Paper BtB - 11

Small Sample Equating: Best Practices using a SAS Macro

Anna M. Kurtz, North Carolina State University, Raleigh, NC
Andrew C. Dwyer, Castle Worldwide, Inc., Morrisville, NC

ABSTRACT

Test score equating, which is considered a critical component of test fairness, is most precise with large samples. Many small-scale testing programs (e.g., highly specialized licensure or certification exams), however, often face the reality of needing to equate when sample sizes fall well below recommended numbers. This study uses resampling methodology with data from a large national certification test to examine the accuracy of several small sample equating methods – mean, Tucker, synthetic, circle-arc, and nominal weights mean.

INTRODUCTION

For several reasons (e.g., test security, test validity), standardized testing programs must replace at least a portion of the test questions, or items, on a regular basis. Because the difficulty level of the new items may not be equivalent to the items being replaced, a statistical process known as test equating is used to be able to compare scores across different versions, or forms, of a test. Many testing programs use a common-item, nonequivalent groups (CINEG) design, where the set of common items across different forms are used to evaluate the relative ability of the examinees taking each form, and the set of unique items is used to determine the relative difficulty of each form. If the new form is found to be easier or harder than the previous form, scores can be adjusted accordingly to ensure the fair treatment of candidates.

Like other statistical procedures, however, common-item equating is subject to sampling error. Equating with large samples of examinees is one way to decrease the amount of sampling error introduced into the equated scores. Kolen and Brennan (2004), for example, recommend sample sizes of at least 400 for linear equating methods and at least 1,500 for equipercentile equating. Because test equating occurs after a new form of a test has been administered and because examinees need to be notified of their scores within a reasonable timeframe, in practice it is often difficult to obtain recommended sample sizes, especially for highly specialized testing programs that offer multiple test administration windows per year. Test equating with samples smaller than 100, and even very small samples below 25, is not unheard of in practice (Kim & Livingston, 2010; Livingston, 1993).

To ensure fairness to examinees it is important to use equating methods that reduce both random and systematic error. Several new equating methods for small samples have been proposed in recent years (Babcock, Albano, & Raymond, 2012; Kim, von Davier, & Haberman, 2008; Livingston & Kim, 2009), and this paper uses a SAS® macro to examine the performance of several equating methods under various sample sizes and differences in test form difficulty.

EQUATING METHODS

Five equating methods—Tucker, mean, synthetic, circle-arc, and nominal weights mean (NWM)—were compared to determine which method(s) produced the most accurate new form score estimates under various testing conditions. Tucker equating is a traditional equating method that assumes a linear relationship between old form scores and new form scores. Nominal weights mean equating, newly introduced, is a simplified linear equating method that does not require score variances or standard deviations to be estimated. Mean equating is a special case of Tucker equating where the slope of the linear component is constrained to be equal to one. Synthetic and circle-arc equating represent methods that have recently been introduced and are suggested for equating with small samples (Babcock, Albano, & Raymond, 2012; Kim et al., 2008; Livingston & Kim, 2009). See Table 1 for suggested sample sizes of these 5 methods.

TUCKER, MEAN, AND NOMINAL WEIGHTS MEAN EQUATING

Tucker, mean, and nominal weights mean equating are linear equating methods, meaning they assume a linear relationship between old form scores and new forms scores (Babcock et al., 2012; Kolen, 1985). Tucker equating makes two important assumptions in order to estimate the equating relationship between the old and new form. First, there is an assumption of equal regression slopes of the total test score on the anchor item set for both examinee populations. Second, there is an assumption of equal variance on the anchor item sets between both examinee populations. Several of the statistics needed to calculate the Tucker slope and intercept coefficients are difficult to estimate well with small samples.

Nominal weights mean (NWM) equating is similar to Tucker equating but does not require the estimation of total and anchor score variances or covariances. Instead, an assumption is made that the sets of total items and common items have similar statistical properties, and the ratio of the total number of items to the number of common items is used in the calculation of the NWM slope and intercept.

Mean equating is actually a special case of Tucker equating where the slope coefficient is assumed to be equal to 1. Mean equating makes the extra assumption that only the means differ between two testing populations and that the score difference between forms is constant along the score scale. Mean equating can be used with samples under 100 (Babcock et al., 2012; Kolen & Brennan, 2004) and even as small as 10 (Kim & Livingston, 2010); however, Tucker, NWM, and mean equating perform best when test forms are nearly identical in difficulty.

