POLYST

A Computer Program for Polytomous IRT Scale Transformation

Version 1.0

May 12, 2003

Seonghoon Kim and Michael J. Kolen

The University of Iowa

POLYST is an ANSI C computer program for implementing four alternative item response theory (IRT) scale transformation methods to place item (and, if necessary, proficiency) parameter estimates from a group of examinees (new group) on a scale determined by a base or reference group of examinees. The four scale transformation methods are the mean/sigma (Marco, 1977), mean/mean (Loyd & Hoover, 1980), Haebara (Haebara, 1980) and Stocking-Lord (Stocking & Lord, 1983) methods. These methods were originally developed for dichotomous IRT models and have been extended to polytomous IRT models. A transformation of IRT scales is often needed under certain test administration conditions such as when the common-item nonequivalent groups design (Kolen & Brennan, 2004) is used. In the common-item nonequivalent groups design, two forms of a test, new and old, are administered independently to two groups of examinees, which are usually nonequivalent in proficiency. If items for the two forms are calibrated separately by group using the same scaling convention (e.g., 0 mean and 1 standard deviation metric), two IRT scales from separate calibrations are group dependent and thus do not have the same origin and unit for measuring proficiency. The two scales are to be linearly related because of the invariance property of IRT modeling. The four scale transformation methods estimate the slope (S) and intercept (I) of a linear transformation relating the two IRT scales resulting from separate calibrations.

POLYST has a capability of handling IRT scale transformation under the following five IRT models: 1) three-parameter logistic (3PL) model, 2) graded response (GR) model, 3) generalized partial credit (GPC) model, 4) nominal response (NR) model, and 5) multiple-choice (MC) model. For the first three IRT models, all of the four scale transformation methods can be implemented. For the NR and MC models, the three scale transformation methods other than the Stocking-Lord method can be implemented,

although the Stocking-Lord method may be used as long as test characteristic curves are defined for the two models. All formulas to implement the four methods are based on Kim and Kolen (2004). For detailed information about the formulas, criterion functions (or, loss functions), and other numerical derivations employed by POLYST, refer to Kim and Kolen (2004) or to the authors.

Several variations of the four scale transformation methods can be, and is, employed in practical applications. For example, for the Haebara and Stocking-Lord methods, different symmetry-related schemes (symmetric versus non-symmetric function) and different summation/integration schemes over examinees have been proposed and used to define the criterion function in question. POLYST provides several options to flexibly define the criterion function in order that users can define their own criterion functions according to their purposes. Detailed explanations about how to define criterion functions with the options POLYST provides are presented in a later section.

Technically, POLYST uses routines described in Dennis and Schnabel (1996) to find the scale transformation constants (i.e., slope and intercept) that minimize the criterion functions for the two characteristic curve methods (Haebara and Stocking-Lord). As was done by Zeng and Kolen (1994), POLYST supports Gauss-Hermite quadrature so that the criterion function may be defined numerically integrating the transformation error over the proficiency continuum instead of summing over a set of discrete proficiency values. There is no limitation in the number of common items and the number of response categories for an item. All items do not need to have the same number of response categories as well.

Contents of Folder

When POLYST is distributed, the following files and folder are provided together:

1) `Readme.txt`	a "Read Me" file for POLYST
2) `POLYST_wc(v1.0).exe`	an executable file for Windows system (95 or above)
or	
`POLYST_m9(v1.0)`	an executable file for Macintosh system (OS9)
3) `POLYST_manual.pdf`	this documentation
4) `Examples`	a subfolder containing examples to illustrate how to create POLYST command files

Program Operation

Two versions of POLYST exist: one is for Windows (95 or above) and the other is for Macintosh (OS9). They are console-based programs. So, POLYST must be run using a console (command prompt window) in Windows and Macintosh. To use POLYST, users should copy all files packed onto their hard drives. How to run each of the Windows and Macintosh versions of POLYST is as follows.

