Statistical Equating Methods

Anthony Albano

March 25, 2010

Abstract

The R package equate (Albano, 2010) contains functions for non-IRT equating under random groups and nonequivalent groups designs. This package vignette introduces these designs and provides an overview of statistical equating with details about each of the supported equating methods. Examples demonstrate the basic functionality of the package.

1 Introduction

Equating is a statistical procedure commonly used in testing programs where administrations across more than one occasion and more than one examinee group can lead to overexposure of items, threatening the security of the test. Another somewhat less common use is in progress monitoring and growth modeling, where administrations occur across multiple time points for the same individuals, and using the same test form is expected to lead to practice effects. In each of these contexts item exposure can be controlled by using alternate test forms; however, these multiple forms lead to multiple score scales for a single test. Despite being designed based on the same specifications, to cover the same content, at the same level of difficulty, these score scales and alternate forms are not identical. Instead, one is likely more difficult than the other. In this case ability differences for examinees taking different forms are confounded by differences in form difficulty. Equating methods can be used to adjust for differences in difficulty across alternate forms, resulting in comparable score scales and more accurate estimates of ability.

The equate package focuses on statistical equating, as opposed to item response theory (IRT) equating (Weeks, 2009, provides a comprehensive R package for IRT equating). Most of the procedures fit under what is called *traditional equating*, but they are more appropriately referred to as non-IRT equating methods. This distinction importantly implies that IRT equating is based on a measurement model (the IRT model, of course) whereas most

traditional methods are not. Although there are many benefits of equating forms using IRT, non-IRT equating can often be a simpler and more practical alternative, one which involves fewer and less demanding assumptions (for further discussion see Kolen & Brennan, 2004; Livingston, 2004).

Statistical equating defines a functional relationship between multiple test score distributions and thereby between multiple score scales. When the test forms have been created according to the same specifications and are similar in statistical characteristics, this functional relationship is referred to as an *equating function* and it serves to translate scores from one scale directly to their equivalent values on another. Whether score distributions are based on samples from a single examinee population or different examinee populations (these are referred to as equating designs, as discussed below), if the appropriate assumptions are met the equating function can be generalized to other examinees (for a detailed discussion see Holland & Dorans, 2006).

2 Equating Designs

An *equating design* refers to the basic structure of an equating study, just as a research design refers to the structure of a research study. The equating study serves to organize all the stages which are essential to and which lead up to the equating process. These stages include creation of test forms, sampling of examinees, and administration of the test. The equating design specifies the administration (i.e., data collection) procedures, and just as the control of variables in a research study depends on design, control of examinee ability (in contrast with form difficulty) depends on the equating design (Holland & Dorans, 2006).

An equating study can take place in a variety of situations, depending on the needs and resources of a testing program. As a result, numerous equating designs have been documented in the literature (Kolen & Brennan, 2004). For simplicity, in this vignette and in the `equate` package, equating designs are categorized as either involving *equivalent groups* or *nonequivalent groups*.

Equivalent Groups

The equivalent groups design consists of either a single group of examinees taking both forms of a test, or two groups sampled randomly from a single population and considered to be randomly equivalent. In either case it is assumed that the two groups are equivalent in ability, thus any differences in scores across forms can be attributed entirely to form difficulty. When forms are administered to a single group administration procedures can be

complicated by order and fatigue effects. Thus the single group design is often not a practical option. Otherwise, because it involves only the examinee population of interest (called the target population), the equivalent groups design is the most efficient, as examinee ability is controlled directly.

Nonequivalent Groups

Without equivalent examinee groups two related problems arise: the target population must be defined indirectly using samples from two different examinee populations, and the ability of these groups must then be controlled. In the nonequivalent groups design[1] these obstacles are both overcome through the use of what is referred to as an *anchor test*, a set of items appearing on both test forms. All non-equivalence is assumed to be removed via these anchor, or common, items. Though this design is often more practical, as nonequivalent groups are more easily obtained than equivalent ones, it also involves additional assumptions, as discussed in the next section (for details see Holland & Dorans, 2006).

As noted above, the equivalent groups is the simpler equating design. The traditional equating types applied with this design are the mean, linear, and equipercentile. More complex extensions of these have been developed for use with the nonequivalent groups design, each of which handles the issues inherent with nonequivalent groups in a slightly different way. These methods are described briefly below, followed by examples of their implementation in the equate package.

3 Equating Types and Methods

Types of Equating

Equatings with the equivalent groups design, that is, equatings in their simplest and most general form, are referred to here and in the equate package as equating *types*. These can be categorized as either linear, including mean and linear equating, or nonlinear, equipercentile equating. An additional nonlinear type supported in the equate package is circle-arc equating, as introduced by Livingston and Kim (2009).

[1]The nonequivalent groups design is also referred to as the nonequivalent groups with anchor test design, the common-item nonequivalent groups design, or simply the anchor- or common-item design.

3.0.1 *Identity equating

The identity equating function simply reproduces the original score value unchanged, and thus un-equated:

$$id_Y(x_i) = x_i. \tag{1}$$

With small samples, and when test forms are believed to be parallel, identity equating, or no equating, has been recommended over other types (Kolen & Brennan, 2004). The identity function can also be combined with any of the functions described below to obtain the synthetic equating function (Kim, von Davier, & Haberman, 2008):

$$s_Y(x_i) = (w_I - 1)g_Y(x_i) + w_I id_Y(x_i), \tag{2}$$

where $s_Y(x_i)$ is a weighted combination of the general equating function $g_Y(x_i)$ and the identity, and w_I is a value between zero and one.

Linear equating

Linear equating defines a linear relationship between scores from forms X and Y, based on the mean and standard deviation of each. In other words, the standardized deviation scores, or z-scores, are set equal for all score points i:

$$\frac{x_i - \hat{\mu}(X)}{\hat{\sigma}(X)} = \frac{y_i - \hat{\mu}(Y)}{\hat{\sigma}(Y)}. \tag{3}$$

When solved for y_i, the linear function $l_Y(x_i)$ can be rewritten in slope-intercept form as

$$l_Y(x_i) = \frac{\hat{\sigma}(Y)}{\hat{\sigma}(X)}x_i - \frac{\hat{\sigma}(Y)}{\hat{\sigma}(X)}\hat{\mu}(X) + \hat{\mu}(Y). \tag{4}$$

Mean equating

Mean equating is a simplification of linear where the slope, or ratio of standard deviations, is not estimated but is instead assumed to be 1. Deviation scores across forms are thus set equal:

$$x_i - \hat{\mu}(X) = y_i - \hat{\mu}(Y), \tag{5}$$

and the resulting mean function $m_Y(x_i)$ for equating X to Y is

$$m_Y(x_i) = x_i - \hat{\mu}(X) + \hat{\mu}(Y). \tag{6}$$

Equipercentile equating

Equipercentile equating defines a nonlinear relationship between score scales by setting equal the percentile ranks for each score point. Specifically, the equipercentile equivalent of a form-X score on the Y scale is calculated by finding the percentile rank in X of score i, and then the form-Y score associated with that form-Y percentile rank:

$$e_Y(x_i) = Q^{-1}[P(x_i)]. \tag{7}$$

Here, $P(x)$ is the percentile rank function in X and $Q^{-1}(x)$ is the inverse percentile rank function in Y. The process is complicated by the fact that scores are discrete, and must be made continuous (for a detailed description see Kolen & Brennan, 2004, ch. 2).