SYNTHETIC EQUATING

Synthetic equating is a hybrid equating method that uses a weighted average of identity equating (i.e., doing nothing) and another equating method of choice. Following a previously published example of synthetic equating, this study used chained linear equating (Kim et al., 2008). The use of identity equating alone leads to a large standard error of equating (SEE) as test forms differ or when equating samples differ, which is typical with small samples. Combining identity equating with another method reduces the SEE from identity equating alone by weighting the relationship with a value between 0 and 1. Equal weighting (0.5) was used in this study, which equally averages the relationship between identity and chained linear equating. Because chained linear equating is a linear method, synthetic equating, as implemented in this study, is also a linear equating method. Synthetic equating will be non-linear, however, if a non-linear method (e.g., circle-arc) is selected.

CIRCLE-ARC EQUATING

Circle-arc equating takes into account the non-linear relationship that can occur between forms and equates scores along the arc of a circle. The circle-arc is estimated using objective upper and lower limits of the test scores as the end points and the mid-point is empirically derived by equating at the middle of the score distribution (Livingston & Kim, 2011).For a full description of the circle-arc formulas please see Appendix A from Livingston and Kim (2011). In previous studies, circle-arc equating performed better than other traditional equating methods (Kim & Livingston, 2010; Livingston & Kim, 2009) for sample sizes between 10 and 100.

Equating Method	Minimum Sample Size Recommendations
Tucker	➢ 50-100 (Babcock, Raymond, & Albano, 2012)
Mean	➢ 50-100 (Babcock, Raymond, & Albano, 2012) ➢ 10-100 performed well at low score ranges (Kim & Livingston, 2010) ➢ ≤ 100 (Kolen & Brennan, 2004)
Synthetic	➢ 50-100 (Kim, von Davier, & Haberman, 2006) ➢ 100+ if form difficulty differs (Babcock, Raymond, & Albano, 2012)
Circle-Arc	➢ As small as 25 (Livingston & Kim, 2009) ➢ 10, 25, 50, & 100 (Kim & Livingston, 2010)
Nominal Weights Mean	➢ 20-80 (Babcock, Raymond, & Albano, 2012)

Table 1. Minimum Sample Size Recommendations for Various Equating Methods

METHODS

This study compares the five equating methods listed above under the common-item nonequivalent groups (CINEG) design, frequently referred to as the nonequivalent groups with anchor test (NEAT) design. The CINEG design uses a representative set of common items, or anchor items, that appear on both the old form (i.e., referent form) and the new form (Kolen & Brennan, 2004). The items from a large national certification test were split into several subsets of items representing different forms of the test. The items were selected in such a way as to create one pair of forms

that were equal in difficulty (0 point difference), one pair that differed slightly in difficulty (2 point difference), and one pair that differed moderately in difficulty (5 point difference). The five equating methods were examined using resampling methodology with small to extremely small samples. In addition, identity equating, which simply assumes test forms are equal in difficulty and makes no adjustment to the new form scores, was also added to the study to give context to the results from the 5 other equating methods.

DATA AND FORM BUILDING

The original certification exam form consisted of 125 items administered to 14,719 examinees. . From these 125 items, subsets of items were selected to create pairs of test forms. Each form contained a total of 50 unique items with 25 anchor items for a total of 75 items per test form. The anchor items were chosen based on content representativeness and covered all task domains of the entire test. The remaining 50 items were split based on coverage of task domains. See Figure 1 for a visual depiction of how the pseudo-forms were built from the original 125 item exam.

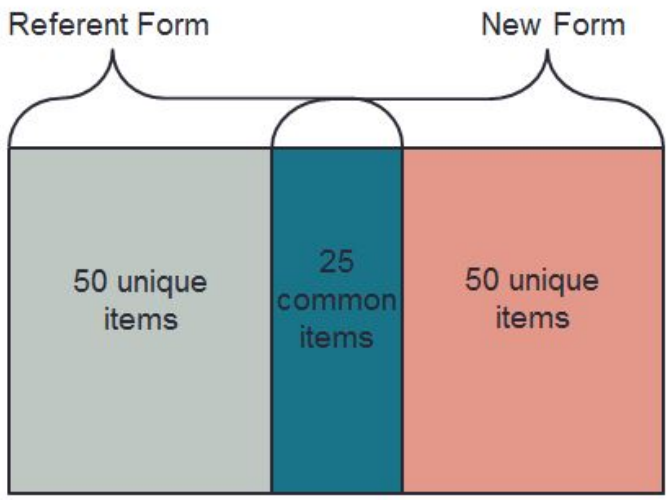

Figure 1. A Visual Depiction of Form Building

In order to examine the relationship between equating and form difficulty, multiple pairs of 75-item exams were created. Pair 0 contained forms of equal difficulty, pair 2 contained forms that differed by 2 points on the total score, and pair 5 contained forms that differed by a total of 5 points on the overall score. Pairs 2 and 5 were built such that the new form is more difficult than the referent form, a phenomenon that is typically observed as test forms change over time. Figure 2 depicts all study variables.