Windows

The command line used to run the POLYST program contains up to two arguments:

```
POLYST_wc(v1.0) input-file [output-file]
```

The first argument is the name of input file and the second argument is optional (indicated by brackets). Input files are also referred to as command files. The input file does not need to be in the same folder in which the executable file exists. The second argument must be distinct from the first argument. If the second argument is not specified, the resulting output file name is made of concatenating the input file name and a chosen extension `out`. Users can use up to 500 characters for an input file name. How to create an input file is explained in the next section. If users run the program without arguments, a command prompt will ask them to enter two file names for input and output, as in Macintosh. Users should enter proper input- and output-file names, as the command prompt to do so.

Macintosh

When the Macintosh version of POLYST is started (by double-clicking the executable file), only a simple console (command prompt window) will be visible, without any menu bar. Then, the command prompt will ask users to enter two file names for input and output, in order. Users should enter proper input- and output-file names, as the command prompts to do so.

In the both versions of POLYST, when the program is successfully executed, a message will show up on the screen to indicate where outputs are saved.

Creating Input Files

POLYST has syntax rules for the input file. Detailed explanations for the rules are presented below. POLYST expects the input file to be a text only file. For the input file it is strongly suggested that white space between keywords or input parameters should be generated typing one or more spaces rather than using tabs. Users may be able to easily get the picture of how to create input files of POLYST through examples provided along with the program, which are in a folder called `Examples`.

POLYST Syntax Rules for Input Files

POLYST has six main keywords and ten option keywords. The six main keywords are MO, NI, NC, NE, OL and BY. The ten option keywords are SC, ST, IT, ND, OD, CW, FS, SY, LM, and OP. Some of the main keywords and option keywords have subkeywords. Three delimiters are used so that the input information may be entered without errors by users: slash '/', left parenthesis '(', and right parenthesis ')'. There is one important principle applied in preparing input files. Every piece of the input information, such as keywords, data, and delimiters, must be separated by space(s) or new line escape character(s). *Never use commas to separate between pieces of the input information!*

An input file must begin with the keyword MO and end with the keyword BY. However, users can give comments by using a C-like comment style before the keyword MO. That is, the usage of */* preceding the body of comments followed by */* permits users to comment their input files. Data after the keyword BY are also ignored by POLYST.

Below is a syntax diagram for the input file. Every keyword consists of two capital letters. Enter keywords as shown, although users can use lowercase or more than two letters. The syntax for each option keyword is provided separately in the last part of this section. After this, specific and detailed explanation for each keyword will be given. The following conventions are applied in the syntax diagram:

Bold-faced Capital letters
indicate the exact same spelling and form as shown should be used.

Italic Small letters
indicate that specific information should be supplied by users.

Vertical bar (|) separating subkeywords and options
indicates only one of the terms separated by it should be chosen.

Bracketed information ([ ])
is optional.

Braces ({ })
indicate that the inside statements should be dealt with as a group.

Three periods (...)
indicate that the same kind of data are entered.

NONE
means that nothing need to be entered.

Syntax Diagram for Input Files

```
/*
 COMMENTS
*/

MO  DR | GR | PC | NR | MC

NI number-of-common-items

NC { AL number-of-categories } |
   { BL number-of-blocks
     / ( number-of-categories number-of-items ) item-list
       ...
     / ( number-of-categories number-of-items ) item-list
     / ( number-of-categories * ) NONE }

NE [ IS | LC ] DI | { FI filename }
item-parameter-list
...
item-parameter-list

OL [ IS | LC ] DI | { FI filename }
item-parameter-list
...
item-parameter-list

[ OPTION ]
...
[ OPTION ]

BY
```

Syntax Diagram for Options

SC *scale-constant*

ST *slope intercept*

IT *number-of-iterations*

OD *n* { **GH** } |

{ **PN** *mean std multiple-of-std* } |

{ **PB** *alpha beta lower-limit upper-limit* } |

{ **RN** *mean std* } |

{ **RU** *lower-limit upper-limit* } |

{ **ED** *starting ending* **/ EQ** | { **WL** *theta-weight-list* } } |

{ **SE** { **DI** *pair-of-value-and-weight-list* } | { **FI** *filename* } }

ND *the same as* **OD**

CW { **AL (** *category-weight-list* **)** } |

{ **BL** *number-of-blocks*

/ (*number-of-cat. number-of-items* **)** *item-list* **(** *category-weight-list* **)**

. . .