Because it involves estimation at each score point, equipercentile equating is especially susceptible to random sampling error. Smoothing methods are typically used to reduce irregularities in either the score distributions or the equating function itself. Two commonly used smoothing methods include polynomial loglinear presmoothing (Holland & Thayer, 2000) and cubic-spline postsmoothing (Kolen, 1984). The equate package currently supports loglinear presmoothing (see Appendix A.2 for details).

Circle-arc equating

Circle-arc equating also defines a nonlinear relationship between score scales, but it requires the estimation of only three points for forms X and Y: the lowest meaningful score (x_1, y_1) which for a multiple-choice test could be the lowest score expected by chance; a midpoint, based on the center (e.g., means) of each form (x_2, y_2); and the maximum possible score on each form (x_3, y_3). The low and high points define the linear component of the function:

$$lin_Y(x_i) = y_1 + \frac{y_3 - y_1}{x_3 - x_1}(x_i - x_1). \tag{8}$$

This linear function is combined with a curvilinear one, a circle-arc that is based on y_{2*}, the distance in Y units of the point (x_2, y_2) from the line $lin_Y(x)$. The center (x_c, y_c) and radius r of the circle define the curvilinear component:

$$arc_Y(x_i) = y_c \pm \sqrt{r^2 - (x_i - x_c)^2}, \tag{9}$$

where the second quantity, under the square root, is added to y_c if y_{2*} is positive (i.e., above the linear function) and subtracted if it is negative (i.e., below the linear function). The

circle-arc function $c_Y(x_i)$ combines the linear and curvilinear components:

$$c_Y(x_i) = lin_y(x_i) + arc_y(x_i). \tag{10}$$

Equations for the center points and radius of the circle are included in Appendix A.3. Livingston and Kim (2009) provide a complete description of the process using helpful graphics.

Equating Methods

The nonequivalent groups design requires that information from anchor items be incorporated into the functions and parameter estimation described above. This is necessary because two populations are involved in the nonequivalent groups design: population 1 taking form X, and 2 taking form Y; however, the equating function itself will be defined for a single population. Since this population is only a hypothetical one, that is, no data exist for it, it is referred to as the *synthetic population* (Braun & Holland, 1982). As described by Kolen and Brennan (2004), the linear equating function from equation (4) can be rewritten in terms of the synthetic population as follows:

$$l_{Y_S}(x_i) = \frac{\hat{\sigma}_S(Y)}{\hat{\sigma}_S(X)}x_i - \frac{\hat{\sigma}_S(Y)}{\hat{\sigma}_S(X)}\hat{\mu}_S(X) + \hat{\mu}_S(Y). \tag{11}$$

Since population S did not take forms X or Y, all of the terms $\hat{\mu}_S$ and $\hat{\sigma}_S$ in this equation must be estimated indirectly using: for the means,

$$\hat{\mu}_S(X) = \hat{\mu}_1(X) - w_2\gamma_1[\hat{\mu}_1(V) - \hat{\mu}_2(V)], \tag{12}$$

$$\hat{\mu}_S(Y) = \hat{\mu}_2(Y) + w_1\gamma_2[\hat{\mu}_1(V) - \hat{\mu}_2(V)]; \tag{13}$$

and for the variances,

$$\hat{\sigma}^2_S(X) = \hat{\sigma}^2_1(X) - w_2\gamma^2_1[\hat{\sigma}^2_1(V) - \hat{\sigma}^2_2(V)] + w_1w_2\gamma^2_1[\hat{\mu}_1(V) - \hat{\mu}_2(V)]^2, \tag{14}$$

$$\hat{\sigma}^2_S(Y) = \hat{\sigma}^2_2(Y) + w_1\gamma^2_2[\hat{\sigma}^2_1(V) - \hat{\sigma}^2_2(V)] + w_1w_2\gamma^2_2[\hat{\mu}_1(V) - \hat{\mu}_2(V)]^2. \tag{15}$$

In these equations the weights w_1 and w_2 sum to 1, and are used to specify the desired influence of populations 1 and 2 in the estimation. The γ terms represent the relationship between total scores on X and Y and the respective anchor test scores on V (described further below). As is clear, γ_1 and γ_2 are used along with the weights to adjust the $\hat{\mu}$ and $\hat{\sigma}^2$ terms for X and Y in order to obtain corresponding estimates for the synthetic population.

For example, setting $w_1 = 0$ and $w_2 = 1$ will force $\hat{\mu}_S(Y)$ to equal $\hat{\mu}_2(Y)$, and conversely $\hat{\mu}_2(X)$ will be adjusted the maximum amount to obtain $\hat{\mu}_S(X)$. The same would occur with the estimation of synthetic variances. Furthermore, the adjustments would be completely removed if $\hat{\mu}_1(V) = \hat{\mu}_2(V)$ and $\hat{\sigma}_1^2(V) = \hat{\sigma}_2^2(V)$.

A variety of techniques have been developed for estimating the γ terms required by equations (12)-(15) above. These techniques are referred to here as equating *methods*. The `equate` package currently supports the Tucker, Levine observed score, Levine true score, Braun/Holland, frequency estimation, and chained equating methods (Kolen & Brennan, 2004, provide a full explanation of the assumptions related to each method, including derivations). Table 1 shows the supported methods that apply to each equating type.

Tucker equating

In Tucker equating the relationship between total and anchor test scores is defined in terms of regression slopes, where γ_1 is the slope resulting from the regression of X on V for population 1, and γ_2 the slope from a regression of Y on V for population 2:

$$\gamma_1 = \frac{\hat{\sigma}_1(X,V)}{\hat{\sigma}_1^2(V)} \qquad \text{and} \qquad \gamma_2 = \frac{\hat{\sigma}_2(Y,V)}{\hat{\sigma}_2^2(V)}. \tag{16}$$

The Tucker method assumes that across populations 1 and 2: 1) the coefficients resulting form a regression of X on V are the same, and 2) the conditional variance of X given V is the same. These assumptions apply to the regression of Y on V and the covariance of Y given V as well.

Table 1: Applicable Equating Types and Methods

	`nominal`	`tucker`	`levine`	`braun`	`frequency`	`chained`
`mean`	√	√	√	√		√
`linear`		√	√	√		√
`equipercentile`					√	√
`circle-arc`	√	√	√	√		√

Nominal weights equating

Nominal weights equating is a simplified version of the Tucker method where the total and anchor tests are assumed to have similar statistical properties and to correlate perfectly

within populations 1 and 2. In this case the γ terms can be approximated by the ratios

$$\gamma_1 = \frac{K(X)}{K(V)} \qquad \text{and} \qquad \gamma_2 = \frac{K(Y)}{K(V)}, \tag{17}$$

where K is the number of items on the test.

Levine equating

Assumptions for the Levine observed score method are stated in terms of true scores (though only observed scores are used), where, across both populations: 1) the correlation between true scores on X and V is 1, as is the correlation between true scores on Y and V; 2) the coefficients resulting form a regression of true scores for X on V are the same, as with true scores for Y on V; and 3) measurement error variance is the same (across populations) for X, Y, and V. These assumptions make possible the estimation of γ as

$$\gamma_1 = \frac{\hat{\sigma}_1^2(X)}{\hat{\sigma}_1(X,V)} \qquad \text{and} \qquad \gamma_2 = \frac{\hat{\sigma}_2^2(Y)}{\hat{\sigma}_2(Y,V)}, \tag{18}$$

which are the inverses of the respective regression slopes for V on X and V on Y. The Levine true score method is based on the same assumptions as the observed score method; however, it uses a slightly different linear equating function:

$$l_Y(x_i) = \frac{\gamma_2}{\gamma_1}(X)[x_i - \hat{\mu}_1(X)] + \hat{\mu}_2(Y) + \gamma_2[\hat{\mu}_1(V) - \hat{\mu}_2(V)]. \tag{19}$$

Hanson (1991) and Kolen and Brennan (2004) provide justifications for using this approach.

