ABSTRACT

Title of Dissertation: AN EMPIRICAL INVESTIGATION OF UNSCALABLE COMPONENTS IN SCALING MODELS

Kristine N. Braaten, Doctor of Philosophy, 2009

Dissertation directed by: Professor C. Mitchell Dayton
Department of Measurement, Statistics, & Evaluation

Guttman (1947) developed a scaling method in which the items measuring an attribute can be ordered according to the strength of the attribute. The Guttman scaling model assumes that every member of the population belongs to a scale type and does not allow for response errors. The Proctor (1970) and the intrusion-omission (Dayton and Macready, 1976) models introduced the notion that observed response patterns deviate from Guttman scale types because of response error. The Goodman (1975) model posited that part of the population is intrinsically unscalable. The extended Proctor and intrusion-omission (Dayton and Macready, 1980) models, commonly called extended Goodman models, include both response error and an intrinsically unscalable class (IUC).

An alternative approach to the Goodman and extended Goodman models is the two-point mixture index of fit developed by Rudas, Clogg, and Lindsay (1994). The index, π^*, is a descriptive measure used to assess fit when the data can be summarized in a contingency table for a hypothesized model. It is defined as the smallest proportion of cases that must be deleted from the observed frequency table to result in a perfect fit for

the postulated model. In addition to contingency tables, π^* can be applied to latent class models, including scaling models for dichotomous data.

This study investigates the unscalable components in the extended Goodman models and the two-point mixture where the hypothesized model is the Proctor or intrusion-omission model. The question of interest is whether the index of fit associated with the Proctor or intrusion-omission model provides a potential alternative to the IUC proportion for the extended Proctor or intrusion-omission model, or in other words, whether or not π^* and the IUC proportion are comparable.

Simulation results in general did not support the notion that π^* and the IUC proportion are comparable. Six-variable extended models outperformed their respective two-point mixture models with regard to the IUC proportion across almost every combination of condition levels. This is also true for the four-variable case except the π^* models showed overall better performance when the true IUC proportion is small. A real data application illustrates the use of the models studied.

AN EMPIRICAL INVESTIGATION OF UNSCALABLE COMPONENTS IN SCALING MODELS

By

Kristine N. Braaten

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2009

Advisory Committee:

Professor C. Mitchell Dayton, Chair
Professor Gregory Hancock
Professor George Macready
Assistant Professor Jeffrey Harring
Professor Eric Slud

DEDICATION

To the memory of my parents, Audrey and Norman Braaten.

ACKNOWLEDGMENTS

I am indebted to Dr. Dayton for his generous sharing of knowledge in the field of latent class analysis and his strong guidance and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SAS PROGRAMS

CHAPTER 1

PURPOSE AND RATIONALE

Background

Scaling models are designed to order subjects with respect to an attribute on a single continuum. To provide information about an attribute, responses from subjects for a set of items are collected. In choosing items, the aim is that the responses of the subjects to the items will vary with respect to positions on the continuum. Guttman (1947) scaling is a procedure for determining whether or not the responses of subjects to the collection of items form a scale. If there are n dichotomous items, there are 2^n response patterns, but if the items form a scale, only $n + 1$ of these patterns should, in theory, occur. Some researchers have referred to the intensity of the attribute as the level of “difficulty” of the latent trait. Subjects having a higher rank ordering on the continuum would score positively on increasingly more difficult items, while subjects having a lower rank ordering would score negatively on the less difficult items. The Guttman scaling model is deterministic in that it does not allow for errors of measurement or misclassification or errors of response. That is, Guttman assumed that every member of the population does in fact belong to one of the Guttman scale types.

Clogg and Sawyer (1981) noted that there are two approaches to addressing the shortcomings of the Guttman model. The first approach is to modify the Guttman model to allow for a misclassification or error process that results in a response pattern that is inconsistent with the subject’s true type. Models developed under this approach assume that the entire population belongs to one of the Guttman scale types. The Proctor model, the Dayton-Macready intrusion-omission model, the variable-specific error rate model, the true-type specific error rate model, and the Lazarsfeld-Henry latent distance models adopt this approach.

The second approach assumes that the scalability of the attribute of interest differs for different segments of the population. The response patterns that do not match one of the Guttman scales are assumed to be scale types for a part of the population where the attribute is associated with a different rank ordering of the items than the rank ordering resulting in the pure Guttman model. Goodman (1975) proposed that an intrinsically unscalable class (IUC) be added to the Guttman scale types. Dayton and Macready (1980) proposed models assuming that response errors could be attributable to an error process that results in responses to items not conforming to the subject's true type as well as assuming that not all parts of the population have the same intrinsic ordering of the population. Dayton and Macready's models, commonly referred to as extended Goodman models, are composed, for example, of the Proctor model with an IUC included, or the intrusion-omission model with an IUC included.

As an alternative to the Goodman and extended Goodman models, the two-point mixture index of fit developed by Rudas, Clogg, and Lindsay (1994) can be considered. The mixture methods discussed in Rudas *et al.* (1994) do not directly pertain to scaling models or to the concept of an intrinsically unscalable class. However, the mixture framework is applicable to the Proctor and intrusion-omission models. The index of fit associated with the Proctor or intrusion-omission model provides a potential alternative to the IUC proportion for the extended Proctor or intrusion-omission model. Before describing the two-point mixture index of fit in the context of scaling models, the original motivation for the measure is discussed.

Traditionally, observed and expected frequencies have been compared to assess the fit of a statistical model associated with a frequency table. The most commonly used goodness-of-fit measure in frequency table analysis is the Pearson chi-squared statistic (Rudas, 2002). Read and Cressie (1988) summarize and discuss the Pearson chi-squared statistic as well as the log likelihood ratio statistic and other goodness-of-fit statistics. These traditional goodness-of-fit measures have various shortcomings in both large and small sample situations. When the sample size is large, the hypothesized model can easily be rejected even though deviations between observed and expected frequencies are small from a substantive point of view. When the sample size is small, the statistics may not follow the asymptotic distributions of the Pearson chi-squared and likelihood ratio statistics. In addition, these statistics may not provide an accurate approximation in some instances, especially with sparse cells.

Rudas *et al.* (1994) developed a descriptive measure used to assess fit when the data can be summarized in a one-way or multi-way cross-classification for a hypothesized model. Their index of fit, π^*, is defined as the smallest proportion of cases that must be deleted from the observed frequency table to result in a perfect fit for the postulated model. In addition to contingency tables and other models involving frequency data, π^* can be applied to latent class models, including scaling models for dichotomous data (Dayton, 1998). The two-point mixture index of fit is not sensitive to the size of the sample (although its standard error is), which makes for an easily interpretable and useful fit measure for many types of frequency models (Dayton, 2003, 2007).

If P denotes a distribution in the population, then the two-point mixture model specifies that $P_h = (1-\pi)\,\Pi_{1h} + \pi\,\Pi_{2h}$, $h = 1, 2, \ldots, N$, where Π_{1h} is the frequency distribution described by a scaling model, and Π_{2h} is a distribution of unrestricted multinomial counts (Rudas *et al.*, 1994). By varying $1-\pi$ from 0 to 1, a class of models can be generated. When $1-\pi$ is close to 1, then all but a small portion of the population is described by the scaling model. Conversely, when $1-\pi$ is close to 0, then the fraction of the population where the scaling model is valid is small. For any distribution P, π^* represents the smallest value of π such that:

$$P = \left(1-\pi^*\right)\Pi_1 + \pi^*\Pi_2 \quad (1)$$

Extended Goodman models and two-point mixture models represent two different modeling approaches to dealing with "observations" outside the model. The applied researcher might question which of these two approaches should be used in a given situation. This research studies the behavior of these scaling model approaches with dichotomous data under varying simulation conditions in order to help answer this question.

Purpose of Study

This study investigates the unscalable components in the extended Goodman models and the two-point mixture where the hypothesized model is the Proctor or intrusion-omission model. For the extended Goodman models, the portion of the population that is unscalable is viewed as belonging to a latent class distinct from those latent classes corresponding to the Guttman scale types, that is, the intrinsically unscalable class. Under the two-point mixture model applied to the Proctor or intrusion-omission models, the portion of the population described by the Proctor or intrusion-omission model is designated as belonging to the first latent class, and the portion of the population described by an unrestricted multinomial probability model is designated as belonging to the second latent class. The two-point mixture index of fit, π^*, is the fraction of the population that lies outside the Proctor or intrusion-omission model and can be interpreted as the fraction of the population unscalable according to the Proctor or intrusion-omission models. Given this interpretation of π^*, the question of interest is whether the index of fit associated with the Proctor or intrusion-omission model provides a reasonable alternative to the IUC proportion for the extended Proctor or intrusion-omission model, or in other words, whether or not π^* and the IUC proportion are comparable. This research begins with the expectation that π^* will overfit the Proctor and intrusion-omission models. The basis for this expectation is that there are no restrictions on the parameters in the unscalable components.

Significance of Study

There has been no previous research to assess the unscalable components of different scaling model approaches. Thus, this study is an initial effort to consider two competing scaling model approaches, the extended Goodman model and the two-point mixture model, to determine whether one approach can be used as a reasonable substitute for the other. To fulfill this effort, the extended Goodman model's IUC proportion and the two-point mixture model's index of fit, π^*, are assessed under varying simulation conditions, including number of items and sample size. Based on simulation results, conclusions are drawn on which of the competing scaling model approaches may be more appropriate for particular condition levels.

CHAPTER 2

REVIEW OF LITERATURE

Two-Point Mixture Index of Fit

Unlike the traditional goodness-of-fit measures for contingency tables, the two-point mixture index of fit is not sensitive to sample size. Rudas *et al.* (1994) developed this descriptive measure to assess fit when the data can be summarized in a one-way or multi-way cross-classification for a hypothesized model. Their index of fit, π^*, is defined as the smallest proportion of cases that must be deleted from the observed frequency table to result in a perfect fit for the postulated model. This index of fit can be computed for virtually any model involving frequency data and has been utilized in applications such as differential item functioning (Rudas and Zwick, 1997), latent class analysis (Dayton, 1998, 2003), rater agreement (Formann, 2000), and the Rasch model (Formann, 2006). Rudas (1999) and Verdes and Rudas (2002) extended the π^* methodology to applications involving continuous variables in linear regression and logistic regression, respectively.

For a given frequency table, consider P to be the true distribution for the cell proportions and consider H to be its postulated model. The two-point mixture model as developed by Rudas *et al.* (1994) can be expressed as:

$$P = (1 - \pi)\Phi + \pi\Psi, \tag{2}$$

where Φ is the probability distribution represented by H, and Ψ is an unrestricted probability distribution. The proportion of the population that is outside of the model H is indicated by the mixture parameter, $0 \le \pi \le 1$. The index of fit, π^*, is described as the minimum value of π when the model is true. Mathematically, π^* is expressed as:

$$\pi^* = \inf \{ \pi | P = (1 - \pi) \Phi + \pi \Psi, \ \Phi \in H \} . \tag{3}$$

The index of fit, π^*, indicates the smallest proportion of cases that must be deleted from the frequency table to result in an exact fit for the postulated model. The smaller this proportion is, the closer the model fits the population of interest. To illustrate the use of π^*, consider the 2×2 contingency table shown below in Table 1. The null hypothesis of independence would be true if the frequency in cell (2,2) were 25 instead of 30. That is, each cell proportion would be equal to the product of its corresponding marginal probabilities. The value of π^* is $5 / 140 = .036$.

Table 1. 2×2 **Frequency Table**

Column/Row	1	2	**Total**
1	40	50	90
2	20	30	50
Total	60	80	140

Rudas *et al.* (1994) demonstrate how to estimate π^* for any observed frequency table by using an iterative procedure to search for the smallest value of π^* where the likelihood ratio statistic G^2 is equal to zero. The steps of the iterative estimation procedure are: (1) set π^* equal to a very small value such as .001, (2) use an expectation-maximization (EM) algorithm to compute maximum likelihood estimates of the parameters in the two-point mixture model, (3) use the maximum likelihood estimates to compute G^2, and (4) repeatedly increase π^* by a small constant such as .001, with parameter re-estimation at each stage. The final estimate of π^* is realized when the value of the likelihood ratio fit statistic, G^2, equals 0 (approximately).

Xi (1994) and Xi and Lindsay (1996) simplified the estimation problem for π^* by using nonlinear programming (NLP) techniques, which are discussed in detail in Chapter 3. The same estimates will be produced whether NLP or the EM algorithm is used (Xi, 1994).

Dayton (2003) applied the Rudas *et al.* (1994) index of fit, π^*, to one-way and two-way contingency tables, latent class analysis, and the Rasch model. Using Excel Solver, Dayton utilized an optimization approach called separable quadratic programming (SQP), which is described by Xi and Lindsay (1996). Xi and Lindsay (1996) estimated the index of fit for an independence model associated with an $I \times J$ two-way contingency table. Although the independence model for a two-way contingency table can be expressed in terms of a loglinear model, Xi and Lindsay (1996) pointed out that, alternatively, the optimization problem can be stated using row and column marginals for the fitted table.

To incorporate the effect of sampling error, Rudas *et al.* (1994) estimated a lower confidence bound for π^*. Their lower bound, $\hat{\pi}_L$, is equal to the value of $\hat{\pi}$ at which the G^2 fit statistic takes on the value of 2.71, which is the 90th percentage point of the chi-square distribution with one degree of freedom. The confidence interval for π^* is one-sided because all values of $\hat{\pi}$ greater than $\hat{\pi}^*$ result in perfect fit of the observed frequencies in Equation 1. In other words, G^2 is equal to zero for all values of $\hat{\pi}$ greater than $\hat{\pi}^*$. For the one-way table model, Dayton (2003) computed a lower confidence bound for π^* using the approach found in Rudas *et al.* (1994) as well as using a jackknife approach. His analysis showed that the confidence lower bounds for each approach were very similar.

Rudas *et al.* (1994) discuss important qualities of π^* which warrant attention. They demonstrated that the maximum likelihood estimate for π^* is unique, defined on the 0-1 interval scale, decreases as a set of nested models become more complex, and is invariant when a contingency table's frequencies are multiplied or divided by a fixed factor. Furthermore, π^* can be a useful index of fit in the comparison of competing models where the same data set is used or in a situation where one model is proposed, but different data sets are used. Finally, no specific guidelines have been established for what constitutes a reasonable value of π^* in any situation, but Rudas *et al.* (1994) state that "10% is reasonable" for a 4 x 4 contingency table.

Proctor Model

The scaling model proposed by Proctor (1970) allowed for measurement error in the Guttman scaling model. In Proctor's model, the probability of an error of measurement occurring is constant over all items and scale types. Unlike the deterministic Guttman scaling model which assumes that the respondent is free of error, the Proctor model is probabilistic.

Proctor's introduction of measurement error allowed one to explain the occurrence of response patterns that do not correspond to the Guttman scale types. The Proctor model can be viewed as a restricted latent class model. If there are k items, then there are $k + 1$ latent classes corresponding to the $k + 1$ permissible response patterns. Proctor, like Guttman, assumed that all persons in the study population belong to one member of the set of permissible scale types but that response patterns that deviate from these permissible scale types are due to errors in measurement.

To describe the mathematical formulation of the Proctor model in a straightforward way, the number of items is fixed at four, namely, A, B, C, and D. Consider (0, 0, 0, 0), (1, 0, 0, 0), (1, 1, 0, 0), (1, 1, 1, 0), and (1, 1, 1, 1) to be the Guttman scale types, where "1" represents agreement or success and "0" represents disagreement or failure. Let π_t^X for $t = 1, 2, 3, 4, 5$ be the probability of occurrence of the t^{th} scale type or latent class. Each of the terms $\pi_{it}^{\bar{A}X}$, $\pi_{it}^{\bar{B}X}$, $\pi_{it}^{\bar{C}X}$, and $\pi_{it}^{\bar{D}X}$ represent the conditional probability of the respondent's responses to items A, B, C, and D, respectively, and each is equal to the value *i* given that the respondent belongs to the t^{th} permissible response pattern. Within a given scale type, the probability of a response that is inconsistent with the permissible response for the latent class is constant for items A, B, C, and D. This probability of an inconsistent response or error is the same across the five latent classes. Let Γ denote the probability of an inconsistent response and let $1 - \Gamma$ denote the probability of a consistent response. The conditional probabilities $\pi_{it}^{\bar{A}X}$, $\pi_{it}^{\bar{B}X}$, $\pi_{it}^{\bar{C}X}$, and $\pi_{it}^{\bar{D}X}$ can be expressed in terms of the parameter Γ as follows:

$$\pi_{11}^{\bar{A}X} = \pi_{11}^{\bar{B}X} = \pi_{11}^{\bar{C}X} = \pi_{11}^{\bar{D}X} = \Gamma$$

$$\pi_{12}^{\bar{A}X} = 1 - \Gamma$$

$$\pi_{12}^{\bar{B}X} = \pi_{12}^{\bar{C}X} = \pi_{12}^{\bar{D}X} = \Gamma$$

$$\pi_{13}^{\bar{A}X} = 1 - \Gamma$$

$$\pi_{13}^{\bar{B}X} = 1 - \Gamma \tag{4}$$

$$\pi_{13}^{\bar{C}X} = \pi_{13}^{\bar{D}X} = \Gamma$$

$$\pi_{14}^{\bar{A}X} = 1 - \Gamma$$

$$\pi_{14}^{\bar{B}X} = 1 - \Gamma$$

$$\pi_{14}^{\bar{C}X} = 1 - \Gamma$$

$$\pi_{14}^{\bar{D}X} = \Gamma$$

$$\pi_{15}^{\bar{A}X} = \pi_{15}^{\bar{B}X} = \pi_{15}^{\bar{C}X} = \pi_{15}^{\bar{D}X} = 1 - \Gamma$$

To express the Proctor model in a general form, Dayton and Macready's (1980) notation is utilized. Let $\mathbf{u}_h$ denote the observed response pattern and let $\mathbf{v}_h$ denote the response vector associated with the scale type. Generally, the probability of the observed response pattern can be written as:

$$P(\mathbf{u}_h) = \sum_{t=1}^{T} \pi_t^X \, \Gamma^{x_{th}} (1-\Gamma)^{k-x_{th}} \tag{5}$$
$$\text{where } x_{th} = (\mathbf{v}_t - \mathbf{u}_h)' (\mathbf{v}_t - \mathbf{u}_h)$$

When the number of items, *n*, is four, the probability of the observed response pattern can be written as:

$$\mathrm{P}(\mathbf{u}_h) = \sum_{t=1}^{5} \pi_t^X \, \Gamma^{x_{th}} (1-\Gamma)^{4-x_{th}} \tag{6}$$

where $x_{th} = (\mathbf{v}_t - \mathbf{u}_h)'(\mathbf{v}_t - \mathbf{u}_h)$

For example, the probability of observing the response pattern (0, 1, 1, 0) is presented. For the five latent classes, the values of the x_{th}'s are as follows:

$$x_{1h} = 2,\ x_{2h} = 3,\ x_{3h} = 2,\ x_{4h} = 1,\ x_{5h} = 2 \tag{7}$$

The expression for P{(0, 1, 1, 0)} is equal to:

$$\pi_1^X \Gamma^2 (1-\Gamma)^2 + \pi_2^X \Gamma^3 (1-\Gamma) + \pi_3^X \Gamma^2 (1-\Gamma)^2 + \pi_4^X \Gamma (1-\Gamma)^3 + \pi_5^X \Gamma^2 (1-\Gamma)^2 \tag{8}$$

Intrusion-Omission (Dayton and Macready) Model

Dayton and Macready (1976) developed a model called the intrusion-omission model as an extension of the Proctor model. They postulated that the error occurring if a respondent answered "1" when a "0" response was consistent with the permissible response pattern was distinct from the error occurring if a respondent answered "0" when a "1" response was consistent with the permissible response pattern. The error of the first type is referred to as an "intrusion" error whose probability is represented by the term β_I, and the second type of error is referred to as an "omission" error whose probability is represented by the term β_O. Again, let the number of items be fixed at four. The conditional probabilities $\pi_{it}^{\bar{A}X}$, $\pi_{it}^{\bar{B}X}$, $\pi_{it}^{\bar{C}X}$, and $\pi_{it}^{\bar{D}X}$ can be expressed in terms of the parameters β_I and β_O as follows:

$$\pi_{11}^{\bar{A}X} = \pi_{11}^{\bar{B}X} = \pi_{11}^{\bar{C}X} = \pi_{11}^{\bar{D}X} = \beta_I$$

$$\pi_{12}^{\bar{A}X} = 1 - \beta_O$$

$$\pi_{12}^{\bar{B}X} = \pi_{12}^{\bar{C}X} = \pi_{12}^{\bar{D}X} = \beta_I$$

$$\pi_{13}^{\bar{A}X} = 1 - \beta_O$$

$$\pi_{13}^{\bar{B}X} = 1 - \beta_O \tag{9}$$

$$\pi_{13}^{\bar{C}X} = \pi_{13}^{\bar{D}X} = \beta_I$$

$$\pi_{14}^{\bar{A}X} = 1 - \beta_O$$

$$\pi_{14}^{\bar{B}X} = 1 - \beta_O$$

$$\pi_{14}^{\bar{C}X} = 1 - \beta_O$$

$$\pi_{14}^{\bar{D}X} = \beta_I$$

$$\pi_{15}^{\bar{A}X} = \pi_{15}^{\bar{B}X} = \pi_{15}^{\bar{C}X} = \pi_{15}^{\bar{D}X} = 1 - \beta_O$$

Using these conditional probabilities and the mixing proportions, the intrusion-omission model can be expressed in mathematical form. Utilizing Dayton and Macready's (1976) notation again, the probability of the observed response pattern $\mathbf{u}_h$ can be written as:

$$P(\mathbf{u}_h) = \sum_{t=1}^{T} \pi_t^X \, \beta_I^{x_{th}} (1 - \beta_I)^{m_t - x_{th}} \beta_O^{y_{th}} (1 - \beta_O)^{k - m_t - y_{th}} \tag{10}$$

When the number of items is four, the intrusion-omission model can be written as:

$$P(\mathbf{u}_h) = \sum_{t=1}^{5} \pi_t^X \, \beta_I^{x_{th}} (1-\beta_I)^{m_t - x_{th}} \beta_O^{y_{th}} (1-\beta_O)^{4-m_t-y_{th}} \tag{11}$$

The term x_{th} denotes the number of intrusion errors in the observed response pattern. The term y_{th} denotes the number of omission errors. Let m_t be the number of chances for an intrusion to occur for the t^{th} scaling category and let $n - m_t$ be the number of chances for an omission to occur. Note that if $\beta_I = \beta_O$, then the intrusion-omission model reduces to the Proctor model.

To illustrate, the probability of occurrence of the response vector (0, 1, 0, 1) can be expressed using the above notation. For the second scale type (1, 0, 0, 0), there are two errors for intrusion ($x_{th} = 2$) and there are three chances for intrusion errors ($m_t = 3$). There is one omission error ($y_{th} = 1$) and there is only one chance for an omission error to occur ($\mathrm{n} - \mathrm{m}_t = 1$). The contribution from the second scale type to the above summation is equal to the expression:

$$\pi_2^X \, \beta_I^2 (1-\beta_I)^{3-2} \beta_O^1 (1-\beta_O)^{4-3-1} \tag{12}$$

Goodman Model

Goodman (1975) formulated a model where a respondent could be classified into *t* mutually exclusive and exhaustive categories. One of these categories is composed of respondents who Goodman viewed as intrinsically unscalable. The remaining respondents Goodman considered intrinsically scalable and their response patterns correspond to their scale category or type. Goodman assumed that the proportion of the population that was intrinsically scalable adhered to the Guttman ordering. For four items, the Guttman scale categories can be represented as (1, 1, 1, 1), (1, 1, 1, 0), (1, 1, 0, 0), (1, 0, 0, 0), and (0, 0, 0, 0). Given that there are 16 possible response patterns for four items, 11 response patterns constitute the unscalable class. Let the subscript, 6, correspond to the IUC. Let the term π_t^X for $t = 1, 2, 3, 4, 5, 6$ denote the probability that a person will be in the t^{th} category. The term $\pi_{it}^{\bar{A}X}$ represents the conditional probability that a person's response to item A will have a value of i given that the respondent is in the t^{th} category. Since most practical applications focus on dichotomous items, let i take on the value of "0" or "1". Let π_{ijkl}^{ABCD} denote the probability of obtaining the response pattern (i, j, k, l) and let $\pi_{ijklt}^{\bar{A}\bar{B}\bar{C}\bar{D}X}$ denote the conditional probability of obtaining the response pattern (i, j, k, l) in the t^{th} category. Due to the definition of the scale types,

$$\pi_{11111}^{\bar{A}\bar{B}\bar{C}\bar{D}X} = \pi_{11102}^{\bar{A}\bar{B}\bar{C}\bar{D}X} = \pi_{11003}^{\bar{A}\bar{B}\bar{C}\bar{D}X} = \pi_{10004}^{\bar{A}\bar{B}\bar{C}\bar{D}X} = \pi_{00005}^{\bar{A}\bar{B}\bar{C}\bar{D}X} = 1\,. \tag{13}$$

For the five response patterns that correspond to the scale types, the probability of obtaining the response pattern is equal to the probability of the person falling in the scale category plus the product of the person falling in the IUC and the conditional probability of the person taking on the values of items A, B, C, and D associated with the response pattern. For example, the expression for the response pattern probability π_{1100}^{ABCD} is $\pi_3^X + \pi_6^X \pi_{16}^{\bar{A}X} \pi_{16}^{\bar{B}X} \pi_{06}^{\bar{C}X} \pi_{06}^{\bar{D}X}$. It is assumed that the items A, B, C, and D are mutually independent within the IUC. Within the IUC category, the probability of the response pattern (i,j,k,l) is:

$$\pi_{ijkl}^{ABCD} = \pi_6^X \pi_{i6}^{\bar{A}X} \pi_{j6}^{\bar{B}X} \pi_{k6}^{\bar{C}X} \pi_{l6}^{\bar{D}X}\ . \tag{14}$$

Goodman (1975) showed that for the scalable categories the expected frequencies are equal to the estimates of the observed frequencies, and only the 11 response patterns comprising the IUC are used to estimate the mixing proportions, i.e., the

π_t^X's for $t = 1, 2, 3, 4, 5, 6$ and $\pi_{i6}^{\bar{A}X}, \pi_{j6}^{\bar{B}X}, \pi_{k6}^{\bar{C}X}, \pi_{l6}^{\bar{D}X}$ with i, j, k, l equal to 0 or 1.

Goodman (1975) expresses the π_{ijkl}^{ABCD} terms in a quasi-independence model such that $\pi_{ijkl}^{ABCD} = \alpha_i \beta_j \gamma_k \delta_l$ and solves for the maximum likelihood estimates of the model parameters. These estimates are then used to compute the mixing proportions, i.e., π_t^X's for $t = 1, 2, 3, 4, 5, 6$ and IUC conditional probabilities $\pi_{i6}^{\bar{A}X}, \pi_{j6}^{\bar{B}X}, \pi_{k6}^{\bar{C}X}, \pi_{l6}^{\bar{D}X}$

with i, j, k, l equal to 0 or 1.

Agresti (1990) discusses how the loglinear version of the quasi-independence model, $\log m_{ijkl} = \lambda + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_l^D$, can be used to solve for the model parameters via Newton-Raphson or iterative proportional fitting algorithms. Chapter 3 describes the computational details of solving for maximum likelihood estimates using the loglinear formulation.

Goodman (1975) notes that the five scalable categories and the one IUC can be treated as six latent classes. Since there are six latent classes but only four items, the parameters in the model will not be identifiable unless some restrictions are imposed. The following restrictions are placed on the parameters:

$$\pi_{11}^{\bar{A}X} = \pi_{11}^{\bar{B}X} = \pi_{11}^{\bar{C}X} = \pi_{11}^{\bar{D}X} = 1$$

$$\pi_{12}^{\bar{A}X} = \pi_{12}^{\bar{B}X} = \pi_{12}^{\bar{C}X} = \pi_{01}^{\bar{D}X} = 1$$

$$\vdots \tag{15}$$

$$\pi_{05}^{\bar{A}X} = \pi_{05}^{\bar{B}X} = \pi_{05}^{\bar{C}X} = \pi_{05}^{\bar{D}X} = 1$$

No restrictions are placed on the IUC conditional probabilities. It should be noted that the Goodman model can be easily extended to cases with two or more intrinsically unscalable classes.

Extended Proctor (Dayton and Macready) Model

Like Goodman (1975), Dayton and Macready (1980) hypothesized that a certain proportion of the study population may not be scalable according to the Guttman scale. However, they extended the Proctor and intrusion-omission models by adding an intrinsically unscalable class (IUC). This latent class's probability of occurrence is denoted by π_6^X , the mixing proportion for the IUC. The conditional probabilities where the four items take on the values of "1" or "0" given the respondent is intrinsically unscalable are denoted by $\pi_{116}, \pi_{106}, \ldots\ldots, \pi_{416}$, and π_{406}. Generally, the extended Proctor model can be written as:

$$P(\mathbf{u}_h) = \pi_{(k+2)}^X \prod_{i=1}^{k} \pi_{ij(k+2)} + \sum_{t=1}^{T} \pi_t^X \, \Gamma^{x_{th}} (1-\Gamma)^{k-x_{th}} \tag{16}$$

When the number of items is four, the extended Proctor model can be written as:

$$P(\mathbf{u}_h) = \pi_6^X \prod_{i=1}^{4} \pi_{ij6} + \sum_{t=1}^{5} \pi_t^X \, \Gamma^{x_{th}} (1-\Gamma)^{4-x_{th}} \tag{17}$$

The π_{ij6} terms, Γ, and the mixing proportions, π_t^X for $t = 1, ..., 6$ are estimated for the extended Proctor model.

Extended Intrusion-Omission (Dayton and Macready) Model

Dayton and Macready's (1976) original formulation of the intrusion-omission model assumed that all respondents in the study population could be assigned to one of the Guttman scale types. Later Dayton and Macready (1980) posited that a certain segment of the study population was intrinsically unscalable and thus extended the intrusion-omission model by adding an IUC. Under the extended intrusion-omission model, the probability of occurrence for the observed response pattern is:

$$P(\mathbf{u}_h) = \pi_6^X \prod_{i=1}^{k} \pi_{ij6} + \sum_{t=1}^{T} \pi_t^X \, \beta_I^{x_{th}} (1-\beta_I)^{m_t - x_{th}} \beta_O^{y_{th}} (1-\beta_O)^{k-m_t-y_{th}} \qquad (18)$$

When the number of items is four, the extended intrusion-omission model is written as:

$$P(\mathbf{u}_h) = \pi_6^X \prod_{i=1}^{4} \pi_{ij6} + \sum_{t=1}^{5} \pi_t^X \, \beta_I^{x_{th}} (1-\beta_I)^{m_t - x_{th}} \beta_O^{y_{th}} (1-\beta_O)^{4-m_t-y_{th}} \qquad (19)$$

Note that $\pi_1^X + \pi_2^X + \pi_3^X + \pi_4^X + \pi_5^X + \pi_6^X = 1$.

Index of Fit for Latent Class Analysis

Rudas *et al.* (1994) used a mixture model approach to evaluate how well a model is describing a set of data. The research focused on contingency table analysis, but Dayton (2003) showed that their mixture model approach was applicable to latent class analysis. Assuming that the study respondents have been administered an instrument containing *k* dichotomous items, there exist $2^k = N$ response patterns and the response pattern probabilities can be written as:

$$P_h = (1-\pi)\Pi_{1h} + \pi\Pi_{2h} \quad with\ h = 1, 2, \ldots, N \tag{20}$$

The term Π_{2h} represents the probability of occurrence of the response under a completely unrestricted model. The term Π_{1h} represents the conditional probability of the occurrence of response pattern h given the data follows the specified latent class model.

The terms π and $1-\pi$ are termed mixing weights and π is the fraction of the population outside of the postulated model. When $1-\pi$ is close to 0, then almost the entire population is characterized by an unrestricted distribution. When $1-\pi$ is close to 1, then the model does well in describing the population. In terms of a mixture model representation, the first latent class consists of that segment of the population described by the model, and the second latent class is that segment of the population conforming to an unrestricted multinomial model. As there are no restrictions that Rudas *et al.* (1994) placed on the first latent class in their mixture model, the Proctor and intrusion-omission models can be used with this approach.

Since the index of fit has applicability to latent class analysis and especially scaling models, this study compares the two-point mixture model and the extended Goodman model in terms of their unscalable components. Specifically, the extended Goodman model's IUC proportion and the two-point mixture model's index of fit, π^*, will be assessed under varying simulation conditions. The generation of the simulated data will be based on the extended Goodman model, since this study's purpose is to determine if the two-point mixture model's index of fit, π^*, is considered a potential alternative to the extended model's IUC proportion. The next chapter describes the methods of estimation and simulation design which will ultimately allow an objective comparison of the two competing scaling model approaches.

CHAPTER 3

STUDY DESIGN AND ESTIMATION

This chapter describes the computational methods used to estimate the parameters in the models described in Chapter 2 as well as the simulation design which generates the data used in the estimation.

Methods of Estimation

Nonlinear programming techniques, specifically the SAS procedure NLP, were utilized to compute maximum likelihood estimates for the parameters in the models of interest. For the computation of the index of fit, π^* , for the Proctor and intrusion-omission models, mixing proportions and measurement error rates were calculated. These were used to compute expected frequencies which were in turn used to compute π^*. For the extended Proctor and intrusion-omission models, the measurement error rates were estimated as well as mixing proportions and the IUC's conditional probabilities. For the Goodman model, primarily in this study for the sake of comparison, the mixing proportions and the IUC's conditional probabilities were estimated. The SAS MACRO language was utilized to conduct the proposed simulation research.

The SAS procedure NLP was utilized to compute maximum likelihood estimates for all five models although it should be noted that other software exists to fit the extended Proctor and intrusion-omission models and the Goodman model. Clogg and Sawyer (1981) discussed the use of the Maximum Likelihood Latent Structure Analysis (MLLSA) software to fit the Goodman and Proctor models. Dayton and Macready

(1980) used the MLLSA software for parameter estimation and goodness-of-fit tests for the extended Proctor and intrusion-omission models. Finally, LEM (Vermunt, 1997) is a software package that can be used to compute maximum likelihood estimates for the extended models and the Goodman model, as well as for a large variety of other loglinear models.

With regard to the two-point mixture index of fit, Xi (1994) and Xi and Lindsay (1996) simplified the estimation problem for π^* by using NLP techniques. Dayton (2003) applied the Rudas *et al.* (1994) index of fit, π^*, to one-way and two-way contingency tables, latent class analysis, and the Rasch model. Using Excel Solver, Dayton utilized an optimization approach called separable quadratic programming (SQP) which is described by Xi and Lindsay (1996). Xi and Lindsay (1996) estimated the index of fit for an independence model associated with an $I \times J$ two-way contingency table.