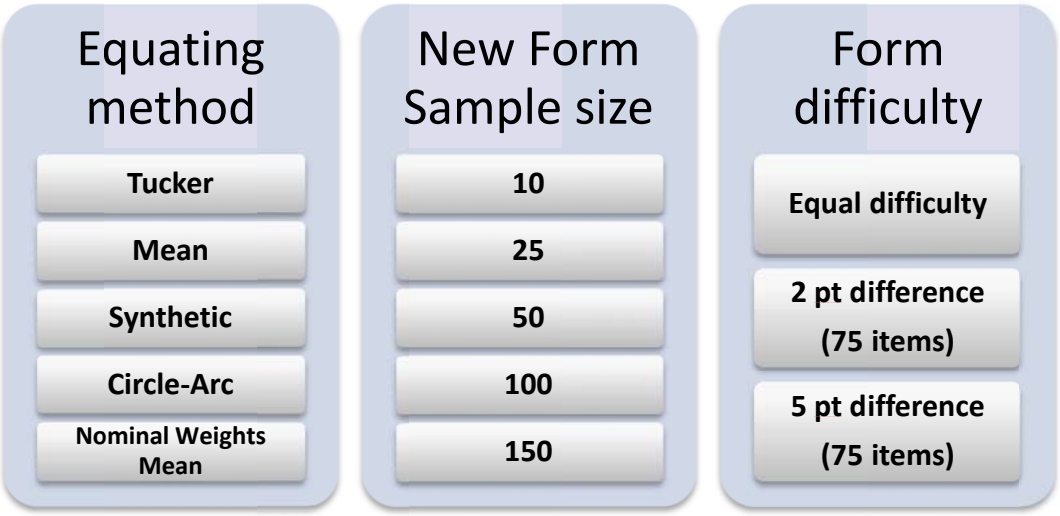

Figure 2. Equating methods and study variables

PROCEDURE

SAS 9.3 software was used to write an equating macro that runs Tucker, mean, NWM, circle-arc, and synthetic equating methods (see the Appendix for the full equating macro). The macro performed 1,000 iterations and used sampling without replacement for drawing the referent and new form samples. New samples were drawn for each iteration of the macro. The referent sample was always 175 examinees and the small samples for the new form were 10, 25, 50, 100, and 150 examinees. Root mean squared difference (RMSD) and bias of equated scores were calculated from the resulting data. These values indicate how close the equated score was to the "true" equating score. The "true" equating score was established by performing equipercentile equating using data from all 14,719 examinees. Because all items were administered to all examinees, it was possible to determine that a score of Y on the referent form was equivalent to a score of X on the new form for each pair of forms.

Macro Invocations for All Pair Difficulties at All Sample Sizes:

```
%EquatingSim(Rep=1000, Ny=175, Nx=25);
%EquatingSim(Rep=1000, Ny=175, Nx=50);
%EquatingSim(Rep=1000, Ny=175, Nx=100);
%EquatingSim(Rep=1000, Ny=175, Nx=10);
%EquatingSim(Rep=1000, Ny=175, Nx=150);
```

RESULTS

The results of this study indicate that for very small samples (i.e., below 50), circle-arc and synthetic equating slightly outperform the other methods, although the difference is minimal and depends on the size of the difference in test form difficulty and the distance of the cut score from the mean of the candidate score distribution. Furthermore, we propose that for well-behaved tests where the primary purpose is to make accurate pass/fail classifications of candidates, a sample size of 50 is sufficient for test equating. Figure 3 (following page) shows example output for RMSD estimates along the score scale for sample sizes of 10 and 100 at all difficulties.

CONCLUSION

Test equating with small samples can be successfully done in practice, though it is important to know how the new test form differs from the referent test form. Differences in difficulty, along with small sample sizes, introduce equating error and bias. Additionally, these equating methods perform at their best around the mean; however, many testing programs set a cut score well above the mean in order to have a highly selective test. When cut scores are above the mean, it is important to use an equating method that can account for differences in test difficulty and small sample sizes. Using a SAS macro to visualize and measure the practical impact of test equating is an added benefit for test developers.

