Paper 208-2012

How Test Length and Sample Size Have an Impact on the Standard Errors for IRT True Score Equating: Integrating SAS® and Other Software

Yi-Fang Wu, Iowa Testing Programs, University of Iowa, Iowa City, IA, USA

ABSTRACT

The standard error of equating is a useful index to quantify the amount of equating error. It is the standard deviation of equated scores over replications of an equating procedure in samples from a population or populations of examines. The current study estimates the SE of item response theory true score equating in the Nonequivalent Groups with Anchor Test design using simulations. Specifically, the test length of the internal anchor and the sample size are of interests. Some specialized programs, such as BILOG-MG 3.0 for item calibration, ST for IRT scale transformations, and PIE for IRT true score equating, are incorporated to accomplish the equatings. The purpose of the paper is to demonstrate such a complicated and repetitive procedure through SAS®.

INTRODUCTION

Test scores or test results are often used in decision-making. The use of different test forms on different occasions (e.g. test dates) leads to the concern about score comparability since forms might differ somewhat in difficulty. For example, the ACT has six administrations yearly and each time an equivalent test form controlled under stringent content and statistical test specifications is given. Equating which is “a statistical process that is used to adjust scores on test forms so that scores on the forms can be used interchangeably” (Kolen & Brennan, 2004, p. 2) plays an important role. Ideally, the more similarity among the test forms, the more valid the score comparability. With the popularity of the item response theory (IRT) models in test development, test scoring and standard setting, test equating involving IRT methodologies become critical in operational work. Also, the flexibility of the Nonequivalent Groups with Anchor Test designs (NEAT; von Davier, Holland, & Thayer, 2004) in practice brings more and more research interests in equating. The current study provides an example of how to estimate the standard errors of equating (SEEs) by means of simulation data under the NEAT design in conjunction with the IRT true score equating method. Once the estimation procedure is established, there can be varieties and extensions in research designs based on different simulation factors of interest.

THE NONEQUIVALENT GROUPS WITH ANCHOR TEST DESIGN

Several data collection designs and equating methodologies have been proposed and implemented widely (see Holland & Dorans, 2006; Kolen & Brennan, 2004). The current study focuses on the NEAT designs, which are sometimes referred to the Common-Item Nonequivalent Groups (CING) designs. These designs are used when more than one form per test date cannot be administered because of test security or other practical concerns. In the NEAT design, Form X (new form) is administered to a sample of population **P** and Form Y (old form) is administered to a sample of population **Q**. The two forms have a set of items in common while one or both forms may also contain a set of unique items. Table 1 shows the configuration of this design. The difference between internal and external anchor items lies on the contribution of the anchor items to examinees’ test scores. The internal anchor items contribute to the test scores while the external anchor items do not. Ideally, the anchor should behave as a mini version of the complete form in many ways. For the internal common items, they are supposed to represent not only the content but the statistical characteristics of the old form (Kolen & Brennan). The current study only considers internal anchors.

Table 1
Configuration of the NEAT design

Population	Sample	X (new form)	A (anchor)	Y (old form)
P	1	✓	✓	
Q	2		✓	✓

Note. Modified from Table 6.4 of Holland & Dorans (2006)

IRT TRUE SCORE EQUATING

A typical IRT true score equating involves three steps. First, item parameters are estimated. Then, parameter estimates are scaled to a base IRT scale using linear transformations. The current study uses the Stocking and Lord method (1983) for developing this common IRT scale. Third, the true scores or the number-correct scores on the new form are transformed to the true score scale on the old form and then to the scale scores, if necessary. The work flows below will demonstrate these steps.

STANDARD ERRORS OF EQUATING

As mentioned, standard errors of equating (SEE) is the criterion used for checking the equating accuracy. The smaller the SEEs, the more the equating accuracy. SEE quantifies random errors due to sampling and is defined as the standard deviation of the equated scores over replications. SEEs differ by score point. Under a single simulation condition, the current study uses 100 replications to get 100 sets of equated true scores and then calculate the standard deviation of the equated true scores across these 100 samples.

DATA STRUCTURE OF SIMULATIONS

The study assumes a two-parameter logistic (2PL) unidimensional IRT model (Lord, 1980) for dichotomously (0/1) scored test forms. The model is represented as

$$p_{ij}(\theta_i; a_j, b_j, c_j)] = c_j + (1-c_j)\frac{\exp[Da_j(\theta_i - b_j)]}{1+\exp[Da_j(\theta_i - b_j)]},$$

where $p_{ij}(\theta_i; a_j, b_j, c_j)$ is the probability that the *i*th examinee with a θ_i ability answers the *j*th item correctly; a_j and b_j denote item discrimination and item difficulty parameters, respectively. *D* equal to 1.7 is the scaling constant. The following conditions are also assumed:

1. a_j ~Uniform(.2, 1.2) and b_j ~Normal(0, 1) for both forms;
2. θ_i ~Normal(0,1) in population **P** and θ_i ~Normal(0, 2) in population **Q**.

The NEAT design has forms with anchor items administered to two independent, nonequivalent groups. The current study considers 10, 20, 30, and 40 internal anchor items, and 40 unique items for the new form (Form X) taken by group 1 and another 40 unique items for the old form (Form Y) administered to group 2. Both Forms X and Y have 50, 60, 70, and 80 items and the total score of each form ranges from 0 to 50, 60, 70, and 80, respectively. In other words, the proportions of the internal anchor to the total test are increasing as follow: 1/5, 2/6, 3/7, and 4/8. See Figure 1 for the configuration of the form structures. Also, the study considers four sample sizes: *N* = 1000, 2000, 5000, and 10000. The same size is assumed for both groups 1 and 2 although the actual numbers of examinees could be different in practice. Theoretically, it is assumed that once the forms are constructed, two nonequivalent samples take the assigned form repeatedly.