The nonlinear programming method utilized in this research is found in the operations research (OR) component of SAS (2004). The specific method used to maximize the nonlinear likelihood functions was quasi-Newton optimization (QUANEW). The SAS Institute recommends QUANEW for general nonlinear optimization for problems with nonlinear and linear constraints. The maximum number of iterations for convergence in the optimization process to occur was set to the default value of 200. Each simulation run consisted of 1,000 replications. For the extended Proctor and extended IO models, the GCONV convergence condition or relative gradient was imposed by SAS. The GCONV condition specifies that the quasi-Newton procedure will terminate when the normalized predicted function reduction is small, that is,

$$\frac{g\left(x^{(k)}\right)^{T}\left(G^{(k)}\right)^{-1} g\left(x^{(k)}\right)}{\left|f\left(x^{(k)}\right)\right|} \leq 0 \tag{21}$$

where $g\left(x^{(k)}\right) =$ the value of the gradient evaluated at the point $x^{(k)}$, k denoting iteration

$G^{(k)} =$ the estimate of the Hessian matrix at the k^{th} iteration

$f\left(x^{(k)}\right) =$ the value of objective function at point $x^{(k)}$

For the Proctor and IO π^{*} models, the FCONV convergence criterion or relative function was utilized by SAS and can be expressed as:

$$\frac{\left|f\left(x^{(k)}\right) - f\left(x^{(k-1)}\right)\right|}{\left|f\left(x^{(k-1)}\right)\right|} < r \tag{22}$$

where $f\left(x^{(k)}\right)$ and $f\left(x^{(k-1)}\right)$ are the values of the objective function at $x^{(k)}$ and $x^{(k-1)}$. The default value of r is 10^{-p} where p is $\log_{10}(\varepsilon)$ where ε is the machine precision.

The next sections discuss the formulation of the likelihood functions in terms of objective functions that were specified in PROC NLP for this study's models.

Estimation for Extended Proctor and Intrusion-Omission Models. The mixture model approach to latent class analysis was used to establish the objective function for the extended Proctor and intrusion-omission models. Assume that four dichotomous items are of interest in the following discussion. As presented in Chapter 2, the probability of the observed response pattern $\mathbf{u}_h$ for the extended Proctor model can be written as:

$$P(\mathbf{u}_h) = \pi_t^X \prod_{i=1}^{k} \pi_{ij6} + \sum_{t=1}^{T} \pi_t^X \, \Gamma^{x_{th}} (1-\Gamma)^{k-x_{th}} \tag{23}$$

Specifically, for four items, the probability of the observed response pattern can be expressed as:

$$P(\mathbf{u}_h) = \pi_t^X \prod_{i=1}^{4} \pi_{ij6} + \sum_{t=1}^{5} \pi_t^X \, \Gamma^{x_{th}} (1-\Gamma)^{4-x_{th}} \tag{24}$$

The log likelihood function can be written as:

$$\log L = \log\left[\prod_{h=1}^{16} P(\mathbf{u}_h)^{k_s}\right]$$

$$= \sum_{h=1}^{16} \left[\log P(\mathbf{u}_h)^{k_s}\right] \tag{25}$$

$$= \sum_{h=1}^{16} \log\left[\left(\pi_t^X \prod_{i=1}^{4} \pi_{ij6} + \sum_{t=1}^{5} \pi_t^X \, \Gamma^{x_{th}} (1-\Gamma)^{4-x_{th}}\right)^{k_s}\right]$$

There are 21 terms contained in the objective function, 20 of which will be in terms of one of the following five expressions:

$$\Gamma^4, \quad (1-\Gamma)^4, \quad \Gamma^3 * (1-\Gamma), \quad \Gamma * (1-\Gamma)^3, \quad \Gamma^2 * (1-\Gamma)^2 \tag{26}$$

The term corresponding to the IUC can be written as:

$$\pi_6^X (\pi_{61} ** y_1 * (1-\pi_{61}) ** (1-y_1) * K\ K * \pi_{64} ** y_4 * (1-\pi_{64}) ** (1-y_4)) \tag{27}$$

The terms $y_{1,}$ $y_{2,}$ $y_{3,}$ and y_4 represent the values of the items A, B, C, and D.

Note that the constraint $\pi_1^X + \pi_2^X + \pi_3^X + \pi_4^X + \pi_5^X + \pi_6^X = 1$ and bounds between 0 and 1 are specified for $\pi_1^X, \pi_2^X, \pi_3^X, \pi_4^X, \pi_5^X, \pi_6^X$, Γ, and $\pi_{61}^{\bar{A}X}, \pi_{62}^{\bar{B}X}, \pi_{63}^{\bar{C}X}, \pi_{64}^{\bar{D}X}$. The complete objective function for the extended intrusion-omission model contains 20 terms which are in terms of the intrusion and omission error rates. Note that the term for the IUC is the same as that for the extended Proctor model.

For those cells with observed frequencies of zero, maximum likelihood estimates could not be computed. Thus, based on the research of Pan (2006), “1” is used as a flattening constant. This adjustment applies to all models in this study.

Estimation for Goodman Model. As cited earlier, Goodman saw his model of scaling response patterns as a quasi-independence model. Agresti (1990) discussed how the loglinear version of the quasi-independence model, $\log m_{ijkl} = \lambda + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_l^D$ can be used to solve for the model parameters via Newton-Raphson or iterative proportional fitting algorithms. Let

$$\log m_{ijkl} = \mu + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_l^D, \quad (i, j, k, l) \in S \tag{28}$$

where S denotes the set of response patterns that do not correspond to the IUCs.

To derive the likelihood function for the quasi-independence model, assume that the observed frequencies, denoted n_{ijkl}, for the response patterns are distributed as Poisson random variates with expected values m_{ijkl}. The joint density function of the set of n_{ijkl}'s is given by:

$$\prod_i \prod_j \prod_k \prod_l \frac{e^{-m_{ijkl}} m_{ijkl}^{n_{ijkl}}}{n_{ijkl}!} \tag{29}$$

Taking the log of this product results in the following expression:

$$\log L = \sum_i \sum_j \sum_k \sum_l n_{ijkl} \log m_{ijkl} - \sum_i \sum_j \sum_k \sum_l m_{ijkl} \tag{30}$$

In terms of the parameters of the loglinear model this expression becomes:

$$\log L = \sum_i \sum_j \sum_k \sum_l n_{ijkl} \left(\mu + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_l^D\right) - \sum_i \sum_j \sum_k \sum_l e^{\mu + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_l^D} \tag{31}$$

In order that the solutions to the maximum likelihood equations are unique, constraints must be placed on the loglinear parameters as follows:

$$\lambda_1^A + \lambda_0^A = 0,\ \ \lambda_1^B + \lambda_0^B = 0,\ \ \lambda_1^C + \lambda_0^C = 0,\ \ \lambda_1^D + \lambda_0^D = 0 \tag{32}$$

The objective function used within the SAS nonlinear programming procedure can be written as:

$$\begin{aligned} n_{ijkl} * \left\{\mu + \lambda_1^A * y_1 + \lambda_0^A * (1 - y_1) + \text{K K} + \lambda_1^D * y_4 + \lambda_0^D * (1 - y_4)\right\} - \\ \exp\left\{\mu + \lambda_1^A * y_1 + \lambda_0^A * (1 - y_1) + \text{K K} + \lambda_1^D * y_4 + \lambda_0^D * (1 - y_4)\right\} \end{aligned} \tag{33}$$

The terms $y_{1,}\ y_{2,}\ y_{3,}$ and y_4 represent the values of the items A, B, C, and D. Note that the constraints were included in the NLP procedure. Once estimates of the loglinear parameters were obtained, these parameter estimates were used to compute the IUC conditional probabilities and the mixing proportions. Note that the expected frequencies were computed as follows:

$$\hat{m}_{ijkl} = \exp\left\{\mu + \hat{\lambda}_i^A + \hat{\lambda}_j^B + \hat{\lambda}_k^C + \hat{\lambda}_l^D\right\} \tag{34}$$

Estimation for the Index of Fit. Dayton (2003) in a latent class application used a two-step optimization approach to find a solution for π^*. The first step involved the minimization of the log likelihood ratio:

$$G^2 = \sum_{i=1}^{S} 2 * n_i * \log \frac{n_i}{\hat{n}_i} \tag{35}$$

where $\hat{n}_i = N\left(v_1 * \pi_{a1}^A * \pi_{b1}^B * \pi_{c1}^C * \pi_{d1}^D + \text{K K K} + v_T * \pi_{aT}^A * \pi_{bT}^B * \pi_{cT}^C * \pi_{dT}^D\right)$

where a, b, c, and d are the values of the items A, B, C, and D for the i-th response pattern. Note that minimizing the log likelihood ratio G^2 is equivalent to maximizing the likelihood function.

The second step involved retaining the estimates of the mixing proportions and the conditional probabilities from the first step and using them in the maximization of the objective function: $\sum_{i=1}^{T} n_i^*$ where the n_i^*'s are the expected frequencies associated with π^*. For this study, SAS code utilizing PROC NLP was developed to compute π^* for the Proctor and intrusion-omission models. For the Proctor model and for the intrusion-omission model, constraints were imposed: $n_i^* \le n_i$ for each of the response patterns. The expected frequencies are functions of conditional probabilities, mixing proportions, and values of the variables. The conditional probabilities are expressed as functions of Γ, β_I, and β_O. The expression for the n_i^* is defined in PROC NLP. The input data set consisted of the response patterns. Using the NLP procedure, SAS built $\sum_{i=1}^{T} n_i^*$, the objective function.

Each constraint $n_i^* \le n_i$ was expressed using the conditional probabilities that are associated with the particular response pattern. For example, consider the expression of the constraint associated with the response pattern (1, 1, 0, 0) under the intrusion-omission model. The values for the variables are substituted into the expression for n_i^* to derive specific functions of β_I and β_O. The expression for the constraint in PROC NLP is:

$$N\{\pi_1^X \beta_I^2 (1-\beta_I)^2 + \pi_2^X (1-\beta_O)\beta_I (1-\beta_I)^2 + \pi_3^X (1-\beta_O)^2 (1-\beta_I)^2 + \pi_4^X \beta_O{}^2 (1-\beta_O)^2 + \pi_5^X (1-\beta_O)^2 \beta_O^2\} \leq n_{(1,1,0,0)} \tag{36}$$

Simulation Design

To achieve this study's purpose, simulations for a variety of scenarios were conducted. Based on varying the mixing proportions (other than the IUC mixing proportion) and IUC conditional probabilities, the major simulation scenarios were: equal mixing proportions/equal conditional probabilities, equal mixing proportions/unequal conditional probabilities, and unequal mixing proportions/unequal conditional probabilities. The simulation design allowed the assessment of π^* and the IUC mixing proportion for each of the mixing proportion/conditional probabilities scenarios while varying number of variables, sample sizes, IUC proportions, and measurement error rates. The specific simulation conditions are shown in Tables 2 and 3 when the number of variables were fixed at 4 and 6, respectively.

Table 2: Simulation Conditions
Extended Proctor and Intrusion/Omission Models
Number of Variables: 4

Size of Sample	240 (average of 15 per cell) 480 (average of 30 per cell) 960 (average of 60 per cell)	
Error Rate(s)		
Extended Proctor Model	.05, .10, .20	
Extended I/O Model		
Intrusion Error Rate Omission Error Rate	.20 .05	.05 .20
IUC Proportion	.10, .25, .40	
Mixing Proportions		
Equal		
IUC = .10	.18, .18, .18, .18, .18	
IUC = .25	.15, .15, .15, .15, .15	
IUC = .40	.12, .12, .12, .12, .12	
Unequal		
IUC = .10	.10, .10, .20, .25, .25	
IUC = .25	.10, .10, .15, .20, .20	
IUC = .40	.05, .05, .10, .20, .20	
Conditional Probabilities		
Equal	.3, .3, .3, .3	
Unequal	.20, .10, .15, .30	

Table 3: Simulation Conditions
Extended Proctor and Intrusion/Omission Models
Number of Variables: 6

Size of Sample	960 (average of 15 per cell) 1920 (average of 30 per cell) 3840 (average of 60 per cell)	
Error Rate(s)		
Extended Proctor Model	.05, .10, .20	
Extended I/O Model		
Intrusion Error Rate Omission Error Rate	.20 .05	.05 .20
IUC Proportion	.10, .25, .40	
Mixing Proportions		
Equal		
IUC = .10	.129, .129, .129, .129, .129, .129, .129	
IUC = .25	.107, .107, .107, .107, .107, .107, .107	
IUC = .40	.086, .086, .086, .086, .086, .086, .086	
Unequal		
IUC = .10	.05, .05, .10, .10, .20, .20, .20	
IUC = .25	.05, .05, .05, .10, .10, .20, .20	
IUC = .40	.05, .05, .10, .10, .10, .10, .10	
Conditional Probabilities		
Equal	.3, .3, .3, .3, .3, .3	
Unequal	.20, .10, .15, .30, .25, .40	

Rationale for Simulation Conditions. The minimum number of variables necessary to ensure identifiability of the models is four, and thus, “four” was chosen as one of the “number of variables” levels. Six was chosen as the other “number of variables” level as it was thought this would be different enough from four to reveal differences in the size of potential differences in estimates. In addition, any “number of variables” greater than six would entail a number of constraints greater than 64, which is the number of constraints for six variables. Other studies of latent class analysis (Hayek, 1978; Holt & Macready, 1989) noted that usually a mean sample size of 60 cases for each response pattern provide reasonable estimates, with smaller sample sizes providing biased estimates. For this research, average sample sizes of 15, 30, and 60 per cell were used, which translate into sample sizes of 240, 480, and 960 for the four-variable models; and 960, 1920, and 3840 for the six-variable models. The IUC proportions of .10, .25, and .40 were selected as they represent a variety of IUC proportions in real world settings. For the Proctor model, the error rates of .05, .10, and .20 were selected as they represent small, moderate, and large error rates. For the IO model, two error rate combinations were selected: .20 for omission error rate/.05 for intrusion error rate and .05 for omission error rate/.20 for intrusion error rate, reflecting contrasting omission and intrusion error levels. The mixing proportion (other than the IUC) and conditional probability values were selected to be similar to other simulation studies and also typical of real world problems.

Generation of Simulated Data. Let π^* denote the index-of-fit measure for the Proctor and intrusion-omission models. Let π_{IUC} denote the mixing proportion for the IUC in the extended Proctor and intrusion-omission models. The simulation compared these estimators by the mixing proportion/conditional probabilities scenarios, sample size, and the values for the measurement errors Γ, β_I, and β_O.

The steps used to implement the simulation comparing π_{IUC} and π^* for the Proctor models when there are four variables follow. (A similar approach would be used for the IO models.) First, a population of 100,000 "respondents" was generated using specified values of Γ (e.g., $\Gamma = .20$), the mixing proportions $\pi_1^X, \pi_2^X, \ldots, \pi_5^X, \pi_{IUC}$, and the IUC conditional probabilities. A uniform random number between 0 and 1 (denoted as R) based on the SAS RANUNI function was generated. Each record was assigned to a latent class as follows:

$$
\begin{aligned}
&\text{If } 0 < R \le \pi_1^X \text{ assign to latent class 1} \\
&\text{If } \pi_1^X < R \le \pi_1^X + \pi_2^X \text{ assign to latent class 2} \\
&\quad\vdots \\
&\text{If } \pi_1^X + \pi_2^X + \ldots + \pi_5^X < R \le 1 \text{ assign to the IUC}
\end{aligned}
$$

For those records falling into the first five latent classes, the conditional probabilities were generated for the variables in terms of Γ. For example, the second latent class corresponds to the scale type (1, 0, 0, 0). Thus, the conditional probabilities in terms of Γ are:

$$\pi_{12}^{\bar{A}X} = 1 - \Gamma$$

$$\pi_{12}^{\bar{B}X} = \Gamma$$

$$\pi_{12}^{\bar{C}X} = \Gamma \qquad (37)$$

$$\pi_{12}^{\bar{D}X} = \Gamma$$

A set of four uniform random numbers denoted $S_{12}, S_{22}, S_{32}, S_{42}$ falling in the interval between 0 and 1 was generated for the second latent class. These random numbers were used to assign the record values for the variables A, B, C, and D. The assignment for the second latent class was as follows:

If $0 < S_{12} \le 1 - \Gamma$ assign a value of 1 to item A
If $1 - \Gamma < S_{12} \le 1$ assign a value of 0 to item A
If $0 < S_{22} \le \Gamma$ assign a value of 1 to item B
If $\Gamma < S_{22} \le 1$ assign a value of 0 to item B
If $0 < S_{32} \le \Gamma$ assign a value of 1 to item C
If $\Gamma < S_{32} \le 1$ assign a value of 0 to item C
If $0 < S_{42} \le \Gamma$ assign a value of 1 to item D
If $\Gamma < S_{42} \le 1$ assign a value of 0 to item D

The assignment of values to the variables for records in the other four latent classes corresponding to the Guttman scale types was done in a similar fashion using the appropriate conditional probabilities.

For the IUC, a set of four uniform random numbers denoted $T_{10}, T_{20}, T_{30}, T_{40}$ falling in the interval between 0 and 1 was generated for the IUC. Let $\alpha, \beta, \chi,$ and δ represent the four IUC conditional probabilities. These random numbers were used to assign the record values for the four variables in the IUC as follows:

If $0 < T_{10} \leq \alpha$ assign a value of 1 to item A
If $\alpha < T_{10} \leq 1$ assign a value of 0 to item A
If $0 < T_{20} \leq \beta$ assign a value of 1 to item B
If $\beta < T_{20} \leq 1$ assign a value of 0 to item B
If $0 < T_{30} \leq \chi$ assign a value of 1 to item C
If $\chi < T_{30} \leq 1$ assign a value of 0 to item C
If $0 < T_{40} \leq \delta$ assign a value of 1 to item D
If $\delta < T_{40} \leq 1$ assign a value of 0 to item D

From the created "population", samples were drawn repeatedly. For each combination of condition levels, one-thousand samples were drawn. For each sample that was selected, observed frequencies were computed for the response patterns. These observed frequencies were input to the NLP procedure to compute the maximum likelihood estimates for the extended Proctor and intrusion-omission models. The observed frequencies were also input to the NLP procedure to maximize the sum of the expected frequencies under the Proctor and intrusion-omission π^* models.

Validation of Universe Creation and Estimation. To ensure that the generation of the universe was implemented correctly, two sets of frequencies were produced: a percentage distribution of the records falling into each of the latent classes, and a percentage distribution showing the crosstabulation of the items for each latent class.

To ensure that the SAS code implementing each estimation procedure is correct, well-known four-item data sets in latent class analysis were input into the SAS programs. The results from the SAS programs were compared to results where different software (LEM and Excel Solver) was used to estimate the same model parameters using the same data sets. The two sets of results for each model were found to be the same. The Stouffer-Toby data set (Goodman, 1975) was used for the validation of estimation pertaining to the extended Proctor, extended IO, and Goodman models. The Lazarsfeld-

Stouffer data set (Goodman, 1975) was used for the validation of the estimation of the Proctor π^* and IO π^* models.

To verify that PROC NLP runs in the simulations met convergence criteria, the SAS log was checked for messages indicating that convergent criteria were satisfied. Once simulations were completed, checks for the reasonableness of parameter estimates were conducted.

The next chapter discusses the results of the summarization of the IUC proportion and π^* estimates across the one-thousand samples.

CHAPTER 4

RESULTS

This chapter presents a summary of the simulation results for IUC and π^* estimation in four-variable and six-variable Proctor and IO models, error rate estimation in four-variable and six-variable Proctor and IO models, recovery of the mixing proportions and conditional probabilities, sampling distributions for the IUC proportion and π^*, and a real world data application. Simulation results are displayed in Appendix Tables A1 through A48.

IUC and π^* Estimation in Four-Variable Proctor Models

For the four-variable extended Proctor and Proctor π^* models, the patterns in the IUC proportion and π^* estimates are overall very similar across variations in IUC proportion, error rate, and sample size across the three scenarios of interest (Tables A1, A5, and A9). For example, IUC proportion and π^* estimates from the equal mixing proportions/unequal conditional probabilities scenario presented in Table A5 are graphed in Figures 1, 2, and 3 for the IUC proportions of .10, .25, and .40, respectively.

Extended Proctor Model. Across all error rates and sample sizes, the IUC proportion is larger than the true IUC proportion of .10, with estimates ranging from .153 to .209 (Figure 1). Within each error rate, usually the IUC proportions become closer to .10 as the sample size increases. For example, for the IUC proportion of .10 and error rate of .10, the IUC proportions are .209, .178, and .158, corresponding to the sample sizes of 240, 480, and 960. In general, for a given sample size, the positive bias increases with the increase in the error rate and, for a given error rate, the bias decreases with sample size.

The extended Proctor model overestimates the true IUC proportion for all error rate/sample size combinations, with estimates ranging from .263 to .298 (Figure 2). The bias generally increases with the error rate level.

The bias in the extended Proctor model IUC proportion is quite small for the error rates of .05 with estimates of .401, .398, and .412 and .10 with estimates of .391, .389, and .397; however, the bias is negative for the error rate of .20 with estimates of .352, .342, and .337 (Figure 3).

Proctor π^* Model. For the IUC proportion of .10 and error rate of .05, the π^* estimate is .099, which is very slightly less than the true value of .10 (Figure 1). As the sample size increases, the π^* estimate becomes smaller, with estimates of .090 and .075 corresponding to sample sizes of 480 and 960, respectively.

For all error rate and sample size combinations, the Proctor π^* model estimates are less than the true IUC proportions (Figures 2 and 3). As the sample size increases for a given error rate, the π^* estimate decreases. For example, for the IUC proportion of .25 and error rate of .05, π^* estimates are .140, .135, and .119 for the sample sizes of 240, 480, and 960, respectively.

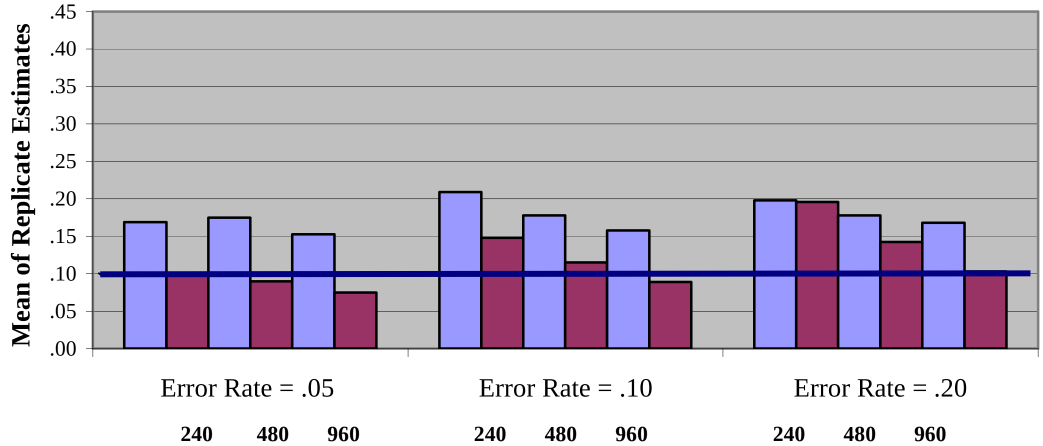

Figure 1. IUC (blue) and π^* (red) estimates for the four-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10.

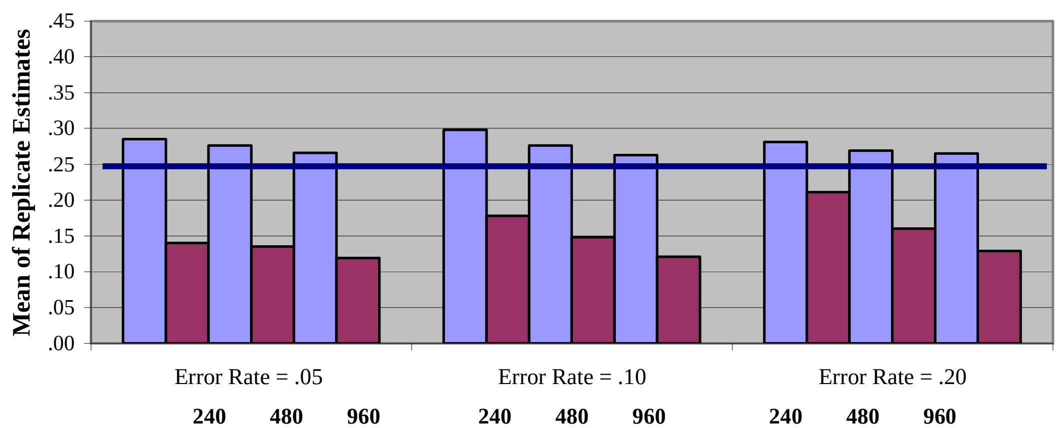

Figure 2. IUC (blue) and π^* (red) estimates for the four-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .25.

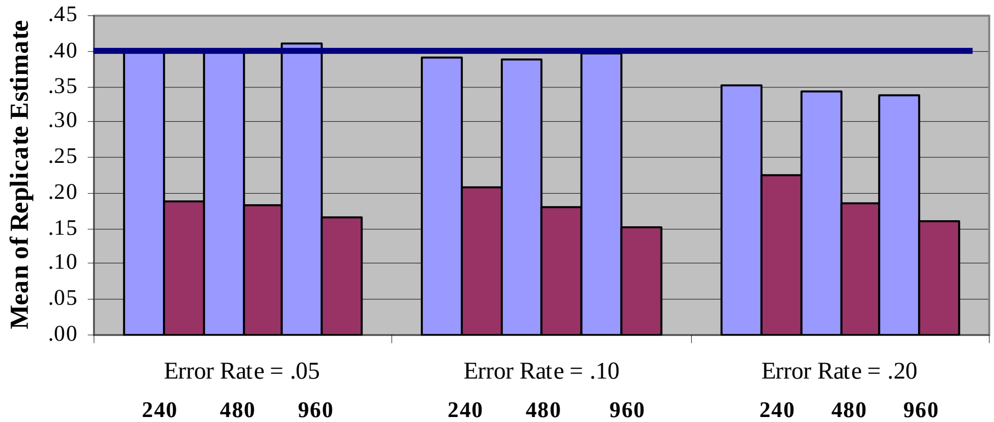

Figure 3. IUC (blue) and π^* (red) estimates for the four-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40

Although the three scenarios are similar, it should be noted that for the IUC proportions of .10 and .40, the IUC proportion under the unequal mixing proportions/unequal conditional probabilities scenario is somewhat less than the corresponding estimate for each of the equal mixing proportions scenarios. Also, note that the estimated IUC mean for the Goodman model is always considerably larger than the corresponding estimates for the extended model and two-point mixture models. This holds true for all combinations of conditions utilized in this study.

IUC and π^* Estimation in Six-Variable Proctor Models

Extended Proctor Model. Overall, the extended Proctor model performs in a similar manner across variations in IUC proportion, error rate, and sample size across all three scenarios of interest (Tables A13, A17, and A21). For example, estimates from the equal mixing proportions/unequal conditional probabilities scenario presented in Table A17 are graphed in Figures 4, 5, and 6 for the IUC proportions of .10, .25, and .40, respectively.

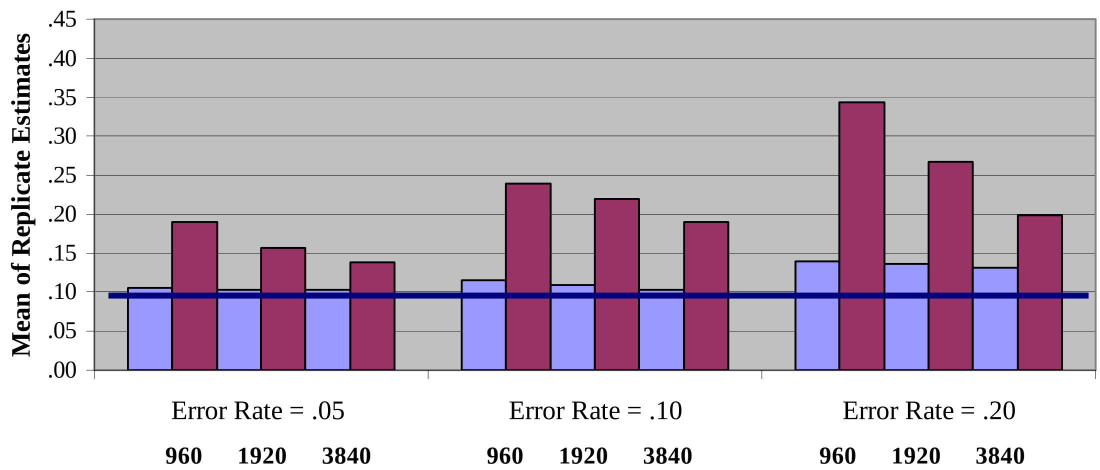

Figure 4. IUC (blue) and π^* (red) estimates for the six-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10.

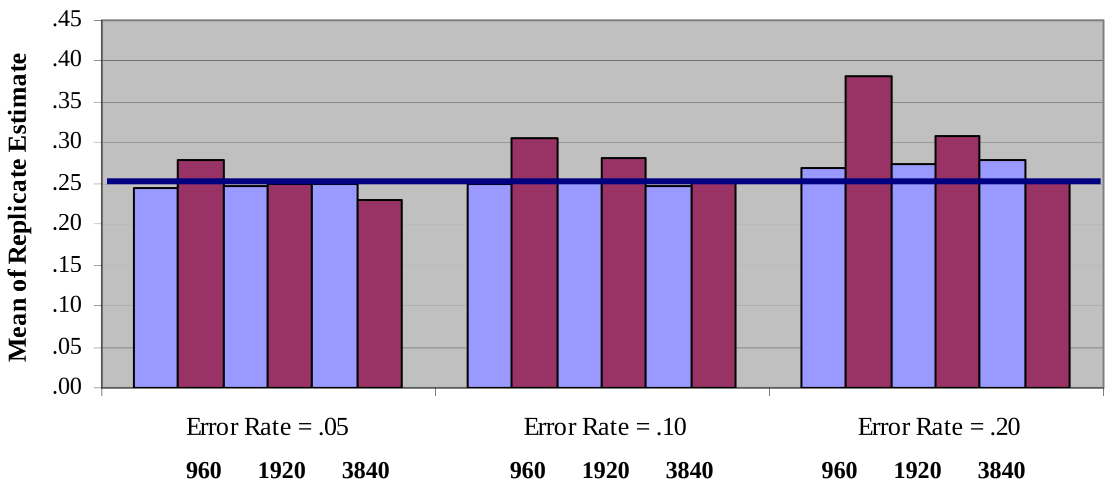

Figure 5. IUC (blue) and π^* (red) estimates for the six-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .25.

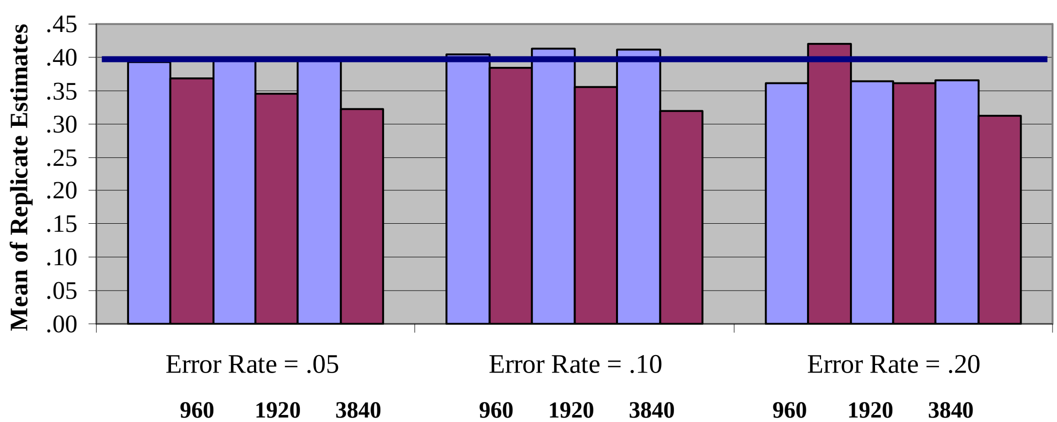

Figure 6. IUC (blue) and π^* (red) estimates for the six-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40.

The difference between the true IUC proportion and IUC proportion estimate is very small across all IUC proportions of .10, .25, and .40 when the error rate is .05 or .10 (Figures 4, 5, and 6). In contrast, when the error rate is .20 for the IUC proportion of .10, the extended Proctor model overestimates the true IUC proportion; those IUC proportions are .140, .136, and .131 for the three sample sizes of 960, 1920, and 3840, respectively. When the error rate is .20 for the IUC proportion of .25, the IUC proportions are slightly overestimated (.269, .273, and .280). When the error rate is .20 for the IUC proportion of .40, the extended Proctor model underestimates the true IUC proportion (.361, .364, and .366 versus .400).

It should be noted that for the IUC proportion of .10 and error rate of .20, the extended Proctor model IUC proportion estimates under unequal mixing proportions/unequal conditional probabilities (.103, .107, and .106 in Table A21) are much closer to .10 than the corresponding means under equal mixing proportions/unequal conditional probabilities (.140, .136, and .131 in Table A17) and those under equal mixing proportions/equal conditional probabilities (.136, .137, and .128 in Table A13).

Proctor π^* Model. The π^* estimate is larger than .10 across all error rates and sample sizes with estimates ranging from .139 to .343 (Figure 4). For a given sample size, the estimated means increase with the size of the error rate. For a given error rate, the π^* estimates decrease (and become closer to .10) as the sample size increases. The π^* estimates range from .230 to .381 (Figure 5). As with the IUC proportion of .10, for a given error rate the π^* estimates decrease as the sample size increases. Across all error rates and sample sizes (with the exception of .420) the π^* estimates are less than the true IUC proportion of .40 (Figure 6). As with the IUC proportions of .10 and .25, for a given error rate, the π^* estimates decrease as the sample size increases for the IUC proportion of .40.

Comparison of Four-Variable and Six-Variable Proctor Models. For the extended Proctor model with IUC proportions of .25 (Figures 2 and 5) and .40 (Figures 3 and 6), there are, in general, small differences in the IUC proportions between the four-variable and six-variable models. For the IUC proportion of .10 (Figures 1 and 4), the differences are more substantial with the six-variable models showing excellent recovery (except with the error rate of .20 in the equal mixing proportions scenarios) than the four-variable models (where estimates range from .132 to .229).

The performance of the Proctor π^* model is more sensitive to the number of variables. It appears that the difference in the performance of the extended Proctor model and the Proctor π^* model lessens as the number of variables increases.