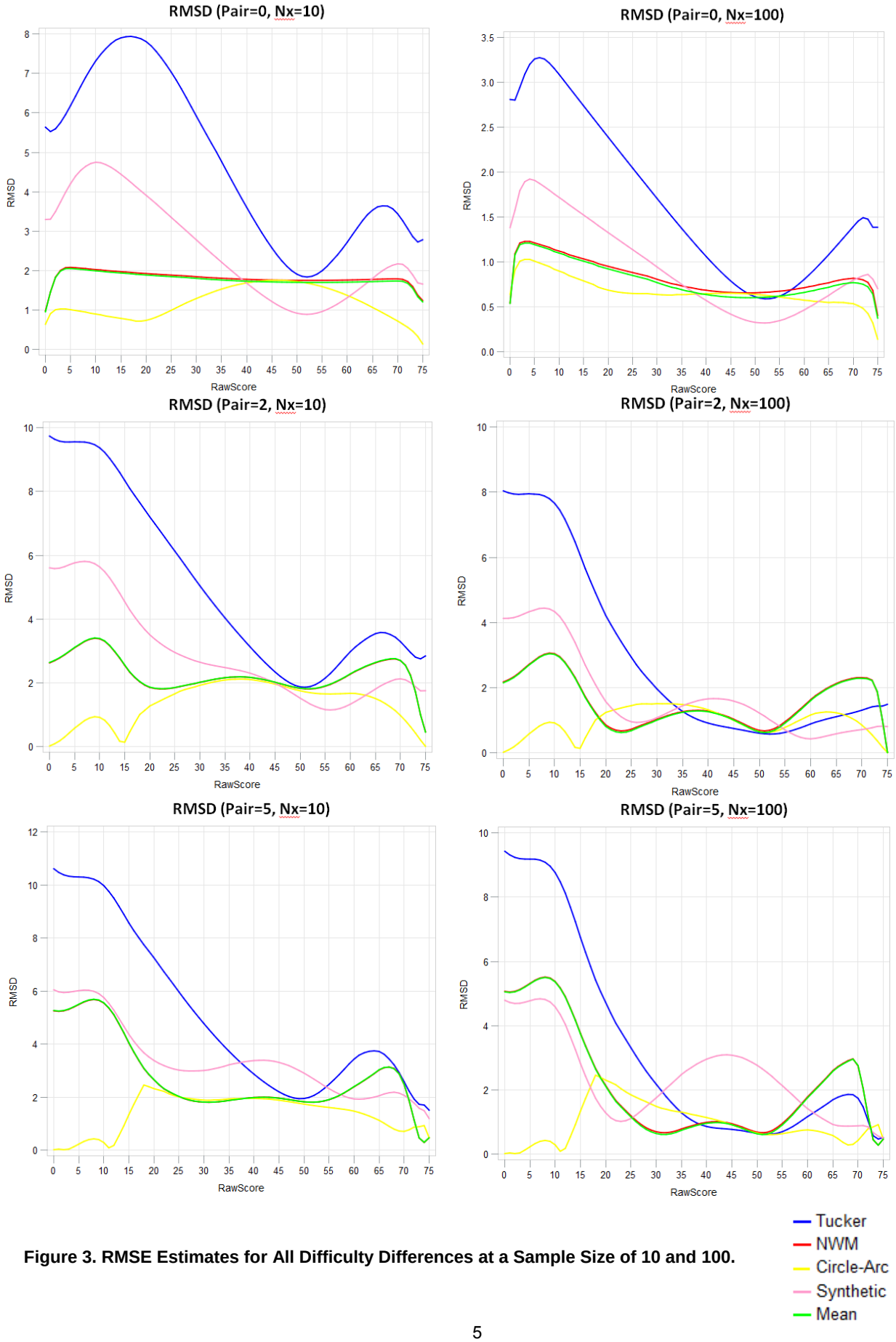

Figure 3. RMSE Estimates for All Difficulty Differences at a Sample Size of 10 and 100.

REFERENCES

Babcock, B., Albano, A., & Raymond, M. (2012). Nominal weights mean equating: A method for very small samples.*Educational And Psychological Measurement*, *72*(4), 608-628. doi:10.1177/0013164411428609

Hanson, B. A. (1996). Testing for differences in test score distributions using loglinear models. *Applied Measurement In Education*, *9*(4), 305-321. doi:10.1207/s15324818ame0904_2

Kim, S., & Livingston, S. A. (2010). Comparisons among small sample equating methods in a common-item design. *Journal Of Educational Measurement*, *47*(3), 286-298. doi:10.1111/j.1745-3984.2010.00114.xLivingston, 1993