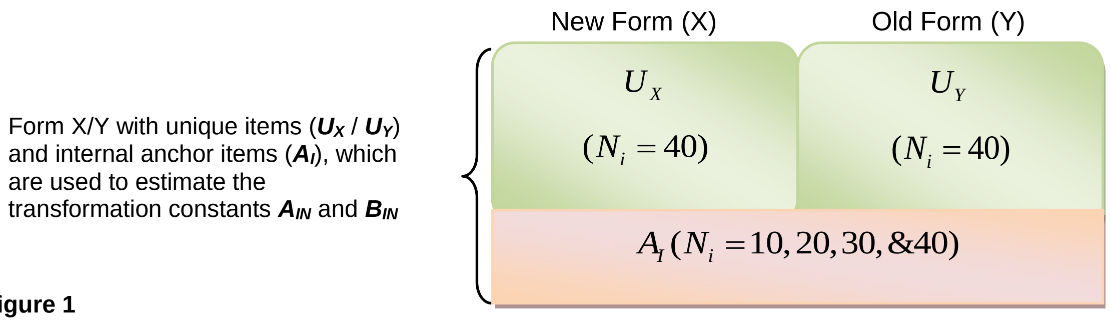
New Form (X)
Old Form (Y)
U_X
$(N_i = 40)$
U_Y
$(N_i = 40)$
$A_I(N_i = 10, 20, 30, \& 40)$
Form X/Y with unique items (U_X / U_Y) and internal anchor items (A_I), which are used to estimate the transformation constants A_{IN} and B_{IN}

Figure 1
The configuration of the form structures in the NEAT design

BEFORE JUMPING INTO SAS®

There is no doubt that the SAS® little man works as hard as he can. However, before programming in SAS®, a directory tree shown in Figure 2 needs to be established. Creating specific folders for different software and programs is helpful for debugging and checking the results before the number of replications increases. Under the root directory, a folder of the project name, **SAS_SEE**, is created. Four subdirectories of sample sizes are placed under the **SAS_SEE** and each of them has another four subdirectories inside it, indicating the different internal anchor lengths. Finally, the last layer of the subdirectory contains names of the three software packages and programs used for IRT true score equatings. Also, by indicating the corresponding path for each simulation condition, an excel file of the equating results, including the equated true scores and SEEs, will be sent here. Note that the SAS® programs are stored in the project folder only. Most importantly, the executable software packages and programs are placed in their folders. That is, BLM1.exe and BLM2.exe are stored in **BILOGMG3**, PIE.exe in **PIE**, and ST.exe in **ST**. After the directory trees are established, now it is time for programmers to see how powerful SAS® is in terms of data generation, data preparation, statistical analyses and result organization.

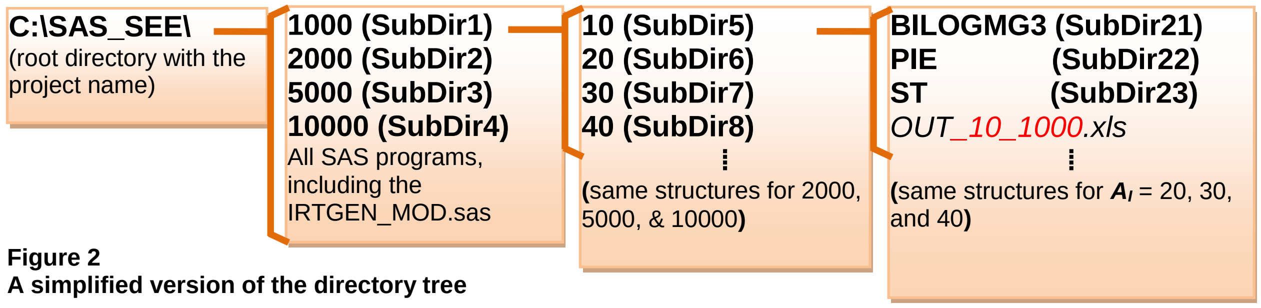

Figure 2
A simplified version of the directory tree

WORK FLOWS IN SAS®

The work flows not only show the data generation and the IRT true score equating procedures but also align with the programming in SAS®.

Step 0: Modificatioin of IRTGEN.sas—IRTGEN_MOD.sas

A well-developed SAS macro program to generate latent trait scores (i.e. θ_i) for commonly used IRT models (Whittaker, Fitzpatrick, Williams, & Dodd, 2003) is modified and used in the study. I only keep the error flags and the code of the 3PL IRT model since setting the pseudo-chance level parameter (c_j) to zero yields the 2PL model. As shown in Appendix A, the MACRO code ***L3GEN*** will generate 0/1 item responses.

Step 1: Data generation form Step1_ResponseGeneration.sas

As mentioned, four internal anchor lengths and four sample sizes are considered (i.e. 16 conditions). In order to get the SEEs, the number of replications for the IRT true score equating is set to 100. That is to say, for each case of the 16 conditions, 100 simulated data sets are obtained. The codes in *Step1_ResponseGeneration.sas* fulfill the need based on the data structures described above. Here I emphasize several bullets as follow.

In ❶, three macro variables—**`SEEDO`**, **`SEEDN`**, and **`SEEDI`**—are defined for feeding seeds to SAS for the unique items of the old form, the unique items of the new form as well as the internal anchor items. The number of unique items for each form is fixed (**`NITEMC = 40`**). The number of replications throughout the study is 100. Unlike the fixed length of the unique part, ❷ shows the internal anchor length actually varies. Recall that 10, 20, 30, and 40 items are of interest. Thus, a DO-LOOP incremented by 10 is used and defined by the macro variable, **`NITEMC`**.

In ❸, a macro program passed by three arguments is declared and defined to generate item parameters. The a_j parameters are assumed to be drawn from the same uniform distribution across the unique part of the old form, the unique part of the new form, and the shared internal anchor items; b_j parameters are from a standard normal distribution. But different seeds should be used since the items are different. As shown in ❹, I use **`RANUNI`** and **`RANNOR`** functions to draw random numbers for a_j and b_j parameters. Multiplying the random variates by 1 and adding up 0.2 gives out random numbers from Uniform(.2, 1.2). Note that c_j parameters are set to 0 for a 2PL IRT model.

In ❺, another macro program passed by six arguments is declared and defined. This program is used to generate the latent trait score distributions (i.e. θ distribution) and for setting up the 100 replications in order to get the 100 simulated data sets with old and new forms paired. As shown in ❻, **`MU`** and **`VAR`**, are passed by reference in conjunction with the **`RANNOR`** function. As seen in ❽, the group 1 is sampled from *N*(0, 1) where the group 2 is sampled from *N*(0, 2). This says the population who took the old form was more variable than the population who took the new form while their average ability was assumed to be the same. At this point, the elements needed for using the modified *IRTGEN.sas* (i.e. *IRTGEN_MOD.sas* in Appendix A) are all set. In ❼, the codes are included and the corresponding parameter values are also declared. I emphasize **`INO`** and **`INN`** in ❽, the parameters passed by the second argument in the third and fourth **`REP100`** calls. This is because the internal anchor item responses are generated from two different theta distributions: *N*(0,1) and *N*(0,2) as stated in ❻; however, they should share the same a_j and b_j parameters since these items are the internal anchor. Hence, the first argument passes the same parameter, **`INT`**, for both groups.