Error Rate Estimation in Four-Variable Proctor Models

For both the extended Proctor and Proctor π^* models, the estimated error rate means for the three scenarios differ very little, overall, from each other (Tables A2, A6, and A10). For example, estimates from the equal mixing proportions/unequal conditional probabilities scenario presented in Table A6 are graphed in Figures 7, 8, and 9 for the IUC proportions of .10, .25, and .40, respectively.

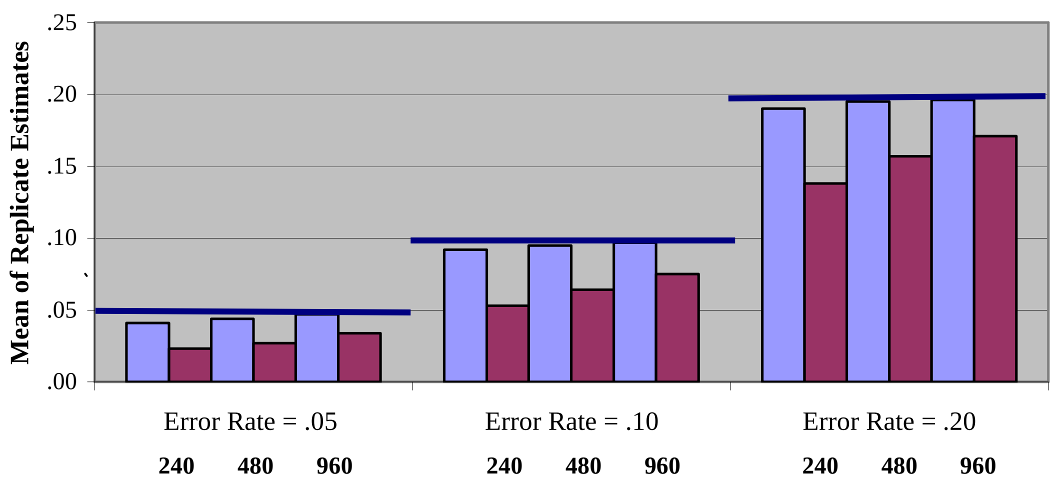

Figure 7. Error rate estimates for the four-variable extended Proctor (blue) and Proctor π^* (red) models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10.

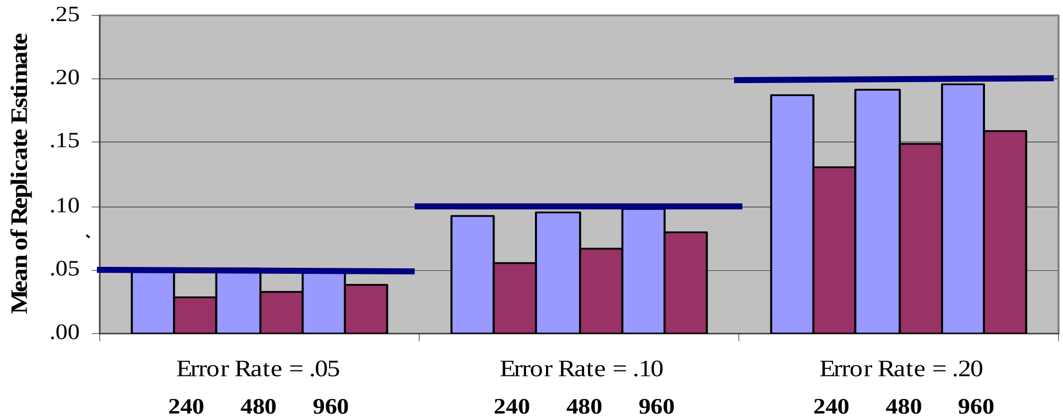

Figure 8. Error rate estimates for the four-variable extended Proctor (blue) and Proctor π^* (red) models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .25.

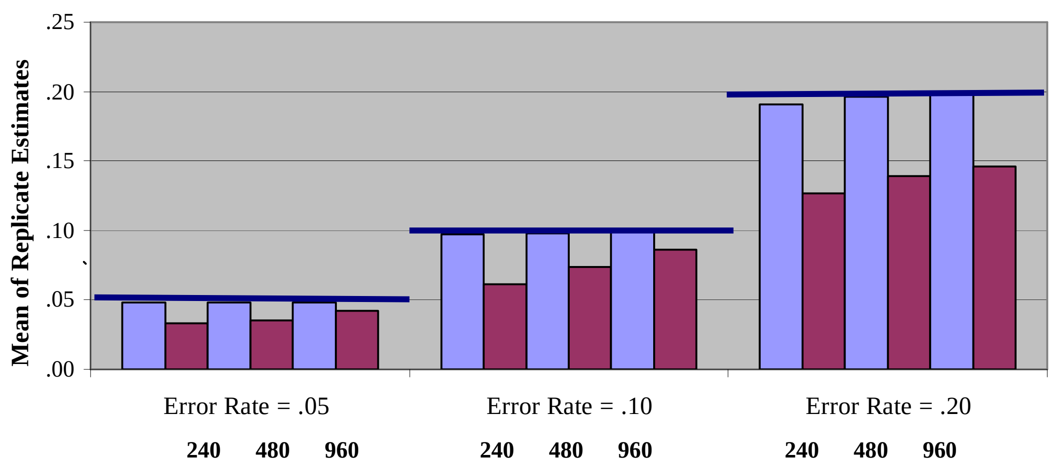

Figure 9. Error rate estimates for the four-variable extended Proctor (blue) and Proctor π^* (red) models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40.

Extended Proctor Model. For the error rate of .05, the extended Proctor model produces an estimate of .041, an underestimate of the true error rate (Figure 7). However, for a sample size of 960, the estimate is .047. It appears that with larger sample sizes the extended Proctor error rate estimated means would converge to the true value.

Estimates are similar across sample sizes and error rates with .048, .051, and .050 for error rate of .05; .093, .095, and .098 for error rate of .10; and .188, .192, and .196 for error rate of .20 (Figure 8). The error rate recovery results for the extended Proctor model in Figure 9 are quite similar to those in Figure 11 (.048, .048, and .048 for error rate of .05; .097, .098, and .099 for error rate of .10; and .191, .196, and .199 for error rate of .20).

Proctor π^* Model. Where the IUC proportion is .10, the error rate is substantially less than the true error rate (Figure 7). For example, for the sample size of 240, the error rates are .023, .053, and .138 and for the sample size of 960, the error rates equal .034, .075, and .171.

The error rates are less than the true error rates at the sample size of 240 with estimates of .028, .055, and .130; 480 with estimates of .032, .067, and .149; and 960 with estimates of .038, .079, and .159 (Figure 8). The error rates are also less than the true error rates (Figure 9). For example, at the sample size of 960, estimates are .042, .086, and .146.

The exception to the overall similarity of the results from the three scenarios is as follows. For the Proctor π^* model, under the condition of equal mixing proportions/unequal conditional probabilities and an error rate of .20, the error rate estimates are somewhat less than those under the other two scenarios (Tables A2, A6, and A10).

Error Rate Estimation in Six-Variable Proctor Models

Extended Proctor Model. The recovery of the error rates for the six-variable extended Proctor model is excellent across all IUC proportions, error rates, and sample sizes for all three scenarios of interest (Tables A14, A18, and A22). For example, error rate estimates from the equal mixing proportions/unequal conditional probabilities scenario are graphed in Figures 10, 11, and 12 for the IUC proportions of .10, .25, and .40, respectively.

Proctor π^* Model. The Proctor π^* model error rates are less than the true error rates for all IUC proportions, error rates, and sample sizes to a similar degree for all three scenarios (estimates ranging widely from .021 to .049 for error rate of .05; .061 to .097 for error rate of .10; and .136 to .195 for error rate of .20) with a few exceptions. These few estimates occur only in the equal mixing proportions/equal conditional probabilities scenario for the IUC proportion of .40 with the sample size of 3840 and their values are reasonably close to the true error rates (Table A14).

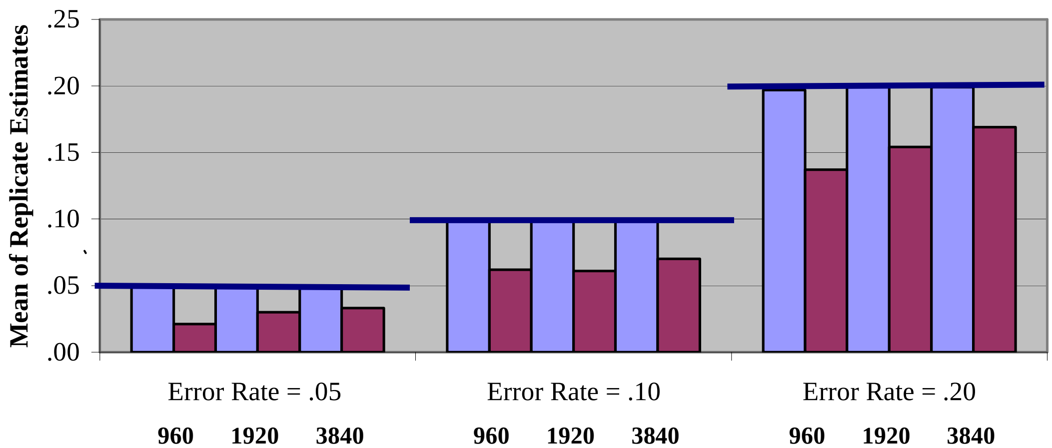

Figure 10. Error rate estimates for the six-variable extended Proctor (blue) and Proctor π^* (red) models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10.

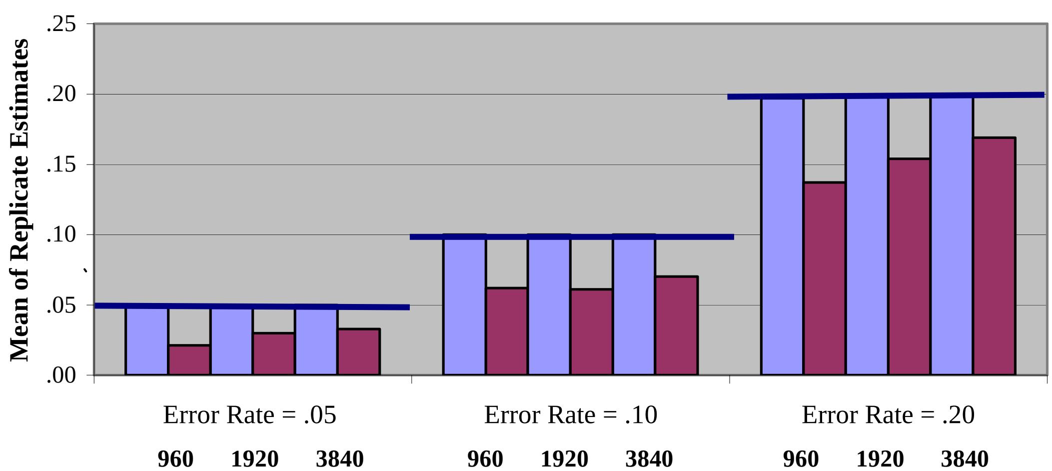

Figure 11. Error rate estimates for the six-variable extended Proctor (blue) and Proctor π^* (red) models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .25.

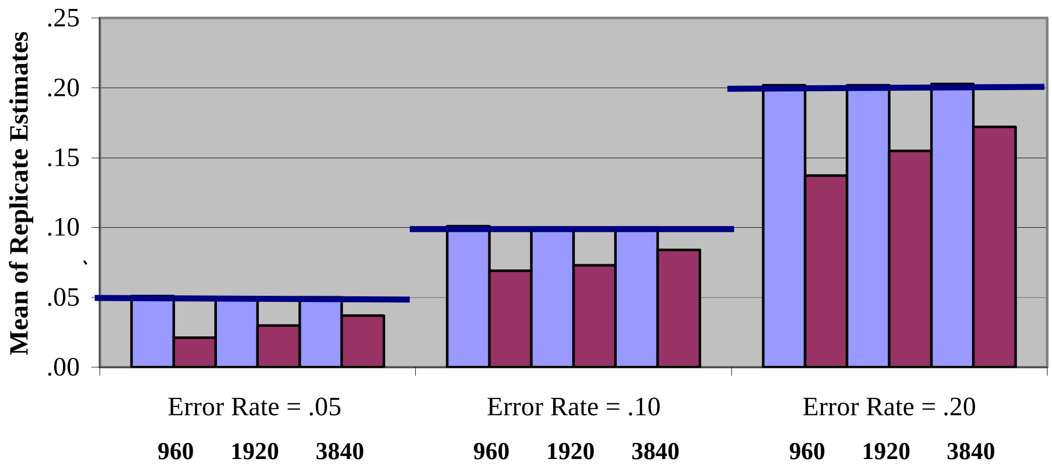

Figure 12. Error rate estimates for the six-variable extended Proctor (blue) and Proctor π^* (red) models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40.

Comparison of Four-Variable and Six-Variable Proctor Models. The patterns found in the error rate estimates for the four-variable Proctor π^* model are similar to the six-variable Proctor π^* model.

IUC and π^* Estimation in Four-Variable IO Models

Extended IO Model. The three scenarios of interest are very similar with regard to their patterns for the extended IO and IO π^* model estimates (Tables A25, A29, and A33). For example, estimates from the equal mixing proportions/unequal conditional probabilities scenario presented in Table A29 are graphed in Figures 13, 14, and 15.

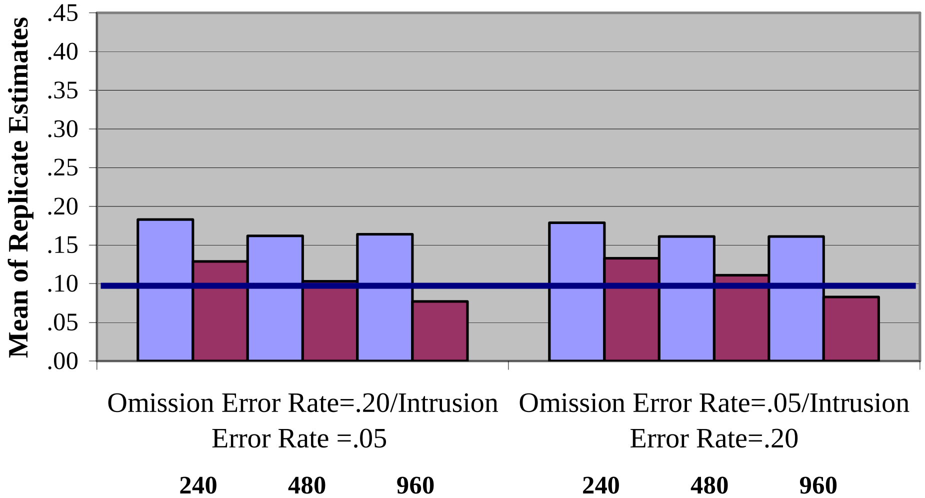

Figure 13. IUC (blue) and π^* (red) estimates for the four-variable extended IO and IO π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10.

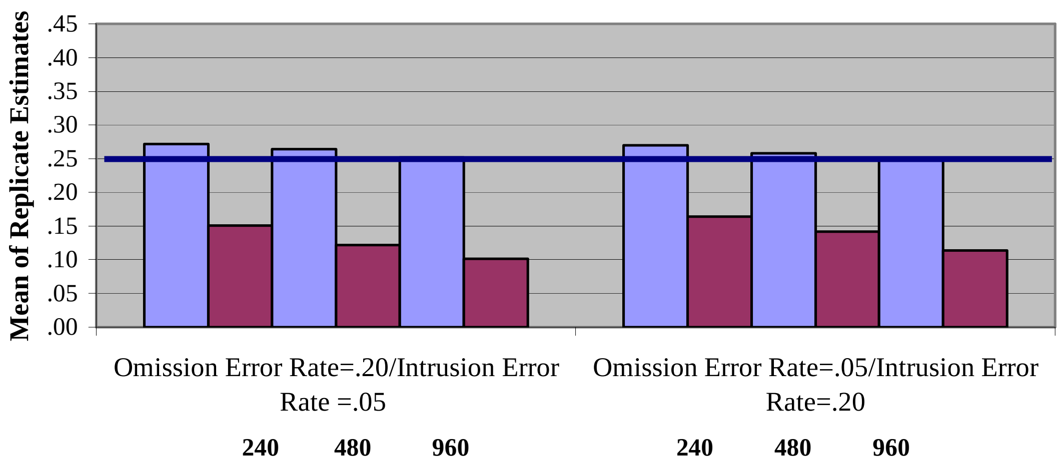

Figure 14. IUC (blue) and π^* (red) estimates for the four-variable extended IO and IO π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .25.

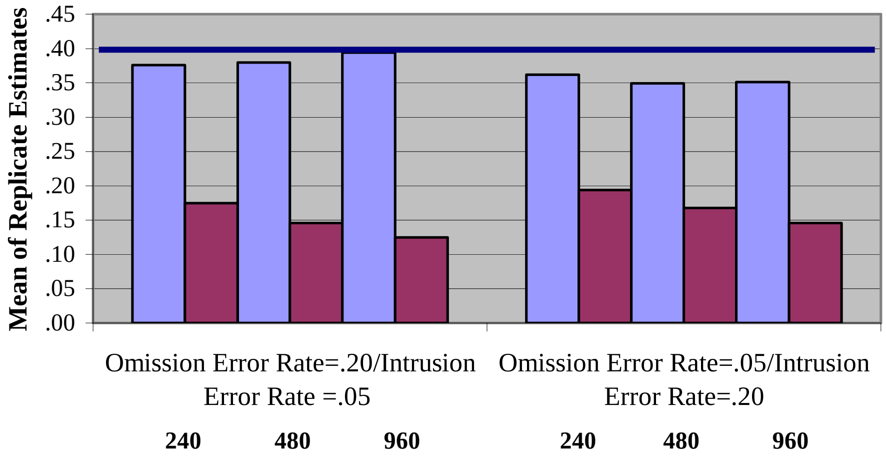

Figure 15. IUC (blue) and π^* (red) estimates for the four-variable extended IO and IO π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40.

The IUC proportions are substantially larger than the true IUC proportion for all sample sizes and for both omission/intrusion error rate combinations with estimates ranging from .161 to .183 (Figure 13). The model usually overestimates the true IUC proportion with estimates ranging from .249 to .272 (Figure 14). However, the degree of the overestimation is considerably less than for the IUC proportion of .10. The model underestimates the true IUC proportion with estimates ranging from .349 to .394 (Figure 15).

IO π^* Model. The π^* estimates are larger than the true IUC proportion for the sample size of 240 with estimates being .129 and .133 (Figure 13). When the sample size is 480, the IUC proportions are close to the IUC proportion (estimates being .103 and .111). When the sample size is 960, the IUC proportions are .077 for the omission error rate of .20 with intrusion error rate of .05 combination and .083 for the omission rate of

.05 with intrusion rate of .20 combination. The π^* estimates are less than the true IUC proportion for the sample size of 240 with estimates .151 and .164, and this relationship becomes more pronounced as the sample size increases with estimates .101 and .114 for the sample size of 960 (Figure 14). The extent of the IO π^* model's discrepancy with the true IUC proportion is at its greatest for the IUC proportion of .40 (with estimates ranging from .194 to .125).

Although the estimated means in Table A33 closely parallel those existing for the other scenarios (Tables A25 and A29), two exceptions for the extended IO model should be noted. Table A33 shows that the extended IO model for the true IUC proportion of .10 produces estimates with a larger positive bias (with estimates ranging from .121 to .154) than the other two scenarios (.161 to .190). Also, for the IUC proportion of .40, the extended IO model produces IUC proportions with a larger negative bias (with estimates ranging from .314 to .333) than the other two scenarios (.349 to .394).

IUC and π^* Estimation in Six-Variable IO Models

The three scenarios of interest are overall very similar with regard to their patterns for the extended IO model estimates and are quite similar with regard to their patterns for the IO π^* model estimates (Tables A37, A41, and A45). For example, estimates from the equal mixing proportions with unequal conditional probabilities scenario presented in Table A41 are graphed in Figures 16, 17, and 18 for the IUC proportions of .10, .25, and .40, respectively.

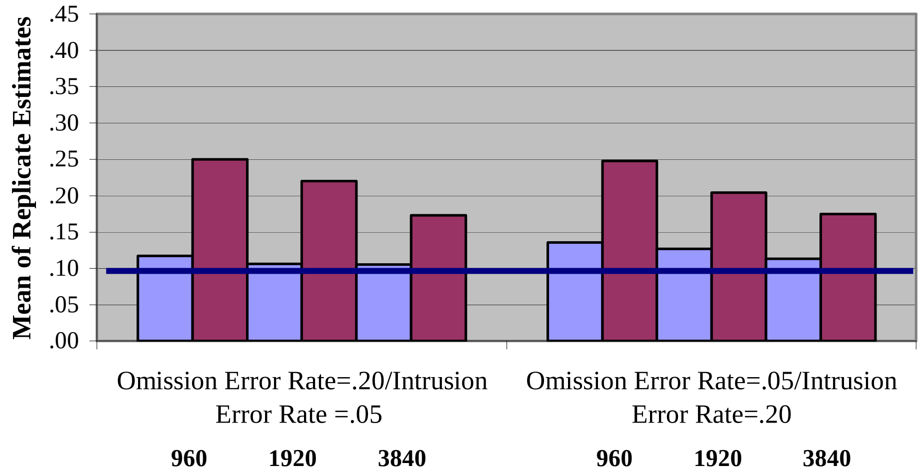

Figure 16. IUC (blue) and π^* (red) estimates for the six-variable extended IO and IO π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10.

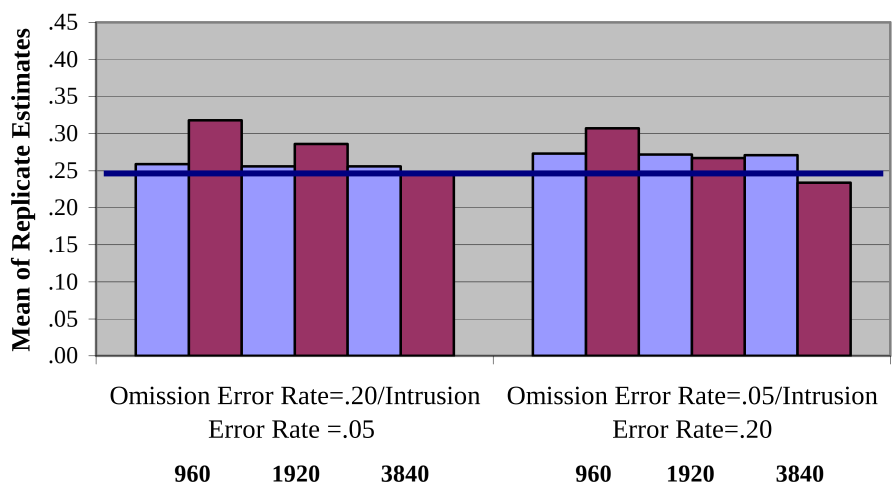

Figure 17. IUC (blue) and π^* (red) estimates for the six-variable extended IO and IO π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .25.

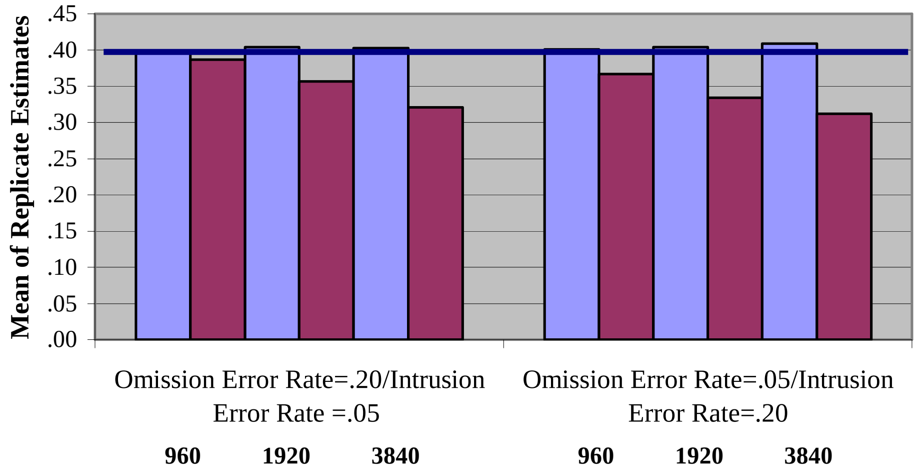

Figure 18. IUC (blue) and π^* (red) estimates for the six-variable extended IO and IO π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40.

Extended IO Model. The extended IO model overestimates the true IUC proportion of 10 (Figure 16). The IUC proportions are .117, .106, and .105 for the omission rate of .20 with intrusion rate of .05 combination; and .136, .127, and .113 for the omission error rate of .05 with intrusion error rate of .20. The IUC proportions for the omission error rate of .20 with intrusion error rate of .05 combination (.259, .256, and .256 for IUC of .25 and .399, .404, and .403 for IUC of .40) and the IUC proportions for the omission error rate of .05 with intrusion error rate of .20 combination (.273, .272, and .271 for IUC proportion of .25; and .401, .404, and .409 for IUC proportion of .40) are all quite close to the true IUC proportion (Figures 17 and 18). Overall, the extended IO model demonstrates good recovery in the estimation of the IUC proportion.

IO π^* Model. The π^* estimates are much larger than the true IUC proportion of .10 for all sample sizes (Figure 16). For the omission rate of .20/intrusion rate of .05 combination, the π^* estimates are .259, .220, and .173; for the omission rate of .05/intrusion combination rate of .20, the π^* estimates are .248, .204, and .175. The magnitude of the estimates decreases strikingly as sample size increases.

The π^* estimates at the sample size of 960 are larger than the true IUC proportion, but at the sample size of 3840, the π^* estimate is less than the true IUC proportion: .318, .286, and .245 for the omission rate of .20/intrusion rate of .05 combination; and .307, .267, and .234 for the omission rate of .05/intrusion rate of .20 combination (Figure 17).

The π^* estimates are less than the true IUC proportion: .387, .357, and .321 for the omission rate of .20/intrusion rate of .05 combination; and .367, .334, and .312 for the omission rate of .05/intrusion rate of .20 combination (Figure 18). Again, the magnitude of the discrepancy increases as the sample size increases.

Comparison of Four-Variable to Six-Variable IO Models. The recovery of the IUC proportions of .25 and .40 for both the four-variable and six-variable extended IO models is overall very good. For the IUC proportion of .10, the estimated means of the six-variable extended IO model (.091 to .136) are overall much closer to the true IUC proportion than the four-variable model (.121 to .190).

For the IUC proportion of .10, the estimated six-variable IO π^* means (.154 to .285) are larger in contrast to the four-variable π^* model means (.072 to .142). For the IUC proportions of .25 and .40, the estimated six-variable IO π^* means are closer to the true values than the four-variable π^* model means.

Error Rate Estimation in Four-Variable IO Models

For the extended IO model, the error rates for the omission error rate of .20 with intrusion error rate of .05 and omission error rate of .05 with intrusion error rate of .20 combinations are very similar for each IUC proportion across all three scenarios of interest (Tables A26, A30, and A34). For the IO π^* model, the patterns in the error rates are for the most part similar for each IUC proportion within each scenario for both error rate combinations. For example, estimates from the equal mixing proportions/unequal conditional probabilities scenario for the omission error rate of .20 with intrusion error rate of .05 combination for the IUC proportion of .25, and for the omission error rate of .05 with intrusion error rate of .20 combination for the IUC proportion of .10 are graphed in Figures 19 and 20, respectively.

Extended IO Model. For the omission error rate of .20 with intrusion error rate of .05 combination, the omission error rates are .182, .192, and .197, and the intrusion error rates are .054, .051, and .050 (Figure 19). For the omission error rate of .05 with intrusion error rate of .20 combination, the omission error rates are .048, .047, and .048, and the intrusion error rates are .200, .204, and .204 (Figure 20).

IO π^* Model. For the omission error rate of .20 with intrusion error rate of .05 combination, the omission error rates are .126, .139, and .147, and the intrusion error rates are .065, .074, and .082 (Figure 19). For the omission error rate of .05 with intrusion error rate of .20 combination, the omission error rates are .025, .030, and .036, and the intrusion error rates are .138, .147, and .154 (Figure 20).

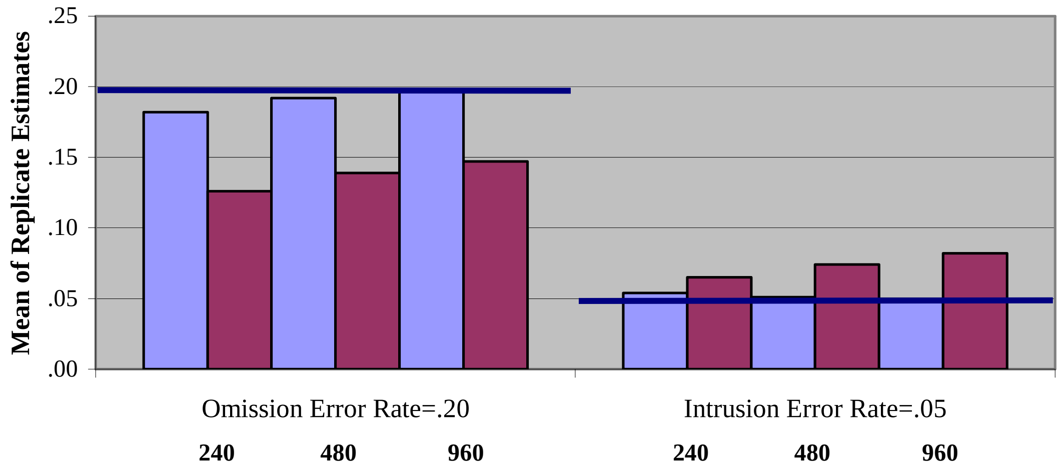

Figure 19. Error rate estimates for the four-variable extended IO (blue) and IO π^* (red) models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .25.

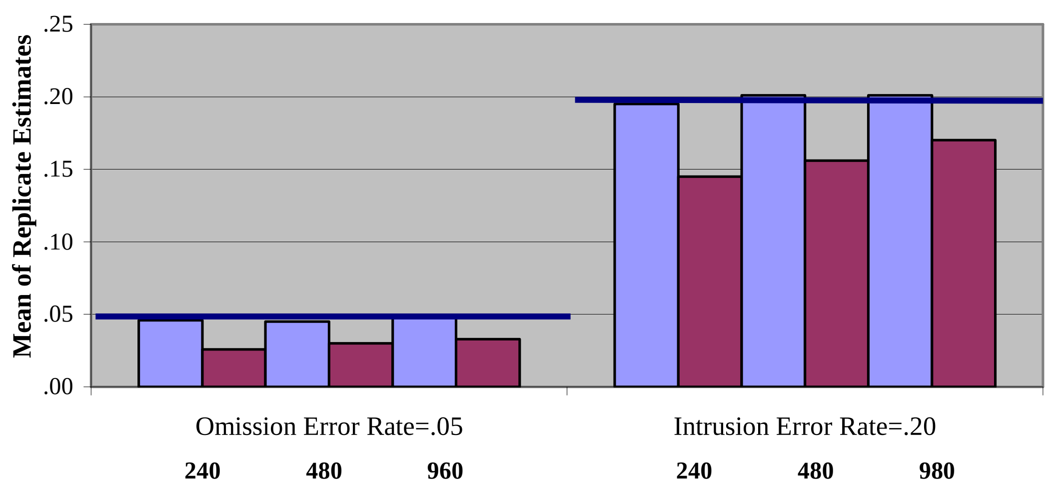

Figure 20. Error rate estimates for the four-variable extended IO (blue) and IO π^* (red) models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10.

Error Rate Estimation in Six-Variable IO Models

The performance of the extended IO model in the recovery of omission and intrusion error rates is excellent across all intrusion/omission error rate combinations, IUC proportions, and sample sizes for all three scenarios of interest, as displayed in Tables A38, A42, and A46. For the IO π^* model, the patterns in the error rates are very similar for each IUC proportion within each scenario for both error rate combinations. For example, estimates from the equal mixing proportions/unequal conditional probabilities scenario for the omission error rate of .20 with intrusion error rate of .05 combination for the IUC proportion of .25 are graphed in Figure 21.

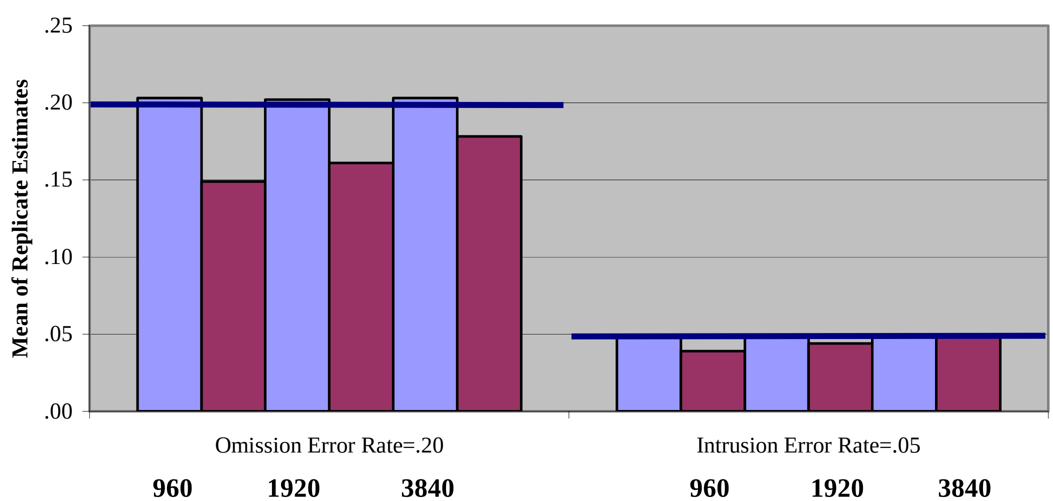

Figure 21. Error rate estimates for the six-variable extended IO (blue) and IO π^* (red) models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .25.

Extended IO Model. For the omission error rate of .20 with intrusion error rate of .05 combination, the omission error rates are .203, .202, and .203, and the intrusion error rates are .048, .049, and .049 (Figure 21). The error rate estimates for the omission error rate of .05 with intrusion error rate of .20 combination are very similar to the estimates shown in Figure 20.

IO π^* Model. For the omission error rate of .20 with intrusion error rate of .05 combination, the omission error rates are .149, .161, and .178, and the intrusion error rates are .039, .044, and .049 (Figure 21). The error rate estimates for the omission error rate of .05 with intrusion error rate of .20 combination are very similar to the estimates shown in Figure 20.

Comparison of Four-Variable to Six-Variable IO Models. For the omission error rate of .20 with intrusion error rate of .05 combination, the six-variable extended IO model error rate recovery is slightly to somewhat better, and the four-variable IO π^* models' intrusion rate is usually greater than the true rate. For the omission error rate of .05 with intrusion error rate of .20 combination, the patterns in the error rates are very similar for both four-variable and six-variable extended IO and IO π^* models.