Kim, S., von Davier, A. A., & Haberman, S. (2008). Small-sample equating using a synthetic linking function. *Journal Of Educational Measurement*, *45*(4), 325-342. doi:10.1111/j.1745-3984.2008.00068.xLivingston & Kim, 2009

Kolen, M. J. (1985). Standard errors of Tucker equating.*Applied Psychological Measurement*, *9*(2), 209-223. doi:10.1177/014662168500900209

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

Livingston, S. A., & Kim, S. (2009). The circle-arc method for equating in small samples. *Journal Of Educational Measurement*, *46*(3), 330-343. doi:10.1111/j.1745-3984.2009.00084.x

Livingston, S. A. (2011). New approaches to equating with small samples. In *Statistical models for test equating, scaling, and linking* (pp. 109-122). Springer Science.

ACKNOWLEDGMENTS

The first author would like to thank Castle Worldwide, Inc. for the opportunity to use SAS in an applied setting, Andrew Dwyer for being a mentor during her internship, and the SESUG Student Grants program.

CONTACT INFORMATION

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

Anna M. Kurtz
North Carolina State University
640 Poe Hall, 2310 Stinson Dr.
Raleigh, NC 27695-7650
(919) 601-2312 (main)
amkurtz@ncsu.edu

Andrew C. Dwyer, Ph.D., Psychometrician
Castle Worldwide, Inc.
900 Perimeter Park Dr., Suite G
Morrisville, NC 27560
919.572.6880 (main)
919.361.2426 (fax)
adwyer@castleworldwide.com