After the 0/1 data sets are generated, one now needs to concatenate the unique part and the internal anchor for each form as shown in ❾. Renaming the internal anchor items is needed before they are merged with the unique part. Note that in ❿, **`TEMPID`** is created for each examinee since the BILOG-MG 3.0 always needs some fixed-length characters in the beginning of each record. A randomly-selected number (i.e. 6666) is used everywhere; the estimated traits (thetas) for examinees are not of interest in the current study. Note that in ⓫, the 100 simulated data sets are stored in the corresponding subdirectory by means of **`&NEXAM`** and **`&NITEMC`**. They are stored in the subdirectory of BILOGMG3 since the next step is to conduct item calibration using the specialized software, BILOG-MG 3.0. Finally, as shown in ⓬, the closing of the **`SIMULATION`** macro program takes four sample size into consideration; 1000, 2000, 5000, and 10000 are passed to **`NEXAM`**, respectively.

```
/*********************Start of Step1_ResponseGeneration.sas**********************/
%LET SEEDO  = 5648574; %LET SEEDN  = 2541458; %LET SEEDI  = 457892; ❶
%LET NITEMU = 40;
%LET REPLICATION = 100;
%MACRO SIMULATION(NEXAM);
    %DO I = 10 %TO 40 %BY 10; ❷
      %LET NITEMC = &I;
      %MACRO FORMDIST(FORM, SEED, NUMITEM); ❸
      DATA AB_par_&FORM;
             KEEP A B C;
             DO I = 1 TO &NUMITEM;
                    A = 1 * RANUNI(&SEED) + 0.2;   A~Uniform(.2, 1.2)
          ❹         B = RANNOR(&SEED);             B~Normal(1,0)
                    C = 0;                         C = 0 for 2PL IRT model
                    OUTPUT;
             END;
      RUN;
      %MEND FORMDIST;
      %FORMDIST (OLD, &SEEDO, &NITEMU);
      %FORMDIST (NEW, &SEEDN, &NITEMU);
      %FORMDIST (INT, &SEEDI, &NITEMC);
```

Define A and B distributions

```
      %MACRO REP100(FORM1, FORM2, MU, VAR, SEED, NUMITEM); ❺
      %DO K = 1 %TO &REPLICATION;
         %LET REP = &K;
             DATA THETA;
                    KEEP THETA;
                    DO I = 1 TO &NEXAM;
          ❻                THETA = &MU + sqrt(&VAR)*RANNOR(&SEED);OUTPUT;   Specify the θ
                    END;                                                      distribution
             RUN;

      %INCLUDE "C:\SAS_SEE\IRTGEN_MOD.sas"; ❼
      %IRTGENL3(MODEL=L3,DATA=AB_par_&FORM1,OUT=&FORM2.&REP,NI=&NUMITEM,NE=&NEXAM);
             DATA &FORM2.&REP; SET &FORM2.&REP; ID = 10000 + _N_; RUN;
      %MEND REP100;
      %REP100(OLD, OLD, 0, 1, &SEEDO, &NITEMU);
      %REP100(NEW, NEW, 0, 2, &SEEDN, &NITEMU);
      %REP100(INT, INO, 0, 1, &SEEDO, &NITEMC); ❽
      %REP100(INT, INN, 0, 2, &SEEDN, &NITEMC);
```

Generate 0/1 item responses

```
      %MACRO OUTTXT(UNIQUE, COMMON);
             DATA &COMMON.&REP; SET &COMMON.&REP;
          ❾        ARRAY R {&NITEMC} R1-R&NITEMC;
                    ARRAY S {&NITEMC} S1-S&NITEMC;
                           DO J = 1 TO &NITEMC;
                                  S(J) = R(J);
                           END;
                    KEEP ID S1-S&NITEMC;
      ❿      DATA &UNIQUE.&REP; MERGE &UNIQUE.&REP &COMMON.&REP;
                 BY ID; TEMPID ="6666"; DROP THETA ID;
             DATA _NULL_; SET &UNIQUE.&REP;
      ⓫      FILE "C:\SAS_SEE\&NEXAM\&NITEMC\BILOGMG3\&UNIQUE.&REP..txt" LRECL = 200;
                 PUT @1 TEMPID @6 R1-R&NITEMU S1-S&NITEMC;
             RUN;
      %END;
      %MEND OUTTXT;
      %OUTTXT(OLD, INO);
      %OUTTXT(NEW, INN);
    %END;
%MEND SIMULATION;
%SIMULATION(1000); %SIMULATION(2000); %SIMULATION(5000); %SIMULATION(10000); ⓬
/*********************End of Step1_ResponseGeneration.sas**********************/
```

Create plain texts for BILOG-MG 3.0

Step 2: Item estimation—Creating BILOG-MG 3.0 command files and calibrating items

As mentioned, the first step in a typical IRT true score equating is item parameter estimation. BILOG-MG 3.0 (Zimowski, Muraki, Mislevy, & Bock, 2003) is run separately for each set of the sample data for groups 1 and 2 obtained above in order to estimate item parameters and the two estimated theta distributions. SAS® is useful for creating the layout needed for other software packages and incorporate multiple software packages into a simulation study (Gagné, Furlow, & Ross, 2009). The codes below show how a bunch of BILOG-MG 3.0 command files can be created and how the Disk Operating System (DOS) can be called without leaving SAS®. The purpose is to create command files with the file extension (**.blm**) and a batch file that can be executed within DOS. In particular, two executable programs in the BILOG-MG 3.0 package are used—BLM1.exe and BLM2.exe—for item parameter estimation.