/ (*number-of-cat. number-of-items* **)** *item-list* **(** *category-weight-list* **)**

/ (*number-of-cat.* * **)** NONE **(** *category-weight-list* **)** }

FS **DO** | **NO** [**DO** | **NO**]

SY **BI** | **NO** | **ON** [**BI** | **NO** | **ON**]

LM *slope intercept*

/ *number-of-searches radius tolerance* **NO** | **IN** | { **FI** *filename* }

OP

Specific explanations for keywords are presented below.

MO **DR** | **GR** | **PC** | **NR** | **MC**

The keyword MO stands for a model chosen for the scale transformation. The subkeyword DR is for the dichotomous response model based on the three-parameter logistic model, GR for Samejima's graded response model, PC for Muraki's generalized partial credit model, NR for Bock's nominal response model, and MC for Thissen & Steinberg's multiple-choice model.

NI *number-of-common-items*

The keyword NI stands for the number of common items used as anchor items. An integer should be supplied for *number-of-common-items*.

NC { **AL** *number-of-categories* } |
{ **BL** *number-of-blocks*
/ **(** *number-of-categories number-of-items* **)** *item-list*
. . .
/ **(** *number-of-categories number-of-items* **)** *item-list*
/ **(** *number-of-categories* * **)** NONE }

The keyword NC stands for the number of categories for each polytomous item. Note that the keyword NC is *not* used in DR. The subkeyword AL (all) can be used if all items have the same number of categories. An integer should be followed for *number-of-categories*. Otherwise, the subkeyword BL (block) will come in handy. The integer after BL, *number-of-blocks*, indicates the number of blocks to be used. Any number of blocks can be made as long as a set of items in each block has the same number of categories. This means that a block with the same number of categories could be divided into smaller blocks. Each block starts with a slash. The first and second integers in the parentheses are the number of categories and the number of items in the block, respectively. Especially, if the second integer is replaced by the character *, this refers to all other items except the items in the above blocks. At this time, any item list after parentheses must not be followed. To conveniently enumerate an item list having consecutive integers by an increment of 1, a special keyword `TO` can be used. For example, to enumerate an item list "1 2 3 5 6 7 8 9", `TO` can be used as follows: `1 TO 3 5 TO 9`. A reasonable combination of numbers and

TO's will facilitate an enumeration of numbers indicating items in a block.

NE [**IS** | **LC**] **DI** | { **FI** *filename* }
item-parameter-list
. . .
item-parameter-list

The keyword NE stands for a new or current form. The subkeywords IS and LC are used only for the GR and PC models. IS stands for "item-step" parameters and LC does for "location and category" parameters. In the GR and PC models, each item has $K - 1$ item-step parameters, where K indicates the number of categories of the item. However, in the Likert version of the GR and PC models, an item-step parameter b_{jk} for category k of item j can be resolved into two parameters b_j and c_{jk} as follows:

$$b_{jk} = b_j - c_{jk},$$

where b_j is called a location parameter and c_{jk} is called a category parameter. If users have item parameters in the form of "location and category" parameters, each item has the same number of item parameters as the number of its categories plus one. In this case, the subkeyword LC should be used.