Recovery of Mixing Proportions and Conditional Probabilities

This research now focuses on the recovery of the mixing proportions associated with the Guttman scale types and the IUC conditional probabilities for the extended models. The mixing proportions estimated for the π^* models (where the first latent class of the two-point mixture model is described by the Proctor or IO model) are not directly comparable to the mixing proportions associated with the extended models. Note that the Proctor and IO π^* models do not possess IUC conditional probabilities.

Four-Variable Models. The mixing proportion results for the three scenarios of interest for the four-variable extended Proctor model are shown in Tables A3, A7, and A11; and these results for the four-variable extended IO model are shown in Tables A27, A31, and A35. First, when the difference between the estimated IUC proportion mean and the true IUC proportion is close to zero, all mixing proportions associated with the Guttman scales are very close to their true values. Second, when the estimated IUC proportion mean is larger than its true IUC proportion, the differences between the estimated mixing proportions and their corresponding true mixing proportions are of a similar extent (except for the first mixing proportion, whose difference is greater). Third, when the estimated IUC proportion mean is smaller than its true IUC proportion, the largest positive difference between the estimated mixing proportion mean and its true mixing proportion usually occurs for the first mixing proportion.

The conditional probability results for the three scenarios of interest for the four-variable extended Proctor model are shown in Tables A4, A8, and A12, and these results for the four-variable extended IO model are shown in Tables A28, A32, and A36. One consistent finding was that for the IUC proportion of .10, the extended Proctor model produces conditional probabilities with a positive bias and for some IUC proportion/error rate combinations the size of the bias can be large. For example, under the unequal mixing proportions/unequal conditional probabilities scenario, with an error rate of .05 and a sample size of 240, the estimated means are .270, .216, .252, and .413 when the true conditional probabilities are .200, .100, .150, and .300.

The relationships in the conditional probabilities found in the Proctor model also hold mostly for the extended IO model. The extended IO model tends to underestimate the IUC mixing proportion when the true IUC proportion is .40. The effect of this underestimation for the other five mixing proportions is that the estimated mixing proportion means are consistently larger than the first mixing proportion's true value. When the IUC proportion is .10, the estimated mixing proportion means are consistently larger than the true value for the first latent class.

For the extended IO model, for the IUC proportion of .10, the extended IO model overestimates the true conditional probabilities for both omission/intrusion error rate combinations and all sample sizes. For the IUC proportions of .25 and .40, there appear to be no consistent relationships between the estimated conditional probability means and their true values.

Six-Variable Models. Tables A15, A19, and A23 show the mixing proportion results for the three scenarios of interest for the six-variable extended Proctor model, and Tables A39, A43, and A47 show these results for the six-variable extended IO model. The recovery of the mixing proportions in both the six-variable extended Proctor and IO models is excellent. It should be noted, however, how the overestimation or underestimation of the IUC mixing proportion affects the estimated first mixing proportions. When the estimated IUC mixing proportion is larger than the true IUC mixing proportion, typically the estimated first mixing proportion mean is less than the corresponding true mixing proportion. When the estimated IUC mixing proportion is less than the true IUC mixing proportion, typically the estimated first mixing proportion mean is larger than the corresponding true mixing proportion.

Tables A16, A20, and A24 show the conditional probability results for the three scenarios of interest for the six-variable extended Proctor model, and Tables A40, A44, and A48 show these results for the six-variable extended IO model. The recovery of the IUC conditional probabilities is very good for the IUC proportions of .25 and .40. However, for the IUC proportion of .10 at smaller sample sizes, the models tend to produce conditional probability estimates that are larger than the corresponding true conditional probabilities.

Sampling Distributions for Proctor Models

Assessment of the difference of the mean of the replicate estimates and the true value of the IUC mixing proportion is the focus of this research. However, it is also of interest to observe how the replicate estimates are distributed about their mean. Since different likelihood functions are being maximized for the extended and π^* models, the sampling distributions of the replicate estimates might differ under the two models. Figures 22 through 29 show the sampling distributions of the replicate estimates for the extended Proctor and Proctor π^* models for various number of variables/IUC proportion/sample size/error rate combinations. These illustrative histograms displaying the sampling distributions of the replicate estimates for the extended IO and IO π^* models allow the comparison of the two models by using the same number of variables/IUC proportion/sample size/error rate combinations. The characteristics of the extended IO and IO π^* distributions were very similar to those of the extended Proctor and Proctor π^* models and thus not included.

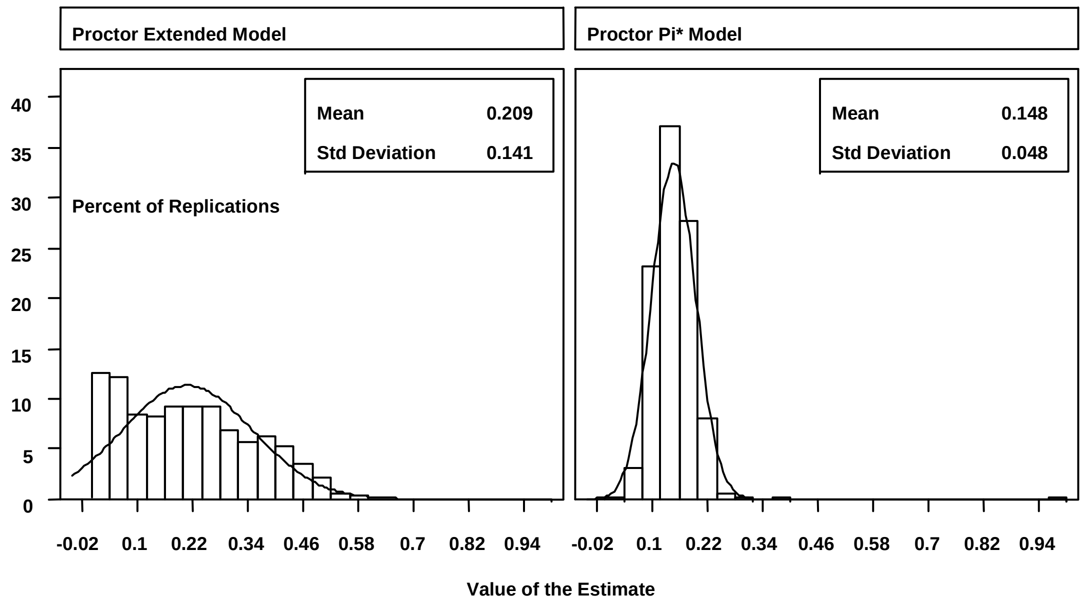

Figure 22. Distributions of IUC proportion and π^* for the four-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10, error rate is .10, and sample size is 240.

Figure 22 shows the sampling distributions for the four-variable extended Proctor and Proctor π^* models for the IUC proportion = .10 with error rate = .10 and sample size = 240 combination for the equal mixing proportions/unequal conditional probabilities scenario. The sampling distribution of the IUC mixing proportion estimates under the extended Proctor model is somewhat skewed to the right. Since most of the class intervals contain between 5% and 10% of the distribution, there is a wide spread to the distribution. The histogram for the Proctor π^* model is in sharp contrast as the distribution for this model shows a very sharp peak. Its distribution is much less variable than the corresponding extended Proctor model and this is reflected in the standard deviations of the replicate estimates for the two models (.141 for the extended model versus .048 for the π^* model). Note that over 30% of the π^* model's distribution falls within the class interval containing the mean of the replicates.

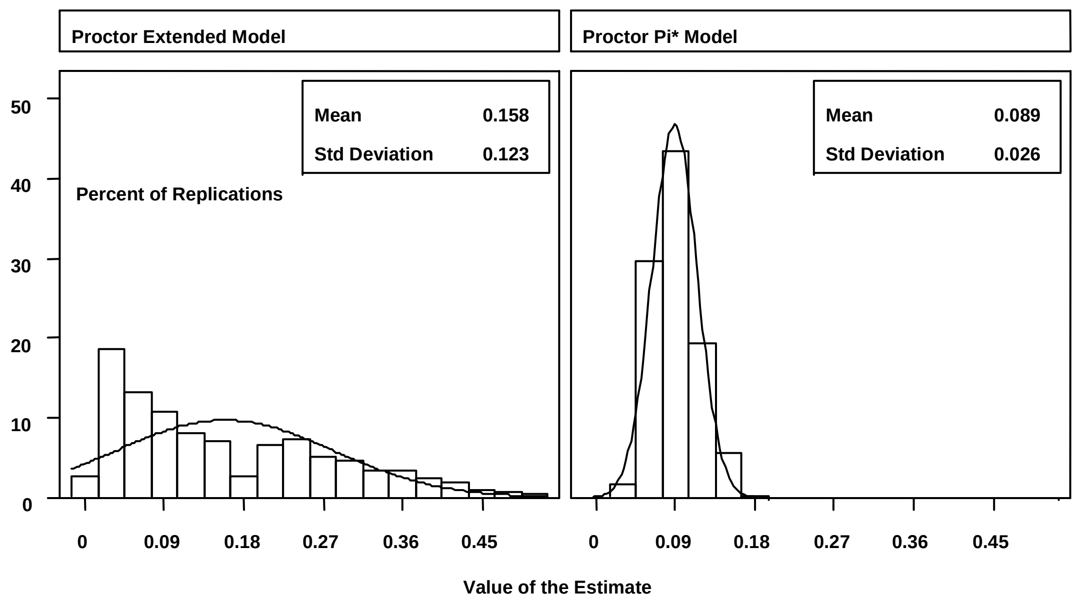

Figure 23. Distributions of IUC proportion and π^* for the four-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10, error rate is .10, and sample size is 960.

Figure 23 shows the sampling distributions for the four-variable extended Proctor and Proctor π^* models for the IUC proportion = .10 with error rate = .10 and sample size = 960 combination for the equal mixing proportions/unequal conditional probabilities scenario. Although the sample size has increased from 240 to 960, the standard deviation has only declined from .141 to .123 for the extended Proctor model. For the Proctor π^* model, the distribution becomes less variable with over 40% of the replicate estimates falling within the interval containing the mean. The standard deviation decreases by almost 50% when the sample size increases from 240 to 960. Also for the π^* model, the distribution has shifted so that the mean of the estimates has decreased from .148 to .089, which is less than the true IUC proportion.

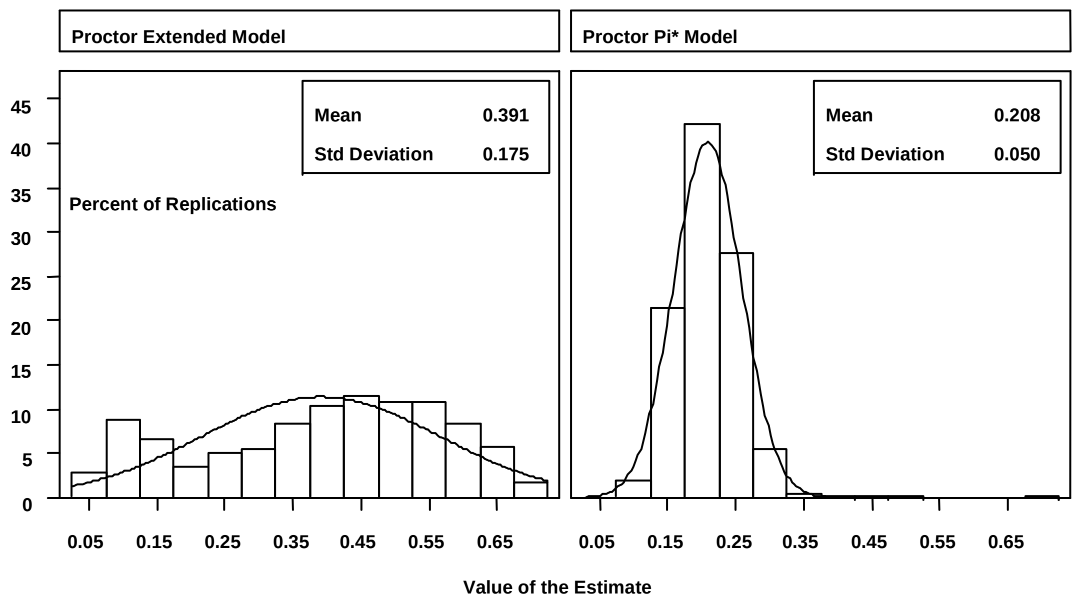

Figure 24. Distributions of IUC proportion and π^* for the four-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40, error rate is .10, and sample size is 240.

Figure 24 shows the sampling distributions for the four-variable extended Proctor and Proctor π^* models for the IUC proportion = .40 with error rate = .10 and sample size = 240 combination for the equal mixing proportions/unequal conditional probabilities scenario. The distribution for the extended Proctor is somewhat skewed to the left and shows a large variability (standard deviation of .175). No class interval contains slightly more than 10% of the distribution. The mean of the distribution is .391, which is very close to the true IUC proportion of .40. The distribution of the Proctor π^* has approximately the same shape as the distribution with IUC = .10; the standard deviations for the two distributions are virtually the same (.050 versus .048). The class interval containing the mean includes over 40% of the distribution.

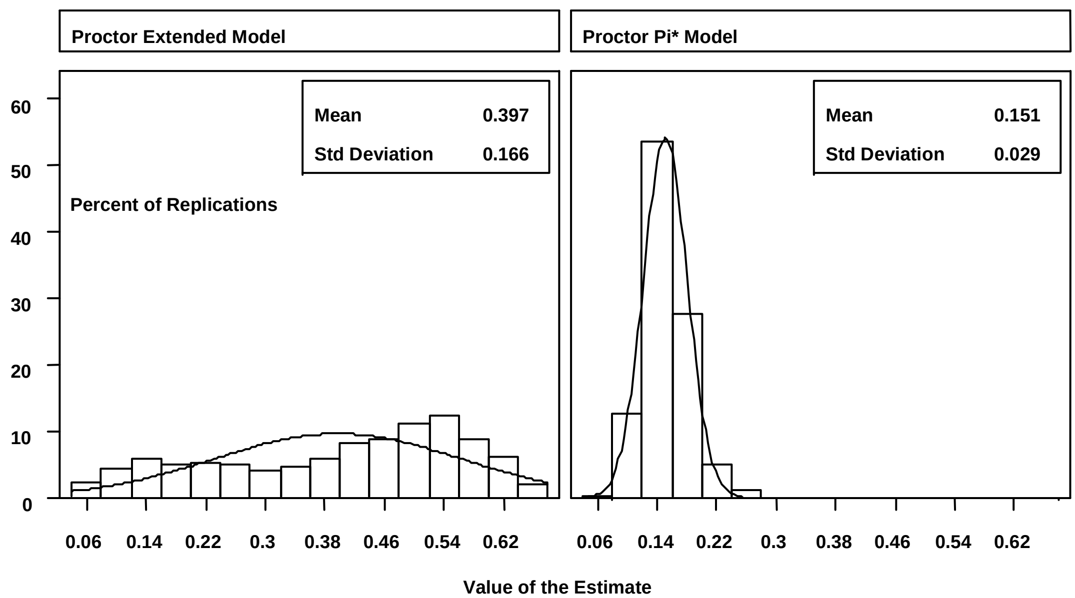

Figure 25. Distributions of IUC proportion and π^* for the four-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40, error rate is .10, and sample size is 960.

Figure 25 shows the sampling distributions for the four-variable extended Proctor and Proctor π^* models for the IUC proportion = .40 with error rate = .10 and sample size = 960 combination for the equal mixing proportions/unequal conditional probabilities scenario. For the extended Proctor model, the shape of the distribution in Figure 25 is a smoother version of the histogram in Figure 24. The standard deviation of the replicate estimates decreases slightly (.175 to .166) as the sample size increases from 240 to 960.

For the Proctor π^* model, the standard deviation drops from .050 to .029. Over 50% of the distribution falls in the interval containing the mean. Comparing Figure 24 to Figure 25 shows that, for the Proctor π^* model, the discrepancy with the true IUC proportion has increased with the increase in sample size (.208 to .151).

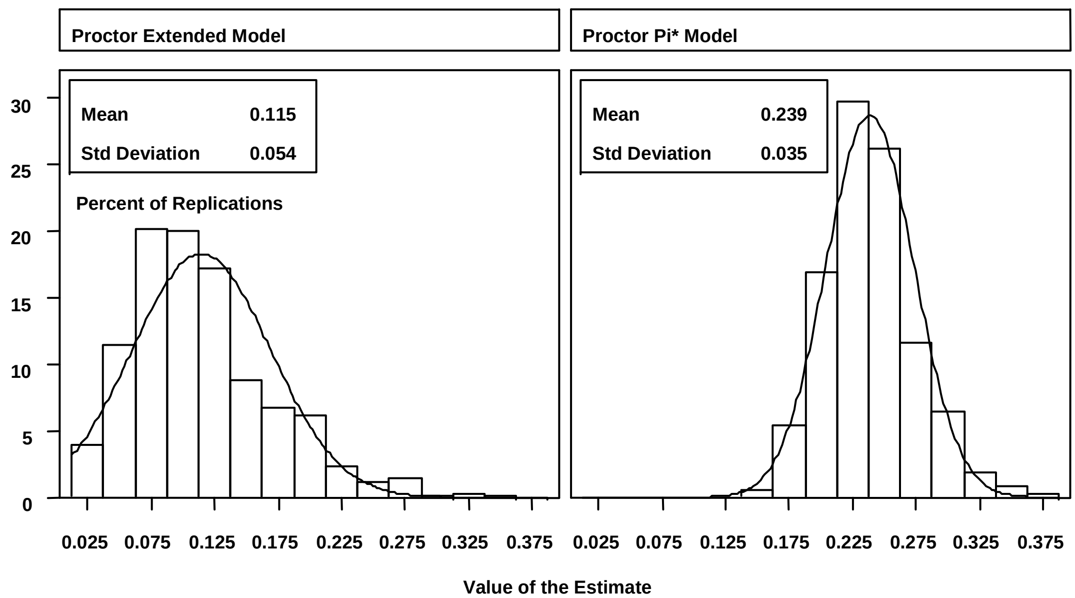

Figure 26. Distributions of IUC proportion and π^* for the six-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10, error rate is .10, and sample size is 960.

Figure 26 shows the sampling distributions for the six-variable extended Proctor and Proctor π^* models for the IUC proportion = .10 with error rate = .10 and sample size = 960 combination for the equal mixing proportions/unequal conditional probabilities scenario. There are striking differences between the extended Proctor model and the same model using four variables. The six-variable distribution is much more peaked than the four-variable distribution in Figure 23. There is a very large concentration of the distribution about the mean of .115 and the standard deviation decreases from .123 to .054. Interestingly, the six-variable Proctor π^* distribution has a mean of .239 compared to .089 for the four-variable case, but the standard deviations are similar (.035 in Figure 26 versus .026 in Figure 23).

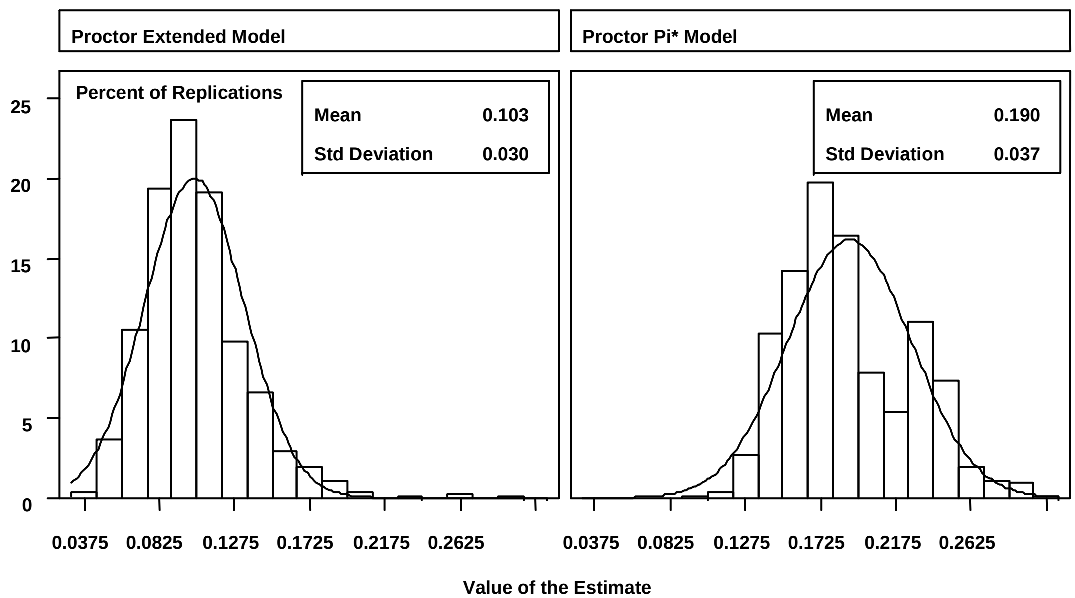

Figure 27. Distributions of IUC proportion and π^* for the six-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .10, error rate is .10, and sample size is 3840.

Figure 27 shows the sampling distributions for the six-variable extended Proctor and Proctor π^* models for the IUC proportion = .10 with error rate = .10 and sample size = 3840 combination for the equal mixing proportions/unequal conditional probabilities scenario. The increase in sample size from 960 to 3840 results in a distribution with less variability than the extended Proctor distribution in Figure 26; the standard deviation decreases from .054 to .030 with the increase in sample size. This distribution has a sharper peak than the distribution in Figure 26 with somewhat less than 25% of the distribution falling in the interval containing the mean. The spread of the distributions for the Proctor π^* models in Figures 25 and 26 is similar (with standard deviations of .030 and .037). The Proctor π^* model produces estimates that are larger than the true IUC proportion (.239 and .190).

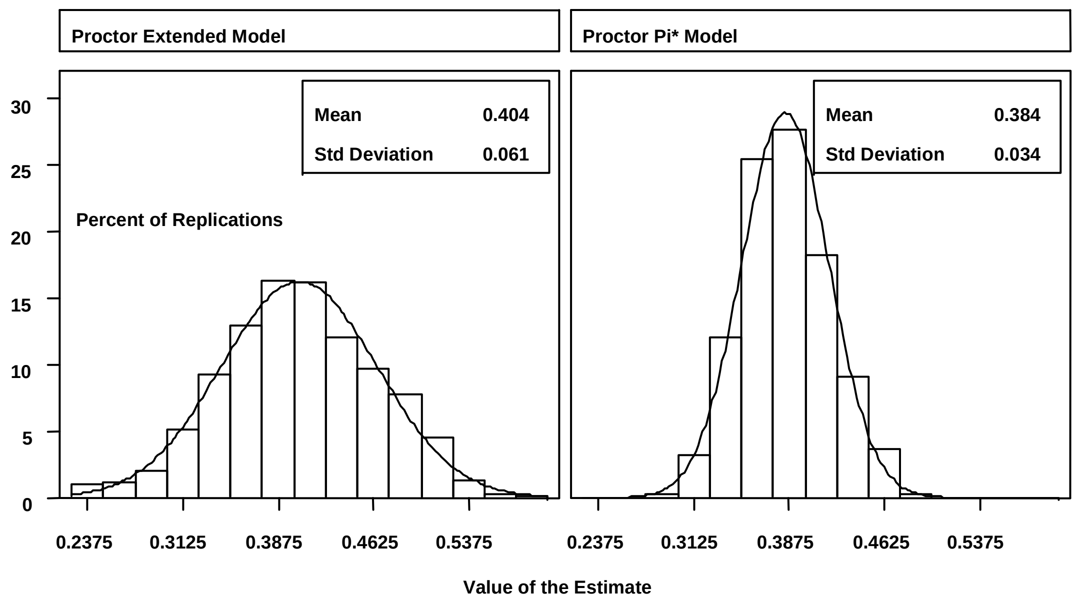

Figure 28. Distributions of IUC proportion and π^* for the six-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40, error rate is .10, and sample size is 960.

Figure 28 shows the sampling distributions for the six-variable extended Proctor and Proctor π^* models for the IUC proportion = .40 with error rate = .10 and sample size = 960 combination for the equal mixing proportions/unequal conditional probabilities scenario. A striking feature of the extended Proctor distribution in Figure 28 is its symmetry. Figure 26's distribution of the extended Proctor model displayed some skewness. The variability of this distribution is only slightly larger than that found in Figure 26 (standard deviations of .061 versus .054). The distribution for the Proctor π^* model is only mildly skewed. The distribution for the Proctor π^* model also shows considerably less variability than the extended Proctor model; standard deviations are .034 for the Proctor π^* and .061 for the extended Proctor.

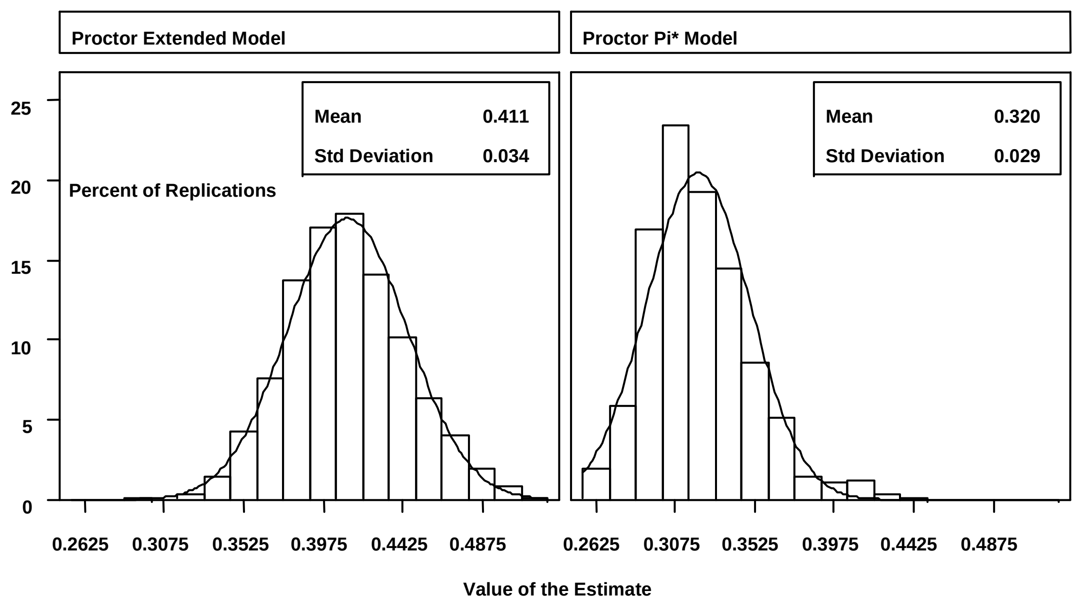

Figure 29. Distributions of IUC proportion and π^* for the six-variable extended Proctor and Proctor π^* models under the equal mixing proportions/unequal conditional probabilities scenario when the IUC proportion is .40, error rate is .10, and sample size is 3840.

Figure 29 shows the sampling distributions for the six-variable extended Proctor and Proctor π^* models for the IUC proportion = .40 with error rate = .10 and sample size = 3840 combination for the equal mixing proportions/unequal conditional probabilities scenario. The extended Proctor model shows symmetry like the distribution in Figure 28, but a much less variable distribution (.061 for a sample size of 960 versus .034 for a sample size of 3840). For the Proctor π^* model, the distribution has shifted to the left and is displaying more skewness than the distribution in Figure 28, with a minimal decrease in the standard deviation (.034 to .029). Considering distance from the true IUC proportion and variability, the Proctor π^* model outperforms the extended Proctor model in Figure 28. The extended Proctor outperforms in Figure 29 as its mean is less discrepant from the true IUC proportion and the standard deviation of its distribution is close to the standard deviation for the Proctor π^* model (.034 versus .029).

Bus Data Example

The real data used as an application for this study are the "Bus" data. This data comes from the International Association for the Evaluation of Educational Achievement (IEA) that has conducted achievement testing of school children around the world (Elley, 1992). A 1991 assessment of reading competency for nine-year old children consisted of a series of paragraphs with accompanying multiple-choice test variables. One paragraph, called "Bus," provided children with bus schedule information and posed four questions about the schedule. The data for the sample of 6,359 United States school children who responded to the four Bus items are presented in Table 4. The four questions increase in difficulty for these children: Dayton (1998) provides the "proportion answering each item correctly" as .690, .516, .272, and .080. Dayton also provides the "average proportion of correct responses for the four items" as .390 and "suggests that these items were quite difficult for the United States school children". Thus, a linear scale seems a natural approach for these data and the fitting of error models seem appropriate.

Goodman Model. First, the Goodman model was considered for fitting to responses to the four Bus data questions. Goodman (1975) formulated a model where a respondent could be classified into mutually exclusive and exhaustive categories. One category is composed of respondents who Goodman viewed as intrinsically unscalable and the remaining respondents Goodman considered intrinsically scalable and their response patterns correspond to a Guttman scale category. For four variables, the Guttman scale categories can be represented as (1, 1, 1, 1), (1, 1, 1, 0), (1, 1, 0, 0), (1, 0, 0, 0), and (0, 0, 0, 0). The estimates of the mixing proportions and the IUC conditional probabilities are shown in Table 5. The probability that a nine-year old is located in the IUC is $\hat{\pi}_0 = .506$. The Goodman model does not fit the bus data very well as the log likelihood value is 138.355 with 6 degrees of freedom. For response patterns (1, 0, 1, 1) and (0, 1, 0, 0), the fitted values are substantially less than the observed frequencies; these response patterns contribute the most to the log likelihood statistic.

Proctor and Extended Proctor Models. The Proctor model was next considered for fitting of the Bus data. The Proctor (1970) model allowed for measurement error in the Guttman scaling model. The probability of an error of measurement occurring is constant over all variables and scale types in the Proctor model. Like Goodman (1975), Dayton and Macready (1980) hypothesized that a certain proportion of the study population may not be scalable according to the Guttman scale. Dayton and Macready (1980) extended the Proctor model by including an IUC. Considering the magnitude of $\hat{\pi}_0$ in the Goodman model, the extended Proctor model seems more appropriate than the original Proctor model.

The estimated error rate for the extended Proctor model is $\hat{\Gamma} = .117$ (Table 5). The estimate of the IUC mixing proportion is .607, which is not much larger than the corresponding estimate under the Goodman model (.506). The log likelihood value for the extended Proctor model is 114.985 with 5 degrees of freedom. The extended Proctor model, like the Goodman model, does not fit the Bus data very well. Comparing the differences between the observed frequencies and the fitted values for the response patterns shows a residual for the (0, 1, 0, 0) response pattern that is larger than the corresponding residual under the Goodman model. Also, very large residuals occur for three of the five Guttman scale types.

Two-Point Mixture Index of Fit. In the Goodman and extended Proctor models, the response pattern (0, 1, 0, 0) contributes heavily to the log likelihood statistic. An alternative approach that could lessen the impact of this response pattern and some of the others is the Rudas *et al.* (1994) two-point mixture model formulation, which indicates how well a model is describing a set of data. Assuming that the children have been administered an instrument containing 4 dichotomous items, there exist $2^4 = 16$ response patterns and the response pattern probabilities can be written as:

$$P_h = (1-\pi)\Pi_{1h} + \pi\,\Pi_{2h} \quad \text{with} \quad h = 1, 2, \ldots, N \tag{38}$$

The term Π_{2h} represents the probability of occurrence of the response under a completely unrestricted model. The term Π_{1h} represents the conditional probability of the occurrence of response pattern *h* given the data follows the specified latent class model, the Proctor model. Here π represents the proportion of the population outside of the Proctor model. The maximum likelihood estimate of π, denoted by $\hat{\pi}^*$, can be found by maximizing the sum of the expected frequencies under the Proctor model subject to the constraint that each expected cell frequency must be less than or equal to the observed cell frequency. The value for $\hat{\pi}^*$ for the Bus data is .163. It is striking that .163 is much lower than the IUC mixing proportion estimate under the extended Proctor model, which is .607. It is also interesting that the error rate estimate of .011 for the Proctor π^* model is much lower than the error rate estimate of .117 for the extended Proctor model.

IO and Extended IO Models. Consider now the extended IO model as postulated by Dayton and Macready (1980). Dayton and Macready (1976) developed a model called the IO model as they believed that Proctor's premise of uniform measurement error was oversimplified. Dayton and Macready (1976) postulated that the error occurring if a respondent answered "1" when a "0" response was consistent with the permissible response pattern was distinct from the error occurring if a respondent answered "0" when a "1" response was consistent with the permissible response pattern. The error of the first type is referred to as an "intrusion" error whose probability is represented by the term β_I, and the second type of error is referred to as an "omission" error whose probability is represented by the term β_O.

The intrusion error rate estimate is .017 and the omission error rate estimate is .140. The IUC mixing proportion estimate is .228. The log likelihood value is 47.969, which is much less than the extended Proctor model's value, but with 4 degrees of freedom, the p-value is less than .0001. There were no extreme positive or negative residuals associated with the extended IO model as there were for the Goodman, extended Proctor, and Proctor π^* models.

For the IO π^* model the estimates of the intrusion error rate (.010) and the omission error rate (.180) are very similar to the extended IO model. The value of $\hat{\pi}^*$ is .079; that is, only 7.9% is the percent of the population where the IO model does not hold true. For the extended IO model, the IUC mixing proportion estimate (.228) is larger than $\hat{\pi}^*$ (.079). The IO π^* model, which describes the population considerably better than the Proctor π^* model, or the extended IO model, which fits better than the Goodman or extended Proctor models, appear to be the preferred models.

