1982) and Wright (1984). Both fit statistics are mean squares that can be approximately normalised by applying the Wilson-Hilferty transformation. The normalised versions of the statistics are referred to as the infit $t$ and the outfit $t$. When the data conform to the model, the $t$-values have a mean near zero and a standard deviation near one.

As with most confirmatory model fitting, the tests of fit provided by Quest are sensitive to sample size—particularly in their normalised form. In practice, we have found that using the mean square fit statistics as effect measures can be a useful way of considering the compatibility of the
model and the data. A fit mean square of $(1+x)$ indicates $(100x)$ per cent more variation between the observed and model-predicted response patterns than would be expected if the data and the model were compatible. Similarly, a fit mean square of $(1-x)$ indicates less variation between the observed and model-predicted response patterns than would be expected if the data and the model were compatible.

As described by Wright (1984), the outfit statistic is sensitive to outlying observations and can sometimes be distorted by a small number of unusual observations. The infit statistic is more robust than the outfit, and it is closely related to item or case discrimination. Under most circumstances, inﬁt and outfit values will be similar.

**EXPRESSIONS OF ITEM DIFFICULTY**

Quest provides three forms of item difficulty estimates that are called deltas, taus, and thresholds.

*Deltas* are the item category delta parameters described by Masters (1982). One delta is provided for each step, and it describes the conditional probability of succeeding on a step provided the previous step has been passed. In terms of the parameters in model (1), the category delta for step *j* in item *i* is $\delta_i + \tau_{ij}$. *Deltas need not be ordered.*

The *tau* parameterisation is the one used in the Quest estimation routine, and it is shown in model (2). Each item is characterised by an item difficulty $\delta_i$ and a set of step parameters $\tau_{ij}$. *Taus need not be ordered.*

The *threshold* is the default representation of item difficulty used by Quest. The Quest thresholds (introduced by Masters, 1988) are analogous to Thurstonian thresholds. A threshold for an item step is the ability level that is required for an individual to have a 50 per cent chance of passing that step. *Item thresholds must be ordered.*
RELIABILITY INDICES

Quest can provide a number of reliability estimates, using both IRT and traditional methods. The item and case summary tables provide the item and person separation reliability indices as defined in Wright and Masters (1982).

Quest's `itanal` command reports the Cronbach alpha index of internal consistency, which is equivalent to KR-20 when applied to binary data.

Quest's `correlate` command can be used to produce split-half and parallel form reliability indices, along with concurrent and predictive validity indices.

DISCRIMINATION INDICES AND DISTRACTER ANALYSIS

Quest's `itanal` command produces the correlation between item score and overall test score for each item; the correlation used is the point-biserial (see Ferguson, 1959).

The item and case infit statistics that are produced by Quest can also be used as indices of discrimination.

As an extension of the notion of discrimination, the `itanal` command produces point-biserial correlations for each observed item response or for each possible item score. The correlation reported for each response is the point-biserial correlation between the binary variable that indicates if a case responded with a particular distractor or not and the total test score.

CORRELATION

The Quest `correlate` command calculates product moment correlations using a pairwise approach to the deletion of missing data. Correlations based upon raw scores are always produced, and correlations based upon Rasch estimates will be computed if the data has been estimated. The count of observations used for the raw score-based and Rasch-based correlations may differ if cases with perfect or zero scores are encountered.
BIBLIOGRAPHY


BIBLIOGRAPHY