In *Step2_ItemCalibration.sas*, it starts a macro program **`BILOGMG3`** with another program **`TEST`** embedded in it. The inner one takes care of two forms: old and new. Together with the outer one, this part creates the format of the BILOG-MG 3.0 control cards. As indicated in ❶, the conditions here are the same as the **`SIMULATION`** macro program in Step 1. We want 100 replications under four internal anchor lengths and four sample sizes. Starting a new one can ease the burden of coding and decrease the complexity in debugging. In ❷, three new macro variables are created. **`PATH1`** simply tells the subdirectory of BILOGMG3. **`NITEMFULL`** as well as **`RES_LENGTH`** are important for BILOG-MG 3.0 since the full length of the response data has to be identified and read in correctly when item parameters are estimated. The purple section indicated by ❸ shows how the BILOG-MG 3.0 command file should look like, if the corresponding parameter values are replaced. Also, the created command files are stored in the subdirectory as **`PATH1`** indicates after **`&NEXAM`** and **`&NITEMC`** are resolved. As ❹ shows, **`.PAR`** and **`.PDIST`** files are saved when BILOG-MG 3.0 is done with the calibration. The information stored in these files is needed for scale transformation and IRT true score equating. Again, they will be stored in the corresponding subdirectory once the BILOG-MG 3.0 is run. In ❺, note that the line `#13 (4A1,1X,&RES_LENGH.A2)` tells BILOG-MG 3.0 how to read in the simulated 0/1 data matrix each time. One should always check the output of the PH1 from the calibration in order to make sure the response strings are identified correctly.

Indicated by ❻, the codes create batch files that will be executed by DOS. Taking **`NEXAM`** = 1000 and **`NITEMC`** = 10 for example, a batch file shown in Figure 3 is created. In ❼, the complied BATCH file in the temporary library needs to be deleted to remove duplicated records, if any. After the batch file is created, store it in the BILOGMG3 folder for each simulation condition as shown in ❽. Figure 4 is an example of the command file when there are 40 anchor items and 10000 examinees. Finally in ❾, automate the BLM1.exe and BLM2.exe using the **`X command`**. The output of PH1, PH2, .PAR and .PDIST will be stored once BILOG-MG 3.0 finishes the item calibration for each simulated data sets.

```
cd C:\SAS_SEE\1000\10\BILOGMG3
"C:\SAS_SEE\1000\10\BILOGMG3\BLM1" OLD_1
"C:\SAS_SEE\1000\10\BILOGMG3\BLM2" OLD_1
⁞
"C:\SAS_SEE\1000\10\BILOGMG3\BLM1" NEW_100
"C:\SAS_SEE\1000\10\BILOGMG3\BLM2" NEW_100
exit
```

400 lines =
2 (executable programs) x 2 (forms) x 100 (replications)

Figure 3
A simplified example of the BILOG-MG 3.0 batch file

```
ITEM PARAMETERS ESTIMATION WITH 2PL IRT MODEL USING BILOG-MG 3.0
USING SIMULATED DATA--NEW.100
>GLOBAL DFNAME='C:\SAS_SEE\10000\40\BILOGMG3\NEW100.TXT', NPARM = 2,
        NTEST =1, SAVE;
>SAVE   PARM='C:\SAS_SEE\10000\40\BILOGMG3\PAR_NEW100.PAR'
        PDISTRIB='C:\SAS_SEE\10000\40\BILOGMG3\PDIST_NEW100.PDIST';
>LENGTH NITEMS=80;
>INPUT  NTOT=80, SAMPLE=10000, NID=4;
>ITEMS  INAMES=(FULL01(1)FULL80),
        INUM=(01(1)80);
>TEST  TNAME=NEW.100,
       INAMES=(FULL01(1)FULL80);
(4A1, 1X, 160A2);
>CALIB NQPT = 40, CYCLES = 200, NEWTON = 20, CRIT = 0.0010, PLOT = 0.0000,
       FIXED, IDIST = 0, NOSPRIOR, TPRIOR;
```

Figure 4
An example of the BILOG-MG 3.0 command file

Create format needed for BILOG-MG 3.0 command files

```
/*************************Start of Step2_ItemCalibration.sas*************************/
       %MACRO BILOGMG3(NEXAM);
              %MACRO TEST(FORM);
                 %DO K = 1 %TO &REPLICATION;  ❶
                 %LET REP = &K;
                     %DO I = 10 %TO 40 %BY 10;
                     %LET NITEMC = &I;
              %LET PATH1 = C:\SAS_SEE\&NEXAM\&NITEMC\BILOGMG3;
              %LET NITEMFULL = %EVAL(&NITEMC + 40);
              %LET RES_LENGH = %EVAL(&NITEMFULL *2);  ❷

           DATA _NULL_;
              FILE "&PATH1\BM_&FORM.&REP..blm";
           PUT #1  "ITEM PARAMETERS ESTIMATION WITH 2PL IRT MODEL BY BILOG-MG 3.0"
                #2  "USING SIMULATED DATA--&FORM..&REP"
                #3  ">GLOBAL DFNAME=" "'" "&PATH1\&FORM.&REP..TXT" "'" ", NPARM = 2,"
                #4  "         NTEST =1, SAVE;"
                #5  ">SAVE    PARM=" "'" "&PATH1\PAR_&FORM.&REP..PAR" "'"  ❹
                #6  "         PDISTRIB=" "'" "&PATH1\PDIST_&FORM.&REP..PDIST" "'" ";"
                #7  ">LENGTH NITEMS=&NITEMFULL;"
                #8  ">INPUT  NTOT=&NITEMFULL, SAMPLE=&NEXAM, NID=4;"           ❸
                #9  ">ITEMS  INAMES=(FULL01(1)FULL&NITEMFULL),"
                #10 "        INUM=(01(1)&NITEMFULL);"
                #11 ">TEST  TNAME=&FORM..&REP,"
                #12 "       INAMES=(FULL01(1)FULL&NITEMFULL);"
            ❺  #13 "(4A1,1X,&RES_LENGH.A2);"
                #14 ">CALIB NQPT=40,CYCLES=200,NEWTON=20,CRIT=0.0010,PLOT = 0.0000,"
                #15 "       FIXED, IDIST = 0, NOSPRIOR, TPRIOR;"
              ;
           RUN;
                     %END;
                 %END;
              %MEND TEST; %TEST(OLD);%TEST(NEW);
       %MEND BILOGMG3;
       %BILOGMG3(1000); %BILOGMG3(2000); %BILOGMG3(5000); %BILOGMG3(10000);
```