Item-parameter lists for the input file can be supplied in two ways. First is to list item parameter estimates for each item *within* the input file after the keyword DI. Second is to read in from an *outside file*, which contains item parameter estimates, using the keyword FI. Whether DI or FI is used, the structure of item parameter estimates input must be the same. Item parameter estimates for each item have to be entered differently according to models as follows:

1) DR (three-parameter logistic model)

$a_j \quad b_j \quad c_j$,

where a_j stands for a slope parameter estimate, b_j for a difficulty parameter estimate, and c_j for a pseudo-guessing parameter estimate. If the two-parameter logistic model is used, values for c_j should be supplied with zeros. If the one-parameter logistic model (i.e., the Rasch model) is used, a constant should be supplied for a_j and values for c_j should be supplied with zeros. In the case of the one-parameter logistic model, two constants from separate calibrations are expected to be different and, in fact, they reflect the difference in spread between the old and new proficiency scales.

2) GR (graded response model)

With **IS**,

$a_j \;\; b_{j2} \;\; b_{j3} \;\ldots\; b_{jK}$,

where b_{j2} to b_{jK} are threshold parameters, where *K* is the number of categories, and the value of b_{jk} is the point on the proficiency scale at which the probability passes 50% that the response is in categories *k* or higher.

With **LC**,

$a_j \;\; b_j \;\; c_{j2} \;\ldots\; c_{jK}$,

where b_j is a location parameter estimate and c_{j2} to c_{jK} are category parameter estimates, which are expressed in terms of the Likert-version of the GR model.

3) PC (generalized partial credit model)

Input format for the PC model is the same as that for the GR model although item parameters are specified differently under each of the two models.

4) NM (nominal response model)

$a_{j1} \;\; a_{j2} \;\ldots\; a_{jK}$

$b_{j1} \;\; b_{j2} \;\ldots\; b_{jK}$,

where a_{j1} to b_{jK} are item parameter estimates under Bock's nominal response model.

5) MC (multiple choice model)

$a_{j0} \;\; a_{j1} \;\; a_{j2} \;\ldots\; a_{jK}$

$b_{j0} \;\; b_{j1} \;\; b_{j2} \;\ldots\; b_{jK}$

$d_{j1} \;\; d_{j2} \;\ldots\; d_{jK}$,

where a_{j0} to d_{jK} are item parameter estimates under Thissen and Steinberg's multiple choice model.

OL [**IS** | **LC**] **DI** | { **FI** *filename* }

item-parameter-list

...

item-parameter-list

The keyword OL stands for an old or target form. The usage of OL is the same as that of NE, except that item parameter estimates calibrated on the old scale should be supplied.

[OPTION]
...
[OPTION]

As described earlier, there are ten option keywords to accommodate users' needs. The order of the option keywords does *not* matter. The detailed explanations for use of options are given below.

BY

The keyword BY stands alone. An input file must end with BY. All data after the keyword will be ignored. Therefore, users can write their additional comments after this keyword.

[OPTION KEYWORDS]

SC *scale-constant*

The keyword SC is used to set a scale constant that is usually used to match the logistic model to the normal ogive model. This keyword is used only for the three IRT models, DR, GR, and PC. A real number should be supplied for *scale-constant*. The default value of the scale constant is 1.7.

ST *slope intercept*

The keyword ST is used to provide "starting" values for the slope and intercept of the linear transformation in the characteristic curve methods. Users should supply two real numbers for the slop and intercept in order. The default values of the slope and intercept are 1.0 and 0.0, respectively, unless this option keyword is used.

IT *number-of-iterations*

The keyword IT is for the maximum number of iterations to obtain solutions that minimize the nonlinear criterion functions for the characteristic curve methods. Users should supply an integer they want. The default maximum number of iterations is 20. If a value of zero for *number-of-iterations* is given, no iteration will be made and the starting values given by the option keyword ST will be plugged into the criterion functions for the characteristic curve methods to lead to the calculated values of the criterion functions. This will be helpful when users want to identify the values of the criterion functions for the

characteristic curve methods at specific values of the slope and intercept.