Frequency estimation equating

The frequency estimation method is used in equipercentile equating under the nonequivalent groups design. It is similar to the methods described above in that it involves a synthetic population. However, in this case score distributions (i.e., percentile ranks) for the synthetic population taking forms X and Y are required:

$$e_{Y_S}(x_i) = Q_S^{-1}[P_S(x_i)]. \tag{20}$$

When the assumption is made that the conditional distribution of total scores on X for a given score point in V is the same across populations 1 and 2 (as with Y and V) the synthetic

distributions can be obtained:

$$f_S(x_i) = w_1 f_1(x_i) + w_2 \sum f_1(x|v) h_2(v), \tag{21}$$

$$g_S(y_i) = w_2 g_2(y_i) + w_1 \sum g_2(y|v) h_1(v) \tag{22}$$

Here, f, g, and h denote the distribution functions for forms X, Y, and V respectively. As before, w_1 and w_2 specify the amount of adjustment to be made to each observed distribution in the estimation of the corresponding synthetic distribution.

Braun/Holland equating

As a kind of extension of the frequency estimation method, the Braun/Holland method defines a linear function relating X and Y that is based on the estimates $\hat{\mu}_S(X)$, $\hat{\mu}_S(Y)$, $\hat{\sigma}_S(X)$, and $\hat{\sigma}_S(Y)$ for the synthetic distributions $f_S(x)$ and $g_S(y)$ obtained via frequency estimation. Thus the full synthetic distributions are estimated, as with frequency estimation, but only in order to obtain the means and standard deviations of each. Though not often used in practice, the method provides an interesting combination of the linear and nonlinear procedures (Braun & Holland, 1982).

Chained equating

Finally, chained equating (Livingston, Dorans, & Wright, 1990) can be applied to both linear and equipercentile equating under the nonequivalent groups with anchor test design. It differs from all other methods discussed here in that it does not reference a synthetic population. Instead, it introduces an additional equating function in the process of estimating score equivalents (see Appendix A.1 for details). For both linear and equipercentile equating the steps are as follows:

1. Define the function relating X to V for population 1, $l_{V1}(x)$ or $e_{V1}(x)$
2. Define the function relating V to Y for population 2, $l_{Y2}(v)$ or $e_{Y2}(v)$
3. Equate X (population 1) to the scale of Y using both equating functions, where

$$lchain_Y(x) = l_{Y2}[l_{V1}(x)] \qquad \text{and} \qquad echain_Y(x) = e_{Y2}[e_{V1}(x)]$$

Methods for circle-arc equating

As discussed above, the circle-arc equating function combines a linear with a curvilinear component based on three points in the X and Y score distributions. The first and third

of these points are determined by the score scale, whereas the midpoint must be estimated. Thus, equating methods used with circle-arc equating apply only to estimation of this midpoint. Livingston and Kim (2009) demonstrate chained linear equating of means, under a nonequivalent groups design. The midpoint could also be estimated using other linear methods, such as Tucker or Levine.

Note that circle-arc equating is defined here as an equating *type*, and equating *methods* are used to estimate the midpoint, which implies a nonequivalent groups design. When groups are considered equivalent (i.e., an anchor test is not used) equating at the midpoint is simply mean equating, as mentioned above (replace x_i with $\hat{\mu}(X)$ in equation 4 to see why this is the case). With scores on an anchor test, both Tucker and Levine equating at the midpoint also reduce to mean equating. However, chained linear equating at the midpoint differs from chained mean (see Appendix A.1).

4 Application Using the **equate** Package

Sample Test Scores

The examples below rely on two data sets, both of which are provided in the equate package. The first, `ACTmath`, is used throughout Kolen and Brennan (2004), and comes from two administrations of the ACT mathematics test. The test scores are based on a random groups design and are contained in a three-column matrix where column one is the 40-point score scale and columns two and three the number of examinees for forms `x` and `y` obtaining each score point.

```
> library(equate)

[1] "equate"    "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"

> head(ACTmath)

     scale xcount ycount
[1,]     0      0      0
[2,]     1      1      1
[3,]     2      1      3
[4,]     3      3     13
[5,]     4      9     42
[6,]     5     18     59
```

The second data set, `KBneat`, is also referenced in Kolen and Brennan (2004). It contains scores for two forms of a 36-item test administered under a nonequivalent groups with anchor

test design. The 12-item anchor test is internal, that is, the total-test score for an examinee includes the score on the anchor items. Thus, the number of non-anchor items, items unique to each form, is 24, and the highest possible score is 36. Unlike the first data set, `KBneat` contains a separate total-test and anchor-test score for each examinee, as is required by the nonequivalent groups equating methods described above. It is a list of length two where the list elements `x` and `y` each consist of a two-column matrix of scores on the total test, and scores on the anchor test `v`.

```
> head(KBneat$x)
```

```
      x xv
[1,]  8  3
[2,] 21  6
[3,] 31 10
[4,]  7  2
[5,] 18  5
[6,] 36 12
```

Preparing the Score Distributions

The `equate` package handles score distributions primarily as frequency tables, as described by the `freqtab` function, which is used to create them. The `ACTmath` data set is an example of a frequency table; scores for over $8,000$ examinees ($N_X = 4,329$, $N_Y = 4,152$) are stored compactly in three columns and 41 rows. The trade-off is that there is no record of scores at the individual level, but this information is not required under the random groups design, as is evident in equations (3)-(10). Frequency tables of class `"freqtab"` are created for the 2 `ACTmath` forms as follows:

```
> act.x <- as.freqtab(ACTmath[, 1], ACTmath[, 2])
> act.y <- as.freqtab(ACTmath[, 1], ACTmath[, 3])
> act.x[1:4, ]
```

```
     x count
[1,] 0     0
[2,] 1     1
[3,] 2     1
[4,] 3     3
```

Here, the command `as.freqtab` is used because the vectors for the score scale and counts are already tabulated, thus they are simply combined and the class changed. The tables can be summarized with the `descript` function:

```
> rbind(descript(act.x), descript(act.y))
```

```
         mean       sd      skew     kurt    n
[1,] 19.85239 8.212585 0.3752283 2.301911 4329
[2,] 18.97977 8.940397 0.3526516 2.145847 4152
```

The function `freqtab` creates a frequency table from scratch, using a vector of scores and the corresponding score scale. With an anchor test this becomes a bivariate frequency table for forms `x` and `y`, and the arguments sent to `freqtab` are the total score scale, vector of total scores, anchor score scale, and vector of anchor scores:

```
> neat.x <- freqtab(0:36, KBneat$x[, 1], 0:12, KBneat$x[, 2])
> neat.y <- freqtab(0:36, KBneat$y[, 1], 0:12, KBneat$y[, 2])
> neat.x[50:55, ]

     x  v count
[1,] 3 10     0
[2,] 3 11     0
[3,] 3 12     0
[4,] 4  0     0
[5,] 4  1     4
[6,] 4  2     3
```

These bivariate tables contain all possible score combinations in columns 1 and 2, along with the number of examinees obtaining each combination in column 3. For example, rows 50 through 55 are displayed above for form X, where counts for 6 X and V score combinations are shown. Based on the scale lengths, tables for `neat.x` and `neat.y` contain $37 \times 13 = 481$ rows of scores, many of which have counts of zero.