Create batch files for BILOG-MG 3.0

```
       DATA EXIT;    LENGTH BLM $200;   BLM = "exit";  ❻
       %MACRO OUTFILE;
              %DO I = 10 %TO 40 %BY 10;
              %LET NITEMC = &I;
                %MACRO BLMBATCH(NEXAM);
                  DATA CD; LENGTH BLM $200; BLM = "cd &PATH1";
                  %DO K = 1 %TO &REPLICATION;
                  %LET REP = &K;

       DATA BATCH_TEMP1; LENGTH BLM $200; BLM ="""&PATH1.\BLM1"" "||" OLD_&REP";
       DATA BATCH_TEMP2; LENGTH BLM $200; BLM ="""&PATH1.\BLM2"" "||" OLD_&REP";
       DATA BATCH_TEMP3; LENGTH BLM $200; BLM ="""&PATH1.\BLM1"" "||" NEW_&REP";
       DATA BATCH_TEMP4; LENGTH BLM $200; BLM ="""&PATH1.\BLM2"" "||" NEW_&REP";
       DATA BATCH_TEMP; SET BATCH_TEMP1 BATCH_TEMP2 BATCH_TEMP3 BATCH_TEMP4;

       PROC DATASETS LIB = WORK NOLIST NODETAILS;
              APPEND BASE = BATCH DATA = BATCH_TEMP; QUIT;
       DATA STBATCH; SET CD BATCH EXIT;
                  %END;
       PROC DATASETS LIB = WORK NOLIST NODETAILS; DELETE BATCH; QUIT;  ❼

       %LET BATCHFILE = &PATH1\BILOG_BATCH.BAT;  ❽
       DATA _NULL_; SET STBATCH;
              FILE "&BATCHFILE";
              PUT @1 BLM $200.;   RUN;
       /**/    x %quote("&BATCHFILE");     /**/  ❾
                %MEND BLMBATCH;
                %BLMBATCH(1000); %BLMBATCH(2000); %BLMBATCH(5000); %BLMBATCH(10000);
              %END;
       %MEND OUTFILE; %OUTFILE;
/*************************End of Step2_ItemCalibration.sas*************************/
```

Execute a batch file which calls BLM1.exe and BLM2.exe

Step 3: Scale transformation—Creating ST command files and conducting the program

The ST program (Hanson & Zeng, 1995a) is used to estimate the Stocking and Lord scale transformation constants A and B by using the internal anchor item parameter estimates and the two estimated theta distributions obtained from Step 2. For a specific sample size, this is done for using the 10, 20, 30, and 40 internal anchor items respectively, which results in the four pairs of transformation constants, the slope (A_{IN}) and intercept (B_{IN}) in Figure 1.

In a similar fashion, I use SAS® to create the command files and the batch files needed for the ST program to conduct the scale transformation. The purpose of the codes below is to create command files with the extension of "**.stin**", to create batch files that can be executed within DOS, and finally to yield **".stout**" files containing the Stocking and Lord transformation constants.

First, the ST command file has its own layout. SAS® is beneficial because of its flexibility in dealing with strings and creating data sets. In this example, stacking techniques are used in order to meet the format of the layout. See Hanson and Zeng (1995a) for the requirement of the layout. Codes shown in ❶ create common part of all the ST command files so they are independent from the macro program. In ❷, a macro variable called **CITEMPOSITIOHN** is initialized conditionally. The values initialized are used to locate item number and the parameter estimates of the internal anchor items depending on the anchor length. As indicated in ❸, the item parameter estimates are brought in for both old and new forms but we are only interested in the estimates of the common items shown in ❷. Thus, in ❹, **CITEMPOSITIOHN** brings in correct item numbers and their item parameter estimates for the internal anchor. They are then used to find the Stocking and Lord linear transformation constants later.

Codes in ❺ and ❻ look very similar. The former takes care of the theta values and the later deals with the weights of the theta distribution (see Hanson & Zeng, 1995a). **.PDIST** files have been created when BILOG-MG 3.0 conducts the PH2 estimation for the items. The format of the layout is fixed so the first valid value always starts at line 17. Also when 40 quadrature points are used, an 8 by 5 matrix with theta or weight values is yielded; thus, the next few lines of codes are just to create the format needed for the ST program. The internal anchor item parameters are estimated separately for old and new forms. In the ST command files, they need to be complied together but stacked in different positions. Thus, a macro variable **STACKPOSITION** as part of the **TEST** program is required as shown in ❼. Once every element needed for the command file of the ST program, 100 **.STIN** files are created in the subdirectory under each condition indicated by ❽. Finally in ❾, the creation of the ST batch files is very similar to the creation of the BILOG-MG 3.0 ones. For convenience, **PATH2** indicated by ❿ directs the command files and the batch file to the corresponding subdirectory by length and by sample size. Up to this point, all the files that the ST program needs for putting the new form (Form X) parameter estimates on the old form (Form Y) scale are all set. By the end of *Step3_ScaleTransformation.sas*, the batch file is automatically executed by the X command line: `x %quote("&BATCHFILE")`. In fact, 16 ST batch files taking care of the 16 simulation conditions are activated after this chuck of codes is run.

```
/*********************Start  of Step3_ScaleTransformation.sas**********************/
DATA ST1; STACK = 1; LENGTH ST $200; ST = "$ Item_Parameters New Form";         ❶
DATA ST3; STACK = 3; LENGTH ST $200; ST = "$ Link_Item_Parameters Old Form";
DATA ST5; STACK = 5; LENGTH ST $200; ST = "$ Theta_Distribution New Form";
DATA ST7; STACK = 7; LENGTH ST $200; ST = "$ Link_Theta_Distribution Old Form";

%MACRO SCALETRAS(NEXAM);
       %DO K = 1 %TO &REPLICATION;
       %LET REP = &K;
              %DO I = 10 %TO 40 %BY 10;
              %LET NITEMC = &I;
/*******************************************************************/
/*   Internal Anchor item positions vary: From 41 to 50/60/70/80    */   ❷
/*******************************************************************/
%IF &NITEMC = 10 %THEN %LET CITEMPOSITION = ('41','42','43','44','45','46','47','48','49','50');
%ELSE %IF &NITEMC = 20 %THEN %LET CITEMPOSITION =
('41','42','43','44','45','46','47','48','49','50','51','52','53','54','55','56','57','58','59','60');
%ELSE %IF &NITEMC = 30 %THEN %LET CITEMPOSITION =
('41','42','43','44','45','46','47','48','49','50','51','52','53','54','55','56','57','58','59','60','6
1','62','63','64','65','66','67','68','69','70');
%ELSE %IF &NITEMC = 40 %THEN %LET CITEMPOSITION =
('41','42','43','44','45','46','47','48','49','50','51','52','53','54','55','56','57','58','59','60','6
1','62','63','64','65','66','67','68','69','70','71','72','73','74','75','76','77','78','79','80');
```