OD *n* { **GH** } |

{ **PN** *mean std multiple-of-std* } |

{ **PB** *alpha beta lower-limit upper-limit* } |

{ **RN** *mean std* } |

{ **RU** *lower-limit upper-limit* } |

{ **ED** *starting ending* **/ EQ** | { **WL** *theta-weight-list* } } |

{ **SE** { **DI** *pair-of-value-and-weight-list* } | { **FI** *filename* } }

ND *the same as* **OD**

The keyword OD is used to specify summation/integration schemes over the proficiency distribution on the old (or base) scale to define criterion functions for the characteristic curve methods. The keyword ND is used to specify summation/integration schemes over the proficiency distribution on the new (current) scale. The integer after OD or ND is the number of proficiency points and should be a number between 1 and 1000, inclusive. When the option keyword SY is used with its subkeyword BI, the proficiency points and their weights provided by the both OD and ND keywords are used. When the option keyword SY is used with its subkeyword NO, only the proficiency points and their weights provided by the keyword OD are used. When the option keyword SY is used with its subkeyword ON, only the proficiency points and their weights provided by the keyword ND are used. For more information about the criterion functions defined on each of new and old scales, refer to Kim and Kolen (2004). The usage of ND and that of OD are the same except that proficiency points and their weights are assigned separately on each of new and old scales.

Specifically, the criterion functions for the characteristic curve methods are subject to the summation (or integration) scheme over examinees to incorporate the proficiency distributions on the old (base) and new (current) scales. Kolen and Brennan (2004) present several ways to specify the proficiency points and their weights to be incorporated into the criterion function in question. The OD and ND support all the ways described in Kolen and Brennan (2004). Each of OD and ND has seven subkeywords: GH, PN, PB, RN, RU, ED, and SE.

First, GH stands for the Gauss-Hermite quadrature points and weights. This

subkeyword can be used properly, if a continuous distribution of proficiency is known or estimated and the summation of the criterion function could be replaced by integration over the proficiency continuum. In the program, the distribution of proficiency is assumed a standard normal one. The possible maximum number of quadrature points is 180. Although more than 180 quadrature points are theoretically possible, authors' experiences suggest that quadrature weights tend to be unstable when trying to obtain more than about 200 quadrature points. According to Zeng and Kolen (1994), even 80 quadrature points seem to be enough to estimate the slope and intercept of a linear transformation to a satisfactory degree.

Second, PN stands for a polygonal approximation to a normal distribution. A polygonal approximation is often encountered in finding areas and evaluating integrals. PN can be used to evaluate *n* proficiency points and their weights to approximate a normal distribution having a value of *mean* and a value of *std,* with a left end point and a right end point being $mean - multiple\text{-}of\text{-}std \times std$ and $mean + multiple\text{-}of\text{-}std \times std$, respectively. More specifically, *n* proficiency points are equally spaced over the range of a left end point to a right end point. At each proficiency point, the density of the *N(mean, std)* is found. Each density is then divided by the sum of all the densities so that the densities are standardized. The standardized densities are then used as the weights. These steps are similar to those used in BILOG (Mislevy & Bock, 1990).

Third, PB stands for a polygonal approximation to a four-parameter beta distribution $Beta(\alpha, \beta, l, u)$, where α and β are two scale parameters and l and u are lower and upper limits. To evaluate *n* proficiency points and their weights, the same logic used in PN is applied except that the interval [*l, u*] is divided into *n* subintervals and then the midpoint of each subinterval is used for a proficiency point.

Fourth, RN stands for random numbers from a normal distribution. With two real numbers for a mean and a standard deviation, RN generates *n* pseudo-random proficiency values sampled from a normal distribution, *N(mean, std)*.

Fifth, RU stands for random numbers from a uniform distribution. With two real numbers for a lower limit and an upper limit, RU generates *n* pseudo-random proficiency values ranging from *lower-limit* to *upper-limit*.