The equate package provides a basic plot method for tables of class `"freqtab"`. Univariate frequency tables (up to two) are plotted together as lines with `type = "h"`. For a single bivariate frequency table a scatter plot with marginal barplots is produced (see Figures 1 and 2).

```
> plot(x = act.x, y = act.y, lwd = 2, xlab = "Score", ylab = "Count")
```

```
> plot(neat.x)
```

Finally, presmoothing options are available for equipercentile equating. Three methods are currently supported, all of which can be requested from within the `equate` function. Two of the methods are designed to adjust (i.e., increase) frequencies falling below a specified threshold. Frequency averaging (described by Moses & Holland, 2008), using `freqavg`, replaces scores falling below `jmin` with averages based on adjacent scores:

```
> cbind(act.x, avg = freqavg(act.x, jmin = 2))[1:5, ]
```

```
     x count  avg
[1,] 0     0 1.25
[2,] 1     1 1.25
[3,] 2     1 1.25
[4,] 3     3 1.25
[5,] 4     9 9.00
```

In columns 1 and 2 are the scale and original counts for `act.x`. Column three contains the adjusted counts which are averaged based on any score points with counts below 2 (scores of 0, 1, and 2), along with the next adjacent value (score of 3, with count of 3). The function `freqbump` simply adds a small relative frequency (`jmin`) to each score point while adjusting the probabilities to sum to one (as described by Kolen & Brennan, 2004, p. 48).

As described above and in Appendix A.2, polynomial loglinear smoothing is a flexible option for reducing irregularities throughout the score distribution. In the equate package a loglinear model is fit using the function `loglinear`. Model terms are specified with either a set of score functions (see `example(loglinear)`), or simply by including the degree of the highest desired polynomial term. Here, the bivariate distribution of X and V is smoothed with `degree=3`, and a frequency table is created from the fitted values. The smoothed distributions in Figure 3 can be compared to the unsmoothed ones in Figure 2. Descriptive statistics show that the smoothed distributions match the unsmoothed in the first three moments.

```
> neat.x.smoothout <- loglinear(neat.x, degree = 3)
> neat.xs <- as.freqtab(neat.x[, 1:2], neat.x.smoothout$fitted)
> rbind(descript(neat.x), descript(neat.xs))

         mean       sd      skew     kurt    n
[1,] 15.82054 6.529799 0.5797331 2.720015 1655
[2,] 15.82054 6.529799 0.5797331 3.270636 1655

> rbind(descript(neat.x[, -1]), descript(neat.xs[, -1]))

         mean       sd      skew     kurt    n
[1,] 5.106344 2.376742 0.4115535 2.766619 1655
[2,] 5.106344 2.376742 0.4115535 2.976297 1655

> plot(neat.xs)
```

The equate Function

Most of the functionality of the equate package can be accessed via `equate`, which integrates the equating types and methods described above into a single function. The random groups

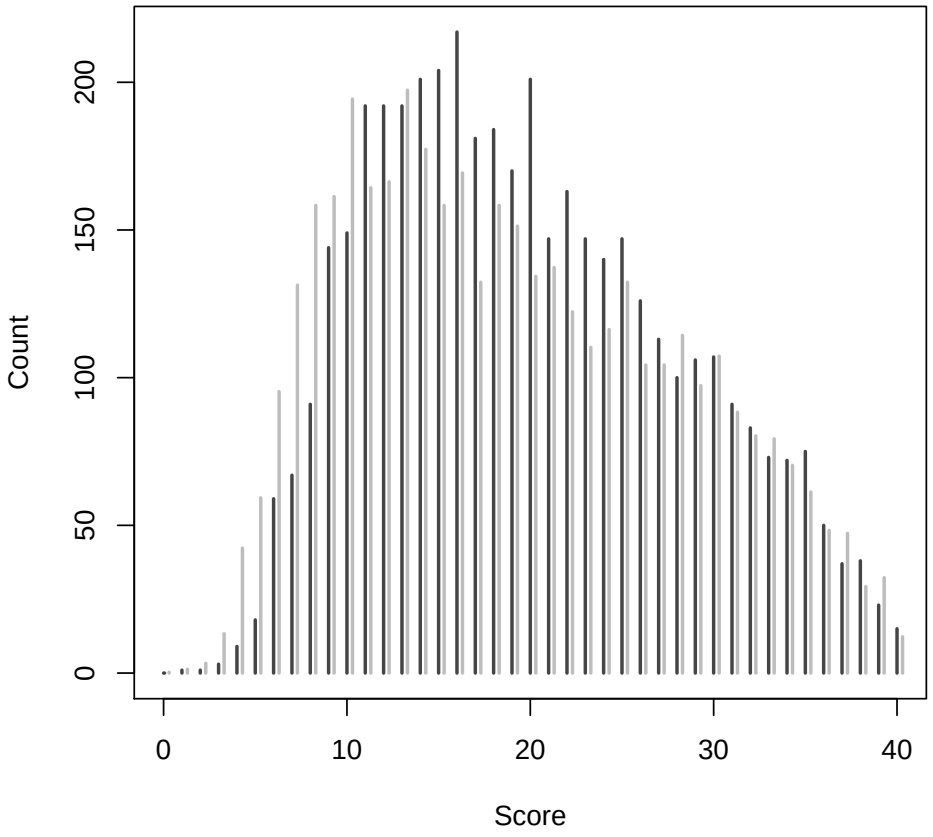

Figure 1: Univariate Plot of ACTmath forms X (dark) and Y (light)