Table 4. Expected Frequencies for Bus Data
Goodman, Proctor, and IO Models

Quest A	Quest B	Quest C	Quest D	Observed Frequency	Goodman	Extended Proctor	Proctor π^*	Extended IO	IO π^*
0	0	0	0	1138	1138.00	1032.55	1138.00	1132.08	1138.00
0	0	0	1	13	18.26	30.41	13.00	14.50	13.00
0	0	1	0	75	108.29	85.78	13.11	83.70	52.34
0	0	1	1	15	8.36	9.07	.19	7.88	5.17
0	1	0	0	502	445.85	600.82	28.27	508.65	305.62
0	1	0	1	9	34.42	12.33	.36	10.60	7.84
0	1	1	0	198	204.07	143.71	9.88	177.01	187.08
0	1	1	1	23	15.75	39.54	3.53	46.79	23.00
1	0	0	0	1532	1532.00	1617.70	1532.00	1545.71	1532.00
1	0	0	1	43	36.56	31.55	17.540	29.42	20.75
1	0	1	0	200	216.80	207.87	27.06	195.10	200.00
1	0	1	1	59	16.74	42.09	3.73	47.13	23.14
1	1	0	0	1354	1354.00	1246.89	1354.00	1339.28	1354.00
1	1	0	1	37	68.91	66.69	18.89	63.80	35.29
1	1	1	0	852	852.00	897.62	852.00	871.08	852.00
1	1	1	1	309	309.00	294.40	309.00	286.26	104.79

Table 5. Parameter Estimates for Bus Data
Goodman, Proctor, and IO Models

Model	Goodman	Extended Proctor	Extended IO		Model	Proctor π^*	IO π^*
IUC	.506	.607	.228		π^*	.163	.079
Mixing Proportion 1	.044	.052	.091		Mixing Proportion 1	.184	.132
Mixing Proportion 2	.070	.038	.202		Mixing Proportion 2	.248	.243
Mixing Proportion 3	.073	.049	.204		Mixing Proportion 3	.218	.276
Mixing Proportion 4	.166	.206	.196		Mixing Proportion 4	.137	.236
Mixing Proportion 5	.142	.048	.078		Mixing Proportion 5	.049	.033
Error Rate		.117			Error Rate	.011	
Omission Error Rate			.140		Omission Error Rate		.180
Intrusion Error Rate			.017		Intrusion Error Rate		.010
Log Likelihood	138.355	114.985	47.969				
Degrees of Freedom	6	5	4				
Cond. Probability A	.667	.635	.446				
Cond. Probability B	.653	.397	.450				
Cond. Probability C	.314	.049	.110				
Cond. Probability D	.072	<.001	<.001				

Discussion. The results of the simulations can be used as a guide to compare the unscalable components of the extended IO and IO π^* models. When the true IUC proportion is .25 under the unequal mixing proportions with unequal conditional probabilities scenario, the mean of the replicate IUC proportion estimates are very close to the true IUC proportion for the omission error rate of .20 with intrusion error rate of .05 combination. In addition, the recovery of the error rates is very good. The simulation results provide evidence that the true IUC proportion is not considerably different from .228, the estimated IUC for the Bus data's extended IO model.

For the IO π^* model, the estimate of π^* of .079 is not too different from .10. Use of the IO π^* model has a tendency to produce π^* estimates that are less than the true IUC proportion. The estimate of π^* for the Bus data's IO π^* model is .079. It is likely that the proportion of the population that is unscalable is larger than .079. Also, the omission error rate of .180 is very close to .20.

From the empirical analysis of the variability, the sampling distribution of the extended IO model's IUC proportion estimates showed much greater variability than the IO π^* model sampling distribution. Considering the sampling distribution and simulation results, it is reasonable to conclude that the IO π^* model is providing a more accurate estimate of the unscalable component for the Bus data.

Large Sample Analysis

Since this study's simulations showed that the π^* estimates decreased as sample sizes increased, it was of interest to investigate if π^* stabilized for extremely large samples sizes. Thus, additional simulations were run for the equal mixing proportions/unequal conditional probabilities scenario with sample sizes of 2560, 5120, and 10240 for the four-variable models and with sample sizes of 10240, 20480, and 40960 for the six-variable models. Tables 6 through 9 present the additional simulation results along with the results of the previous smaller sample sizes.

Four-Variable Extended Proctor and IO Models. For the true IUC proportion of .10 across all error rates, IUC estimates continue to decrease with the larger sample sizes with estimates ranging from .151 to .121 for the extended Proctor model (Table 6) and with estimates ranging from .155 to .116 for the extended IO model (Table 7). When the IUC proportion is small, it appears that extremely large sample sizes will be needed to produce IUC estimates with minimal bias.

For the IUC proportions of .25 and .40 across all error rates, the IUC estimates for the larger sample sizes generally are close to the true IUC proportion (and similar to the smaller sample size estimates). When the IUC proportion is .25, estimates range from .269 to .235 for the extended Proctor model (Table 6) and from .260 to .243 for the extended IO model (Table 7). When the IUC proportion is .40, estimates range from .413 to .375 for the extended Proctor model (Table 6) and from .417 to .364 for the extended IO model (Table 7).

Four-Variable Proctor and IO π^* Models. The π^* estimates continue to decrease as sample size increases (Tables 6 and 7). With the larger sample sizes, the π^* estimates show considerably less variability than the estimates resulting from the smaller sample sizes. For both the Proctor and IO π^* models, the estimates for the two largest sample sizes (5120 and 10240) for the true IUC proportions of .10, .25, and .40 are very close.

For the Proctor π^* model, with the IUC proportion of .10, the estimates for the sample sizes of 5120 and 10240 are .049 and .043, .050 and .042, and .061 and .054 for the error rates of .05, .10, and .20, respectively. For the IUC proportion of .25, these estimates are .094 and .089, .083 and .076, and .100 and .097 for the error rates of .05, .10, and .20, respectively. For the IUC proportion of .40, these estimates are .136 and .131, .116 and .110, and .147 and .147 for the error rates of .05, .10, and .20, respectively.

For the IO π^* model, with the IUC proportion of .10, the estimates for the sample sizes of 5120 and 10240 are .043 and .036 (omission rate of .20 with intrusion rate of .05), and .053 and .050 (omission rate of .05 with intrusion rate of .20). For the IUC proportion of .25, these estimates are .072 and .067 (omission rate of .20 with intrusion rate of .05), and .091 and .094 (omission rate of .05 with intrusion rate of .20). For the IUC proportion of .40, these estimates are .099 and .095 (omission rate of .20 with intrusion rate of .05) and .141 and .139 (omission rate of .05 with intrusion rate of .20).

Across all the true IUC proportions for both models, the decreases in the π^* estimates appear to stabilize as the sample size increases from 2560 to 10240. In other words, the π^* estimates are stabilizing with the very large sample sizes.

Table 6. IUC and π Estimates for Large Sample Sizes
Four-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model IUC			Extended Proctor Model IUC			Proctor Model π			Proctor Model π		
Sample Size	**240**	**480**	**960**	**2560**	**5120**	**10240**	**240**	**480**	**960**	**2560**	**5120**	**10240**
IUC = .10												
Rate = .05	.169	.175	.153	.151	.137	.129	.099	.090	.075	.056	.049	.043
Rate = .10	.209	.178	.158	.148	.141	.138	.148	.115	.089	.062	.050	.042
Rate = .20	.198	.178	.168	.144	.134	.121	.196	.142	.103	.074	.061	.054
IUC = .25												
Rate = .05	.285	.276	.266	.256	.251	.247	.140	.135	.119	.100	.094	.089
Rate = .10	.298	.276	.263	.269	.264	.254	.178	.148	.121	.095	.083	.076
Rate = .20	.281	.269	.265	.251	.241	.235	.211	.160	.129	.106	.100	.097
IUC = .40												
Rate = .05	.401	.398	.412	.413	.410	.409	.189	.182	.165	.143	.136	.131
Rate = .10	.391	.389	.397	.375	.391	.399	.208	.179	.151	.125	.116	.110
Rate = .20	.352	.342	.337	.375	.388	.387	.226	.185	.159	.149	.147	.147

Table 7. IUC and π Estimates for Large Sample Sizes
Four-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended IO Model IUC			Extended IO Model IUC			IO Model π			IO Model π		
Sample Size	**240**	**480**	**960**	**2560**	**5120**	**10240**	**240**	**480**	**960**	**2560**	**5120**	**10240**
IUC = .10												
Om. Rate=.20 Int. Rate=.05	.183	.162	.164	.150	.137	.116	.129	.103	.077	.054	.043	.036
Om. Rate=.05 Int. Rate=.20	.179	.161	.161	.155	.146	.148	.133	.111	.083	.061	.053	.050
IUC = .25												
Om. Rate=.20 Int. Rate=.05	.272	.264	.252	.260	.250	.243	.151	.122	.101	.079	.072	.067
Om. Rate=.05 Int. Rate=.20	.270	.258	.249	.254	.253	.244	.164	.142	.114	.091	.091	.094
IUC = .40												
Om. Rate=.20 Int. Rate=.05	.376	.380	.394	.406	.421	.417	.175	.146	.125	.105	.099	.095
Om. Rate=.05 Int. Rate=.20	.362	.349	.351	.364	.364	.374	.194	.168	.146	.141	.141	.139

Six-Variable Extended Proctor and IO Models. Across all true IUC proportions and error rate condition levels, the IUC proportion estimates show excellent recovery for the larger sample sizes as they did for the smaller sample sizes (Tables 8 and 9). The IUC proportions change minimally as the sample size increases from 10480 to 40960. For the extended Proctor model, with IUC proportions of .10, .25, and .40, the estimates range from .107 to .095, .250 to .248, and .406 to .396, respectively. For the extended IO model, with IUC proportions of .10, .25, and .40, the estimates range from .109 to .091, .273 to .252, and .405 to .391, respectively.

An interesting observation for the six-variable extended Proctor model is that the IUC proportion estimates decrease sizably from sample size 3840 to sample size 10240 when the error rate is .20.

Six-Variable Proctor and IO π^* Models. For the IUC proportion of .10, π^* estimates are close to the true IUC proportion for the sample sizes of 20,480 and 40,960 (Tables 8 and 9). For the IUC proportion of .10, these Proctor π^* estimates are .108 and .094, .106 and .093, and .111 and .097 for the error rates of .05, .10, and .20, respectively. For the IUC proportions of .25 and .40, the π^* estimates continue to decrease from the true IUC proportion as the sample size increases. However, the size of the decrease becomes smaller as the sample size increases from 10240 to 40960.

The estimates for the two largest sample sizes (20480 and 40960) for the IUC proportions of .25 and .40 are very close. For the Proctor π^* model, with the IUC proportion of .25, these estimates are .193 and .186, .186 and .175, and .183 and .172 for the error rates of .05, .10, and .20, respectively. For the IUC proportion of .40, these estimates are .291 and .285, .259 and .250, and .268 and .260 for the error rates of .05,

.10, and .20, respectively. For the IO π^* model, with the IUC proportion of .25, these estimates are .197 and .190 (omission rate of .20 with intrusion rate of .05) and .182 and .174 (omission rate of .05 with intrusion rate of .20). For the IUC proportion of .40, these estimates are .281 and .273 (omission rate of .20 with intrusion rate of .05) and .277 and .272 (omission rate of .05 with intrusion rate of .20).

Across all the true IUC proportions for both models, the decreases in the π^* estimates appear to stabilize as the sample size increases from 10240 to 40960. As with the four-variable models, π^* estimates for the six-variable models are stabilizing with very large sample sizes.

Table 8. IUC and π Estimates for Large Sample Sizes
Six-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model IUC			Extended Proctor Model IUC			Proctor Model π			Proctor Model π		
Sample Size	**960**	**1920**	**3840**	**10240**	**20480**	**40960**	**960**	**1920**	**3840**	**10240**	**20480**	**40960**
IUC = .10												
Rate = .05	.106	.103	.103	.095	.095	.095	.190	.157	.139	.129	.108	.094
Rate = .10	.115	.109	.103	.103	.103	.102	.239	.219	.190	.131	.106	.093
Rate = .20	.140	.136	.131	.107	.101	.099	.343	.267	.199	.135	.111	.097
IUC = .25												
Rate = .05	.245	.248	.249	.250	.250	.250	.280	.250	.230	.205	.193	.186
Rate = .10	.249	.251	.247	.249	.250	.250	.305	.281	.254	.206	.186	.175
Rate = .20	.269	.273	.280	.249	.249	.248	.381	.309	.255	.200	.183	.172
IUC = .40												
Rate = .05	.393	.397	.397	.396	.396	.396	.369	.345	.323	.300	.291	.285
Rate = .10	.404	.413	.411	.397	.398	.398	.384	.355	.320	.275	.259	.250
Rate = .20	.361	.364	.366	.406	.403	.404	.420	.361	.312	.279	.268	.260

Table 9. IUC and π Estimates for Large Sample Sizes
Six-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended IO Model IUC			Extended IO Model IUC			IO Model π			IO Model π		
Sample Size	**960**	**1920**	**3840**	**10240**	**20480**	**40960**	**960**	**1920**	**3840**	**10240**	**20480**	**40960**
IUC = .10												
Om. Rate=.20 Int. Rate=.05	.117	.106	.105	.109	.109	.108	.250	.220	.173	.130	.111	.101
Om. Rate=.05 Int. Rate=.20	.136	.127	.113	.098	.094	.091	.248	.204	.175	.124	.101	.088
IUC = .25												
Om. Rate=.20 Int. Rate=.05	.259	.256	.256	.252	.253	.253	.318	.286	.245	.211	.197	.190
Om. Rate=.05 Int. Rate=.20	.273	.272	.271	.273	.272	.269	.307	.267	.234	.195	.182	.174
IUC = .40												
Om. Rate=.20 Int. Rate=.05	.399	.404	.403	.405	.405	.405	.387	.357	.321	.291	.281	.273
Om. Rate=.05 Int. Rate=.20	.401	.404	.409	.391	.391	.391	.367	.334	.312	.283	.277	.272

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

This study assessed the comparability of unscalable components for: (a) a two-point mixture scaling model and (b) a scaling model that incorporates an intrinsically unscalable class (IUC). The specific models investigated are the Proctor π^* and extended Proctor models, as well as the intrusion-omission π^* and extended intrusion-omission models. This chapter presents the study's conclusions based on the simulation results, makes model recommendations to researchers, and suggests ideas for future research.

Conclusions for IUC and π^* Estimation

- For the four-variable case when the IUC proportion is small, Proctor π^* estimates show a better approximation to the true IUC proportion compared to extended Proctor estimates, that tend to be highly positively biased. For larger IUC proportions, Proctor π^* estimates are substantially different from the true IUC proportions, whereas extended Proctor estimates generally are closer to the true values. Similar conclusions hold for the four-variable IO π^* and extended IO models.
- In terms of estimating the IUC proportion for the six-variable case, the extended Proctor model outperforms the Proctor π^* model across almost every combination of IUC proportion/error rate/sample size in all mixing proportion/conditional probability scenarios. The extended Proctor model shows some prominent trends in the IUC proportion estimated means. First, the

differences between the estimated means and true IUC proportions are smallest for the lowest error rate. Second, these differences tend to decrease as sample size increases. Third, the extended Proctor model shows good stability as the true IUC proportion increases.

- For the extended Proctor model with larger IUC proportions, there are, in general, only minimal differences in the mean IUC proportion estimated when comparing the four-variable and six-variable models. For small IUC proportions, the differences are more substantial with the six-variable models showing superior recovery compared to the four-variable models. For larger IUC proportions, the six-variable Proctor π^* model outperforms the four-variable Proctor π^* model.
- The six-variable extended IO model, in general, outperforms the six-variable IO π^* model with respect to the recovery of the true IUC proportion across all IUC proportions, mixing proportion/conditional probability scenarios, omission/intrusion error rate combinations, and sample sizes. There is a gradual change in the mean IUC proportion estimated from the smallest sample size to the largest sample size for both extended models. This could be informative to researchers who may be willing to tolerate a small amount of bias with a smaller sample size and lower costs.
- The estimated means of the six-variable extended IO model approach the true IUC proportion more closely than the four-variable extended model, with only one exception at the smallest IUC proportion. For larger IUC proportions, the six-variable IO π^* model outperforms the four-variable IO π^* model.

Conclusions for Error Rate Estimation

- For the four-variable extended Proctor model, the recovery of the error rate, overall, is very good. In general, at small sample sizes there is a slight underestimation of the error rate; however, as the sample size increases, the amount of underestimation becomes negligible. The error rates are very stable across the true IUC proportions. With a few exceptions, the Proctor π^* model yields estimates less than the true error rate, worsening for small sample sizes. Based on these simulation results, the four-variable extended Proctor model outperforms the four-variable Proctor π^* model in terms of error rate estimation.
- The four-variable extended IO model tends to produce negatively biased estimates for the omission and intrusion error rates at smaller sample sizes for the smallest IUC proportion. For the larger IUC proportions, the results are mixed for the extended IO model with some overestimation and underestimation. However, for all IUC proportions, with only a few exceptions, the error rate estimated means are close to the true error rate at the largest sample size. In general, the four-variable IO π^* model produces omission and intrusion error rates that are different from the true error rates.
- For the six-variable models, the error rate recovery for the extended Proctor and the extended IO models is excellent across all IUC proportions, error rates, and sample sizes for all three scenarios of interest. In fact, the error rate recovery for the smallest sample size is almost identical to the error rate recovery at the largest sample size. The Proctor π^* model tends to produce estimates that are less than the true error rates. The IO π^* model, more often than not, tends to produce

estimates that are less than the true omission and intrusion error rates. There are a few exceptions where the estimates are larger than the true error rates, most notable under the equal mixing proportions/equal conditional probabilities scenario for the largest IUC proportion for both omission error/intrusion error rate combinations. Patterns found in the error rate estimated means for the four-variable IO π^* model are similar to the six-variable IO π^*.

General Recommendation

The extended model (Proctor or IO) should not be replaced by the corresponding π^* model as the simulation results demonstrated that π^* is not comparable to the extended models' IUC proportion. This study's expectation was realized in that even with very large samples sizes π^* overfitted.

Suggestions for Future Research

Sampling Errors. A direction for further research is efficient methods for estimating sampling errors for π^* estimates. Rudas *et al.* (1994) did not directly compute a standard error for π^* in their seminal work, but, rather, utilized the premise that the log likelihood of π^* is asymptotically distributed as a chi-square distribution. They estimated a lower confidence bound for π^*, equal to the value of $\hat{\pi}$ at which the G^2 fit statistic is associated with the 90th percentage point. The confidence interval for π^* is one-sided because all values of $\hat{\pi}$ greater than $\hat{\pi}^*$ result in perfect fit of the observed frequencies. For the one-way contingency table model, Dayton (2003) computed a lower confidence bound for π^* using the approach found in Rudas *et al.* (1994) as well as using a jackknife approach, which could also be applied to the π^* models in this paper. For the extended models, techniques for estimating the standard errors of the IUC mixing proportion are found in de Menezes (1999).

Alternate IUC Construction. The simulations conducted for this research were based on a universe where the condition of conditional independence was assumed for the IUC. The second latent class of the Rudas *et al.* (1994) mixture model is that segment of the population conforming to an unrestricted multinomial model. Further research may want to utilize different probability models in the IUC and study how the performance of the extended and π^* models is affected.

Number of Variables. Finally, a suggestion for further research is to see how the π^* models perform as the number of variables increases. This paper concluded that the performance of the Proctor and IO π^* models improves when the number of variables increases from four to six. Research with a greater number of variables could provide further evidence to support this pattern. Note though that with, say eight variables, the number of computations is dramatically increased as there will be 256 constraints. Further research with an increased number of variables would also provide the benefit of more evidence about the performance of the extended Proctor and IO models.

APPENDIX A: RESULTS TABLES

Table A1. IUC and π^* Estimates
Four-Variable Extended Proctor Model
Equal Mixing Proportions, Equal Conditional Probabilities

Estimated Mean	Extended Proctor Model IUC			Proctor Model Pistar			$\pi_{IUC} - \pi^*$			$\lvert \pi_{IUC} - \pi^* \rvert$			Scaled Value of $\lvert \pi_{IUC} - \pi^* \rvert$			Goodman Model IUC		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10																		
Rate = .05	.167	.145	.133	.108	.090	.074	.059	.055	.059	.098	.084	.073	.978	.838	.731	.298	.289	.288
Rate = .10	.201	.182	.167	.152	.115	.088	.049	.066	.079	.130	.125	.111	1.304	1.249	1.109	.464	.458	.456
Rate = .20	.229	.219	.204	.198	.143	.099	.031	.076	.105	.141	.143	.147	1.412	1.435	1.467	.724	.719	.717
IUC = .25																		
Rate = .05	.281	.277	.269	.161	.142	.123	.120	.135	.146	.143	.144	.148	.571	.578	.593	.411	.401	.400
Rate = .10	.294	.282	.265	.183	.148	.120	.110	.134	.145	.162	.161	.154	.647	.645	.617	.541	.538	.534
Rate = .20	.312	.317	.318	.210	.150	.105	.102	.167	.213	.174	.204	.231	.695	.815	.924	.745	.742	.739
IUC = .40																		
Rate = .05	.394	.383	.388	.216	.188	.168	.178	.196	.220	.195	.202	.220	.487	.505	.550	.527	.517	.519
Rate = .10	.424	.417	.409	.222	.180	.147	.202	.238	.262	.231	.249	.266	.577	.622	.666	.622	.618	.616
Rate = .20	.413	.426	.448	.229	.161	.119	.184	.265	.329	.232	.285	.338	.581	.713	.845	.781	.778	.776

Table A2. Error Rate Recovery
Four-Variable Extended Proctor Model
Equal Mixing Proportions, Equal Conditional Probabilities

Estimated Mean	Extended Proctor Model Error Rate			Proctor Pistar Model Error Rate			$\Gamma - \Gamma^*$			$\lvert \Gamma - \Gamma^* \rvert$			Scaled Value of $\lvert \Gamma - \Gamma^* \rvert$		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10															
Rate = .05	.041	.045	.047	.026	.034	.041	.015	.011	.006	.020	.016	.011	.400	.324	.222
Rate = .10	.083	.087	.091	.056	.071	.083	.027	.015	.007	.036	.026	.018	.356	.259	.181
Rate = .20	.177	.177	.182	.143	.164	.181	.034	.013	.0009	.053	.041	.031	.267	.204	.157
IUC = .25															
Rate = .05	.044	.045	.046	.036	.045	.053	.008	.0003	-.007	.021	.017	.014	.429	.338	.284
Rate = .10	.088	.091	.096	.075	.089	.102	.013	.002	-.006	.032	.025	.021	.321	.251	.206
Rate = .20	.177	.181	.183	.158	.183	.199	.019	-.002	-.016	.056	.048	.039	.281	.240	.195
IUC = .40															
Rate = .05	.052	.053	.051	.051	.063	.072	.0008	-.009	-.020	.026	.024	.025	.514	.472	.491
Rate = .10	.088	.095	.098	.090	.111	.129	-.002	-.016	-.030	.036	.032	.036	.359	.324	.363
Rate = .20	.167	.171	.173	.166	.198	.216	.0009	-.027	-.043	.062	.059	.056	.309	.297	.278

Table A3. Mixing Proportion Recovery
Four-Variable Extended Proctor Model
Equal Mixing Proportions, Equal Conditional Probabilities

Model		Extended Proctor						Proctor Pistar					
IUC = .10		**.180**	**.180**	**.180**	**.180**	**.180**	**.100**						$\hat{\pi}^*$
Rate = .05	**n = 240**	.154	.163	.168	.174	.174	.167	.193	.181	.174	.175	.169	.108
	n = 480	.160	.167	.174	.178	.176	.145	.198	.184	.178	.178	.171	.090
	n = 960	.164	.170	.175	.179	.179	.133	.203	.186	.180	.181	.177	.074
Rate = .10	**n = 240**	.144	.157	.161	.168	.168	.201	.186	.176	.167	.165	.153	.152
	n = 480	.155	.157	.166	.171	.169	.182	.198	.178	.174	.173	.162	.115
	n = 960	.160	.163	.166	.173	.172	.167	.206	.184	.176	.178	.168	.088
Rate = .20	**n = 240**	.130	.147	.163	.174	.157	.229	.181	.162	.163	.162	.134	.198
	n = 480	.143	.147	.162	.169	.160	.219	.202	.169	.167	.173	.146	.143
	n = 960	.150	.147	.165	.172	.162	.204	.218	.174	.176	.178	.155	.099
IUC = .25		**.150**	**.150**	**.150**	**.150**	**.150**	**.250**						$\hat{\pi}^*$
Rate = .05	**n = 240**	.136	.146	.143	.146	.148	.281	.214	.175	.155	.150	.145	.161
	n = 480	.135	.146	.148	.145	.149	.277	.220	.176	.161	.151	.150	.142
	n = 960	.140	.147	.147	.148	.149	.269	.227	.179	.162	.156	.154	.123
Rate = .10	**n = 240**	.143	.140	.141	.141	.141	.294	.220	.167	.155	.144	.131	.183
	n = 480	.142	.145	.143	.145	.144	.282	.232	.172	.158	.151	.138	.148
	n = 960	.146	.150	.145	.147	.147	.265	.244	.177	.160	.156	.143	.120
Rate = .20	**n = 240**	.147	.128	.140	.139	.134	.312	.243	.149	.143	.138	.117	.210
	n = 480	.147	.123	.137	.138	.138	.317	.274	.157	.145	.148	.126	.150
	n = 960	.145	.123	.137	.137	.140	.318	.295	.161	.150	.154	.135	.105
IUC = .40		**.120**	**.120**	**.120**	**.120**	**.120**	**.400**						$\hat{\pi}^*$
Rate = .05	**n = 240**	.116	.129	.123	.118	.122	.394	.237	.163	.140	.124	.120	.216
	n = 480	.121	.130	.123	.119	.123	.383	.248	.168	.141	.129	.126	.188
	n = 960	.121	.128	.123	.119	.121	.388	.254	.170	.145	.132	.130	.168
Rate = .10	**n = 240**	.120	.114	.114	.112	.116	.424	.257	.156	.135	.124	.107	.222
	n = 480	.110	.123	.118	.116	.117	.417	.275	.168	.137	.131	.110	.180
	n = 960	.113	.124	.119	.117	.119	.409	.295	.171	.138	.138	.111	.147
Rate = .20	**n = 240**	.140	.118	.115	.111	.104	.413	.300	.138	.126	.117	.090	.229
	n = 480	.132	.115	.110	.110	.106	.426	.346	.140	.127	.127	.100	.161
	n = 960	.117	.112	.107	.106	.110	.448	.374	.138	.128	.133	.108	.119

Table A4. Conditional Probability Recovery
Four-Variable Extended Proctor Model
Equal Mixing Proportions, Equal Conditional Probabilities

Model		Extended Proctor			
IUC = .10		**.300**	**.300**	**.300**	**.300**
Rate = .05	**n = 240**	.323	.336	.364	.383
	n = 480	.296	.302	.341	.371
	n = 960	.284	.289	.319	.325
Rate = .10	**n = 240**	.337	.366	.345	.368
	n = 480	.341	.359	.358	.372
	n = 960	.330	.359	.352	.362
Rate = .20	**n = 240**	.336	.352	.336	.351
	n = 480	.353	.363	.344	.343
	n = 960	.349	.347	.326	.346
IUC = .25		**.300**	**.300**	**.300**	**.300**
Rate = .05	**n = 240**	.273	.312	.323	.331
	n = 480	.272	.299	.308	.310
	n = 960	.283	.296	.300	.297
Rate = .10	**n = 240**	.284	.338	.343	.368
	n = 480	.262	.334	.327	.364
	n = 960	.255	.322	.319	.338
Rate = .20	**n = 240**	.314	.322	.340	.348
	n = 480	.324	.319	.323	.337
	n = 960	.321	.315	.339	.343
IUC = .40		**.300**	**.300**	**.300**	**.300**
Rate = .05	**n = 240**	.245	.298	.316	.338
	n = 480	.248	.295	.311	.322
	n = 960	.271	.300	.309	.309
Rate = .10	**n = 240**	.288	.315	.327	.335
	n = 480	.262	.301	.314	.325
	n = 960	.260	.299	.308	.315
Rate = .20	**n = 240**	.283	.328	.342	.331
	n = 480	.285	.326	.324	.328
	n = 960	.285	.334	.337	.330

Table A5. IUC and π^* Estimates
Four-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model IUC			Proctor Model Pistar			$\pi_{IUC} - \pi^*$			$\lvert \pi_{IUC} - \pi^* \rvert$			Scaled Value of $\lvert \pi_{IUC} - \pi^* \rvert$			Goodman Model IUC		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10																		
Rate = .05	.169	.175	.153	.099	.090	.075	.070	.085	.079	.103	.111	.093	1.032	1.110	.932	.278	.267	.265
Rate = .10	.209	.178	.158	.148	.115	.089	.061	.063	.069	.122	.108	.103	1.224	1.079	1.032	.435	.429	.424
Rate = .20	.198	.178	.168	.196	.142	.103	.002	.037	.065	.110	.099	.098	1.099	.993	.977	.678	.678	.675
IUC = .25																		
Rate = .05	.285	.276	.266	.140	.135	.119	.145	.141	.147	.163	.159	.153	.653	.638	.612	.371	.360	.358
Rate = .10	.298	.276	.263	.178	.148	.121	.121	.129	.142	.157	.157	.161	.627	.627	.643	.486	.478	.474
Rate = .20	.281	.269	.265	.211	.160	.129	.070	.108	.136	.127	.137	.151	.508	.550	.605	.677	.670	.670
IUC = .40																		
Rate = .05	.401	.398	.412	.189	.182	.165	.212	.216	.246	.229	.226	.248	.574	.564	.619	.486	.472	.471
Rate = .10	.391	.389	.397	.208	.179	.151	.182	.210	.246	.213	.229	.254	.532	.573	.634	.561	.554	.551
Rate = .20	.352	.342	.337	.226	.185	.159	.126	.157	.178	.166	.174	.186	.414	.436	.464	.708	.699	.699

Table A6. Error Rate Recovery
Four-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model Error Rate			Proctor Pistar Model Error Rate			$\Gamma - \Gamma^*$			$\lvert \Gamma - \Gamma^* \rvert$			Scaled Value of $\lvert \Gamma - \Gamma^* \rvert$		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10															
Rate = .05	.041	.044	.047	.023	.027	.034	.018	.018	.013	.020	.018	.014	.408	.368	.273
Rate = .10	.092	.095	.097	.053	.064	.075	.039	.030	.022	.042	.032	.023	.417	.318	.232
Rate = .20	.190	.195	.196	.138	.157	.171	.052	.038	.025	.060	.045	.030	.300	.226	.150
IUC = .25															
Rate = .05	.048	.051	.050	.028	.032	.038	.019	.018	.012	.022	.019	.012	.437	.389	.248
Rate = .10	.093	.095	.098	.055	.067	.079	.037	.028	.019	.039	.029	.020	.394	.293	.198
Rate = .20	.188	.192	.196	.130	.149	.159	.058	.043	.036	.066	.049	.040	.329	.244	.198
IUC = .40															
Rate = .05	.048	.048	.048	.033	.035	.042	.016	.013	.006	.021	.016	.010	.412	.325	.209
Rate = .10	.097	.098	.099	.061	.074	.086	.036	.023	.013	.040	.027	.018	.399	.274	.181
Rate = .20	.191	.196	.199	.127	.139	.146	.065	.058	.053	.074	.064	.054	.370	.318	.271

Table A7. Mixing Proportion Recovery
Four-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Proctor						Proctor Pistar					
IUC = .10		**.180**	**.180**	**.180**	**.180**	**.180**	**.100**						$\hat{\pi}^*$
Rate = .05	**n = 240**	.149	.164	.171	.173	.174	.169	.213	.181	.172	.170	.166	.099
	n = 480	.134	.159	.176	.178	.178	.175	.215	.182	.173	.173	.168	.090
	n = 960	.146	.166	.177	.179	.179	.153	.219	.184	.175	.175	.172	.075
Rate = .10	**n = 240**	.124	.144	.173	.176	.174	.209	.204	.170	.165	.162	.151	.148
	n = 480	.137	.154	.175	.179	.177	.178	.216	.175	.168	.169	.157	.115
	n = 960	.144	.160	.179	.180	.178	.158	.224	.180	.172	.172	.163	.089
Rate = .20	**n = 240**	.128	.154	.174	.175	.171	.198	.209	.163	.151	.148	.134	.196
	n = 480	.144	.150	.181	.174	.173	.178	.231	.169	.158	.154	.147	.142
	n = 960	.141	.159	.180	.176	.176	.168	.243	.177	.158	.160	.158	.103
IUC = .25		**.150**	**.150**	**.150**	**.150**	**.150**	**.250**						$\hat{\pi}^*$
Rate = .05	**n = 240**	.129	.143	.147	.148	.148	.285	.261	.169	.146	.143	.140	.140
	n = 480	.131	.144	.150	.149	.149	.276	.264	.170	.147	.144	.140	.135
	n = 960	.139	.146	.150	.150	.149	.266	.269	.173	.149	.147	.143	.119
Rate = .10	**n = 240**	.124	.139	.146	.148	.144	.298	.264	.161	.139	.135	.124	.178
	n = 480	.133	.146	.148	.151	.145	.276	.276	.163	.142	.141	.130	.148
	n = 960	.139	.151	.148	.151	.147	.263	.287	.168	.145	.145	.135	.121
Rate = .20	**n = 240**	.144	.142	.144	.146	.143	.281	.290	.149	.121	.120	.109	.211
	n = 480	.148	.143	.149	.148	.144	.269	.315	.156	.122	.127	.120	.160
	n = 960	.144	.144	.153	.145	.148	.265	.330	.158	.124	.128	.130	.129
IUC = .40		**.120**	**.120**	**.120**	**.120**	**.120**	**.400**						$\hat{\pi}^*$
Rate = .05	**n = 240**	.125	.119	.117	.118	.119	.401	.308	.157	.118	.115	.112	.189
	n = 480	.126	.119	.118	.120	.118	.398	.311	.158	.119	.117	.113	.182
	n = 960	.113	.118	.119	.120	.118	.412	.318	.161	.121	.119	.116	.165
Rate = .10	**n = 240**	.133	.121	.117	.120	.118	.391	.325	.148	.111	.109	.098	.208
	n = 480	.137	.118	.118	.121	.117	.389	.342	.150	.114	.113	.101	.179
	n = 960	.127	.118	.118	.121	.119	.397	.357	.152	.116	.116	.108	.151
Rate = .20	**n = 240**	.177	.124	.117	.120	.110	.352	.367	.136	.093	.095	.082	.226
	n = 480	.179	.125	.121	.119	.114	.342	.393	.135	.095	.099	.092	.185
	n = 960	.181	.124	.121	.119	.118	.337	.406	.138	.094	.103	.100	.159

Table A8. Conditional Probability Recovery
Four-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Proctor			
IUC = .10		**.200**	**.100**	**.150**	**.300**
Rate = .05	**n = 240**	.263	.224	.248	.410
	n = 480	.238	.149	.191	.360
	n = 960	.223	.122	.169	.354
Rate = .10	**n = 240**	.277	.175	.211	.334
	n = 480	.255	.147	.177	.358
	n = 960	.244	.130	.165	.399
Rate = .20	**n = 240**	.263	.194	.200	.343
	n = 480	.264	.180	.184	.361
	n = 960	.242	.162	.164	.373
IUC = .25		**.200**	**.100**	**.150**	**.300**
Rate = .05	**n = 240**	.199	.127	.178	.374
	n = 480	.191	.100	.158	.375
	n = 960	.193	.097	.159	.344
Rate = .10	**n = 240**	.198	.117	.170	.381
	n = 480	.183	.109	.168	.409
	n = 960	.169	.100	.159	.411
Rate = .20	**n = 240**	.211	.133	.160	.385
	n = 480	.205	.121	.157	.389
	n = 960	.193	.108	.148	.389
IUC = .40		**.200**	**.100**	**.150**	**.300**
Rate = .05	**n = 240**	.192	.111	.164	.382
	n = 480	.194	.104	.158	.366
	n = 960	.199	.102	.152	.326
Rate = .10	**n = 240**	.183	.099	.159	.392
	n = 480	.193	.102	.160	.389
	n = 960	.192	.100	.159	.370
Rate = .20	**n = 240**	.208	.100	.148	.400
	n = 480	.194	.083	.142	.382
	n = 960	.192	.074	.137	.376