Sixth, ED stands for equal distance in the intervals between two points on the proficiency scale. Users should supply two real numbers for a starting point and an ending point. The proficiency continuum ranging from the starting point to the ending point is divided into $n-1$ intervals with an equal length. Two subkeywords, EQ and WL, preceded

by a slash are prepared to provide weights associated with their proficiency values. EQ stands alone and is used to give the same constant weight 1.0. WL is used to list weights associated with their proficiency values, and the weights should be supplied with values users want.

Seventh, SE is used to specify *n* selected proficiency values and their weights in pairs. As in the keyword NC, DI or FI can be used. When using DI, selected proficiency values and their weights are input as follows.

```
-2.0 0.001
-1.5 0.015
 ...
 1.5 0.015
 2.0 0.001
```

When selected proficiency values and their weights are read in from an outside file, FI is used and the input format must be the same as that used in DI. The output file does not need to be located in a folder in which the executable file exists.

The default setting for proficiency values and their weights is that 25 proficiency values, which are equally spaced between -3.0 and 3.0, are used with the same weight 1.0 for all proficiency values. The default setting can be expressed as 25 ED -3.0 3.0 / EQ.

CW { **AL (** *category-weight-list* **)** } |
{ **BL** *number-of-blocks*
/ (*number-of-cat.* *number-of-items* **)** *item-list* **(** *category-weight-list* **)**
...
/ (*number-of-cat.* *number-of-items* **)** *item-list* **(** *category-weight-list* **)**
/ (*number-of-cat.* * **)** NONE **(** *category-weight-list* **)** }

The keyword CW is used only for the Stocking-Lord method under the two models, GR and PC. In other words, CW is prepared to give POLYST information about the different category weights (i.e., scoring functions) assigned to each item in order that the test characteristic curves may be defined. Often, two scoring functions are used, which are $U_{jk} = k$ and $U_{jk} = k - 1$, where U_{jk} is a weight allocated to the response category k of item j. POLYST uses the scoring function $U_{jk} = k - 1$ as a default scoring function, that is, *0 1 2 ... K – 1*, where *K* is the number of categories of an item. If users want to assign other scoring functions different from the default scoring function to any item, two subkeywords AL and BL can be used. Users can use the subkeyword AL (all) if all items

have the same scoring function. Users should supply a list of real numbers for it. Otherwise, the subkeyword BL (block) will come in handy. The usage of BL is the same as that of BL under the keyword NC except that a list of category weights, which is enclosed with parentheses, follows a list of items in each block. To conveniently enumerate a list of category weights having consecutive integers by an increment of 1, the special keyword `TO` can be used.

FS DO | NO [DO | NO]

The option keyword FS is used for standardizing the criterion functions for the characteristic curve methods. Two subkeywords, DO and NO, are used to specify options. The first DO or NO is for the Haebara method and the second DO or NO is for the Stocking-Lord method. DO means that standardization should be done whereas NO means that no standardization is done. What is meant by standardization of the criterion function is that one divides a sum of squared differences between characteristic curves in the criterion function by the number of the squared differences or the sum of weights assigned to the differences. For example, usually, to standardize the criterion function for the Stocking-Lord method, one divides the sum of squared differences between test characteristic curves by the number of proficiency values (or examinees). More specifically, for another example, a criterion function for the Haebara method can be expressed as follows:

$$\mathbf{Q} = \mathbf{Q_1} + \mathbf{Q_2},$$

where

$$\mathbf{Q_1} = \frac{1}{n\sum_{i=1}^{G_O} W(\theta_{iO})} \sum_{i=1}^{G_O} W(\theta_{iO}) \sum_{j=1}^{n} \left(\hat{P}_{ijO} - \hat{P}^{*}_{ijN}\right)^2, \text{ and}$$

$$\mathbf{Q_2} = \frac{1}{n\sum_{i=1}^{G_N} W(\theta_{iN})} \sum_{i=1}^{G_N} W(\theta_{iN}) \sum_{j=1}^{n} \left(\hat{P}_{ijN} - \hat{P}^{\#}_{ijO}\right)^2,$$

where $W(\theta_{iO})$ is a weight given to a value of θ_{iO} on the old scale, and $W(\theta_{iN})$ is a weight given to a value of θ_{iN} on the new scale. In the above expressions, $\mathbf{Q_1}$ and $\mathbf{Q_2}$ are standardized by their respective standardization factors, $n\sum_{i=1}^{G_O} W(\theta_{iO})$ and $n\sum_{i=1}^{G_N} W(\theta_{iN})$.