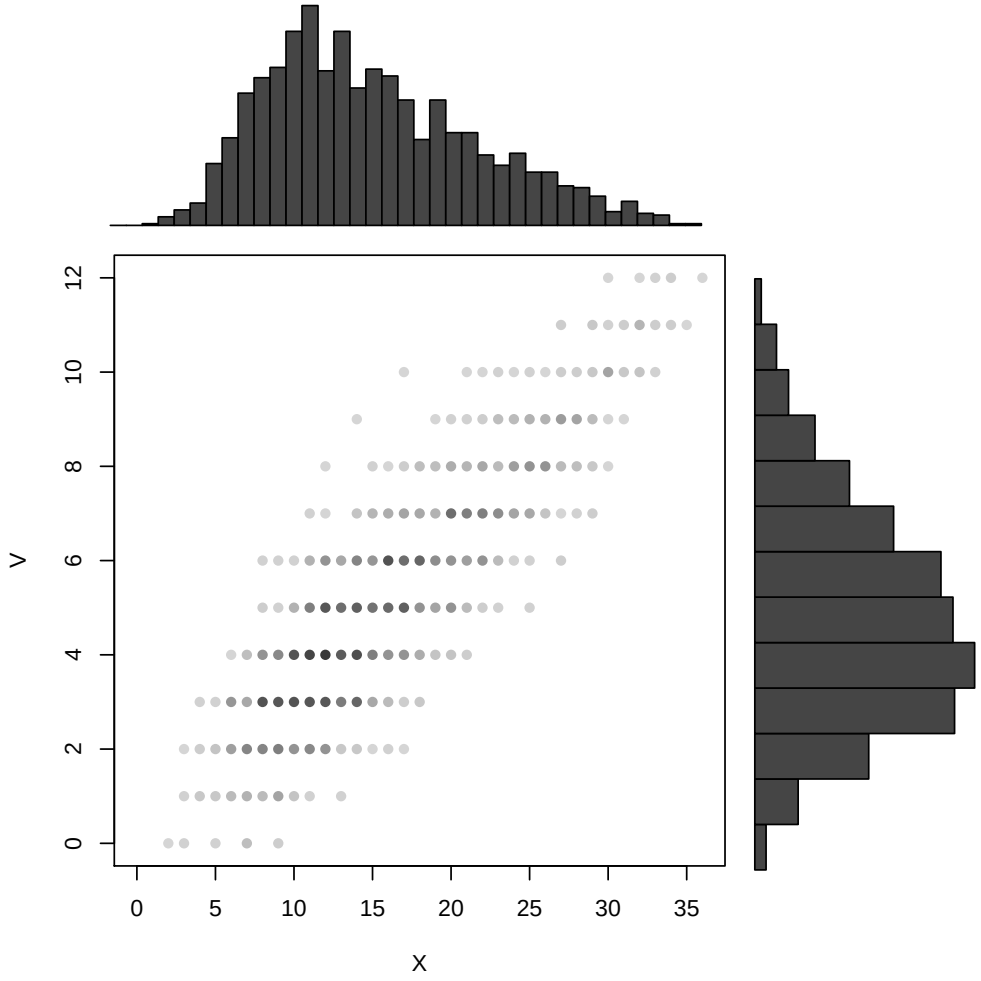

Figure 2: Bivariate Plot of KBneat Total (X) and Anchor (V) Distributions

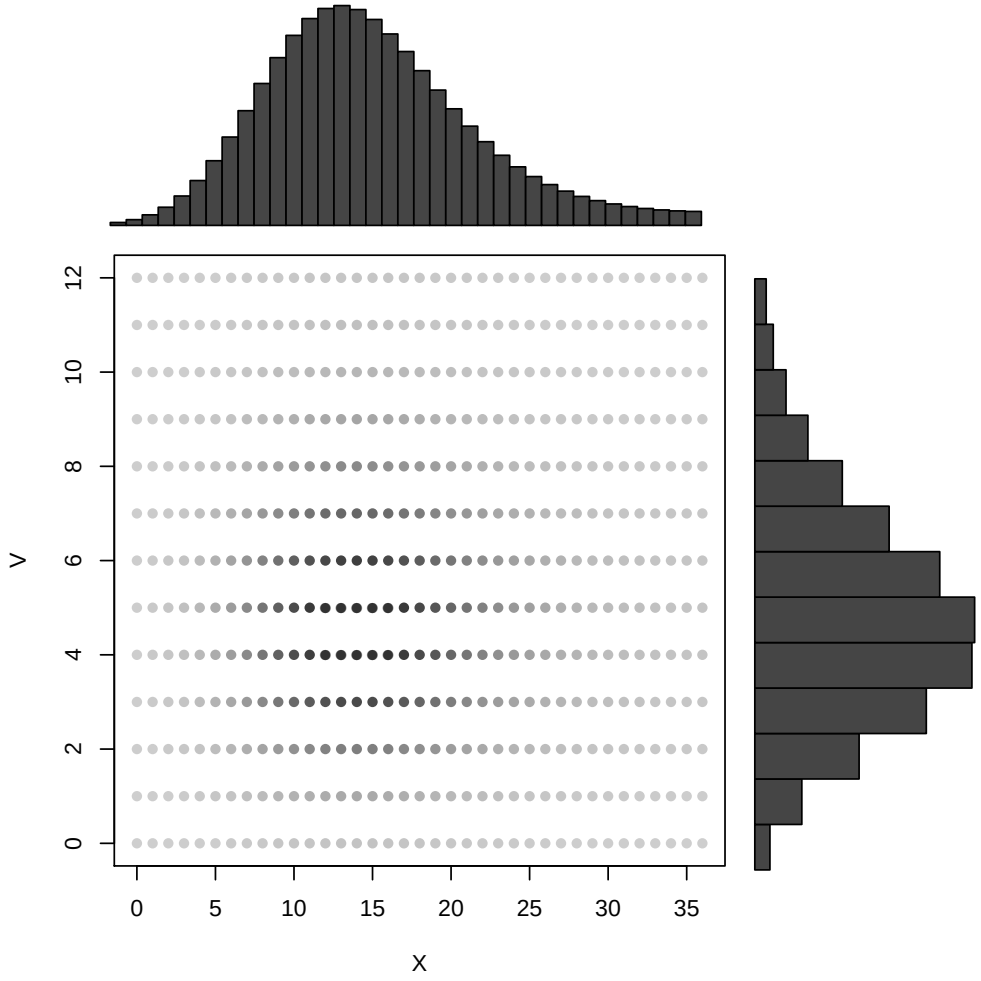

Figure 3: Bivariate Plot of Smoothed KBneat Total (X) and Anchor (V) Distributions

design provides a simple example, where, besides the frequency tables, only the equating `type` need be specified:

```
> equate(act.x, act.y, type = "mean")

Mean Equating: Random Groups

Summary Statistics:
      mean     sd   skew   kurt    n
x  19.8524 8.2126 0.3752 2.3019 4329
y  18.9798 8.9404 0.3527 2.1458 4152
yx 18.9798 8.2126 0.3752 2.3019 4329

Coefficients:
intercept     slope
  -0.8726    1.0000
```

Summary statistics and the intercept and slope are printed (for a full description of available output see `?equate`). The nonequivalent groups design is requested by specifying an equating `method`:

```
> neat.e.c <- equate(neat.x, neat.y, type = "equip", method = "chained")

Chained Equipercentile Equating: Nonequivalent Groups

Summary Statistics:
      mean     sd   skew   kurt    n
x  15.8205 6.5298 0.5797 2.7200 1655
y  18.6728 6.8805 0.2051 2.3014 1638
yx 16.5556 6.5909 0.5439 2.6925 1655
xv  5.1063 2.3767 0.4116 2.7666 1655
yv  5.8626 2.4522 0.1072 2.5089 1638
```

Table 1 above summarizes the equating methods that apply to each equating type in the nonequivalent groups design. For convenience, these may all be specified in the `equate` function using only the first letter, as in `type="c"` for circle-arc equating. Levine true-score equating (`lts`) is requested by including the additional argument `lts=TRUE`.

The `equate` function can also be used to convert scores from one scale to another based on the function from a previous equating. For example, scores on Y for a few more examinees taking `KBneat` form X could be obtained:

```
> cbind(newx = c(3, 29, 8, 7, 13), yx = equate(x = c(3, 29, 8,
+     7, 13), y = neat.e.c))
```

```
     newx        yx
[1,]    3  4.321349
[2,]   29 31.319486
[3,]    8  9.679070
[4,]    7  8.230876
[5,]   13 16.256316
```

Here, the argument `y` passed to `equate` is the chained equipercentile equating from above, which is an object of class `"equate"`. The `equate` function recognizes it as such and attempts to perform the conversion. Note that since the equating function from `neat.e.c` relates scores on X to the scale of Y, anchor test scores are not needed for the examinees `newx`.

Comparing Equatings

There are many considerations involved in choosing a type and method for equating two test forms (see Kolen & Brennan, 2004, ch. 8). However, sample size is paramount, as statistical equating involves the estimation of different numbers of parameters, and accurate estimation depends on adequate and representative samples. As shown above, each equating type and method creates an equating function using different estimates of the score distributions. The equated equivalent at a given score point can vary substantially across equating methods, and within a single equating method across examinee samples.