Table A9. IUC and π^* Estimates
Four-Variable Extended Proctor Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model IUC			Proctor Model Pistar			$\pi_{IUC} - \pi^*$			$\lvert \pi_{IUC} - \pi^* \rvert$			Scaled Value of $\lvert \pi_{IUC} - \pi^* \rvert$			Goodman Model IUC		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10																		
Rate = .05	.144	.141	.132	.101	.087	.070	.043	.054	.063	.073	.078	.075	.732	.777	.753	.261	.252	.248
Rate = .10	.160	.136	.134	.143	.111	.085	.017	.025	.048	.080	.073	.074	.800	.732	.741	.424	.416	.415
Rate = .20	.161	.142	.139	.195	.138	.103	-.034	.004	.035	.102	.076	.071	1.015	.758	.710	.675	.672	.672
IUC = .25																		
Rate = .05	.272	.267	.270	.149	.133	.118	.124	.133	.152	.144	.145	.155	.577	.578	.620	.349	.341	.337
Rate = .10	.271	.256	.259	.175	.141	.118	.096	.115	.141	.133	.138	.157	.532	.553	.630	.451	.450	.445
Rate = .20	.249	.236	.236	.208	.160	.130	.041	.076	.106	.101	.107	.120	.405	.428	.480	.656	.653	.649
IUC = .40																		
Rate = .05	.365	.380	.401	.196	.182	.166	.170	.198	.235	.180	.200	.236	.451	.501	.590	.433	.426	.424
Rate = .10	.326	.336	.341	.209	.180	.156	.117	.156	.185	.153	.174	.195	.383	.435	.488	.497	.488	.491
Rate = .20	.316	.303	.315	.234	.192	.172	.082	.111	.143	.123	.130	.150	.308	.326	.374	.644	.639	.639

Table A10. Error Rate Recovery
Four-Variable Extended Proctor Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model Error Rate			Proctor Pistar Model Error Rate			$\Gamma - \Gamma^*$			$\lvert \Gamma - \Gamma^* \rvert$			Scaled Value of $\lvert \Gamma - \Gamma^* \rvert$		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10															
Rate = .05	.043	.048	.049	.023	.029	.036	.020	.019	.013	.022	.019	.013	.437	.383	.263
Rate = .10	.093	.096	.098	.055	.067	.077	.038	.030	.022	.040	.031	.022	.403	.306	.219
Rate = .20	.189	.194	.195	.136	.158	.170	.053	.036	.024	.059	.042	.031	.294	.210	.155
IUC = .25															
Rate = .05	.049	.050	.051	.026	.032	.040	.023	.018	.011	.024	.018	.012	.478	.368	.240
Rate = .10	.093	.095	.097	.056	.068	.078	.037	.028	.019	.040	.029	.020	.395	.288	.199
Rate = .20	.189	.195	.199	.130	.152	.164	.058	.043	.035	.064	.047	.038	.318	.236	.188
IUC = .40															
Rate = .05	.047	.048	.048	.028	.034	.042	.020	.013	.006	.022	.015	.008	.436	.290	.163
Rate = .10	.097	.101	.102	.059	.070	.080	.039	.031	.021	.040	.031	.022	.402	.315	.217
Rate = .20	.194	.198	.200	.125	.144	.152	.068	.053	.048	.071	.055	.048	.356	.272	.240

Table A11. Mixing Proportion Recovery
Four-Variable Extended Proctor Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Proctor						Proctor Pistar					
IUC = .10		**.100**	**.100**	**.200**	**.250**	**.250**	**.100**						$\hat{\pi}^*$
Rate = .05	**n = 240**	.091	.091	.191	.242	.241	.144	. 141	.107	.189	.234	.228	.101
	n = 480	.078	.091	.196	.250	.245	.141	.143	.108	.191	.240	.232	.087
	n = 960	.081	.092	.199	.250	.247	.132	.147	.109	.195	.243	.237	.070
Rate = .10	**n = 240**	.073	.087	.194	.242	.244	.160	.135	.102	.182	.223	.215	.143
	n = 480	.084	.090	.197	.244	.248	.136	.144	.105	.186	.228	.225	.111
	n = 960	.083	.090	.199	.246	.249	.134	.151	.106	.190	.235	.233	.085
Rate = .20	**n = 240**	.079	.094	.183	.245	.238	.161	.144	.102	.159	.206	.194	.195
	n = 480	.090	.088	.190	.251	.239	.142	.164	.103	.165	.223	.207	.138
	n = 960	.093	.088	.191	.246	.244	.139	.172	.105	.169	.226	.224	.103
IUC = .25		**.100**	**.100**	**.150**	**.200**	**.200**	**.250**						$\hat{\pi}^*$
Rate = .05	**n = 240**	.088	.098	.147	.201	.194	.272	.209	.125	.145	.191	.181	.149
	n = 480	.086	.099	.150	.203	.196	.267	.214	.125	.147	.195	.185	.133
	n = 960	.082	.098	.149	.203	.198	.270	.220	.125	.148	.198	.191	.118
Rate = .10	**n = 240**	.103	.089	.144	.199	.194	.271	.219	.117	.138	.182	.170	.175
	n = 480	.107	.092	.148	.201	.197	.256	.231	.119	.143	.188	.179	.141
	n = 960	.099	.094	.149	.202	.198	.259	.239	.121	.145	.192	.185	.118
Rate = .20	**n = 240**	.121	.100	.143	.197	.189	.249	.249	.115	.119	.160	.150	.208
	n = 480	.126	.098	.145	.201	.194	.236	.268	.121	.114	.173	.164	.160
	n = 960	.127	.093	.149	.200	.195	.236	.284	.122	.113	.180	.172	.130
IUC = .40		**.050**	**.050**	**.100**	**.200**	**.200**	**.400**						$\hat{\pi}^*$
Rate = .05	**n = 240**	.082	.059	.099	.199	.197	.365	.235	.093	.101	.189	.186	.196
	n = 480	.072	.055	.098	.199	.196	.380	.242	.093	.101	.193	.189	.182
	n = 960	.055	.050	.098	.200	.196	.401	.250	.092	.101	.198	.193	.166
Rate = .10	**n = 240**	.117	.064	.098	.199	.196	.326	.259	.089	.095	.177	.171	.209
	n = 480	.105	.061	.100	.200	.198	.336	.271	.090	.098	.183	.179	.180
	n = 960	.101	.058	.101	.200	.200	.341	.283	.089	.098	.188	.186	.156
Rate = .20	**n = 240**	.123	.076	.097	.194	.194	.316	.301	.088	.079	.151	.147	.234
	n = 480	.133	.076	.098	.191	.200	.303	.326	.088	.073	.158	.163	.192
	n = 960	.124	.069	.098	.195	.200	.315	.337	.089	.064	.171	.168	.172

Table A12. Conditional Probability Recovery
Four-Variable Extended Proctor Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Proctor			
IUC = .10		**.200**	**.100**	**.150**	**.300**
Rate = .05	**n = 240**	.270	.216	.252	.413
	n = 480	.221	.136	.177	.367
	n = 960	.216	.110	.161	.363
Rate = .10	**n = 240**	.241	.167	.204	.356
	n = 480	.232	.147	.195	.398
	n = 960	.229	.129	.174	.391
Rate = .20	**n = 240**	.238	.191	.212	.356
	n = 480	.248	.170	.201	.359
	n = 960	.252	.151	.194	.367
IUC = .25		**.200**	**.100**	**.150**	**.300**
Rate = .05	**n = 240**	.199	.120	.168	.382
	n = 480	.188	.099	.158	.353
	n = 960	.188	.098	.152	.320
Rate = .10	**n = 240**	.226	.132	.175	.399
	n = 480	.221	.117	.166	.395
	n = 960	.212	.107	.159	.394
Rate = .20	**n = 240**	.225	.136	.166	.386
	n = 480	.225	.115	.152	.389
	n = 960	.229	.102	.143	.372
IUC = .40		**.200**	**.100**	**.150**	**.300**
Rate = .05	**n = 240**	.196	.116	.176	.379
	n = 480	.201	.110	.168	.349
	n = 960	.204	.106	.158	.319
Rate = .10	**n = 240**	.198	.105	.171	.439
	n = 480	.195	.091	.164	.411
	n = 960	.197	.090	.162	.407
Rate = .20	**n = 240**	.177	.094	.157	.403
	n = 480	.173	.080	.144	.393
	n = 960	.171	.073	.137	.369

Table A13. IUC and π^* Estimates
Six-Variable Extended Proctor Model
Equal Mixing Proportions, Equal Conditional Probabilities

Estimated Mean	Extended Proctor Model IUC			Proctor Model Pistar			$\pi_{IUC} - \pi^*$			$\left\| \pi_{IUC} - \pi^* \right\|$			Scaled Value of $\left\| \pi_{IUC} - \pi^* \right\|$			Goodman Model IUC		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10																		
Rate = .05	.113	.102	.101	.189	.155	.135	-.077	-.053	-034	.077	.053	.034	.767	.530	.341	.336	.329	.327
Rate = .10	.110	.102	.100	.233	.210	.175	-.123	-.109	-.075	.125	.110	.076	1.246	1.098	.763	.519	.516	.515
Rate = .20	.136	.137	.128	.336	.255	.180	-.199	-.118	-.053	.208	.140	.094	2.081	1.401	.941	.772	.771	.770
IUC = .25																		
Rate = .05	.252	.249	.250	.277	.246	.220	-.025	.003	.026	.034	.024	.029	.136	.096	.118	.432	.430	.429
Rate = .10	.254	.254	.253	.301	.266	.223	-.047	-.012	.030	.061	.037	.038	.246	.149	.150	.581	.579	.580
Rate = .20	.263	.258	.269	.352	.268	.200	-.089	-.010	.068	.141	.107	.097	.563	.428	.389	.791	.791	.791
IUC = .40																		
Rate = .05	.398	.400	.398	.367	.339	.313	.031	.061	.085	.039	.062	.085	.098	.156	.213	.540	.539	.539
Rate = .10	.396	.401	.400	.365	.324	.284	.031	.077	.116	.053	.079	.116	.133	.198	.291	.659	.657	.657
Rate = .20	.365	.396	.404	.371	.280	.213	-.005	.116	.191	.139	.150	.195	.347	.375	.487	.824	.823	.823

Table A14. Error Rate Recovery
Six-Variable Extended Proctor Model
Equal Mixing Proportions, Equal Conditional Probabilities

Estimated Mean	Extended Proctor Model Error Rate			Proctor Pistar Model Error Rate			$\Gamma - \Gamma^*$			$\lvert \Gamma - \Gamma^* \rvert$			Scaled Value of $\lvert \Gamma - \Gamma^* \rvert$		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10															
Rate = .05	.048	.049	.049	.023	.032	.036	.025	.017	.013	.025	.017	.013	.504	.344	.260
Rate = .10	.100	.100	.100	.065	.066	.077	.035	.034	.023	.035	.034	.023	.345	.341	.230
Rate = .20	.196	.197	.197	.143	.162	.177	.053	.035	.020	.054	.035	.020	.268	.176	.101
IUC = .25															
Rate = .05	.049	.050	.050	.026	.036	.042	.023	.014	.008	.023	.015	.009	.464	.292	.170
Rate = .10	.100	.100	.100	.074	.081	.094	.027	.020	.007	.027	.021	.010	.269	.205	.096
Rate = .20	.197	.200	.199	.158	.178	.195	.039	.021	.004	.040	.025	.012	.202	.123	.059
IUC = .40															
Rate = .05	.051	.051	.051	.030	.040	.049	.020	.010	.002	.021	.012	.007	.415	.241	.135
Rate = .10	.101	.101	.101	.085	.097	.110	.016	.004	-.009	.020	.015	.013	.200	.148	.130
Rate = .20	.197	.197	.198	.170	.195	.212	.027	.002	-.014	.035	.020	.018	.174	.102	.091

Table A15. Mixing Proportion Recovery
Six-Variable Extended Proctor Model
Equal Mixing Proportions, Equal Conditional Probabilities

Model		Extended Proctor								Proctor Pistar							
IUC = .10		**.129**	**.129**	**.129**	**.129**	**.129**	**.129**	**.129**	**.100**								$\hat{\pi}^*$
Rate = .05	**n = 960**	.132	.123	.127	.126	.127	.126	.127	.113	.125	.117	.118	.115	.116	.113	.107	.189
	n = 1920	.130	.126	.129	.128	.130	.128	.130	.102	.131	.123	.122	.120	.122	.117	.111	.155
	n = 3840	.130	.126	.130	.128	.130	.128	.130	.101	.135	.125	.126	.121	.124	.120	.115	.135
Rate = .10	**n = 960**	.126	.127	.128	.127	.128	.127	.128	.110	.123	.113	.110	.110	.108	.105	.096	.233
	n = 1920	.126	.129	.130	.128	.130	.127	.130	.102	.126	.116	.114	.112	.112	.109	.101	.210
	n = 3840	.126	.129	.130	.129	.129	.128	.129	.100	.133	.122	.119	.117	.118	.115	.101	.175
Rate = .20	**n = 960**	.122	.112	.128	.129	.122	.124	.122	.136	.114	.095	.101	.098	.090	.093	.073	.336
	n = 1920	.122	.107	.129	.130	.123	.124	.123	.137	.132	.105	.110	.110	.100	.105	.084	.255
	n = 3840	.125	.110	.130	.132	.124	.124	.124	.128	.148	.114	.119	.119	.109	.112	.098	.180
IUC = .25		**.107**	**.107**	**.107**	**.107**	**.107**	**.107**	**.107**	**.250**								$\hat{\pi}^*$
Rate = .05	**n = 960**	.107	.108	.106	.106	.107	.106	.107	.252	.127	.112	.102	.099	.098	.094	.092	.277
	n = 1920	.105	.109	.108	.107	.106	.106	.108	.249	.134	.117	.107	.102	.102	.097	.094	.246
	n = 3840	.105	.109	.108	.106	.107	.106	.107	.250	.138	.120	.110	.105	.105	.102	.096	.224
Rate = .10	**n = 960**	.109	.104	.108	.106	.104	.106	.104	.254	.140	.108	.100	.094	.090	.088	.080	.301
	n = 1920	.108	.104	.108	.106	.105	.107	.105	.254	.146	.113	.105	.099	.094	.093	.085	.266
	n = 3840	.109	.104	.108	.106	.105	.107	.105	.253	.157	.119	.111	.103	.099	.098	.090	.223
Rate = .20	**n = 960**	.112	.100	.103	.103	.110	.103	.110	.263	.162	.094	.090	.085	.083	.079	.054	.352
	n = 1920	.101	.106	.107	.107	.111	.104	.111	.258	.185	.105	.099	.096	.093	.091	.063	.268
	n = 3840	.088	.107	.108	.106	.111	.105	.111	.269	.206	.114	.104	.103	.100	.097	.075	.200
IUC = .40		**.086**	**.086**	**.086**	**.086**	**.086**	**.086**	**.086**	**.400**								$\hat{\pi}^*$
Rate = .05	**n = 960**	.085	.085	.087	.086	.085	.087	.085	.398	.132	.100	.090	.082	.080	.079	.071	.367
	n = 1920	.084	.084	.087	.086	.086	.087	.086	.400	.139	.104	.094	.085	.084	.082	.073	.339
	n = 3840	.085	.084	.087	.086	.086	.087	.086	.398	.146	.108	.097	.089	.087	.086	.073	.313
Rate = .10	**n = 960**	.086	.081	.087	.085	.083	.086	.086	.396	.157	.108	.088	.078	.075	.070	.057	.365
	n = 1920	.082	.089	.087	.084	.084	.086	.086	.401	.168	.114	.091	.082	.080	.077	.063	.324
	n = 3840	.083	.090	.088	.084	.084	.086	.086	.400	.181	.121	.098	.084	.085	.079	.068	.284
Rate = .20	**n = 960**	.118	.095	.084	.084	.086	.084	.085	.365	.216	.091	.077	.073	.067	.065	.040	.371
	n = 1920	.092	.086	.083	.086	.084	.086	.086	.396	.258	.099	.084	.085	.073	.075	.047	.280
	n = 3840	.085	.083	.085	.085	.085	.086	.086	.404	.289	.107	.087	.091	.079	.081	.054	.213

Table A16. Conditional Probability Recovery
Six-Variable Extended Proctor Model
Equal Mixing Proportions, Equal Conditional Probabilities

Model		Extended Proctor					
IUC = .10		**.300**	**.300**	**.300**	**.300**	**.300**	**.300**
Rate = .05	**n = 960**	.354	.353	.357	.363	.365	.358
	n = 1920	.311	.308	.315	.329	.329	.309
	n = 3840	.313	.300	.309	.316	.314	.296
Rate = .10	**n = 960**	.276	.322	.326	.343	.345	.365
	n = 1920	.258	.290	.300	.314	.314	.321
	n = 3840	.270	.289	.291	.304	.299	.304
Rate = .20	**n = 960**	.347	.307	.330	.367	.388	.347
	n = 1920	.345	.298	.330	.357	.386	.347
	n = 3840	.347	.295	.328	.370	.400	.354
IUC = .25		**.300**	**.300**	**.300**	**.300**	**.300**	**.300**
Rate = .05	**n = 960**	.307	.311	.315	.310	.312	.316
	n = 1920	.297	.302	.302	.300	.303	.304
	n = 3840	.298	.301	.302	.298	.302	.302
Rate = .10	**n = 960**	.297	.303	.306	.303	.308	.317
	n = 1920	.301	.298	.300	.294	.298	.304
	n = 3840	.304	.299	.300	.294	.299	.304
Rate = .20	**n = 960**	.287	.327	.312	.331	.336	.340
	n = 1920	.262	.315	.309	.324	.332	.337
	n = 3840	.270	.300	.298	.307	.308	.315
IUC = .40		**.300**	**.300**	**.300**	**.300**	**.300**	**. 300**
Rate = .05	**n = 960**	.303	.302	.298	.298	.307	.308
	n = 1920	.304	.300	.295	.293	.305	.301
	n = 3840	.303	.301	.295	.294	.304	.301
Rate = .10	**n = 960**	.294	.298	.303	.304	.299	.308
	n = 1920	.296	.294	.299	.300	.294	.302
	n = 3840	.295	.295	.300	.300	.295	.303
Rate = .20	**n = 960**	.269	.328	.320	.317	.320	.338
	n = 1920	.283	.317	.304	.301	.306	.318
	n = 3840	.299	.310	.301	.296	.300	.305

Table A17. IUC and π^* Estimates
Six-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model IUC			Proctor Model Pistar			$\pi_{IUC} - \pi^*$			$\lvert \pi_{IUC} - \pi^* \rvert$			Scaled Value of $\lvert \pi_{IUC} - \pi^* \rvert$			Goodman Model IUC		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10																		
Rate = .05	.106	.103	.103	.190	.157	.139	-.085	-.054	-.035	.085	.054	.036	.846	.538	.356	.318	.311	.307
Rate = .10	.115	.109	.103	.239	.219	.190	-.124	-.110	-.086	.126	.111	.087	1.257	1.114	.873	.507	.503	.503
Rate = .20	.140	.136	.131	.343	.267	.199	-.202	-.131	-.068	.206	.144	.093	2.058	1.443	.928	.755	.755	.756
IUC = .25																		
Rate = .05	.245	.248	.249	.280	.250	.230	-.035	-.003	.018	.040	.022	.022	.160	.089	.090	.417	.411	.410
Rate = .10	.249	.251	.247	.305	.281	.254	-.056	-.030	-.007	.067	.044	.033	.269	.175	.131	.552	.549	.549
Rate = .20	.269	.273	.280	.381	.309	.255	-.112	-.037	.025	.129	.081	.060	.516	.324	.240	.764	.764	.763
IUC = .40																		
Rate = .05	.393	.397	.397	.369	.345	.323	.024	.053	.074	.034	.053	.074	.085	.134	.186	.525	.521	.521
Rate = .10	.404	.413	.411	.384	.355	.320	.020	.058	.091	.049	.062	.091	.122	.155	.228	.632	.630	.630
Rate = .20	.361	.364	.366	.420	.361	.312	-.059	.003	.054	.103	.072	.067	.258	.180	.167	.787	.787	.787

Table A18. Error Rate Recovery
Six-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model Error Rate			Proctor Pistar Model Error Rate			$\Gamma - \Gamma^*$			$\lvert \Gamma - \Gamma^* \rvert$			Scaled Value of $\lvert \Gamma - \Gamma^* \rvert$		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10															
Rate = .05	.049	.050	.050	.021	.030	.033	.029	.020	.017	.029	.020	.017	.570	.400	.334
Rate = .10	.100	.100	.100	.062	.061	.070	.037	.039	.030	.037	.039	.030	.374	.369	.298
Rate = .20	.197	.199	.199	.137	.154	.169	.060	.045	.029	.060	.045	.029	.300	.224	.147
IUC = .25															
Rate = .05	.050	.050	.050	.021	.031	.035	.029	.020	.015	.029	.020	.015	.116	.393	.297
Rate = .10	.100	.099	.099	.065	.066	.075	.035	.033	.025	.035	.033	.025	.346	.326	.248
Rate = .20	.201	.201	.201	.138	.156	.170	.062	.045	.031	.062	.045	.031	.312	.224	.154
IUC = .40															
Rate = .05	.051	.050	.050	.021	.030	.037	.029	.020	.013	.029	.020	.013	.582	.398	.263
Rate = .10	.101	.100	.100	.069	.073	.084	.033	.028	.017	.033	.028	.017	.327	.277	.169
Rate = .20	.202	.202	.203	.137	.155	.172	.065	.047	.031	.065	.047	.031	.325	.235	.154

Table A19. Mixing Proportion Recovery
Six-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Proctor								Proctor Pistar							
IUC = .10		**.129**	**.129**	**.129**	**.129**	**.129**	**.129**	**.129**	**.100**								$\hat{\pi}^*$
Rate = .05	**n = 960**	.136	.125	.123	.128	.126	.128	.129	.106	.133	.118	.111	.114	.112	.112	.108	.190
	n = 1920	.128	.127	.126	.129	.128	.129	.130	.103	.139	.123	.116	.119	.117	.116	.114	.157
	n = 3840	.126	.127	.127	.129	.128	.130	.130	.103	.141	.124	.118	.121	.119	.120	.117	.139
Rate = .10	**n = 960**	.118	.130	.125	.125	.128	.130	.130	.115	.130	.115	.105	.104	.106	.105	.096	.239
	n = 1920	.117	.130	.128	.127	.129	.130	.131	.109	.130	.115	.108	.107	.109	.108	.103	.219
	n = 3840	.121	.132	.128	.126	.129	.130	.130	.103	.136	.121	.113	.110	.114	.112	.105	.190
Rate = .20	**n = 960**	.110	.112	.131	.124	.130	.128	.125	.140	.128	.095	.092	.089	.094	.089	.069	.343
	n = 1920	.106	.116	.133	.124	.131	.129	.126	.136	.146	.106	.098	.100	.103	.100	.080	.267
	n = 3840	.108	.118	.133	.123	.130	.129	.127	.131	.162	.113	.106	.106	.111	.109	.094	.199
IUC = .25		**.107**	**.107**	**.107**	**.107**	**.107**	**.107**	**.107**	**.250**								$\hat{\pi}^*$
Rate = .05	**n = 960**	.118	.108	.105	.106	.105	.107	.107	.245	.149	.109	.094	.094	.093	.093	.088	.280
	n = 1920	.111	.108	.107	.106	.106	.108	.107	.248	.156	.113	.098	.098	.097	.097	.092	.250
	n = 3840	.110	.108	.107	.106	.106	.108	.107	.249	.160	.114	.101	.100	.099	.100	.096	.230
Rate = .10	**n = 960**	.105	.112	.106	.107	.107	.106	.109	.249	.159	.109	.088	.089	.088	.083	.080	.305
	n = 1920	.102	.112	.107	.107	.107	.105	.108	.251	.161	.110	.093	.091	.091	.088	.085	.281
	n = 3840	.104	.114	.107	.106	.108	.105	.108	.247	.168	.115	.095	.094	.096	.093	.085	.254
Rate = .20	**n = 960**	.095	.100	.108	.107	.105	.110	.106	.269	.182	.088	.066	.074	.074	.074	.060	.381
	n = 1920	.088	.103	.107	.108	.106	.110	.105	.273	.204	.099	.068	.083	.085	.084	.068	.309
	n = 3840	.079	.102	.109	.109	.105	.110	.106	.280	.222	.108	.061	.090	.090	.093	.080	.255
IUC = .40		**.086**	**.086**	**.086**	**.086**	**.086**	**.086**	**.086**	**.400**								$\hat{\pi}^*$
Rate = .05	**n = 960**	.094	.088	.085	.085	.084	.087	.085	.393	.162	.097	.077	.075	.075	.075	.069	.369
	n = 1920	.088	.087	.085	.085	.085	.087	.085	.397	.171	.100	.080	.077	.078	.077	.072	.345
	n = 3840	.087	.087	.086	.085	.085	.087	.085	.397	.177	.102	.082	.080	.080	.080	.075	.323
Rate = .10	**n = 960**	.081	.089	.085	.083	.085	.085	.087	.404	.187	.097	.070	.068	.070	.065	.060	.384
	n = 1920	.073	.087	.085	.084	.085	.087	.086	.413	.191	.098	.074	.072	.073	.072	.066	.355
	n = 3840	.074	.088	.085	.084	.085	.087	.086	.411	.203	.103	.077	.076	.077	.076	.067	.320
Rate = .20	**n = 960**	.126	.091	.082	.086	.089	.081	.085	.361	.235	.083	.045	.056	.061	.052	.048	.420
	n = 1920	.121	.091	.083	.085	.090	.081	.086	.364	.258	.093	.042	.061	.071	.060	.052	.361
	n = 3840	.116	.092	.083	.086	.090	.081	.086	.366	.275	.104	.028	.070	.078	.068	.064	.312

Table A20. Conditional Probability Recovery
Six-Variable Extended Proctor Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Proctor
IUC = .10		**.200 .100 .150 .300 .250 .400**
Rate = .05	**n = 960**	.246 .185 .213 .345 .310 .449
	n = 1920	.201 .126 .159 .304 .258 .402
	n = 3840	.192 .109 .145 .292 .245 .388
Rate = .10	**n = 960**	.198 .138 .179 .319 .274 .451
	n = 1920	.181 .099 .153 .296 .249 .409
	n = 3840	.186 .092 .151 .295 .248 .404
Rate = .20	**n = 960**	.230 .116 .168 .307 .274 .465
	n = 1920	.206 .094 .150 .293 .277 .442
	n = 3840	.208 .087 .152 .291 .270 .422
IUC = .25		**.200 .100 .150 .300 .250 .400**
Rate = .05	**n = 960**	.213 .122 .170 .318 .268 .426
	n = 1920	.201 .103 .156 .306 .253 .407
	n = 3840	.203 .099 .152 .303 .249 .404
Rate = .10	**n = 960**	.198 .107 .157 .316 .257 .414
	n = 1920	.197 .102 .151 .309 .250 .401
	n = 3840	.195 .098 .151 .307 .250 .400
Rate = .20	**n = 960**	.199 .094 .140 .304 .248 .421
	n = 1920	.199 .095 .141 .296 .246 .408
	n = 3840	.203 .099 .144 .292 .245 .394
IUC = .40		**.200 .100 .150 .300 .250 .400**
Rate = .05	**n = 960**	.201 .110 .158 .313 .258 .414
	n = 1920	.197 .101 .151 .304 .248 .403
	n = 3840	.196 .100 .149 .304 .247 .402
Rate = .10	**n = 960**	.194 .104 .154 .304 .256 .402
	n = 1920	.196 .102 .150 .299 .248 .394
	n = 3840	.196 .102 .150 .298 .247 .393
Rate = .20	**n = 960**	.181 .079 .136 .314 .253 .430
	n = 1920	.187 .083 .139 .310 .253 .424
	n = 3840	.185 .085 .141 .308 .252 .417

Table A21. IUC and π^* Estimates
Six-Variable Extended Proctor Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model IUC			Proctor Model Pistar			$\pi_{IUC} - \pi^*$			$\lvert \pi_{IUC} - \pi^* \rvert$			Scaled Value of $\lvert \pi_{IUC} - \pi^* \rvert$			Goodman Model IUC		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10																		
Rate = .05	.104	.099	.098	.181	.150	.139	.-077	-.050	-.041	.077	.051	.041	.771	.506	.410	.310	.303	.300
Rate = .10	.108	.102	.101	.238	.223	.188	-.130	-.121	-.087	.130	.121	.087	1.300	1.206	.870	.500	.497	.496
Rate = .20	.103	.107	.106	.349	.270	.200	-.246	-.162	-.093	.246	.163	.094	2.458	1.625	.944	.762	.762	.761
IUC = .25																		
Rate = .05	.248	.250	.250	.273	.245	.229	-.025	.006	.022	.032	.021	.023	.129	.084	.094	.391	.385	.385
Rate = .10	.260	.262	.259	.308	.291	.250	-.049	-.030	.009	.056	.040	.031	.225	.160	.123	.530	.527	.530
Rate = .20	.233	.245	.250	.392	.312	.256	-.159	-.067	-.006	.160	.076	.040	.641	.306	.161	.742	.743	.741
IUC = .40																		
Rate = .05	.397	.402	.403	.369	.346	.327	.028	.056	.077	.037	.057	.077	.093	.141	.192	.515	.511	.511
Rate = .10	.390	.395	.393	.377	.352	.322	.013	.043	.071	.045	.050	.073	.112	.126	.182	.619	.616	.617
Rate = .20	.368	.380	.389	.424	.367	.326	-.056	.013	.064	.087	.059	.071	.217	.147	.176	.780	.780	.780

Table A22. Error Rate Recovery
Six-Variable Extended Proctor Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Proctor Model Error Rate			Proctor Pistar Model Error Rate			$\Gamma - \Gamma^*$			$\lvert \Gamma - \Gamma^* \rvert$			Scaled Value of $\lvert \Gamma - \Gamma^* \rvert$		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10															
Rate = .05	.050	.050	.050	.024	.031	.032	.026	.019	.018	.026	.019	.018	.526	.380	.352
Rate = .10	.100	.100	.100	.062	.060	.070	.039	.041	.030	.039	.041	.030	.389	.405	.304
Rate = .20	.200	.200	.200	.137	.154	.170	.063	.046	.030	.063	.046	.030	.314	.228	.150
IUC = .25															
Rate = .05	.050	.050	.049	.024	.032	.035	.026	.017	.015	.026	.017	.015	.525	.350	.290
Rate = .10	.100	.100	.100	.064	.064	.076	.037	.036	.024	.037	.036	.024	.366	.361	.240
Rate = .20	.202	.201	.202	.137	.158	.173	.065	.043	.028	.065	.043	.028	.325	.216	.141
IUC = .40															
Rate = .05	.050	.049	.049	.021	.030	.036	.029	.020	.014	.029	.020	.014	.579	.391	.271
Rate = .10	.102	.101	.101	.068	.070	.080	.034	.030	.021	.034	.030	.021	.341	.303	.208
Rate = .20	.201	.201	.201	.136	.155	.172	.065	.046	.029	.065	.046	.029	.327	.229	.147

Table A23. Mixing Proportion Recovery
Six-Variable Extended Proctor Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Proctor								Proctor Pistar							
IUC = .10		**.050**	**.050**	**.100**	**.100**	**.200**	**.200**	**.200**	**.100**								$\hat{\pi}^*$
Rate = .05	**n = 960**	.059	.048	.097	.097	.199	.198	.199	.104	.066	.048	.085	.087	.178	.179	.175	.181
	n = 1920	.054	.049	.100	.098	.201	.200	.201	.099	.070	.050	.089	.090	.184	.185	.182	.150
	n = 3840	.053	.049	.100	.098	.201	.201	.201	.098	.071	.052	.092	.092	.186	.186	.184	.139
Rate = .10	**n = 960**	.049	.048	.100	.099	.199	.199	.199	.108	.068	.045	.077	.083	.165	.165	.157	.238
	n = 1920	.050	.049	.101	.100	.200	.200	.200	.102	.069	.048	.082	.087	.165	.167	.160	.223
	n = 3840	.050	.049	.101	.099	.200	.200	.200	.101	.072	.050	.086	.088	.173	.174	.168	.188
Rate = .20	**n = 960**	.055	.046	.098	.102	.201	.195	.201	.103	.072	.042	.061	.073	.144	.135	.123	.349
	n = 1920	.051	.043	.099	.104	.200	.196	.200	.107	.085	.045	.069	.082	.156	.152	.142	.270
	n = 3840	.050	.043	.100	.103	.201	.196	.201	.106	.095	.049	.074	.086	.170	.166	.161	.200
IUC = .25		**.050**	**.050**	**.050**	**.100**	**.100**	**.200**	**.200**	**.250**								$\hat{\pi}^*$
Rate = .05	**n = 960**	.057	.048	.051	.097	.099	.200	.201	.248	.100	.056	.046	.084	.088	.179	.176	.273
	n = 1920	.053	.048	.051	.097	.099	.201	.200	.250	.105	.057	.045	.087	.092	.186	.183	.245
	n = 3840	.052	.048	.051	.098	.099	.201	.201	.250	.107	.058	.049	.090	.094	.188	.186	.229
Rate = .10	**n = 960**	.045	.050	.049	.098	.099	.201	.197	.260	.110	.056	.038	.080	.083	.166	.158	.308
	n = 1920	.043	.050	.051	.098	.100	.200	.198	.262	.111	.057	.044	.081	.088	.168	.160	.291
	n = 3840	.045	.050	.050	.098	.100	.200	.198	.259	.120	.059	.045	.086	.091	.178	.170	.250
Rate = .20	**n = 960**	.062	.055	.049	.094	.101	.205	.201	.233	.129	.052	.026	.063	.076	.140	.122	.392
	n = 1920	.050	.055	.049	.095	.102	.204	.200	.245	.146	.060	.025	.070	.085	.159	.142	.312
	n = 3840	.044	.057	.049	.096	.101	.204	.200	.250	.158	.066	.020	.076	.092	.171	.161	.256
IUC = .40		**.050**	**.050**	**.100**	**.100**	**.100**	**.100**	**.100**	**.400**								$\hat{\pi}^*$
Rate = .05	**n = 960**	.056	.047	.100	.098	.102	.098	.102	.397	.130	.063	.090	.087	.090	.085	.085	.369
	n = 1920	.050	.047	.100	.099	.102	.098	.102	.402	.137	.064	.093	.090	.094	.087	.088	.346
	n = 3840	.049	.047	.100	.099	.102	.098	.102	.403	.141	.066	.095	.093	.096	.090	.092	.327
Rate = .10	**n = 960**	.057	.053	.099	.099	.100	.101	.101	.390	.159	.066	.081	.081	.083	.081	.071	.377
	n = 1920	.053	.051	.100	.099	.101	.100	.101	.395	.161	.066	.085	.086	.087	.084	.079	.352
	n = 3840	.053	.051	.100	.100	.101	.101	.101	.393	.171	.068	.089	.089	.091	.089	.080	.322
Rate = .20	**n = 960**	.080	.051	.103	.097	.101	.100	.100	.368	.207	.060	.050	.065	.069	.067	.058	.424
	n = 1920	.067	.049	.104	.098	.102	.101	.100	.380	.227	.068	.039	.075	.081	.078	.065	.367
	n = 3840	.060	.048	.104	.098	.100	.101	.100	.389	.243	.076	.023	.080	.088	.086	.078	.326