APPENDIX

Equating Macro

```
LIBNAME Data 'file extension';

*--------------------------*
*  Define Parameters       *
*--------------------------*;
%Let TotalItems=75;
%Let AnchorItems=25;
%Let Upper=&TotalItems;
%Let Choice=4;

*----------------------------------------------------------*
*  Write Log to External File (to not use too much memory)  *
*----------------------------------------------------------*;
PROC PRINTTO new log="file extension"; run;

*================================================*
*  Macro for:                                    *
*   - all 5 equating methods                     *
*   - all 3 pairs (0, 2, 5)                      *
*   - varying sample sizes                       *
*================================================*;
%Macro EquatingSim(Rep, Ny, Nx);

*------------------------------------------------------*
*  Delete Any Pre-existing Results dataset             *
*------------------------------------------------------*;
Proc Datasets Lib=Work Nolist;
   Delete AllResults_1(gennum=all);
Run;

*-------------------------------------*
*  Start i-loop                       *
*-------------------------------------*;
%Do i=1 %To &Rep;

*---------------------------------------------*
*  Sampling data for both Ref and New form    *
*---------------------------------------------*;
Proc SurveySelect Data=Data.Fulldata_scores Noprint Method=SRS
      Sampsize=&Ny Seed=0 Out=Select1 Outall;
Run;
Data Refdata; Set Select1;
      If Selected = 1;
Run;
Proc SurveySelect  Data=Select1 Noprint Method=SRS
      Sampsize=&Nx Seed=0 Out=Newdata;
      Where Selected=0;
Run;

*--------------------------*
*  Ref Form Summary Stats  *
*--------------------------*;
PROC MEANS DATA=Refdata NOPRINT;
```

```
      OUTPUT OUT=RefStats
      MEAN(Ref_0_Total Ref_2_Total Ref_5_Total Anchor)   = MY0   MY2   MY5
MVy
      STDDEV(Ref_0_Total Ref_2_Total Ref_5_Total Anchor) = SDY0  SDY2  SDY5
SDVy
      VAR(Ref_0_Total Ref_2_Total Ref_5_Total Anchor)    = VarY0 VarY2 VarY5
VarVy
      N=NY ;
RUN;
PROC CORR COV DATA=Refdata NOPRINT NOCORR NOSIMPLE OUT=RefCov;
      Var Anchor;
      With Ref_0_Total Ref_2_Total Ref_5_Total;
RUN;
DATA RefCov; SET RefCov;
      IF _TYPE_ = "COV";
RUN;
PROC Transpose Data=RefCov Out=RefCov;
      ID _NAME_;
Run;
DATA RefCov(Keep=RefCovYV0 RefCovYV2 RefCovYV5); SET RefCov;
      RefCovYV0 = Ref_0_Total;
      RefCovYV2 = Ref_2_Total;
      RefCovYV5 = Ref_5_Total;
RUN;

*--------------------------*
*  New Form Summary Stats  *
*--------------------------*;
PROC MEANS DATA=Newdata NOPRINT;
      OUTPUT OUT=NewStats
      MEAN(New_0_Total New_2_Total New_5_Total Anchor)   = MX0   MX2   MX5
MVx
      STDDEV(New_0_Total New_2_Total New_5_Total Anchor) = SDX0  SDX2  SDX5
SDVx
      VAR(New_0_Total New_2_Total New_5_Total Anchor)    = VarX0 VarX2 VarX5
VarVx
      N=NX ;
RUN;
PROC CORR COV DATA=Newdata NOPRINT NOCORR NOSIMPLE OUT=NewCov;
      Var Anchor;
      With New_0_Total New_2_Total New_5_Total;
RUN;
DATA NewCov; SET NewCov;
      IF _TYPE_ = "COV";
RUN;
PROC Transpose Data=NewCov Out=NewCov;
      ID _NAME_;
Run;
DATA NewCov(Keep=NewCovXV0 NewCovXV2 NewCovXV5); SET NewCov;
      NewCovXV0 = New_0_Total;
      NewCovXV2 = New_2_Total;
      NewCovXV5 = New_5_Total;
RUN;

*----------------------*
*  All Summary Stats  *
*----------------------*;
```

```
DATA AllStats;
      MERGE RefStats RefCov NewStats NewCov;
      Drop _TYPE_ _FREQ_;
RUN;

*--------------------*
*  Tucker Equating  *
*--------------------*;
%Macro Tucker(J);
      Data Tucker_&j; Set Allstats;
            GammaX&j=(NewCovXV&j)/(VarVx);
            GammaY&j=(RefCovYV&j)/(VarVy);
            wX=(NX)/(NX+NY);
            wY=(NY)/(NX+NY);
            MXs&j=(MX&j)-(wY*GammaX&j)*(MVx-MVy);
            MYs&j=(MY&j)+(wX*GammaY&j)*(MVx-MVy);
            VarXs&j=(VarX&j)-(wY*GammaX&j**2)*(VarVx-
VarVy)+(wX*wY*GammaX&j**2)*((MVx-MVy)**2);
            VarYs&j=(VarY&j)+(wX*GammaY&j**2)*(VarVx-
VarVy)+(wX*wY*GammaY&j**2)*((MVx-MVy)**2);
            Slope&j=(SQRT(VarYs&j))/(SQRT(VarXs&j));
            Int&j=(MYs&j)-(MXs&j*Slope&j);
      Run;
      Data TuckerTable_&j(Keep=RawScore TuckerScore_&j); Set Tucker_&j;
            Do RawScore=0 To &Upper;
                  TuckerScore_&j=Round((Slope&j*RawScore)+(Int&j),.01);