Theoretically, standardization of criterion functions should not affect the solutions of the characteristic curve methods. However, in practice, it could affect the solutions since the minimization algorithm used for nonlinear problems is affected by the magnitude of a

criterion function due to its stopping rules. If users do *not* want to standardize the criterion functions for both the Haebara and Stocking-Lord methods, they need to specify as follows:

```
FS NO NO
```

For the two models, NR and MC, under which the Stocking-Lord method are not considered, users must supply only one subkeyword, such as `FS DO`. The default setting is `FS DO [DO]`, which means that for both the Haebara and Stocking-Lord methods standardization is made. For users' information, the program ST (Hanson and Zeng, 1995) does not standardize the criterion functions for the characteristic curve methods.

SY { **BI** | **NO** | **ON** } [**BI** | **NO** | **ON**]

The option keyword SY is used to define criterion functions as non-symmetric or symmetric. Three subkeywords, BI, NO, and ON, are used to specify the symmetry-related directions of the criterion functions. The first BI, NO, or ON is for the Haebara method and the second BI, NO, or ON is for the Stocking-Lord method. Theoretically, the criterion function for each characteristic curve method can be defined in three symmetry-related schemes. The first is one in which the criterion function is defined only on the old scale (new-to-old direction: NO). The second is one in which the criterion function is defined only on the new scale (old-to-new direction: ON). The third is one in which the criterion function is defined on the both old and new scales (new-to-old and old-to-new; i.e., bi-directional: BI). More specifically, based on the criterion functions, $\mathbf{Q_1}$ and $\mathbf{Q_2}$, as described in the option keyword FS, BI corresponds to defining the criterion function by using both $\mathbf{Q_1}$ and $\mathbf{Q_2}$. NO corresponds to defining the criterion function by using $\mathbf{Q_1}$ alone. ON corresponds to defining the criterion function by using $\mathbf{Q_2}$ alone.

LM *slope intercept*
/ *number-of-searches radius tolerance* **NO** | **IN** | { **FI** *filename* }

The option keyword LM is used to search for possible local minimum solutions for the scale transformation constants after the first solutions for the characteristic curve methods are obtained. Three subkeywords, NO, IN, and FI are prepared to instruct the program how and where to show the resulting history of local minimum search. If NO is used, no history is shown in an output file. If IN is used, the resulting history is shown within the main output file specified by users or opened by the program. If FI followed by a file name is

used, the resulting history is saved separately in the file, which does not need to be located in the folder having the executable file of POLYST.

The criterion function defined by either of the Haebara and Stocking-Lord methods is nonlinear with respect to the scale transformation constants. Nonlinear systems could yield a local minimum but not a global minimum. To make sure that the obtained solutions (slope and intercept) are globally minimal, it is suggested that the option keyword LM be used. The first two real numbers, *slope* and *intercept*, are the values of the slope and intercept used to designate a center point of search. The center point is fixed and used for all iterations of local minimum search. An integer value, *number-of-searches*, after a slash is the number of iterations for local minimum search, which should range from 0 to 1000. A real number, *radius*, is a radius for local minimum search. With the center point being an origin, *number-of-searches* pseudo-random points within the radius are selected. If a selected point (slope, intercept) is within the radius but the value of slope is negative, the *absolute* value of slope is used. Then, the resulting point is used as the starting point for each of iterations of local minimum search. The last real number, *tolerance*, represents a tolerance limit for each of the slope and intercept, within which all solutions for the slope and intercept are regarded as the same. For example, suppose that the tolerance limit is given as 0.01. If the solution of the slope at the first search is 1.21 and the slope at the second search is 1.23, two solutions are regarded as different from each other and, in turn, the local minimum points from the two searches are regarded as different from each other. In this case, a * will appear in a table that presents a history of the local minimum search.