When samples are small[2] or inadequate for a specific method, random sampling error becomes a major concern. This type of error can be indexed by the standard error of equating (SEE), which is defined as the standard deviation of equated scores for a given x_i over multiple repeated equatings (systematic error is an equally important consideration, but is not as easily estimated; see Appendix A.4). The `equate` package provides estimates of linear and equipercentile SEE under the random groups design, based on equations derived by Braun and Holland (1982) and Lord (1982, p. 168). Additionally, bootstrap standard errors are obtained through the `equate` function using the argument `bootse`:

```
> boots <- equate(act.x, act.y, type = "lin", bootse = TRUE)$bootse
> round(boots, 4)

 [1] 0.2503 0.2422 0.2344 0.2271 0.2203 0.2140 0.2082 0.2031 0.1986 0.1948
[11] 0.1918 0.1895 0.1881 0.1875 0.1877 0.1888 0.1906 0.1933 0.1967 0.2009
[21] 0.2058 0.2112 0.2173 0.2239 0.2310 0.2385 0.2465 0.2548 0.2634 0.2724
[31] 0.2816 0.2911 0.3008 0.3107 0.3208 0.3310 0.3415 0.3520 0.3627 0.3735
[41] 0.3843
```

[2]Kolen and Brennan (2004) refer to "small" as less than 100. Other literature discusses small-sample equating with 20-30 examinees per form, for example Livingston (1993) and Skaggs (2005)

The sample size taken with each bootstrap replication is specified via `xn` and `yn`, the number of replications via `reps`, and the matrix of equated scores (one column per replication) is requested by setting `returnboots=TRUE` (see `?se.boot` for details).

The example below compares mean and linear Tucker and Levine equating, frequency estimation and chained equipercentile equating, and circle-arc chained (linear) and Tucker (mean) equating of the forms `neat.x` and `neat.y`. Thus there are eight separate nonequivalent groups equatings (see Appendix B.1 for R code). Table 2 contains Y equivalents of scores on X for each (R code in Appendix B.2). The conversion table reveals that equated scores vary somewhat by method. Equipercentile equating with frequency estimation (e.f) produced the highest scores of any method between $X = 5$ and $X = 32$. The largest difference between equated scores was between e.f and mean Levine (m.l) at $X = 21$, a difference of 3.25 points on Y. Across methods the smallest equated scores came from circle-arc Tucker equating (c.t) at score points $X < 3, X > 31$, linear Levine (l.l) at points $2 < X < 16$, and mean Levine (m.l) at scores of $15 > X < 32$.

In Figure 4 are plotted the bootstrap standard errors (code for this plot is found in Appendix B.3). The four equating types exhibit a clear trend in SEE across the score scale. As expected, SEE for both mean equatings do not vary by score point, since the scores are equated by a constant amount. Also as expected, random error for linear equating is lowest in the center (slightly lower than estimates for mean) and increases in the tails of the distribution. Overall SEE for equipercentile equating appear to be the largest, despite the fact that the raw score distributions were smoothed. Finally, random error for circle-arc equating is lowest overall, though values increase toward the center of the distribution.

Since equating methods do not extend across all types, they are most easily compared within equating type. Tucker mean outperforms Levine mean; however, the opposite is true for linear equating where Levine SEE are smaller than Tucker across the scale. Until a score on X of 10, values for the two equipercentile methods are comparable. Beyond $X = 10$ random error for chained equating is much lower. Finally, circle-arc equating using the Tucker method to obtain the midpoint results in the lowest SEE of all, values about half as large as those of the chained circle-arc.

Again, it is important to note that random sampling error paints only half the picture when describing equating accuracy. Though a method such as Tucker circle-arc results in some SEE of nearly zero, it may very well be that the estimates are stable (i.e., not varying) around a point that is far from the true equated score. Nevertheless, this example serves to demonstrate the ease with which multiple equatings can be conducted and compared using the `equate` function.

Table 2: Form Y Equivalents for Eight Nonequivalent Groups Equatings

scale	m.t	m.l	l.t	l.l	e.f	e.c	c.c	c.t
0	0.995	0.428	0.537	0.251	0.191	0.038	0.000	0.000
1	1.995	1.428	1.566	1.263	1.486	1.142	1.233	1.109
2	2.995	2.428	2.595	2.274	2.677	2.295	2.451	2.212
3	3.995	3.428	3.624	3.285	3.879	3.472	3.656	3.309
4	4.995	4.428	4.653	4.296	5.097	4.535	4.847	4.400
5	5.995	5.428	5.682	5.307	6.326	5.554	6.025	5.484
6	6.995	6.428	6.710	6.318	7.546	6.593	7.189	6.562
7	7.995	7.428	7.739	7.330	8.731	7.610	8.339	7.633
8	8.995	8.428	8.768	8.341	9.925	8.620	9.477	8.699
9	9.995	9.428	9.797	9.352	11.123	9.631	10.601	9.758
10	10.995	10.428	10.826	10.363	12.320	10.661	11.711	10.811
11	11.995	11.428	11.855	11.374	13.511	11.659	12.809	11.857
12	12.995	12.428	12.884	12.385	14.682	12.692	13.893	12.898
13	13.995	13.428	13.913	13.396	15.850	13.722	14.965	13.932
14	14.995	14.428	14.942	14.408	17.010	14.696	16.023	14.960
15	15.995	15.428	15.971	15.419	18.159	15.781	17.068	15.982
16	16.995	16.428	17.000	16.430	19.294	16.797	18.101	16.997
17	17.995	17.428	18.029	17.441	20.411	17.808	19.120	18.006
18	18.995	18.428	19.058	18.452	21.509	18.881	20.127	19.010
19	19.995	19.428	20.087	19.463	22.591	19.878	21.120	20.006
20	20.995	20.428	21.116	20.475	23.647	20.918	22.101	20.997
21	21.995	21.428	22.145	21.486	24.675	21.967	23.068	21.982
22	22.995	22.428	23.174	22.497	25.669	22.948	24.023	22.960
23	23.995	23.428	24.203	23.508	26.626	23.980	24.965	23.932
24	24.995	24.428	25.232	24.519	27.542	24.998	25.893	24.898
25	25.995	25.428	26.260	25.530	28.425	25.963	26.809	25.857
26	26.995	26.428	27.289	26.542	29.272	26.947	27.711	26.811
27	27.995	27.428	28.318	27.553	30.075	27.921	28.601	27.758
28	28.995	28.428	29.347	28.564	30.829	28.864	29.477	28.699
29	29.995	29.428	30.376	29.575	31.534	29.788	30.339	29.633
30	30.995	30.428	31.405	30.586	32.235	30.705	31.189	30.562
31	31.995	31.428	32.434	31.597	32.898	31.621	32.025	31.484
32	32.995	32.428	33.463	32.609	33.522	32.524	32.847	32.400
33	33.995	33.428	34.492	33.620	34.171	33.402	33.656	33.309
34	34.995	34.428	35.521	34.631	34.805	34.323	34.451	34.212
35	35.995	35.428	36.550	35.642	35.445	35.205	35.233	35.109
36	36.995	36.428	37.579	36.653	36.150	36.057	36.000	36.000