Bring in anchor item parameter estimates

```
%MACRO TEST(FORM, STACKPOSITION);
DATA &FORM._&REP;
       INFILE "&PATH1\PAR_&FORM.&REP..PAR" MISSOVER PAD FIRSTOBS = 5;   ❸
       INPUT ITEMNAME $1-7  A 37-46 B 57-66 C 97-106;
DATA COM&FORM._&REP; SET &FORM._&REP;
       IF SUBSTR(ITEMNAME, 5, 2) IN &CITEMPOSITION;   ❹
```

```
/**********************Step3_ScaleTransformation.sas(Cont'd)************************/
```

❺ Bring in the theta distribution from PH2

```
     DATA &FORM.THETA1_&REP;
          INFILE "&PATH1\PDIST_&FORM.&REP..PDIST" PAD MISSOVER FIRSTOBS=7 OBS=14
          INPUT   @2 t1 $CHAR11. @14 t2 $CHAR11. @26 t3 $CHAR11. @38 t4 $CHAR11. @50 t5 $CHAR11.;
     PROC TRANSPOSE DATA = &FORM.THETA1_&REP
                    OUT = &FORM.THETA1_T_&REP(KEEP = COL1-COL8); VAR t1-t5; RUN;
               %MACRO ITE1;
                    %DO J = 1 %TO 8;
                      %LET VAR = &J;

                      DATA TEMP&VAR; SET &FORM.THETA1_T_&REP (KEEP = COL&VAR);
                      DATA TEMP&VAR; SET TEMP&VAR(RENAME = (COL&VAR = COL));

                      PROC APPEND BASE = TEMPALL_&FORM DATA = TEMP&VAR;RUN;QUIT;
                    %END;
               %MEND ITE1;
     PROC DATASETS LIB = WORK NOLIST NODETAILS; DELETE TEMPALL_&FORM; QUIT;
               %ITE1;
     DATA &FORM.THETA_&REP; SET TEMPALL_&FORM (RENAME = (COL = PTHETA)); RUN;
```

❻ Bring in the weight distribution from PH2

```
     DATA &FORM.WEIGHT1_&REP;
          INFILE "&PATH1\PDIST_&FORM.&REP..PDIST" PAD MISSOVER FIRSTOBS=17 OBS=24;
          INPUT   @2 t1 $CHAR11. @14 t2 $CHAR11. @26 t3 $CHAR11. @38 t4 $CHAR11. @50 t5 $CHAR11.;
     PROC TRANSPOSE DATA = &FORM.WEIGHT1_&REP
                    OUT = &FORM.WEIGHT1_T_&REP(KEEP = COL1-COL8); VAR t1-t5; RUN;
               %MACRO ITE1;
                    %DO J = 1 %TO 8;
                    %LET VAR = &J;

                    DATA TEMP&VAR; SET &FORM.WEIGHT1_T_&REP (KEEP = COL&VAR);
                    DATA TEMP&VAR; SET TEMP&VAR(RENAME = (COL&VAR = COL));

                    PROC APPEND BASE = TEMPALL_&FORM DATA = TEMP&VAR;RUN;QUIT;
                    %END;
               %MEND ITE1;
     PROC DATASETS LIB = WORK NOLIST NODETAILS; DELETE TEMPALL_&FORM; QUIT;
               %ITE1;
     DATA &FORM.WEIGHT _&REP; SET TEMPALL_&FORM (RENAME = (COL = PWEIGHT)); RUN;
```

Compile theta and weight estimates from both forms

```
     DATA &FORM.ALL_&REP;
          MERGE &FORM.THETA_&REP &FORM.WEIGHT_&REP;
          LENGTH ST $200;
          ST = LEFT(PTHETA) || " " ||PWEIGHT; STACK = 6; KEEP STACK ST; RUN; QUIT;
     DATA COM&FORM._&REP; SET COM&FORM._&REP;
          N = _N_;
          STACK = &STACKPOSITION;
          ST = LEFT(N) || " " || A || " " || B || " " || C;
          KEEP STACK ST; RUN;
   %MEND TEST;
   %TEST(OLD,4); TEST(NEW,2);
```

❼

```
   DATA FINALST_&REP;
          SET ST1 COMNEW_&REP ST3 COMOLD_&REP ST5 NEWALL_&REP ST7 OLDALL_&REP;
          FILE "C:\SAS_SEE\&NEXAM\&NITEMC\ST\OLDNEW_&REP..STIN";
          PUT @1 (ST)($200.);
   RUN;
          %END;
   %END;
%MEND SCALETRAS;
%SCALETRAS(1000); %SCALETRAS(2000); %SCALETRAS(5000); %SCALETRAS(10000);
```

❽

Create batch files for ST program

```
/**********************Step3_ScaleTransformation.sas(Cont'd)************************/
       DATA EXIT; LENGTH ST $200; ST = "exit";
       %MACRO OUTFILE;
9             %DO I = 10 %TO 40 %BY 10;
              %LET NITEMC = &I;
                %MACRO STBATCH(NEXAM);
                %LET PATH2 = C:\SAS_SEE\&NEXAM\&NITEMC\ST;  10
                  DATA CD;  LENGTH ST $200; ST = "cd &PATH2";
                  %DO K = 1 %TO &REPLICATION;
                  %LET REP = &K;

                  DATA BATCH_TEMP; LENGTH ST $200;
                     ST ="""&PATH2.\ST"" "||" OLDNEW_&REP..STIN OLDNEW_&REP..STOUT";
                  PROC DATASETS LIB = WORK NOLIST NODETAILS;
                         APPEND BASE = BATCH DATA = BATCH_TEMP; QUIT;
                  DATA STBATCH; SET CD BATCH EXIT; RUN;

                  %END;
                  PROC DATASETS LIB = WORK NOLIST NODETAILS; DELETE BATCH; QUIT;

               %LET BATCHFILE = &PATH2\ST_BATCH.BAT;
                  DATA _NULL_; SET STBATCH;
                         FILE "&BATCHFILE";
                         PUT @1 ST $200.;
                  RUN;
       x %quote("&BATCHFILE");          → Execute a batch file which calls ST.exe
                %MEND STBATCH;
                %STBATCH(1000); %STBATCH(2000); %STBATCH(5000); %STBATCH(10000);
              %END;
       %MEND OUTFILE; %OUTFILE;
/*********************End of Step3_ScaleTransformation.sas**********************/
```

Step 4: IRT true score equating—Creating PIE command files and conducting the program

After the scale transformation is done, it is time for the IRT true score equating. The PIE program (Hanson & Zeng, 1995b) is used to obtain Form Y true score equivalents for Form X using item parameter estimates for both forms, with rescaled item parameters on Form X that have been obtained by Step 3.