                  If TuckerScore_&j<0 Then TuckerScore_&j=0;
                  If TuckerScore_&j>&Upper Then TuckerScore_&j=&Upper;
                  Output;
            End;
      Run;
      Proc SQL; Drop Table Tucker_&j; Quit;
%Mend;

*----------------*
*  NWM Equating  *
*----------------*;
%Macro NWM(J);
      Data NWM_&j; Set Allstats;
            Ratio=(&Upper/&AnchorItems);
            wX=(NX)/(NX+NY);
            wY=(NY)/(NX+NY);
            MXs&j=(MX&j)-(wY*Ratio)*(MVx-MVy);
            MYs&j=(MY&j)+(wX*Ratio)*(MVx-MVy);
      Run;
      Data NWMTable_&j(Keep=RawScore NWMScore_&j); Set NWM_&j;
            Do RawScore=0 To &Upper;
                  NWMScore_&j=Round((RawScore-MXs&j+MYs&j),.01);
                  If NWMScore_&j<0 Then NWMScore_&j=0;
                  If NWMScore_&j>&Upper Then NWMScore_&j=&Upper;
                  Output;
            End;
      Run;
      Proc SQL; Drop Table NWM_&j; Quit;
%Mend;

*----------------------*
```

```
*  Circle-Arc Equating  *
*-----------------------*;
%Macro Circle(J);
  %Let Lower=%Eval(&Upper/&Choice);
      Data Circle_&j; Set Allstats;
            MidXt&j=MX&j;
            MidY&j=MY&j+(SDY&j/SDVy)*(MVx-
MVy)+(SDY&j/SDVy)*(SDVx/SDX&j)*(MidXt&j-MX&j);
            MidYt&j=MidY&j-MidXt&j;
            LowYt=&Lower-&Lower;
            LowXt=&Lower;
            UpperYt=&Upper-&Upper;
            UpperXt=&Upper;
            CirMidX=((UpperXt**2)-(LowXt**2))/(2*(UpperXt-LowXt));
            CirMidY&j=((LowXt**2)*(UpperXt-MidXt&j)-
((MidXt&j**2)+(MidYt&j**2))*(UpperXt-LowXt)+(UpperXt**2)*(MidXt&j-
LowXt))/(2*(MidYt&j*(LowXt-UpperXt)));
            Radius&j=SQRT((CirMidX-LowXt)**2+(CirMidY&j**2));
      Run;
      DATA CircleTable_&j(Keep=RawScore CircleScore_&j); Set Circle_&j;
            Do RawScore=0 To &Upper;
                  If MidYt&j>0 Then
                        Do;
                              If RawScore<=&Lower Then
CircleScore_&j=RawScore;
                              IF RawScore>&Lower  Then
CircleScore_&j=Round(RawScore+CirMidY&j+SQRT(Radius&j**2-(RawScore-
CirMidX)**2),.01);
                        End;
                  If MidYt&j<=0 Then
                        Do;
                              If RawScore<=&Lower Then
CircleScore_&j=RawScore;
                              If RawScore>&Lower  Then
CircleScore_&j=Round(RawScore+CirMidY&j-SQRT(Radius&j**2-(RawScore-
CirMidX)**2),.01);
                        End;
                  Output;
            End;
      Run;
      Proc SQL; Drop Table Circle_&j; Quit;
%Mend;

*-----------------------------------------------------*
*  Synthetic Equating using Chained Linear Equating  *
*-----------------------------------------------------*;
%Macro Syn(J);
      Data SynTable_&j(Keep=RawScore SynScore_&j); Set Allstats;
            Do RawScore=0 To &Upper;
                  SynScore_&j=Round(.5*(MY&j+(SDY&j/SDVy)*(MVx-
MVy)+(SDY&j/SDVy)*(SDVx/SDX&j)*(RawScore-MX&j))+.5*RawScore,.01);
                  If SynScore_&j<0 Then SynScore_&j=0;
                  If SynScore_&j>&Upper Then SynScore_&j=&Upper;
                  Output;
            End;
      Run;
%Mend;
```

```
*-----------------*
*  Mean Equating  *
*-----------------*;
%Macro MeanEq(J);
      Data MeanEq_&j; Set Allstats;
            GammaX&j=(NewCovXV&j)/(VarVx);
            GammaY&j=(RefCovYV&j)/(VarVy);
            wX=(NX)/(NX+NY);
            wY=(NY)/(NX+NY);
            MXs&j=(MX&j)-(wY*GammaX&j)*(MVx-MVy);
            MYs&j=(MY&j)+(wX*GammaY&j)*(MVx-MVy);
            ScaleFactor=(&Upper/&Upper);
            Slope=1*ScaleFactor;
            Intercept&j=MYs&j-ScaleFactor*MXs&j;
      Run;
      Data MeanTable_&j(Keep=RawScore MeanEqScore_&j); SET MeanEq_&j;
            Do RawScore=0 to &Upper;
                  MeanEqScore_&j=Round((Slope*RawScore)+(Intercept&j),.01);
                  If MeanEqScore_&j<0 Then MeanEqScore_&j=0;
            If MeanEqScore_&j>&Upper Then MeanEqScore_&j=&Upper;
            Output;
            End;
      Run;
      Proc SQL; Drop Table MeanEq_&j; Quit;
%Mend;

*-----------------------------------------------*
*  Execute all equating methods for all pairs   *
*-----------------------------------------------*;
%Tucker(0);
%Tucker(2);
%Tucker(5);
%NWM(0);
%NWM(2);
%NWM(5);
%Circle(0);
%Circle(2);
%Circle(5);
%Syn(0);
%Syn(2);
%Syn(5);
%MeanEq(0);
%MeanEq(2);