OP

The option keyword OP is used when users want to see in detail how options were given to the program. OP stands alone. Without OP, the section [OPTIONS AND DEFAULTS] will not show up in a main output file.

Errors in Running POLYST

POLYST gives a specific error message when it does not understand the syntax used in the input file. The error message will show up on the screen and tell possible sources of the error. Based on the message shown on the screen, users could find out any syntax errors in their input files of POLYST. Users could also get a hint by opening the output file, which is not successfully closed. In most cases, the last part of the output file, which is properly

read in, will give a hint enough to guess where a syntax error happened.

Reading Output Files

Reading an output file is straightforward. The output file first replicates the contents of the input file, and so users can verify if the program read the data properly. If the option keyword OP were used, the options given to the program would show up. Then, means and standard deviations of the common item parameter estimates are presented. Next, the solutions for the moment methods are presented and the first solutions for the characteristic curve methods are presented. If the local minimum search is tried through the use of the option keyword LM, a history of the search shows up. All solutions by method are summarized again at the end of the output file. Especially, how to read termination codes ranging from 1 to 5, which are passed from the minimization algorithm used in the program, is provided to give users information about the characteristics of the obtained solutions. If the program generates any termination code other than 1 in the Haebara and Stocking-Lord methods, it is strongly suggested that users try again with different starting values for the solutions or with the LM option.

Notification of Authors

It is requested that users of POLYST provide Seonghoon Kim or Michael J. Kolen with their email addresses by sending them a brief email message at

seonghoonkim@empal.com or michael-kolen@uiowa.edu.

This will permit contacting users about any errors, additions, or explanations relevant to the program.

Distribution

POLYST may be distributed to others without obtaining the permission of the authors. However, it is requested that anyone who uses this program contact the authors in the manner indicated above.

Disclaimer

No warranties are made that POLYST is free of error, that it is consistent with any particular standard, or that it will meet the requirements of any particular application. The authors disclaim any direct or consequential damages resulting from use of this program.

References

Dennis, J.E., & Schnabel, R.B. (1996). *Numerical methods for unconstrained optimization and nonlinear equations*. Philadelphia: Society for Industrial and Applied Mathematics.

Haebara, T. (1980). Equating logistic ability scales by a weighted least squares method. *Japanese Psychological Research, 22*, 144-149.

Hanson, B.A. & Zeng, L. (1995). ST: A computer program for IRT scale transformation.

Kim, S., & Kolen, M.J. (2004). *Methods for obtaining a common scale under unidimensional IRT models: A technical review and further extensions* (Occasional Paper). Iowa City, IA: Iowa Testing Programs.

Kolen, M.J., & Brennan, R.L. (2004). *Test equating, scaling, and linking: Methods and practices*. New York: Springer.

Loyd, B.H. & Hoover, H.D. (1980). Vertical equating using the Rasch model. *Journal of Educational Measurement, 17*, 179-193.

Marco, G.L. (1977). Item characteristic curve solutions to three intractable testing problems. *Journal of Educational Measurement, 14*, 139-160.

Mislevy, R.J., & Bock, R.D. (1990). *BILOG 3: Item analysis and test scoring with binary logistic models*(2nd ed.). Mooresville, IN: Scientific Software Inc.

Stocking, M., & Lord, F.M. (1983). Developing a common metric in item response theory. *Applied Psychological Measurement, 7*, 207-210.

Zeng, L., & Kolen, M.J. (1994). *IRT scale transformations using numerical integration*. Paper presented at the annual meeting of the American Educational Research Association, New Orleans.