The creation of the command and batch files for the PIE program is very similar to the creation for the ST program and thus is not provided. The purposes are to create command files with the extension of "**.piein**", to create batch files that can be executed within DOS, and finally to yield "**.pieout**" files containing the equated scores. The key is to read in the **.stout** files from the previous step in order to get the transformation constants, A and B; they are named "intercept" and "slope" by the ST program. Therefore, carrying the variable names from ST, one must include the following lines to rescale the a_j and b_j parameter estimates for the new form (Form X) before IRT true score equatings are actually done. Six digits are reserved for precisions.

```
    A = ROUND(A/SLOPE,.000001);
    B = ROUND(B*SLOPE + INTERCEPT,.000001);
    C = ROUND(C, .000001);
```

Step 5: Calculating the standard errors of equating

Steps 2-4 are repeated for 100 times by using the 100 simulation data sets derived from Step 1. The standard deviation of the Form Y true score equivalents is computed over 100 replications to obtain the empirical standard error estimates of IRT true score equating at each raw score point for Form X. In sum, Steps 1 to 5 are done for the four sample sizes considered.

The codes used to calculate the standard errors of equating are not provided. The basic idea is straightforward: Using **DATA** steps within macro programs to get the equated scores (e.g. equated true scores) at each raw score point from the 100 replications of IRT true score equating. Taking raw score (RS) of 1 for example, **PROC MEANS** is used to get the standard deviation of the 100 equated true scores for this score point. That is, the resulted standard deviation is the SEE at RS = 1. For the test length of 50, replicate this process for 51 times would result in the SEEs for true scores ranging from 0 to 50. For other test lengths, the process is replicated in a similar fashion.

Step 6: Plotting the standard errors of equating

Finally, SEEs based on the factors of interest can be plotted for visualizing the differences and for the purpose of comparisons. To systematically compare the effect of sample size and the internal anchor length, the same population values of item parameters (i.e. the same forms) and θ distributions are used. So, the seeds are fixed as if the forms are fixed throughout the study. One can change the seeds and replicate the whole study as if a pair of different old and new forms is used. For simplicity, the codes used to compile the SEEs for the 16 conditions are not shown. Assuming a data set contains the old form true score points (`RS`) and the SEEs of the equated true scores, the following codes can be used to plot the SEEs for different sample sizes and internal anchor lengths. As an example, Figure 5 shows the comparisons of SEEs for the four sample sizes when 10 internal anchor items are used. Finally, including the `TITLE` command in the end prevents the carrying of the title specified here.

```
%MACRO PLOT1 (NITEMC);
PROC SGPLOT DATA = COMPILE NOAUTOLEGEND;
   WHERE NITEMC = "&NITEMC";
     SERIES X=RS Y=STD_&NITEMC._10000/LINEATTRS=(COLOR="GREEN" THICKNESS=2 PATTERN=SOLID);
     SERIES X=RS Y=STD_&NITEMC._5000 /LINEATTRS=(COLOR="BLUE"  THICKNESS=2 PATTERN=SOLID);
     SERIES X=RS Y=STD_&NITEMC._2000 /LINEATTRS=(COLOR="RED"   THICKNESS=2 PATTERN=SOLID);
     SERIES X=RS Y=STD_&NITEMC._1000 /LINEATTRS=(COLOR="BLACK" THICKNESS=2 PATTERN=SOLID);
     XAXIS LABEL = 'Raw Score';
     YAXIS LABEL = 'Standard Error of Equating';
     INSET "Length of Internal Anchor is &NITEMC"/POSITION=BOTTOMRIGHT NOBORDER;
     INSET "Black=1000; Red=2000; Blue=5000; Green=10000"/POSITION=TOPLEFT NOBORDER;
TITLE "Comparisons of Standard Error of Equating for N = 1000, 2000, 5000, & 10000";
RUN;
%MEND PLOT1; %PLOT1(10); %PLOT1(20); %PLOT1(30); %PLOT1(40); TITLE;
```

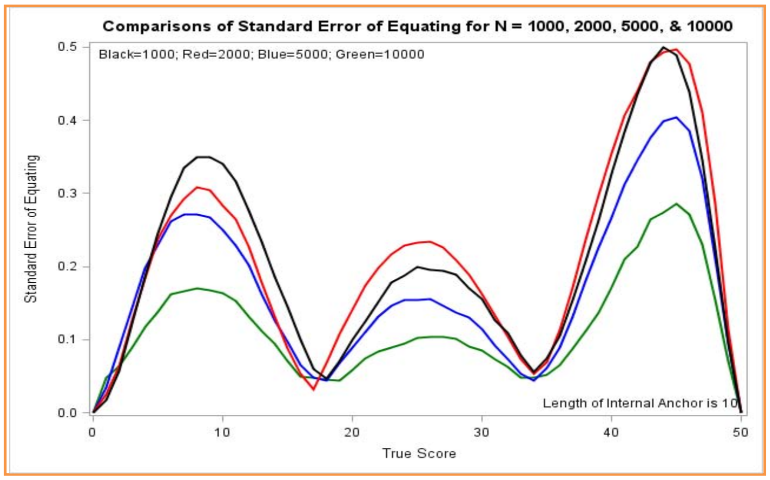

Figure 5
SEEs for 10 internal anchor items and different sample sizes

Step 7: A quick implementation of IRT true score equating

It is not necessarily to include Step 7 in the work flows. However, if one changes the seeds in *Step1_DataGeneration.sas* and implements the following codes, the study can be conducted again. Conceptually, using a different set of seeds implies introducing a new pair of old and new forms. One can replicate the study in such a way to see whether there is a systematic pattern by varying the internal anchor length and the sample size.

```
%include "C:\SAS_SEE\Step1_ResponseGeneration.sas";
%include "C:\SAS_SEE\Step2_ItemCalibration.sas";
%include "C:\SAS_SEE\Step3_ScaleTransformation.sas";
%include "C:\SAS_SEE\Step4_IRTequating.sas";
%include "C:\SAS_SEE\Step5_SEEquating.sas";
%include "C:\SAS_SEE\Step6_PlotSEE.sas";
```

CONCLUSION

Based on one pair of forms, the results of the IRT true score equating show that as sample sizes increase, the SEEs decrease although there might be fluctuations at different true score points. The fluctuations could come from sampling errors and possible convergence issues in item parameter estimation. This pattern is shown across the four lengths of the internal anchors considered in the study. Interestingly, the SEEs based on different proportions of the internal anchor items do not vary by much in terms of the magnitude. The findings above, however, are only based on one pair of old and new forms. More simulations are needed to reach conclusive findings. To achieve this goal, the power of SAS® to work with other software packages and programs makes it easier and more friendly.