Table A24. Conditional Probability Recovery
Six-Variable Extended Proctor Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Proctor
IUC = .10		**.200 .100 .150 .300 .250 .400**
Rate = .05	**n = 960**	.243 .175 .211 .337 .304 .456
	n = 1920	.209 .121 .162 .307 .262 .420
	n = 3840	.207 .108 .154 .301 .252 .408
Rate = .10	**n = 960**	.212 .123 .166 .317 .274 .444
	n = 1920	.202 .099 .149 .309 .256 .427
	n = 3840	.204 .092 .145 .304 .254 .418
Rate = .20	**n = 960**	.213 .111 .148 .320 .260 .471
	n = 1920	.214 .097 .139 .311 .247 .443
	n = 3840	.213 .085 .136 .302 .243 .421
IUC = .25		**.200 .100 .150 .300 .250 .400**
Rate = .05	**n = 960**	.211 .118 .168 .311 .265 .419
	n = 1920	.205 .105 .158 .301 .254 .406
	n = 3840	.203 .103 .157 .300 .253 .402
Rate = .10	**n = 960**	.204 .106 .156 .302 .253 .403
	n = 1920	.207 .101 .153 .300 .248 .396
	n = 3840	.205 .100 .153 .300 .247 .395
Rate = .20	**n = 960**	.188 .086 .142 .312 .253 .435
	n = 1920	.189 .096 .144 .304 .250 .415
	n = 3840	.188 .098 .145 .299 .248 .405
IUC = .40		**.200 .100 .150 .300 .250 .400**
Rate = .05	**n = 960**	.204 .112 .159 .310 .260 .410
	n = 1920	.200 .104 .153 .302 .252 .401
	n = 3840	.200 .103 .152 .302 .250 .398
Rate = .10	**n = 960**	.196 .098 .155 .307 .255 .412
	n = 1920	.197 .096 .154 .303 .251 .406
	n = 3840	.196 .095 .153 .303 .250 .404
Rate = .20	**n = 960**	.188 .085 .147 .315 .252 .430
	n = 1920	.193 .089 .148 .309 .248 .417
	n = 3840	.195 .091 .150 .306 .248 .410

Table A25. IUC and π^* Estimates
Four-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Equal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model IUC			Intrusion-Omission Model Pistar			$\pi_{IUC} - \pi^*$			$\left\| \pi_{IUC} - \pi^* \right\|$			Scaled Value of $\left\| \pi_{IUC} - \pi^* \right\|$			Goodman Model IUC		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10																		
Rates: Omission=.20 Intrusion=.05	.177	.173	.175	.130	.103	.076	.047	.070	.098	.111	.113	.123	1.114	1.127	1.229	.509	.507	.502
Rates: Omission=.05 Intrusion=.20	.190	.164	.163	.133	.100	.072	.057	.064	.091	.112	.110	.117	1.118	1.095	1.170	.527	.522	.521
IUC = .25																		
Rates: Omission=.20 Intrusion=.05	.276	.274	.264	.158	.123	.099	.118	.151	.165	.149	.169	.173	.596	.677	.691	.564	.561	.557
Rates: Omission=.05 Intrusion=.20	.277	.283	.282	.157	.117	.086	.120	.166	.196	.158	.187	.206	.632	.749	.822	.596	.594	.592
IUC = .40																		
Rates: Omission=.20 Intrusion=.05	.367	.382	.370	.178	.142	.119	.188	.240	.251	.213	.250	.256	.532	.625	.640	.648	.641	.642
Rates: Omission=.05 Intrusion=.20	.379	.384	.390	.177	.128	.094	.202	.256	.296	.225	.269	.302	.564	.674	.754	.663	.664	.658

Table A26. Error Rate Recovery
Four-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Equal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model Error Rate			Intrusion-Omission Pistar Model Error Rate			$\beta - \beta^*$			$\lvert \beta - \beta^* \rvert$			Scaled Value of $\lvert \beta - \beta^* \rvert$		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10															
Rates:															
Omission = .20	.169	.177	.178	.140	.159	.173	.030	.018	.005	.049	.041	.033	.243	.203	.165
Intrusion = .05	.049	.046	.045	.048	.055	.062	.0006	-.009	-.016	.027	.026	.025	.546	.512	.500
Rates:															
Omission = .05	.040	.041	.041	.031	.037	.040	.009	.005	.001	.021	.017	.014	.428	.335	.283
Intrusion = .20	.190	.192	.196	.147	.163	.179	.043	.029	.017	.063	.046	.033	.313	.232	.164
IUC = .25															
Rates:															
Omission = .20	.171	.183	.191	.160	.188	.205	.011	-.005	-.014	.056	.050	.041	.282	.250	.207
Intrusion = .05	.053	.049	.050	.069	.079	.085	-.016	-.030	-.035	.039	.041	.043	.783	.812	.857
Rates:															
Omission = .05	.043	.042	.039	.038	.040	.042	.005	.002	-.003	.026	.023	.022	.517	.468	.448
Intrusion = .20	.184	.184	.187	.154	.181	.202	.030	.003	-.015	.065	.050	.038	.327	.250	.191
IUC = .40															
Rates:															
Omission = .20	.173	.174	.184	.184	.209	.224	-.011	-.035	-.039	.063	.071	.061	.317	.357	.307
Intrusion = .05	.063	.058	.059	.092	.109	.117	-.030	-.051	-.059	.053	.062	.064	1.059	1.243	1.276
Rates:															
Omission = .05	.044	.039	.041	.041	.039	.041	.003	.0001	-.0001	.029	.027	.025	.572	.532	.510
Intrusion = .20	.182	.180	.185	.168	.203	.223	.014	-.023	-.038	.077	.065	.058	.387	.327	.291

Table A27. Mixing Proportion Recovery
Four-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Equal Conditional Probabilities

Model		Extended Intrusion-Omission						Intrusion-Omission Pistar					
IUC = .10		**.180**	**.180**	**.180**	**.180**	**.180**	**.100**						$\hat{\pi}^*$
Rates:	**n = 240**	.169	.167	.164	.163	.160	.177	.222	.192	.173	.157	.126	.130
Omission = .20	**n = 480**	.166	.168	.163	.167	.163	.173	.225	.196	.176	.167	.133	.103
Intrusion = .05	**n = 960**	.168	.165	.165	.165	.162	.175	.228	.198	.181	.172	.145	.076
Rates:	**n = 240**	.137	.153	.171	.172	.177	.190	.172	.165	.169	.176	.184	.133
Omission = .05	**n = 480**	.151	.161	.174	.174	.177	.164	.191	.172	.175	.179	.184	.100
Intrusion = .20	**n = 960**	.149	.163	.174	.175	.177	.163	.211	.177	.176	.180	.183	.072
IUC = .25		**.150**	**.150**	**.150**	**.150**	**.150**	**.250**						$\hat{\pi}^*$
Rates:	**n = 240**	.159	.148	.137	.139	.141	.276	.254	.184	.153	.146	.104	.158
Omission = .20	**n = 480**	.152	.147	.141	.140	.145	.274	.258	.188	.162	.151	.118	.123
Intrusion = .05	**n = 960**	.152	.150	.143	.144	.148	.264	.258	.193	.164	.158	.128	.099
Rates:	**n = 240**	.147	.140	.142	.148	.147	.277	.233	.155	.146	.154	.155	.157
Omission = .05	**n = 480**	.144	.139	.140	.146	.147	.283	.269	.161	.149	.154	.151	.117
Intrusion = .20	**n = 960**	.149	.141	.140	.144	.144	.282	.301	.165	.149	.154	.146	.086
IUC = .40		**.120**	**.120**	**.120**	**.120**	**.120**	**.400**						$\hat{\pi}^*$
Rates:	**n = 240**	.153	.123	.121	.119	.117	.367	.284	.168	.146	.136	.087	.178
Omission = .20	**n = 480**	.145	.122	.117	.118	.116	.382	.302	.175	.145	.147	.089	.142
Intrusion = .05	**n = 960**	.141	.128	.123	.119	.119	.370	.305	.180	.152	.150	.093	.119
Rates:	**n = 240**	.147	.119	.116	.118	.120	.379	.312	.135	.125	.125	.126	.177
Omission = .05	**n = 480**	.155	.116	.112	.115	.117	.384	.373	.135	.122	.125	.117	.128
Intrusion = .20	**n = 960**	.145	.119	.113	.116	.117	.390	.414	.135	.120	.124	.113	.094

Table A28. Conditional Probability Recovery
Four-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Equal Conditional Probabilities

Model		Extended Intrusion-Omission			
IUC = .10		**.300**	**.300**	**.300**	**.300**
Rates:	**n = 240**	.333	.361	.352	.361
Omission = .20	**n = 480**	.307	.351	.350	.355
Intrusion = .05	**n = 960**	.309	.357	.358	.333
Rates:	**n = 240**	.332	.335	.331	.327
Omission = .05	**n = 480**	.321	.344	.326	.352
Intrusion = .20	**n = 960**	.322	.346	.332	.338
IUC = .25		**.300**	**.300**	**.300**	**.300**
Rates:	**n = 240**	.281	.332	.333	.349
Omission = .20	**n = 480**	.263	.318	.334	.351
Intrusion = .05	**n = 960**	.251	.317	.332	.340
Rates:	**n = 240**	.293	.326	.336	.324
Omission = .05	**n = 480**	.293	.342	.332	.332
Intrusion = .20	**n = 960**	.290	.370	.354	.345
IUC = .40		**.300**	**.300**	**.300**	**.300**
Rates:	**n = 240**	.270	.325	.332	.348
Omission = .20	**n = 480**	.267	.320	.333	.326
Intrusion = .05	**n = 960**	.249	.312	.326	.333
Rates:	**n = 240**	.273	.305	.309	.319
Omission = .05	**n = 480**	.286	.333	.334	.331
Intrusion = .20	**n = 960**	.278	.333	.325	.322

Table A29. IUC and π^* Estimates
Four-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model IUC			Intrusion-Omission Model Pistar			$\pi_{IUC} - \pi^*$			$\left\| \pi_{IUC} - \pi^* \right\|$			Scaled Value of $\left\| \pi_{IUC} - \pi^* \right\|$			Goodman Model IUC		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10																		
Rates: Omission=.20 Intrusion=.05	.183	.162	.164	.129	.103	.077	.055	.059	.087	.107	.099	.109	1.067	.994	1.091	.472	.466	.464
Rates: Omission=.05 Intrusion=.20	.179	.161	.161	.133	.111	.083	.046	.049	.078	.094	.089	.098	.938	.891	.981	.503	.497	.493
IUC = .25																		
Rates: Omission=.20 Intrusion=.05	.272	.264	.252	.151	.122	.101	.121	.141	.152	.147	.161	.163	.587	.644	.653	.505	.499	.500
Rates: Omission=.05 Intrusion=.20	.270	.258	.249	.164	.142	.114	.107	.116	.135	.132	.132	.143	.530	.530	.572	.552	.545	.541
IUC = .40																		
Rates: Omission=.20 Intrusion=.05	.376	.380	.394	.175	.146	.125	.201	.234	.269	.219	.247	.274	.548	.618	.685	.581	.571	.570
Rates: Omission=.05 Intrusion=.20	.362	.349	.351	.194	.168	.146	.168	.181	.205	.188	.193	.209	.471	.482	.521	.611	.598	.600

Table A30. Error Rate Recovery
Four-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model Error Rate			Intrusion-Omission Pistar Model Error Rate			$\beta - \beta^*$			$\lvert \beta - \beta^* \rvert$			Scaled Value of $\lvert \beta - \beta^* \rvert$		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10															
Rates:															
Omission = .20	.180	.188	.193	.134	.142	.160	.046	.046	.033	.057	.054	.040	.287	.271	.202
Intrusion = .05	.045	.047	.043	.047	.055	.061	-.002	-.009	-.017	.026	.024	.024	.528	.487	.472
Rates:															
Omission = .05	.046	.045	.048	.026	.030	.033	.020	.016	.015	.025	.020	.018	.503	.397	.350
Intrusion = .20	.195	.201	.201	.145	.156	.170	.050	.046	.031	.062	.053	.038	.309	.263	.191
IUC = .25															
Rates:															
Omission = .20	.182	.192	.197	.126	.139	.147	.056	.053	.051	.066	.062	.055	.330	.310	.275
Intrusion = .05	.054	.051	.050	.065	.074	.082	-.010	-.023	-.032	.039	.039	.040	.774	.774	.790
Rates:															
Omission = .05	.048	.047	.048	.025	.030	.036	.023	.017	.012	.028	.021	.017	.569	.428	.342
Intrusion = .20	.200	.204	.204	.138	.147	.154	.062	.057	.051	.075	.066	.056	.375	.330	.278
IUC = .40															
Rates:															
Omission = .20	.176	.189	.195	.115	.116	.119	.060	.074	.076	.078	.082	.080	.388	.412	.399
Intrusion = .05	.062	.054	.050	.079	.093	.102	-.017	-.039	-.051	.050	.054	.058	1.002	1.077	1.162
Rates:															
Omission = .05	.048	.047	.048	.025	.032	.041	.022	.015	.007	.030	.024	.018	.591	.472	.363
Intrusion = .20	.183	.190	.198	.124	.135	.134	.060	.056	.064	.081	.072	.069	.403	.358	.345

Table A31. Mixing Proportion Recovery
Four-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Intrusion-Omission						Intrusion-Omission Pistar					
IUC = .10		**.180**	**.180**	**.180**	**.180**	**.180**	**.100**						$\hat{\pi}^*$
Rates:	**n = 240**	.151	.157	.171	.170	.169	.183	.251	.187	.169	.152	.114	.129
Omission = .20	**n = 480**	.154	.161	.177	.176	.169	.162	.258	.190	.172	.157	.121	.103
Intrusion = .05	**n = 960**	.148	.158	.178	.175	.176	.164	.258	.191	.177	.161	.136	.077
Rates:	**n = 240**	.130	.157	.180	.177	.178	.179	.186	.166	.165	.172	.178	.133
Omission = .05	**n = 480**	.140	.166	.178	.177	.177	.161	.200	.171	.165	.174	.179	.111
Intrusion = .20	**n = 960**	.133	.166	.182	.178	.180	.161	.221	.176	.169	.174	.178	.083
IUC = .25		**.150**	**.150**	**.150**	**.150**	**.150**	**.250**						$\hat{\pi}^*$
Rates:	**n = 240**	.152	.148	.147	.143	.138	.272	.320	.181	.141	.123	.085	.151
Omission = .20	**n = 480**	.153	.145	.148	.147	.143	.264	.333	.183	.143	.127	.092	.122
Intrusion = .05	**n = 960**	.155	.149	.148	.149	.146	.252	.342	.186	.144	.129	.099	.101
Rates:	**n = 240**	.147	.137	.150	.147	.148	.270	.266	.146	.134	.142	.148	.164
Omission = .05	**n = 480**	.156	.143	.150	.149	.145	.258	.278	.153	.134	.145	.149	.142
Intrusion = .20	**n = 960**	.164	.139	.151	.149	.148	.249	.299	.151	.136	.147	.153	.114
IUC = .40		**.120**	**.120**	**.120**	**.120**	**.120**	**.400**						$\hat{\pi}^*$
Rates:	**n = 240**	.155	.130	.117	.113	.109	.376	.388	.168	.110	.096	.064	.175
Omission = .20	**n = 480**	.144	.128	.117	.116	.115	.380	.413	.170	.109	.095	.068	.146
Intrusion = .05	**n = 960**	.130	.124	.117	.118	.118	.394	.429	.172	.108	.094	.073	.125
Rates:	**n = 240**	.165	.115	.120	.120	.119	.362	.334	.129	.108	.116	.119	.194
Omission = .05	**n = 480**	.174	.121	.119	.119	.119	.349	.353	.132	.108	.118	.122	.168
Intrusion = .20	**n = 960**	.174	.118	.121	.120	.117	.351	.369	.131	.108	.121	.125	.146

Table A32. Conditional Probability Recovery
Four-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Intrusion-Omission			
IUC = .10		**.200**	**.100**	**.150**	**.300**
Rates:	**n = 240**	.246	.180	.219	.356
Omission = .20	**n = 480**	.230	.149	.207	.383
Intrusion = .05	**n = 960**	.210	.139	.190	.370
Rates:	**n = 240**	.248	.162	.216	.351
Omission = .05	**n = 480**	.241	.152	.196	.369
Intrusion = .20	**n = 960**	.235	.123	.182	.368
IUC = .25		**.200**	**.100**	**.150**	**.300**
Rates:	**n = 240**	.194	.133	.186	.386
Omission = .20	**n = 480**	.191	.111	.160	.402
Intrusion = .05	**n = 960**	.179	.100	.161	.412
Rates:	**n = 240**	.219	.107	.154	.378
Omission = .05	**n = 480**	.207	.091	.144	.368
Intrusion = .20	**n = 960**	.206	.082	.132	.367
IUC = .40		**.200**	**.100**	**.150**	**.300**
Rates:	**n = 240**	.187	.106	.166	.386
Omission = .20	**n = 480**	.181	.100	.165	.391
Intrusion = .05	**n = 960**	.182	.092	.154	.370
Rates:	**n = 240**	.200	.097	.150	.406
Omission = .05	**n = 480**	.190	.086	.146	.397
Intrusion = .20	**n = 960**	.191	.080	.137	.372

Table A33. IUC and π^* Estimates
Four-Variable Extended Intrusion-Omission Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model IUC			Intrusion-Omission Model Pistar			$\pi_{IUC} - \pi^*$			$\left\| \pi_{IUC} - \pi^* \right\|$			Scaled Value of $\left\| \pi_{IUC} - \pi^* \right\|$			Goodman Model IUC		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10																		
Rates: Omission=.20 Intrusion=.05	.154	.148	.141	.142	.113	.083	.012	.034	.058	.083	.081	.080	.828	.814	.797	.552	.546	.543
Rates: Omission=.05 Intrusion=.20	.143	.123	.121	.120	.099	.077	.023	.023	.044	.067	.063	.067	.666	.632	.673	.403	.394	.390
IUC = .25																		
Rates: Omission=.20 Intrusion=.05	.251	.244	.254	.163	.129	.105	.088	.115	.149	.117	.136	.159	.468	.543	.637	.528	.527	.525
Rates: Omission=.05 Intrusion=.20	.243	.231	.229	.158	.134	.111	.085	.098	.117	.108	.113	.125	.430	.451	.499	.457	.454	.451
IUC = .40																		
Rates: Omission=.20 Intrusion=.05	.328	.328	.333	.187	.154	.130	.141	.174	.203	.161	.186	.207	.401	.465	.518	.579	.578	.578
Rates: Omission=.05 Intrusion=.20	.326	.314	.322	.194	.162	.144	.132	.152	.178	.147	.160	.180	.368	.401	.451	.474	.469	.467

Table A34. Error Rate Recovery
Four-Variable Extended Intrusion-Omission Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model Error Rate			Intrusion-Omission Pistar Model Error Rate			$\beta - \beta^*$			$\lvert\beta - \beta^*\rvert$			Scaled Value of $\lvert\beta - \beta^*\rvert$		
Sample Size	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**	**240**	**480**	**960**
IUC = .10															
Rates:															
Omission = .20	.185	.190	.196	.138	.152	.170	.047	.038	.025	.055	.044	.031	.275	.221	.154
Intrusion = .05	.052	.045	.045	.053	.058	.061	-.001	-.012	-.017	.034	.030	.026	.676	.609	.510
Rates:															
Omission = .05	.046	.048	.049	.025	.032	.036	.021	.017	.013	.024	.018	.014	.474	.357	.276
Intrusion = .20	.193	.194	.193	.156	.153	.167	.036	.041	.026	.054	.050	.036	.270	.252	.179
IUC = .25															
Rates:															
Omission = .20	.181	.191	.197	.129	.145	.162	.052	.045	.035	.061	.051	.039	.307	.255	.197
Intrusion = .05	.063	.056	.053	.073	.080	.087	-.010	-.024	-.034	.043	.043	.045	.869	.863	.909
Rates:															
Omission = .05	.046	.046	.048	.023	.029	.036	.023	.017	.012	.026	.020	.014	.514	.391	.283
Intrusion = .20	.181	.188	.186	.134	.143	.145	.047	.045	.042	.068	.059	.050	.338	.294	.252
IUC = .40															
Rates:															
Omission = .20	.185	.192	.196	.132	.150	.164	.052	.042	.033	.063	.049	.036	.317	.243	.182
Intrusion = .05	.074	.072	.067	.092	.101	.104	-.017	-.029	-.037	.061	.064	.060	1.221	1.280	1.203
Rates:															
Omission = .05	.044	.045	.047	.023	.032	.040	.021	.014	.007	.024	.018	.012	.486	.351	.245
Intrusion = .20	.209	.200	.207	.143	.128	.128	.066	.072	.079	.091	.090	.086	.457	.450	.430

Table A35. Mixing Proportion Recovery
Four-Variable Extended Intrusion-Omission Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Intrusion-Omission	Intrusion-Omission Pistar
IUC = .10		**.100 .100 .200 .250 .250 .100**	$\hat{\pi}^*$
Rates:	**n = 240**	.093 .093 .191 .237 .232 .154	.168 .121 .189 .212 .169 .142
Omission = .20	**n = 480**	.090 .091 .194 .238 .239 .148	.169 .121 .195 .218 .184 .113
Intrusion = .05	**n = 960**	.085 .089 .196 .243 .246 .141	.168 .118 .198 .229 .204 .083
Rates:	**n = 240**	.079 .092 .193 .247 .245 .143	.123 .099 .183 .235 .239 .120
Omission = .05	**n = 480**	.089 .092 .198 .249 .250 .123	.132 .102 .181 .238 .249 .099
Intrusion = .20	**n = 960**	.087 .094 .197 .249 .252 .121	.145 .103 .185 .241 .249 .077
IUC = .25		**.100 .100 .150 .200 .200 .250**	$\hat{\pi}^*$
Rates:	**n = 240**	.121 .108 .146 .190 .183 .251	.266 .138 .143 .168 .122 .163
Omission = .20	**n = 480**	.114 .106 .147 .195 .194 .244	.273 .139 .146 .177 .137 .129
Intrusion = .05	**n = 960**	.100 .102 .147 .197 .199 .254	.278 .139 .149 .184 .144 .105
Rates:	**n = 240**	.116 .097 .144 .197 .203 .243	.220 .107 .132 .187 .197 .158
Omission = .05	**n = 480**	.126 .097 .146 .197 .202 .231	.234 .110 .133 .189 .200 .134
Intrusion = .20	**n = 960**	.128 .095 .147 .197 .204 .229	.249 .109 .134 .191 .206 .111
IUC = .40		**.050 .050 .100 .200 .200 .400**	$\hat{\pi}^*$
Rates:	**n = 240**	.109 .078 .104 .202 .180 .328	.312 .102 .102 .179 .118 .187
Omission = .20	**n = 480**	.108 .074 .104 .200 .186 .328	.324 .100 .104 .183 .135 .154
Intrusion = .05	**n = 960**	.098 .072 .103 .202 .191 .333	.329 .100 .104 .191 .146 .130
Rates:	**n = 240**	.120 .068 .099 .201 .187 .326	.262 .080 .089 .185 .190 .194
Omission = .05	**n = 480**	.134 .061 .099 .197 .193 .314	.288 .076 .088 .184 .202 .162
Intrusion = .20	**n = 960**	.128 .057 .100 .199 .194 .322	.295 .076 .086 .188 .210 .144

Table A36. Conditional Probability Recovery
Four-Variable Extended Intrusion-Omission Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Intrusion-Omission			
IUC = .10		**.200**	**.100**	**.150**	**.300**
Rates:	**n = 240**	.242	.178	.215	.351
Omission = .20	**n = 480**	.238	.158	.211	.372
Intrusion = .05	**n = 960**	.217	.129	.185	.372
Rates:	**n = 240**	.242	.167	.201	.371
Omission = .05	**n = 480**	.239	.127	.164	.400
Intrusion = .20	**n = 960**	.218	.124	.157	.403
IUC = .25		**.200**	**.100**	**.150**	**.300**
Rates:	**n = 240**	.205	.142	.190	.391
Omission = .20	**n = 480**	.188	.115	.174	.405
Intrusion = .05	**n = 960**	.179	.097	.157	.378
Rates:	**n = 240**	.206	.105	.168	.402
Omission = .05	**n = 480**	.211	.088	.164	.404
Intrusion = .20	**n = 960**	.206	.081	.156	.402
IUC = .40		**.200**	**.100**	**.150**	**.300**
Rates:	**n = 240**	.171	.111	.180	.408
Omission = .20	**n = 480**	.166	.096	.166	.405
Intrusion = .05	**n = 960**	.160	.091	.161	.387
Rates:	**n = 240**	.180	.083	.142	.377
Omission = .05	**n = 480**	.187	.071	.135	.379
Intrusion = .20	**n = 960**	.190	.070	.133	.351

Table A37. IUC and π^* Estimates
Six-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Equal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model IUC			Intrusion-Omission Model Pistar			$\pi_{IUC} - \pi^*$			$\lvert \pi_{IUC} - \pi^* \rvert$			Scaled Value of $\lvert \pi_{IUC} - \pi^* \rvert$			Goodman Model IUC		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10																		
Rates: Omission=.20 Intrusion=.05	.125	.116	.111	.244	.207	.160	-.119	-.091	-.049	.123	.093	.053	1.229	.930	.528	.562	.561	.560
Rates: Omission=.05 Intrusion=.20	.136	.131	.134	.240	.193	.154	-.104	-.062	-.019	.126	.107	.080	1.256	1.075	.801	.575	.572	.572
IUC = .25																		
Rates: Omission=.20 Intrusion=.05	.256	.253	.258	.300	.256	.215	-.044	-.003	.043	.066	.043	.049	.266	.174	.194	.616	.615	.615
Rates: Omission=.05 Intrusion=.20	.238	.241	.248	.272	.221	.172	-.034	.020	.076	.103	.090	.095	.412	.358	.378	.629	.627	.626
IUC = .40																		
Rates: Omission=.20 Intrusion=.05	.396	.393	.397	.350	.303	.269	.046	.090	.128	.069	.093	.128	.172	.231	.320	.683	.683	.683
Rates: Omission=.05 Intrusion=.20	.351	.376	.388	.302	.244	.192	.049	.132	.196	.131	.156	.199	.328	.389	.498	.696	.696	.695

Table A38. Error Rate Recovery
Six-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Equal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model Error Rate			Intrusion-Omission Pistar Model Error Rate			$\beta - \beta^*$			$\lvert \beta - \beta^* \rvert$			Scaled Value of $\lvert \beta - \beta^* \rvert$		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10															
Rates:															
Omission = .20	.201	.201	.202	.153	.165	.180	.048	.036	.022	.050	.038	.024	.249	.189	.120
Intrusion = .05	.045	.047	.047	.034	.040	.046	.011	.006	.001	.017	.012	.009	.337	.243	.175
Rates:															
Omission = .05	.049	.049	.050	.026	.028	.032	.022	.022	.018	.023	.022	.018	.456	.437	.355
Intrusion = .20	.197	.199	.199	.144	.161	.175	.053	.038	.024	.053	.038	.025	.267	.191	.123
IUC = .25															
Rates:															
Omission = .20	.199	.200	.200	.174	.187	.204	.026	.013	-.005	.038	.027	.018	.189	.133	.088
Intrusion = .05	.048	.049	.048	.044	.052	.055	.004	-.003	-.007	.018	.015	.012	.355	.306	.247
Rates:															
Omission = .05	.049	.050	.050	.026	.029	.034	.023	.021	.016	.023	.021	.016	.468	.414	.327
Intrusion = .20	.201	.202	.202	.165	.180	.199	.036	.021	.004	.039	.027	.014	.194	.134	.072
IUC = .40															
Rates:															
Omission = .20	.202	.203	.203	.209	.222	.239	-.007	-.020	-.036	.040	.035	.040	.201	.177	.198
Intrusion = .05	.046	.047	.046	.057	.067	.069	-.012	-.021	-.023	.024	.025	.026	.475	.504	.513
Rates:															
Omission = .05	.048	.048	.049	.022	.026	.031	.025	.022	.018	.026	.022	.018	.518	.446	.358
Intrusion = .20	.202	.202	.202	.181	.201	.221	.021	.001	-.019	.033	.024	.024	.164	.118	.121

Table A39. Mixing Proportion Recovery
Six-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Equal Conditional Probabilities

Model		Extended Intrusion-Omission								Intrusion-Omission Pistar							
IUC = .10		**.129**	**.129**	**.129**	**.129**	**.129**	**.129**	**.129**	**.100**								$\hat{\pi}^*$
Rates:	**n = 960**	.125	.120	.125	.126	.126	.126	.127	.125	.142	.126	.120	.113	.105	.088	.062	.244
Omission = .20	**n = 1920**	.126	.122	.127	.127	.127	.127	.128	.116	.146	.129	.124	.117	.111	.103	.063	.207
Intrusion = .05	**n = 3840**	.126	.122	.127	.127	.128	.128	.129	.111	.148	.132	.128	.122	.118	.112	.080	.160
Rates:	**n = 960**	.117	.117	.122	.128	.125	.126	.130	.136	.105	.093	.100	.110	.113	.116	.123	.240
Omission = .05	**n = 1920**	.113	.119	.125	.128	.127	.127	.129	.131	.125	.106	.109	.114	.115	.118	.120	.193
Intrusion = .20	**n = 3840**	.106	.120	.126	.129	.128	.127	.130	.134	.141	.113	.115	.119	.119	.119	.121	.154
IUC = .25		**.107**	**.107**	**.107**	**.107**	**.107**	**.107**	**.107**	**.250**								$\hat{\pi}^*$
Rates:	**n = 960**	.106	.109	.103	.104	.107	.106	.109	.256	.145	.123	.107	.101	.095	.079	.051	.300
Omission = .20	**n = 1920**	.107	.110	.104	.105	.106	.107	.109	.253	.150	.127	.111	.105	.101	.091	.059	.256
Intrusion = .05	**n = 3840**	.105	.108	.103	.105	.106	.107	.109	.258	.148	.128	.113	.111	.108	.098	.077	.215
Rates:	**n = 960**	.125	.108	.105	.109	.103	.105	.107	.238	.169	.089	.086	.096	.093	.097	.098	.272
Omission = .05	**n = 1920**	.112	.112	.107	.109	.103	.108	.109	.241	.191	.101	.094	.099	.095	.100	.099	.221
Intrusion = .20	**n = 3840**	.104	.111	.107	.110	.104	.107	.109	.248	.218	.110	.098	.103	.098	.100	.099	.172
IUC = .40		**.086**	**.086**	**.086**	**.086**	**.086**	**.086**	**.086**	**.400**								$\hat{\pi}^*$
Rates:	**n = 960**	.086	.083	.087	.087	.085	.088	.089	.396	.154	.112	.103	.095	.082	.067	.038	.350
Omission = .20	**n = 1920**	.087	.084	.087	.088	.085	.088	.088	.393	.162	.117	.105	.100	.086	.080	.047	.303
Intrusion = .05	**n = 3840**	.085	.082	.088	.087	.085	.088	.088	.397	.159	.117	.110	.103	.092	.085	.066	.269
Rates:	**n = 960**	.126	.098	.086	.085	.085	.085	.084	.351	.236	.087	.071	.075	.076	.078	.074	.302
Omission = .05	**n = 1920**	.103	.094	.085	.086	.085	.086	.084	.376	.275	.095	.078	.077	.079	.080	.073	.244
Intrusion = .20	**n = 3840**	.093	.091	.085	.085	.086	.086	.085	.388	.318	.101	.078	.076	.081	.081	.073	.192

Table A40. Conditional Probability Recovery
Six-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Equal Conditional Probabilities

Model		Extended Intrusion-Omission					
IUC = .10		**.300**	**.300**	**.300**	**.300**	**.300**	**.300**
Rates:	**n = 960**	.296	.294	.323	.313	.321	.330
Omission = .20	**n = 1920**	.292	.286	.308	.288	.300	.299
Intrusion = .05	**n = 3840**	.301	.278	.295	.283	.289	.290
Rates:	**n = 960**	.340	.361	.369	.371	.372	.349
Omission = .05	**n = 1920**	.303	.336	.345	.345	.347	.322
Intrusion = .20	**n = 3840**	.293	.319	.334	.336	.318	.303
IUC = .25		**.300**	**.300**	**.300**	**.300**	**.300**	**.300**
Rates:	**n = 960**	.279	.296	.302	.304	.312	.310
Omission = .20	**n = 1920**	.284	.299	.299	.302	.307	.304
Intrusion = .05	**n = 3840**	.296	.298	.298	.298	.304	.300
Rates:	**n = 960**	.275	.349	.333	.358	.350	.343
Omission = .05	**n = 1920**	.253	.324	.309	.327	.328	.331
Intrusion = .20	**n = 3840**	.265	.315	.298	.315	.312	.312
IUC = .40		**.300**	**.300**	**.300**	**.300**	**.300**	**.300**
Rates:	**n = 960**	.289	.294	.297	.303	.300	.305
Omission = .20	**n = 1920**	.292	.295	.298	.304	.299	.303
Intrusion = .05	**n = 3840**	.296	.294	.297	.301	.299	.302
Rates:	**n = 960**	.267	.336	.333	.336	.338	.329
Omission = .05	**n = 1920**	.273	.315	.310	.317	.318	.309
Intrusion = .20	**n = 3840**	.284	.307	.302	.307	.308	.303