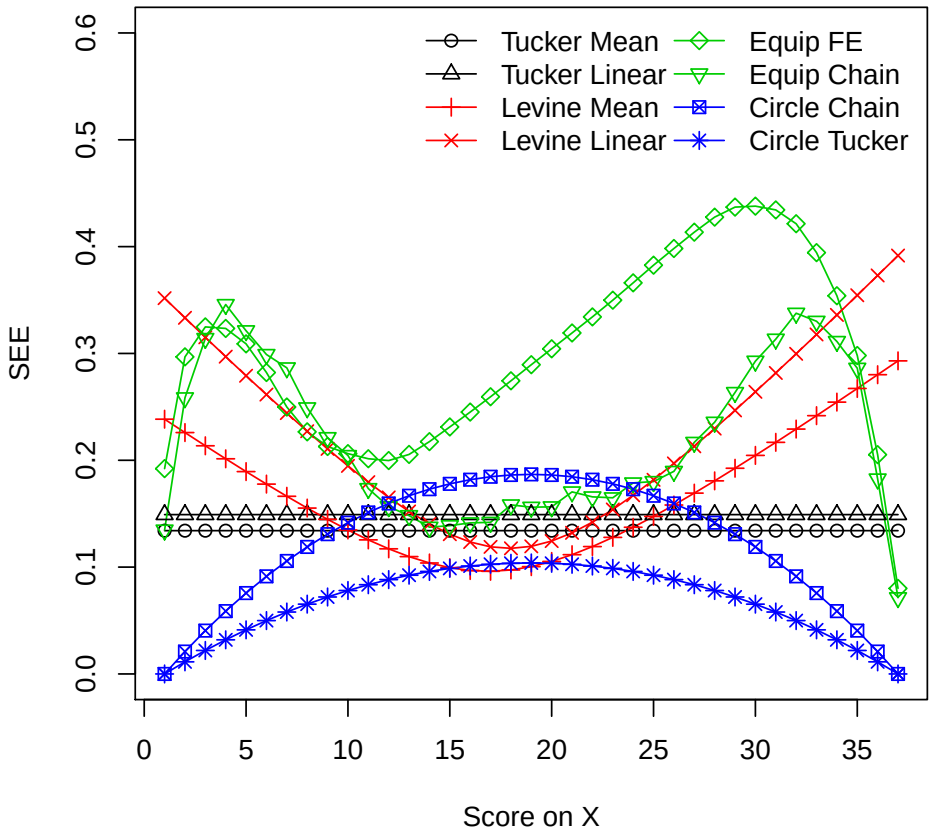

Figure 4: Bootstrap Standard Errors for Eight Nonequivalent Groups Equatings

A Additional Equations

A.1 Chained Linear Equating

Chained linear equating involves two separate linear functions. In the equations below the anchor test V is distinguished by population (1 taking form X and 2 taking form Y), though the items on V do not change. The first linear function in slope-intercept form converts X to the scale of V_1:

$$l_{V_1}(x_i) = \frac{\hat{\sigma}(V_1)}{\hat{\sigma}(X)}x_i - \frac{\hat{\sigma}(V_1)}{\hat{\sigma}(X)}\hat{\mu}(X) + \hat{\mu}(V_1). \tag{23}$$

The second function converts V_2 to the scale of Y:

$$l_Y(v_{2i}) = \frac{\hat{\sigma}(Y)}{\hat{\sigma}(V_2)}v_2 i - \frac{\hat{\sigma}(Y)}{\hat{\sigma}(V_2)}\hat{\mu}(V_2) + \hat{\mu}(Y). \tag{24}$$

These functions are combined, where the first, $l_{V_1}(x_i)$, takes the place of $v_2 i$ in the second to obtain:

$$lchain_Y(x_i) = \frac{\hat{\sigma}(Y)}{\hat{\sigma}(V_2)}\left[\frac{\hat{\sigma}(V_1)}{\hat{\sigma}(X)}x_i - \frac{\hat{\sigma}(V_1)}{\hat{\sigma}(X)}\hat{\mu}(X) + \hat{\mu}(V_1)\right] - \frac{\hat{\sigma}(Y)}{\hat{\sigma}(V_2)}\hat{\mu}(V_2) + \hat{\mu}(Y), \tag{25}$$

or, in slope-intercept form, after some rearranging:

$$lchain_Y(x_i) = \frac{\hat{\sigma}(Y)}{\hat{\sigma}(V_2)}\frac{\hat{\sigma}(V_1)}{\hat{\sigma}(X)}x_i + \frac{\hat{\sigma}(Y)}{\hat{\sigma}(V_2)}\left[\hat{\mu}(V_1) - \frac{\hat{\sigma}(V_1)}{\hat{\sigma}(X)}\hat{\mu}(X) - \hat{\mu}(V_2)\right] + \hat{\mu}(Y). \tag{26}$$

Finally, for chained mean equating this reduces to:

$$mchain_Y(x_i) = x_i + \hat{\mu}(V_1) - \hat{\mu}(X) - \hat{\mu}(V_2) + \hat{\mu}(Y). \tag{27}$$

When used to obtain the midpoint coordinates in circle-arc equating, the chained method reduces even further, since x_i is $\hat{\mu}(X)$. Here, the linear and mean functions simplify to

$$lchain_Y(x_i) = \frac{\hat{\sigma}(Y)}{\hat{\sigma}(V_2)}\hat{\mu}(V_1) - \frac{\hat{\sigma}(Y)}{\hat{\sigma}(V_2)}\hat{\mu}(V_2) + \hat{\mu}(Y), \tag{28}$$

and

$$mchain_Y(x_i) = \hat{\mu}(V_1) - \hat{\mu}(V_2) + \hat{\mu}(Y). \tag{29}$$

A.2 Loglinear Presmoothing

Polynomial loglinear modeling is a flexible method for smoothing distributions of various shapes to varying degrees; the structure of a distribution can either be maintained or ignored depending on the complexity of the model, where the degree of the polynomial term included determines the moment of the raw score distribution to be preserved. For example, a model with terms to the first, second, and third powers would create a smoothed distribution which matches the raw in mean, variance, and skewness. In the model below, the log of the expected relative frequency (p_i) at score point i is expressed in terms of a normalizing constant (β_0) and three weighted score functions (x_1, x_2, x_3) of the possible score values of test X:

$$log(p_i) = \beta_0 + \beta_1 x_i^1 + \beta_2 x_i^2 + \beta_3 x_i^3. \tag{30}$$

Indicator variables may also be included to preserve specific moments for specific score points. In the next model the mean and variance of a sub-distribution are preserved, in addition to the first three moments of the full distribution. Scores with indicator function $S_i = 1$ are included in this sub-distribution, whereas scores with $S_i = 0$ are ignored:

$$log(p_i) = \beta_0 + \beta_1 x_i^1 + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_{S0} S_i + \beta_{S1} x_i^1 S_i + \beta_{S2} x_i^2 S_i. \tag{31}$$

An acceptable degree of smoothing is typically achieved by comparing multiple models with different numbers of polynomial terms based on their fit to the data (Kolen & Brennan, 2004). In the equate package, the function `loglinear` produces fitted (i.e., smoothed) values for univariate and bivariate distributions, and can be used to compare models based on a number of fit indices. It uses a Newton-Raphson maximum likelihood procedure modeled after a SAS macro written by (Moses & von Davier, 2006). Comparable fitted estimates can be obtained using the `glm` function.