EXTENDED READING

This research involves the comparison of SEEs for IRT true score equating based on the internal and external anchors. Simulations and resamplings are done using SAS® integrated with other software packages and programs. Wu, Y. F., & Li, D. (2012). The standard errors for IRT true score equating based on internal and external anchors. Paper presented at the 2012 NCME annual meeting, Vancouver, British Columbia, Canada.

REFERENCES

- Gagné, P., Furlow, C. & Ross, T. (2009). Increasing the number of replications in item response theory simulations: Automation through SAS and disk operating system. *Educational and Psychological Measurement, 69*, 79-84. DOI: 10.1177/0013164408324461
- Hanson, B. A., & Zeng, L. (1995a). ST: A Computer Program for IRT Scale Transformation (Version 2.0). [computer software]. Available at http://www.education.uiowa.edu/centers/casma/computer-programs.aspx#equating
- Hanson, B. A., & Zeng, L. (1995b). PIE: A Computer Program for IRT Equating (GUI Version). [computer software]. Available at http://www.education.uiowa.edu/centers/casma/computer-programs.aspx#equating
- Holland, P., & Dorans, N. (2006). Linking and equating. In R. L. Brennan (Ed.), *Educational Measurement* (4th ed., pp. 187–220). Westport, CT: American Council on Education and Praeger.
- Kolen, M. J., & Brennan, R. L. (2004). *Testing equating, scaling, and linking: Methods and practices* (2nd ed.). New York: Springer-Verlag.
- Lord, F. M. (1980). *Applications of item response theory to practical testing problems.* Hillsdale, NJ: Erlbaum Associates.
- SAS Institute Inc. (2009). *Base SAS® 9.2 Procedures Guide*. Cary, NC: SAS Institute Inc.
- SAS Institute Inc. (2008). SAS/IML user's guide, Version 9.1. Available at http://support.sas.com/91doc/docMainpage.jsp
- Stocking , M. L., & Lord, F. M. (1983). Developing a common metric in item response theory. *Applied Psychological Measurement, 7*, 201-210.
- von Davier, A. A., Holland, P. W., & Thayer, D. T. (2004). *The kernel method of test equating*. New York: Springer.
- Whittaker, T. A., Fitzpatrick, S. J., Williams, N. J., & Dodd, G. G.(2003). IRTGEN: A SAS macro program to generate known trait scores and item responses for commonly used item response theory models. *Applied Psychological Measurement, 27*, 299-300. DOI: 10.1177/0146621603027004005
- Zimowski, M., Muraki, E., Mislevy, R., & Bock, D. (2003). BILOG-MG (Version 3.0) [computer software]. Lincolnwood, IL: Scientific Software International.

ACKNOWLEDGMENTS

The author would like to thank Dr. Nathan Wall from eMetric, LLC, who generously shared some initial ideas of the ST and PIE codes.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Yi-Fang Wu
Educational Measurement and Statistics, University of Iowa
340 Lindquist Center
Iowa City, IA 52241
319-335-5721
yi-fang-wu@uiowa.edu

APPENDIX A: IRTGEN_MOD.SAS

```
/*********************************************************************************/
/*                                                                               */
/*   Modified from:                                                              */
/*   Whittaker, T.A., Fitzpatrick, S.J., Williams, N.J., & Dodd, G.G.(2003).     */
/*   IRTGEN: A SAS Macro Program to Generate Known Trait Scores                  */
/*   and Item Responses for Commonly Used Item Response Theory Models.           */
/*                                                                               */
/*********************************************************************************/
/*                                                                               */
/*   IRTGEN_MOD.SAS only includes the 3PL IRT model. C is set to 0 for 2PL model.  */
/*                                                                               */
/***************************MACRO IRTGENL3 BEGINS*********************************/
%MACRO IRTGENL3(MODEL=, DATA=, OUT=, NI=, NE=);
/***************************MACRO L3GEN***********************************/
    %MACRO L3GEN;
          EU=EXP(A*(THETA-B));
                P=C+((1-C)*(EU/(1+EU)));
                ARRAY R {&NUMITEM} R1-R&NUMITEM;
                     DO J = 1 TO &NUMITEM;
                            IF P GE RANUNI(-1) THEN R(J)=1;
                            ELSE R(J)=0;
                     END;
    %MEND L3GEN;
/***************************MACRO IRTGEN RESUMES*************************/
%LET FLAG=0;
       %IF %LENGTH(&MODEL)=0 %THEN %DO;
       %PUT;
       %PUT ***** ERROR ***** YOU MUST SPECIFY A MODEL *****;
       %PUT;
       %LET FLAG=1;
    %END;
%LET MODEL=%UPCASE(&MODEL);
    %IF  &MODEL=L3  %THEN %LET MDL=L3GEN;
    %ELSE %DO;
       %PUT;
       %PUT ***** ERROR IN MODEL SPECIFICATION: &MODEL *****;
       %PUT;
       %LET FLAG=1;
    %END;
    %IF %LENGTH(&NI)=0 OR &NI=0 %THEN %DO;
          %PUT;
          %PUT ***** ERROR ***** YOU MUST SPECIFY NUMBER OF ITEMS *****;
          %PUT;
          %LET FLAG=1;
    %END;
    %IF %LENGTH(&NE)=0 OR &NE=0 %THEN %DO;
          %PUT;
          %PUT ***** ERROR ***** YOU MUST SPECIFY NUMBER OF EXAMINEES *****;
          %PUT;
          %LET FLAG=1;
       %END;
       DATA &OUT;
          KEEP THETA R1-R&NI;
             SET THETA;
          DO J=1 TO &NI;
           SET &DATA POINT=J;
                %&MDL;
          END;
       RUN;
%MEND IRTGENL3;
```