%MeanEq(5);

*-----------------------------------------------*
*  Merging Outputs from All 5 Equating Methods  *
*-----------------------------------------------*;
Data AllResults_&i;
      Merge TuckerTable_0 TuckerTable_2 TuckerTable_5
            NWMTable_0 NWMTable_2 NWMTable_5
            CircleTable_0 CircleTable_2 CircleTable_5
            SynTable_0 SynTable_2 SynTable_5
            MeanTable_0 MeanTable_2 MeanTable_5;
      By RawScore;
      Rep=&i;
```

```
Run;

*----------------------------*
*  Appending Data from Reps  *
*----------------------------*;
  %If &i>1 %Then
      %Do;
            Proc Append Base=AllResults_1 Data=AllResults_&i; Run;
            Proc SQL; Drop Table AllResults_&i; Quit;
      %End;

*-------------------------------------*
*  End i-loop                         *
*-------------------------------------*;
%End;

*-----------------------*
*  Addding True Scores  *
*-----------------------*;
Proc SQL; Create Table AllResults_&Ny._&Nx As
      Select t1.*, True_0, True_2, True_5
      From AllResults_1 t1, Data.Trueequating_allpairs t2
      Where t1.RawScore=t2.RawScore;
      Drop Table AllResults_1;
Quit;

Data AllResults_&Ny._&Nx; Set AllResults_&Ny._&Nx;
  Ny = &Ny;
  Nx = &Nx;
Run;

*-------------------------------------*
*  End macro                          *
*-------------------------------------*;
%Mend EquatingSim;

*---------------------------------------------*
*  Invoke macro for various sample sizes      *
*---------------------------------------------*;
%EquatingSim(Rep=500, Ny=175, Nx=25);
%EquatingSim(Rep=500, Ny=175, Nx=50);
%EquatingSim(Rep=500, Ny=175, Nx=100);
%EquatingSim(Rep=500, Ny=175, Nx=10);
%EquatingSim(Rep=500, Ny=175, Nx=150);

*---------------------------------------------*
*  Combine all results                        *
*---------------------------------------------*;
Data Data.AllResults;
  Set AllResults_175_100
      AllResults_175_10
      AllResults_175_25
      AllResults_175_50
      AllResults_175_150;
Run;

*-------------------------------------------------------------------------------*
```

```
* Compute Differences and Squared Differences between Estimates and Truth  *
*-------------------------------------------------------------------------*;
%Macro Diff(J);
  Data Data.AllResults; Set Data.AllResults;
    TuckerDiff_&j = TuckerScore_&j - True_&j;   TuckerDiff2_&j =
(TuckerScore_&j - True_&j)**2;
    NWMDiff_&j    = NWMScore_&j    - True_&j;   NWMDiff2_&j    = (NWMScore_&j
- True_&j)**2;
    CircleDiff_&j = CircleScore_&j - True_&j;   CircleDiff2_&j =
(CircleScore_&j - True_&j)**2;
    SynDiff_&j    = SynScore_&j    - True_&j;   SynDiff2_&j    = (SynScore_&j
- True_&j)**2;
    MeanEqDiff_&j = MeanEqScore_&j - True_&j;   MeanEqDiff2_&j =
(MeanEqScore_&j - True_&j)**2;
  Run;
%Mend;
%Diff(0);
%Diff(2);
%Diff(5);

*-------------------------------------------------------------*
* Compute RMSD at each score point                            *
*-------------------------------------------------------------*;
Proc Sort Data=Data.AllResults;
 By RawScore Nx;
Run;
%Macro RMSD(J);
  Proc Means Data=Data.AllResults Noprint;
      Output Out=RMSD_&j
      Sum(TuckerDiff_&j  NWMDiff_&j  CircleDiff_&j  SynDiff_&j  MeanEqDiff_&j
        TuckerDiff2_&j NWMDiff2_&j CircleDiff2_&j SynDiff2_&j MeanEqDiff2_&j)
      = TuckerSum_&j   NWMSum_&j   CircleSum_&j   SynSum_&j   MeanEqSum_&j
        TuckerSum2_&j  NWMSum2_&j  CircleSum2_&j  SynSum2_&j  MeanEqSum2_&j;
      By RawScore Nx;
  Run;
  Data RMSD_&j; Set RMSD_&j;
    TuckerRMSD_&j = Sqrt(TuckerSum2_&j/_FREQ_);
      NwmRMSD_&j    = Sqrt(NWMSum2_&j/_FREQ_);
      CircleRMSD_&j = Sqrt(CircleSum2_&j/_FREQ_);
      SynRMSD_&j    = Sqrt(SynSum2_&j/_FREQ_);
      MeanEqRMSD_&j = Sqrt(MeanEqSum2_&j/_FREQ_);
    TuckerBias_&j = TuckerSum_&j/_FREQ_;
      NwmBias_&j    = NWMSum_&j/_FREQ_;
      CircleBias_&j = CircleSum_&j/_FREQ_;
      SynBias_&j    = SynSum_&j/_FREQ_;
      MeanEqBias_&j = MeanEqSum_&j/_FREQ_;
  Run;

%Mend;
%RMSD(0);
%RMSD(2);
%RMSD(5);

*-------------------------------------------------------------*
* Combine RMSD files                                          *
*-------------------------------------------------------------*;
Data Data.RMSD;
```

```
 Merge RMSD_0 RMSD_2 RMSD_5;
 By RawScore Nx;
Run;
```