A.3 Circle-Arc Equating

The circle-arc in circle-arc equating is a section of the circle that is defined by the distance of the three points (x_1, y_1), (x_2, y_2), and (x_3, y_3) from the line $lin_Y(x)$. Since the low and high points define the line $lin_Y(x)$, they reduce to $(x_1, 0)$ and $(x_3, 0)$. The new midpoint is identified as (x_2, y_{2*}). These three points are used to determine the coordinates x_c and y_c for the center of the circle:

$$x_c = \frac{(x_3^2 - x_1^2)}{2(x_3 - x_1)}, \tag{32}$$

$$y_c = \frac{(x_1^2)(x_3 - x_2) - (x_2^2 + y_{2*}^2)(x_3 - x_1) + (x_3^2)(x_2 - x_1)}{2[y_{2*}(x_1 - x_3)]}. \tag{33}$$

These center points are then used to obtain the radius

$$r^2 = (x_1 - x_c)^2 + (y_{1*} - y_c)^2. \tag{34}$$

Since $y_{1*} = 0$ (x_3 and y_3 could also be used) this reduces to

$$r = \sqrt{(x_1 - x_c)^2 + (y_c)^2}. \tag{35}$$

A.4 Error in Equating

In the literature, equatings are typically compared based on both random and systematic error, where the first is estimated by the standard error of equating (SEE or simply SE) and the second by the $Bias$. As demonstrated above, estimates of SEE can be obtained through bootstrap resampling from the sample score distributions. However, both the SE and $Bias$ are defined in terms of the population equating function. Using a generic equating function $g_Y(x_i)$ to represent a score on X equated to Y, the systematic error is calculated as

$$Bias = \hat{\bar{g}}_Y(x_i) - g_Y(x_i), \tag{36}$$

where $g_Y(x_i)$ is the population equating equivalent and

$$\hat{\bar{g}}_Y(x_i) = \frac{1}{R}\sum_{r=1}^{R} \hat{g}_{Yr}(x_i) \tag{37}$$

is the average estimated equivalent over R samples. The random error is defined as

$$SE = \frac{1}{R}\sqrt{\sum_{r=1}^{R}[\hat{g}_{Yr}(x_i) - \hat{\bar{g}}_Y(x_i)]^2}. \tag{38}$$

And combining both systematic error and random error, the root mean squared error ($RMSE$) is defined as

$$RMSE = \sqrt{\{Bias\}^2 + \{SE\}^2}, \tag{39}$$

B Additional R Code

B.1 Eight Equatings

```
> neat.m.t <- equate(neat.x, neat.y, type = "m", method = "t",
+     bootse = TRUE)
> neat.m.l <- equate(neat.x, neat.y, type = "m", method = "l",
+     bootse = TRUE)
> neat.l.t <- equate(neat.x, neat.y, type = "l", method = "t",
+     bootse = TRUE)
> neat.l.l <- equate(neat.x, neat.y, type = "l", method = "l",
+     bootse = TRUE)
> neat.e.f <- equate(neat.x, neat.y, type = "e", method = "f",
+     bootse = TRUE, smooth = "loglin", degree = 3)
> neat.e.c <- equate(neat.x, neat.y, type = "e", method = "c",
+     bootse = TRUE, smooth = "loglin", degree = 3)
> neat.c.c <- equate(neat.x, neat.y, type = "c", method = "c",
+     bootse = TRUE)
> neat.c.t <- equate(neat.x, neat.y, type = "c", method = "t",
+     bootse = TRUE)
```

B.2 Concordance Table

```
> concordance <- cbind(neat.m.t$conc, neat.m.l$conc[, 2], neat.l.t$conc[,
+     2], neat.l.l$conc[, 2], neat.e.f$conc[, 2], neat.e.c$conc[,
+     2], neat.c.c$conc[, 2], neat.c.t$conc[, 2])
> colnames(concordance)[-1] <- c("m.t", "m.l", "l.t", "l.l", "e.f",
+     "e.c", "c.c", "c.t")
```

B.3 Plotting Bootstrap SEE

```
> plot(c(1, 37), c(0, 0.6), type = "n", xlab = "Score on X", ylab = "SEE")
> points(neat.m.t$bootsee, col = 1, type = "l")
> points(neat.m.l$bootsee, col = 1, type = "l")
> points(neat.l.t$bootsee, col = 2, type = "l")
> points(neat.l.l$bootsee, col = 2, type = "l")
> points(neat.e.f$bootsee, col = 3, type = "l")
> points(neat.e.c$bootsee, col = 3, type = "l")
> points(neat.c.c$bootsee, col = 4, type = "l")
> points(neat.c.t$bootsee, col = 4, type = "l")
> points(neat.m.t$bootsee, col = 1, type = "p", pch = 1)
> points(neat.m.l$bootsee, col = 1, type = "p", pch = 2)
> points(neat.l.t$bootsee, col = 2, type = "p", pch = 3)
> points(neat.l.l$bootsee, col = 2, type = "p", pch = 4)
> points(neat.e.f$bootsee, col = 3, type = "p", pch = 5)
> points(neat.e.c$bootsee, col = 3, type = "p", pch = 6)
```

```
> points(neat.c.c$bootsee, col = 4, type = "p", pch = 7)
> points(neat.c.t$bootsee, col = 4, type = "p", pch = 8)
> legend("topright", legend = c("Tucker Mean", "Tucker Linear",
+     "Levine Mean", "Levine Linear", "Equip FE", "Equip Chain",
+     "Circle Chain", "Circle Tucker"), col = rep(1:4, each = 2),
+     pch = 1:8, lty = 1, bty = "n", ncol = 2)
```

References

Albano, A. D. (2010). equate: Statistical methods for test equating [Computer software manual]. Available from `http://CRAN.R-project.org/package=equate` (R package)

Braun, H. I., & Holland, P. W. (1982). Observed-score test equating: A mathematical analysis of some ETS equating procedures. In P. W. Holland & D. B. Rubin (Eds.), *Test equating* (pp. 9–49). New York, NY: Academic.

Hanson, B. A. (1991). A note on Levine's formula for equating unequally reliable tests using data from the common item nonequivalent groups design. *Journal of Educational and Behavioral Statistics*, *16*, 93.

Holland, P. W., & Dorans, N. J. (2006). Linking and equating. In R. L. Brennan (Ed.), *Educational measurement* (4th ed., pp. 187–220). Westport, CT: Greenwood.

Holland, P. W., & Thayer, D. T. (2000). Univariate and bivariate loglinear models for discrete test score distributions. *Journal of Educational and Behavioral Statistics*, *25*(2), 133-183.

Kim, S., von Davier, A. A., & Haberman, S. (2008). Small-sample equating using a synthetic linking function. *Journal of Educational Measurement*, *45*, 325–342.

Kolen, M. J. (1984). Effectiveness of analytic smoothing in equipercentile equating. *Journal of Educational and Behavioral Statistics*, *9*, 25–44.

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

Livingston, S. A. (1993). Small-sample equating with log-linear smoothing. *Journal of Educational Measurement*, 23–39.

Livingston, S. A. (2004). *Equating Test Scores (Without IRT).* Princeton, NJ: ETS.

Livingston, S. A., Dorans, N. J., & Wright, N. K. (1990). What combination of sampling and equating methods works best? *Applied Measurement in Education*, *3*, 73–95.

Livingston, S. A., & Kim, S. (2009). The circle-arc method for equating in small samples. *Journal of Educational Measurement*, *46*, 330–343.

Lord, F. M. (1982). The standard error of equipercentile equating. *Journal of Educational Statistics*, *7*, 165–174.

Moses, T. P., & Holland, P. W. (2008). *Notes on a general framework for observed score equating (ETS Research Rep. No. RR-08-59).* Princeton, NJ: ETS.

Moses, T. P., & von Davier, A. A. (2006). *A SAS macro for loglinear smoothing: Applications and implications (ETS Research Rep. No. RR-06-05).* Princeton, NJ: ETS.

Skaggs, G. (2005). Accuracy of random groups equating with very small samples. *Journal of Educational Measurement*, *42*, 309–330.

Weeks, J. P. (2009). plink: IRT separate calibration linking methods [Computer software manual]. Available from `http://CRAN.R-project.org/package=plink` (R package)