Table A41. IUC and π^* Estimates
Six-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model IUC			Intrusion-Omission Model Pistar			$\pi_{IUC} - \pi^*$			$\lvert \pi_{IUC} - \pi^* \rvert$			Scaled Value of $\lvert \pi_{IUC} - \pi^* \rvert$			Goodman Model IUC		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10																		
Rates: Omission=.20 Intrusion=.05	.117	.106	.105	.250	.220	.173	-.133	-.114	-.068	.133	.114	.068	1.333	1.144	.684	.547	.545	.545
Rates: Omission=.05 Intrusion=.20	.136	.127	.113	.248	.204	.175	-.113	-.077	-.062	.126	.101	.088	1.264	1.008	.877	.552	.550	.548
IUC = .25																		
Rates: Omission=.20 Intrusion=.05	.259	.256	.256	.318	.286	.245	-.059	-.030	.011	.063	.039	.025	.254	.156	.101	.591	.590	.590
Rates: Omission=.05 Intrusion=.20	.273	.272	.271	.307	.267	.234	-.034	.005	.037	.085	.069	.062	.342	.278	.249	.599	.597	.596
IUC = .40																		
Rates: Omission=.20 Intrusion=.05	.399	.404	.403	.387	.357	.321	.012	.047	.082	.044	.052	.082	.109	.130	.205	.650	.648	.649
Rates: Omission=.05 Intrusion=.20	.401	.404	.409	.367	.334	.312	.034	.070	.098	.081	.083	.100	.204	.208	.250	.662	.660	.661

Table A42. Error Rate Recovery
Six-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model Error Rate			Intrusion-Omission Pistar Model Error Rate			$\beta - \beta^*$			$\lvert \beta - \beta^* \rvert$			Scaled Value of $\lvert \beta - \beta^* \rvert$		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10															
Rates:															
Omission = .20	.199	.200	.199	.139	.152	.167	.061	.047	.033	.061	.047	.033	.304	.237	.163
Intrusion = .05	.049	.051	.051	.036	.038	.044	.013	.013	.006	.016	.015	.010	.316	.297	.198
Rates:															
Omission = .05	.051	.050	.050	.023	.023	.026	.027	.027	.023	.027	.027	.023	.547	.547	.466
Intrusion = .20	.199	.198	.200	.140	.156	.166	.059	.042	.034	.059	.042	.034	.294	.210	.170
IUC = .25															
Rates:															
Omission = .20	.203	.202	.203	.149	.161	.178	.053	.041	.025	.055	.042	.025	.274	.211	.127
Intrusion = .05	.048	.049	.049	.039	.044	.049	.009	.004	-.0001	.015	.013	.010	.299	.256	.200
Rates:															
Omission = .05	.054	.052	.052	.023	.024	.028	.031	.029	.024	.031	.029	.024	.617	.571	.484
Intrusion = .20	.192	.194	.193	.137	.153	.167	.056	.041	.027	.056	.041	.027	.281	.207	.136
IUC = .40															
Rates:															
Omission = .20	.201	.199	.199	.156	.158	.177	.045	.041	.022	.050	.044	.025	.252	.218	.127
Intrusion = .05	.049	.048	.048	.045	.052	.057	.004	-.004	-.009	.018	.017	.016	.354	.345	.319
Rates:															
Omission = .05	.050	.049	.050	.017	.021	.025	.033	.028	.024	.033	.028	.024	.653	.564	.489
Intrusion = .20	.198	.198	.199	.137	.150	.170	.060	.048	.029	.061	.048	.030	.303	.239	.148

Table A43. Mixing Proportion Recovery
Six-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Intrusion-Omission								Intrusion-Omission Pistar							
IUC = .10		**.129**	**.129**	**.129**	**.129**	**.129**	**.129**	**.129**	**.100**								$\hat{\pi}^*$
Rates:	**n = 960**	.123	.123	.125	.126	.128	.127	.130	.117	.153	.127	.115	.107	.102	.082	.065	.250
Omission = .20	**n = 1920**	.127	.126	.127	.128	.129	.128	.129	.106	.152	.128	.119	.110	.108	.096	.067	.220
Intrusion = .05	**n = 3840**	.127	.127	.128	.128	.129	.128	.129	.105	.155	.131	.122	.116	.114	.107	.082	.173
Rates:	**n = 960**	.106	.121	.127	.127	.125	.130	.128	.136	.109	.096	.096	.106	.109	.118	.118	.248
Omission = .05	**n = 1920**	.107	.123	.130	.128	.125	.131	.129	.127	.130	.107	.105	.110	.111	.118	.115	.204
Intrusion = .20	**n = 3840**	.118	.127	.130	.128	.126	.131	.128	.113	.139	.114	.110	.113	.114	.120	.116	.175
IUC = .25		**.107**	**.107**	**.107**	**.107**	**.107**	**.107**	**.107**	**.250**								$\hat{\pi}^*$
Rates:	**n = 960**	.098	.106	.105	.106	.110	.106	.110	.259	.166	.118	.098	.092	.088	.068	.052	.318
Omission = .20	**n = 1920**	.099	.107	.106	.107	.110	.106	.109	.256	.170	.121	.100	.096	.094	.081	.053	.286
Intrusion = .05	**n = 3840**	.098	.107	.107	.107	.110	.106	.109	.256	.172	.122	.104	.100	.100	.088	.069	.245
Rates:	**n = 960**	.095	.094	.105	.109	.108	.106	.111	.273	.155	.085	.077	.088	.094	.095	.099	.307
Omission = .05	**n = 1920**	.092	.097	.106	.109	.108	.106	.111	.272	.174	.094	.084	.092	.096	.095	.098	.267
Intrusion = .20	**n = 3840**	.093	.096	.106	.109	.108	.106	.110	.271	.185	.102	.088	.096	.099	.097	.098	.234
IUC = .40		**.086**	**.086**	**.086**	**.086**	**.086**	**.086**	**.086**	**.400**								$\hat{\pi}^*$
Rates:	**n = 960**	.087	.087	.084	.084	.084	.087	.089	.399	.189	.109	.080	.072	.069	.053	.041	.387
Omission = .20	**n = 1920**	.085	.085	.085	.084	.083	.086	.087	.404	.199	.112	.079	.075	.072	.063	.043	.357
Intrusion = .05	**n = 3840**	.086	.085	.085	.084	.083	.086	.088	.403	.202	.114	.081	.079	.077	.071	.055	.321
Rates:	**n = 960**	.089	.085	.085	.085	.087	.084	.085	.401	.199	.080	.061	.067	.074	.075	.076	.367
Omission = .05	**n = 1920**	.082	.088	.086	.085	.086	.084	.084	.404	.213	.088	.065	.071	.076	.076	.076	.334
Intrusion = .20	**n = 3840**	.078	.086	.087	.085	.087	.084	.084	.409	.215	.097	.070	.075	.078	.077	.076	.312

Table A44. Conditional Probability Recovery
Six-Variable Extended Intrusion-Omission Model
Equal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Intrusion-Omission					
IUC = .10		**.200**	**.100**	**.150**	**.300**	**.250**	**.400**
Rates:	**n = 960**	.209	.124	.171	.320	.265	.416
Omission = .20	**n = 1920**	.193	.098	.151	.308	.258	.402
Intrusion = .05	**n = 3840**	.195	.093	.151	.301	.252	.391
Rates:	**n = 960**	.233	.153	.175	.327	.280	.489
Omission = .05	**n = 1920**	.205	.118	.149	.311	.251	.457
Intrusion = .20	**n = 3840**	.196	.108	.144	.307	.249	.448
IUC = .25		**.200**	**.100**	**.150**	**.300**	**.250**	**.400**
Rates:	**n = 960**	.186	.097	.149	.298	.255	.396
Omission = .20	**n = 1920**	.187	.095	.149	.297	.253	.390
Intrusion = .05	**n = 3840**	.188	.095	.147	.295	.251	.389
Rates:	**n = 960**	.217	.100	.152	.309	.264	.416
Omission = .05	**n = 1920**	.214	.094	.150	.302	.259	.403
Intrusion = .20	**n = 3840**	.224	.096	.154	.299	.260	.398
IUC = .40		**.200**	**.100**	**.150**	**.300**	**.250**	**.400**
Rates:	**n = 960**	.193	.101	.153	.299	.250	.409
Omission = .20	**n = 1920**	.197	.102	.155	.298	.250	.404
Intrusion = .05	**n = 3840**	.198	.102	.155	.297	.249	.403
Rates:	**n = 960**	.195	.098	.147	.308	.256	.416
Omission = .05	**n = 1920**	.193	.099	.147	.304	.253	.408
Intrusion = .20	**n = 3840**	.196	.100	.148	.303	.253	.402

Table A45. IUC and π^* Estimates
Six-Variable Extended Intrusion-Omission Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model IUC			Intrusion-Omission Model Pistar			$\pi_{IUC} - \pi^*$			$\left\| \pi_{IUC} - \pi^* \right\|$			Scaled Value of $\left\| \pi_{IUC} - \pi^* \right\|$			Goodman Model IUC		
Sample Size	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**	**960**	**1920**	**3840**
IUC = .10																		
Rates: Omission=.20 Intrusion=.05	.108	.104	.103	.285	.232	.184	-.176	-.127	-.080	.176	.127	.080	1.763	1.274	.804	.639	.637	.637
Rates: Omission=.05 Intrusion=.20	.108	.096	.091	.207	.189	.166	-.099	-.093	-.075	.099	.094	.077	.995	.936	.766	.463	.460	.458
IUC = .25																		
Rates: Omission=.20 Intrusion=.05	.254	.255	.255	.351	.297	.252	-.098	-.041	.002	.098	.046	.023	.393	.184	.093	.646	.646	.646
Rates: Omission=.05 Intrusion=.20	.244	.240	.245	.283	.256	.231	-.039	-.016	.014	.061	.047	.040	.242	.187	.158	.495	.493	.493
IUC = .40																		
Rates: Omission=.20 Intrusion=.05	.395	.393	.395	.394	.362	.326	.001	.031	.070	.037	.041	.070	.093	.101	.175	.673	.672	.672
Rates: Omission=.05 Intrusion=.20	.375	.377	.378	.358	.327	.299	.017	.050	.079	.068	.068	.081	.171	.170	.203	.630	.627	.627

Table A46. Error Rate Recovery
Six-Variable Extended Intrusion-Omission Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Estimated Mean	Extended Intrusion-Omission Model Error Rate	Intrusion-Omission Pistar Model Error Rate	$\beta - \beta^*$	$\lvert \beta - \beta^* \rvert$	Scaled Value of $\lvert \beta - \beta^* \rvert$
Sample Size	**960 1920 3840**	**960 1920 3840**	**960 1920 3840**	**960 1920 3840**	**960 1920 3840**
IUC = .10					
Rates:					
Omission = .20	.198 .198 .198	.139 .158 .170	.059 .040 .028	.059 .040 .028	.294 .201 .140
Intrusion = .05	.051 .052 .052	.036 .037 .044	.016 .015 .008	.021 .019 .012	.411 .377 .248
Rates:					
Omission = .05	.051 .050 .050	.025 .029 .028	.026 .021 .022	.026 .021 .022	.511 .428 .444
Intrusion = .20	.198 .200 .200	.151 .156 .169	.047 .044 .031	.048 .045 .032	.242 .225 .161
IUC = .25					
Rates:					
Omission = .20	.197 .197 .197	.134 .154 .170	.063 .043 .028	.063 .043 .028	.316 .217 .139
Intrusion = .05	.055 .055 .056	.055 .062 .066	-.001 -.007 -.011	.021 .021 .018	.429 .426 .362
Rates:					
Omission = .05	.052 .051 .051	.025 .028 .030	.027 .023 .020	.027 .023 .020	.536 .458 .407
Intrusion = .20	.200 .201 .199	.138 .143 .156	.062 .057 .043	.064 .058 .044	.322 .291 .218
IUC = .40					
Rates:					
Omission = .20	.196 .196 .195	.153 .162 .178	.043 .034 .017	.046 .037 .020	.232 .183 .101
Intrusion = .05	.052 .053 .053	.046 .054 .056	.006 -.0001 -.003	.017 .015 .012	.349 .296 .233
Rates:					
Omission = .05	.051 .051 .050	.019 .022 .026	.032 .029 .024	.032 .029 .024	.638 .576 .489
Intrusion = .20	.202 .202 .202	.139 .153 .172	.062 .049 .030	.063 .049 .031	.315 .245 .154

Table A47. Mixing Proportion Recovery
Six-Variable Extended Intrusion-Omission Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Intrusion-Omission								Intrusion-Omission Pistar							
IUC = .10		**.050**	**.050**	**.100**	**.100**	**.200**	**.200**	**.200**	**.100**								$\hat{\pi}^*$
Rates:	**n = 960**	.048	.051	.102	.099	.197	.198	.196	.108	.074	.057	.093	.090	.157	.144	.100	.285
Omission = .20	**n = 1920**	.048	.050	.103	.101	.198	.199	.197	.104	.073	.056	.094	.093	.169	.161	.122	.232
Intrusion = .05	**n = 3840**	.048	.050	.103	.101	.198	.199	.197	.103	.075	.057	.098	.096	.178	.171	.140	.184
Rates:	**n = 960**	.048	.049	.099	.097	.198	.198	.203	.108	.062	.037	.073	.083	.172	.181	.186	.207
Omission = .05	**n = 1920**	.053	.052	.099	.097	.201	.199	.203	.096	.058	.041	.080	.085	.176	.183	.189	.189
Intrusion = .20	**n = 3840**	.055	.053	.100	.098	.200	.200	.203	.091	.066	.049	.085	.088	.178	.183	.184	.166
IUC = .25		**.050**	**.050**	**.050**	**.100**	**.100**	**.200**	**.200**	**.250**								$\hat{\pi}^*$
Rates:	**n = 960**	.055	.049	.049	.100	.101	.195	.198	.254	.126	.065	.047	.086	.085	.141	.099	.351
Omission = .20	**n = 1920**	.052	.049	.049	.101	.101	.195	.199	.255	.129	.068	.047	.092	.090	.157	.121	.297
Intrusion = .05	**n = 3840**	.052	.049	.050	.100	.100	.195	.198	.255	.130	.069	.049	.096	.095	.168	.141	.252
Rates:	**n = 960**	.056	.048	.049	.099	.100	.204	.200	.244	.111	.045	.034	.078	.087	.178	.184	.283
Omission = .05	**n = 1920**	.058	.048	.051	.099	.100	.204	.200	.240	.123	.047	.037	.080	.089	.180	.187	.256
Intrusion = .20	**n = 3840**	.053	.047	.051	.099	.101	.204	.199	.245	.131	.053	.040	.083	.093	.183	.186	.231
IUC = .40		**.050**	**.050**	**.100**	**.100**	**.100**	**.100**	**.100**	**.400**								$\hat{\pi}^*$
Rates:	**n = 960**	.057	.053	.097	.100	.099	.100	.098	.395	.153	.078	.092	.087	.083	.066	.046	.394
Omission = .20	**n = 1920**	.056	.054	.099	.101	.100	.100	.096	.393	.160	.080	.095	.092	.088	.076	.047	.362
Intrusion = .05	**n = 3840**	.055	.054	.099	.100	.100	.100	.096	.395	.160	.080	.099	.095	.093	.084	.063	.326
Rates:	**n = 960**	.079	.051	.097	.100	.101	.099	.099	.375	.175	.056	.068	.079	.086	.088	.089	.358
Omission = .05	**n = 1920**	.076	.050	.098	.101	.101	.099	.098	.377	.189	.060	.073	.085	.088	.089	.089	.327
Intrusion = .20	**n = 3840**	.075	.051	.098	.101	.101	.099	.098	.378	.197	.067	.079	.088	.091	.090	.089	.299

Table A48. Conditional Probability Recovery
Six-Variable Extended Intrusion-Omission Model
Unequal Mixing Proportions, Unequal Conditional Probabilities

Model		Extended Intrusion-Omission					
IUC = .10		**.200**	**.100**	**.150**	**.300**	**.250**	**.400**
Rates:	**n = 960**	.192	.122	.166	.314	.260	.415
Omission = .20	**n = 1920**	.189	.107	.156	.309	.251	.397
Intrusion = .05	**n = 3840**	.191	.108	.155	.302	.246	.391
Rates:	**n = 960**	.204	.127	.174	.325	.278	.478
Omission = .05	**n = 1920**	.186	.091	.147	.313	.253	.466
Intrusion = .20	**n = 3840**	.176	.081	.142	.311	.250	.444
IUC = .25		**.200**	**.100**	**.150**	**.300**	**.250**	**.400**
Rates:	**n = 960**	.200	.104	.155	.306	.256	.408
Omission = .20	**n = 1920**	.201	.101	.155	.301	.251	.399
Intrusion = .05	**n = 3840**	.202	.102	.156	.302	.252	.400
Rates:	**n = 960**	.197	.095	.149	.309	.253	.419
Omission = .05	**n = 1920**	.198	.087	.145	.308	.250	.414
Intrusion = .20	**n = 3840**	.203	.091	.147	.304	.250	.408
IUC = .40		**.200**	**.100**	**.150**	**.300**	**.250**	**.400**
Rates:	**n = 960**	.204	.107	.152	.306	.256	.406
Omission = .20	**n = 1920**	.203	.104	.151	.305	.255	.404
Intrusion = .05	**n = 3840**	.204	.105	.152	.304	.255	.402
Rates:	**n = 960**	.199	.093	.145	.311	.259	.418
Omission = .05	**n = 1920**	.200	.090	.145	.310	.256	.412
Intrusion = .20	**n = 3840**	.201	.092	.146	.308	.255	.408

APPENDIX B: SAS PROGRAMS

```
*SAS program for extended Proctor and Proctor pistar models ;
libname pr_univ 'c:\nlp programs\proctor4\g20_iuc40_e_e' ;
filename pr_log 'c:\nlp programs\proctor4\proclog.txt' ;
filename pr_out 'c:\nlp programs\proctor4\procout.txt' ;
proc printto log = pr_log  print = pr_out ;
run ;

*response patterns used as input into PROC NLP ;
data resppat ;
input vara varb varc vard pattern ;
cards ;
1  1  1  1  1
1  1  1  0  2
1  1  0  1  6
1  1  0  0  3
1  0  1  1  6
1  0  1  0  6
1  0  0  1  6
1  0  0  0  4
0  1  1  1  6
0  1  1  0  6
0  1  0  1  6
0  1  0  0  6
0  0  1  1  6
0  0  1  0  6
0  0  0  1  6
0  0  0  0  5
 ;
run ;

*universe creation ;
data pr_univ.proctordata ;
pi_error = 0.20 ;
  do j = 1 to 100000 ;
    latnum = ranuni(0) ;
      if 0 lt latnum le 0.12 then do ;
        latclass = 1 ;
        varanum = ranuni(0) ;
        varbnum = ranuni(0) ;
        varcnum = ranuni(0) ;
        vardnum = ranuni(0) ;
          if 0 lt varanum le pi_error then vara = 1 ;
            else if pi_error lt varanum le 1.00000 then vara = 0 ;
          if 0 lt varbnum le pi_error then varb = 1 ;
```

```
    else if pi_error lt varbnum le 1.00000 then varb = 0 ;
   if 0 lt varcnum le pi_error then varc = 1 ;
    else if pi_error lt varcnum le 1.00000 then varc = 0 ;
   if 0 lt vardnum le pi_error then vard = 1 ;
    else if pi_error lt vardnum le 1.00000 then vard = 0 ;
end ;

else if 0.12 lt latnum le 0.24 then do ;
  latclass = 2 ;
  varanum = ranuni(0) ;
  varbnum = ranuni(0) ;
  varcnum = ranuni(0) ;
  vardnum = ranuni(0) ;
   if 0 lt varanum le (1 - pi_error) then vara = 1 ;
    else if (1 - pi_error) lt varanum le 1.00000 then vara = 0 ;
   if 0 lt varbnum le pi_error then varb = 1 ;
    else if pi_error lt varbnum le 1.00000 then varb = 0 ;
   if 0 lt varcnum le pi_error then varc = 1 ;
    else if pi_error lt varcnum le 1.00000 then varc = 0 ;
   if 0 lt vardnum le pi_error then vard = 1 ;
    else if pi_error lt vardnum le 1.00000 then vard = 0 ;
end ;

else if 0.24 lt latnum le 0.36 then do ;
  latclass = 3 ;
  varanum = ranuni(0) ;
  varbnum = ranuni(0) ;
  varcnum = ranuni(0) ;
  vardnum = ranuni(0) ;
   if 0 lt varanum le (1 - pi_error) then vara = 1 ;
    else if (1 - pi_error) lt varanum le 1.00000 then vara = 0 ;
   if 0 lt varbnum le (1 - pi_error)  then varb = 1 ;
    else if (1 - pi_error) lt varbnum le 1.00000 then varb = 0 ;
   if 0 lt varcnum le pi_error then varc = 1 ;
    else if pi_error lt varcnum le 1.00000 then varc = 0 ;
   if 0 lt vardnum le pi_error then vard = 1 ;
    else if pi_error lt vardnum le 1.00000 then vard = 0 ;
end ;

else if 0.36 lt latnum le 0.48 then do ;
  latclass = 4 ;
  varanum = ranuni(0) ;
  varbnum = ranuni(0) ;
  varcnum = ranuni(0) ;
  vardnum = ranuni(0) ;
   if 0 lt varanum le (1 - pi_error) then vara = 1 ;
```

```
        else if (1 - pi_error) lt varanum le 1.00000 then vara = 0 ;
      if 0 lt varbnum le (1 - pi_error) then varb = 1 ;
        else if (1 - pi_error) lt varbnum le 1.00000 then varb = 0 ;
      if 0 lt varcnum le (1 - pi_error) then varc = 1 ;
        else if (1 - pi_error) lt varcnum le 1.00000 then varc = 0 ;
      if 0 lt vardnum le pi_error then vard = 1 ;
        else if pi_error lt vardnum le 1.00000 then vard = 0 ;
    end ;

    else if 0.48 lt latnum le 0.60 then do ;
     latclass = 5 ;
     varanum = ranuni(0) ;
     varbnum = ranuni(0) ;
     varcnum = ranuni(0) ;
     vardnum = ranuni(0) ;
      if 0 lt varanum le (1 - pi_error) then vara = 1 ;
        else if (1 - pi_error) lt varanum le 1.00000 then vara = 0 ;
      if 0 lt varbnum le (1 - pi_error) then varb = 1 ;
        else if (1 - pi_error) lt varbnum le 1.00000 then varb = 0 ;
      if 0 lt varcnum le (1 - pi_error) then varc = 1 ;
        else if (1 - pi_error) lt varcnum le 1.00000 then varc = 0 ;
      if 0 lt vardnum le (1 - pi_error) then vard = 1 ;
        else if (1 - pi_error) lt vardnum le 1.00000 then vard = 0 ;
    end ;

    else if 0.60 lt latnum le 1.0000 then do ;
     latclass = 6 ;
     varanum = ranuni(0) ;
     varbnum = ranuni(0) ;
     varcnum = ranuni(0) ;
     vardnum = ranuni(0) ;

     if varanum le 0.30 then vara = 1 ;
      else if varanum gt 0.30 then vara = 0 ;
     if varbnum le 0.30 then varb = 1 ;
      else if varbnum gt 0.30 then varb = 0 ;
     if varcnum le 0.30 then varc = 1 ;
      else if varcnum gt 0.30 then varc = 0 ;
     if vardnum le 0.30 then vard = 1 ;
      else if vardnum gt 0.30 then vard = 0 ;
    end ;
   output ;
  end ;
run ;

*options symbolgen mlogic mprint ;
```

```
*macro for sampling, estimation, and summarization of replicate statistics ;
%macro proctorsim ;
%do m = 1 %to 3 ;
%let l = %eval(120 * 2 ** &m) ;
%do i = 1 %to 1000 ;

proc surveyselect data = pr_univ.proctordata method = urs n = &l
                        out = proctorsamp outhits noprint ;
run ;

proc freq data = proctorsamp noprint ;
  tables vara * varb * varc * vard /sparse out = proctdata (drop = percent) ;
run ;

proc sort data = proctdata ;
  by vara varb varc vard ;
run ;

data transfreq ;
array obsfreq{16} d0000 d0001 d0010 d0011 d0100 d0101 d0110 d0111
                  d1000 d1001 d1010 d1011 d1100 d1101 d1110 d1111 ;
do m = 1 to 16 ;
  set proctdata ;
    if count > 0 then obsfreq{m} = count ;
      else if count = 0 then obsfreq{m} = 1 ;
      *in case of sampling zeros, Pan (2006) suggests using 1 as a flattening constant  ;
end ;
drop vara varb varc vard m count ;
run ;

data _null_ ;
set proctdata ;
if count > 0 then countr = count ;
  else countr = 1 ;
convara = left(put(vara,1.)) ;
convarb = left(put(varb,1.)) ;
convarc = left(put(varc,1.)) ;
convard = left(put(vard,1.)) ;
call symput('c'||convara||convarb||convarc||convard, trim(left(put(countr,3.)))) ;
run ;

data procabridge ;
set proctdata ;
if count > 0 then countr = count ;
  else countr = 1 ;
run ;
```

```
*extended model ;
proc nlp data = procabridge tech = dbldog outest = proctor_est
(drop = _iter_ _name_ _rhs_ _tech_) noprint ;

max loglik ;

parms gamma = 0.20 ,
          theta1 = 0.12 ,
          theta2 = 0.12 ,
          theta3 = 0.12 ,
          theta4 = 0.12 ,
          theta5 = 0.12 ,
          theta6 = 0.40 ,
          alpha = 0.30 ,
          beta = 0.30 ,
          delta = 0.30 ,
          sigma = 0.30 ;

bounds 0 < theta1 - theta6 < 1 ,
           0 < gamma < 1 ,
           0 < alpha < 1 ,
           0 < beta < 1 ,
           0 < delta < 1 ,
           0 < sigma < 1 ;

lincon theta1 + theta2 + theta3 + theta4 + theta5 + theta6 = 1 ;

loglik = countr *(log(theta6 * alpha ** vara * (1 - alpha) ** (1 - vara)
                                    * beta ** varb * (1 - beta) ** (1 - varb)
                                    * delta ** varc * (1 - delta) ** (1 - varc)
                                    * sigma ** vard * (1 - sigma) ** (1 - vard)

                        + theta1 * gamma ** vara * (1 - gamma) ** (1 - vara)
                                    * gamma ** varb * (1 - gamma) ** (1 - varb)
                                    * gamma ** varc * (1 - gamma) ** (1 - varc)
                                    * gamma ** vard * (1 - gamma) ** (1 - vard)

                        + theta2 * (1 - gamma) ** vara * gamma ** (1 - vara)
                                    * gamma ** varb * (1 - gamma) ** (1 - varb)
                                    * gamma ** varc * (1 - gamma) ** (1 - varc)
                                    * gamma ** vard * (1 - gamma) ** (1 - vard)

                        + theta3 * (1 - gamma) ** vara * gamma ** (1 - vara)
                                    * (1 - gamma) ** varb * gamma ** (1 - varb)
                                    * gamma ** varc * (1 - gamma) ** (1 - varc)
                                    * gamma ** vard * (1 - gamma) ** (1 - vard)
```

```
                    + theta4 * (1 - gamma) ** vara * gamma ** (1 - vara)
                           * (1 - gamma) ** varb * gamma ** (1 - varb)
                           * (1 - gamma) ** varc * gamma ** (1 - varc)
                           * gamma ** vard * (1- gamma)** (1 - vard)

                    + theta5 * (1 - gamma) ** vara * gamma ** (1 - vara)
                           * (1 - gamma) ** varb * gamma ** (1 - varb)
                           * (1 - gamma) ** varc * gamma ** (1 - varc)
                           * (1 - gamma) ** vard * gamma ** (1 - vard))) ;
run ;

data procextfreq ;
set proctor_est ;
where _type_ = 'PARMS' ;
c0000 = &l * (theta1 * (1 - gamma) ** 4
          + theta2 * gamma * (1 - gamma) ** 3
          + theta3 * gamma **2 * (1 - gamma) ** 2
          + theta4 * gamma ** 3 * (1 - gamma)
          + theta5 * gamma ** 4
          + theta6 * (1 - alpha) * (1 - beta) * (1 - delta) * (1 - sigma)) ;

c1000 = &l * (theta1 * gamma * (1 - gamma) ** 3
          + theta2 * (1 - gamma) ** 4
          + theta3 * (1 - gamma) ** 3 * gamma
          + theta4 * (1 - gamma) ** 2 * gamma ** 2
          + theta5 * (1 - gamma) * gamma ** 3
          + theta6 * alpha * (1 - beta) * (1 - delta) * (1 - sigma)) ;

c1100 = &l * (theta1 * gamma ** 2 * (1 - gamma) ** 2
          + theta2 * (1 - gamma) ** 3 * gamma
          + theta3 * (1 - gamma) ** 4
          + theta4 * (1- gamma) ** 3 * gamma
          + theta5 * (1 - gamma) ** 2 * gamma ** 2
          + theta6 * alpha * beta * (1 - delta) * (1 - sigma) ) ;

c1110 = &l * (theta1 * gamma ** 3 * (1 - gamma)
          + theta2 * (1 - gamma) ** 2 * gamma ** 2
          + theta3 * (1 - gamma) ** 3 * gamma
          + theta4 * (1 - gamma) ** 4
          + theta5 * (1- gamma) ** 3 * gamma
          + theta6 * alpha * beta * delta * (1 - sigma)) ;

c1111 = &l * (theta1 * gamma ** 4
          + theta2 * (1 - gamma) * gamma ** 3
          + theta3 * (1 - gamma) ** 2 * gamma ** 2
          + theta4 * (1 - gamma) ** 3 * gamma
```

```
            + theta5 * (1 - gamma) ** 4
            + theta6 * alpha * beta * delta * sigma) ;

c0100 = &l * (theta1 * (1 - gamma) ** 3 * gamma
            + theta2 * gamma ** 2 * (1 - gamma) ** 2
            + theta3 * (1 - gamma) ** 3 * gamma
            + theta4 * gamma ** 2 * (1 - gamma) ** 2
            + theta5 * gamma ** 3 * (1 - gamma)
            + theta6 * (1 - alpha) * beta * (1 - delta) * (1 - sigma)) ;

c0110 = &l * (theta1 * (1 - gamma) ** 2 * gamma ** 2
            + theta2 * gamma ** 3 * (1 - gamma)
            + theta3 * gamma ** 2 * (1 - gamma) ** 2
            + theta4 * gamma * (1 - gamma) ** 3
            + theta5 * gamma ** 2 * (1 - gamma) ** 2
            + theta6 * (1 - alpha) * beta * delta * (1 - sigma)) ;

c0111 = &l * (theta1 * (1 - gamma) * gamma ** 3
            + theta2 * gamma ** 4
            + theta3 * gamma ** 3 * (1 - gamma)
            + theta4 * gamma ** 2 * (1 - gamma) ** 2
            + theta5 * gamma * (1 - gamma) ** 3
            + theta6 * (1 - alpha) * beta * delta * sigma ) ;

c1001 = &l * (theta1 * gamma ** 2 * ( 1 - gamma) ** 2
            + theta2 * (1 - gamma) ** 3 * gamma
            + theta3 * (1 - gamma) ** 2 * gamma ** 2
            + theta4 * (1 - gamma) * gamma ** 3
            + theta5 * (1 - gamma) ** 2 * gamma ** 2
            + theta6 * alpha * (1 - beta) * (1 - delta) * sigma ) ;

c0101 = &l * (theta1 * (1 - gamma) ** 2 * gamma ** 2
            + theta2 * (1 - gamma) * gamma ** 3
            + theta3 * gamma ** 2 * (1 - gamma) ** 2
            + theta4 * gamma ** 3 * (1 - gamma)
            + theta5 * (1 - gamma) ** 2 * gamma ** 2
            + theta6 * (1 - alpha) * beta * (1 - delta) * sigma ) ;

c0001 = &l * (theta1 * (1 - gamma) ** 3 * gamma
            + theta2 * gamma ** 2 * (1 - gamma) ** 2
            + theta3 * gamma ** 3 * (1 - gamma)
            + theta4 * gamma ** 4
            + theta5 * gamma ** 3 * (1 - gamma)
            + theta6 * (1 - alpha) * (1 - beta) * (1 - delta) * sigma ) ;

c0011 = &l * (theta1 * (1 - gamma) ** 2 * gamma ** 2
```

```
                + theta2 * gamma ** 3 * (1 - gamma)
                + theta3 * gamma ** 4
                + theta4 * gamma ** 3 * (1 - gamma)
                + theta5 * (1 - gamma) ** 2 * gamma ** 2
                + theta6 * (1 - alpha) * (1 - beta) * delta * sigma ) ;

c1010 = &l * (theta1 * gamma ** 2 * (1 - gamma) ** 2
                + theta2 * (1 - gamma) ** 3 * gamma
                + theta3 * (1 - gamma) ** 2 * gamma ** 2
                + theta4 * (1 - gamma) ** 3 * gamma
                + theta5 * gamma ** 2 * (1 - gamma)** 2
                + theta6 * alpha * (1 - beta) * delta * (1 - sigma) ) ;

c1101 = &l * (theta1 * gamma ** 3 * (1 - gamma)
                + theta2 * (1 - gamma) ** 2 * gamma ** 2
                + theta3 * gamma * (1 - gamma) ** 3
                + theta4 * (1 - gamma) ** 2 * gamma ** 2
                + theta5 * (1 - gamma) ** 3 * gamma
                + theta6 * alpha * beta * (1 - delta) * sigma ) ;

c1011 = &l * (theta1 * gamma ** 3 * (1 - gamma)
                + theta2 * (1 - gamma) ** 2 * gamma ** 2
                + theta3 * gamma ** 3 * (1 - gamma)
                + theta4 * (1 - gamma) ** 2 * gamma ** 2
                + theta5 * (1 - gamma) ** 3 * gamma
                + theta6 * alpha * (1 - beta) * delta * sigma ) ;

c0010 = &l * (theta1 * (1 - gamma) ** 3 * gamma
                + theta2 * (1 - gamma) ** 2 * gamma ** 2
                + theta3 * gamma ** 3 * (1 - gamma)
                + theta4 * gamma ** 2 * (1 - gamma) ** 2
                + theta5 * gamma ** 3 * (1 - gamma)
                + theta6 * (1 - alpha) * (1 - beta) * delta * (1 - sigma) ) ;
run ;

*creating macro variables to serve as starting values for pistar solution ;
proc sql noprint ;
select theta1, theta2, theta3, theta4, theta5, gamma
into :start_theta1, :start_theta2, :start_theta3, :start_theta4,
:start_theta5, :start_gamma from work.procextfreq ;
quit ;

*creating data sets for input into PROC UNIVARIATE to compute replicate statistics ;
%if &i = 1 %then %do ;
data pr_univ.procextend&m ;
array obscnt{16} d0000 d1000 d1100 d1110 d1111 d0100 d0010 d0001 d0101
```

```
                        d0110 d0011 d0111 d1001 d1010 d1011 d1101 ;
array expcnt{16} c0000 c1000 c1100 c1110 c1111 c0100 c0010 c0001 c0101
                        c0110 c0011 c0111 c1001 c1010 c1011 c1101 ;
merge procextfreq (in = a) transfreq (in = b) ;
iter = &i ;
sampsize = &l ;
run ;
%end ;

%else %if &i > 1 %then %do ;
data prextobsexp ;
array obscnt{16} d0000 d1000 d1100 d1110 d1111 d0100 d0010 d0001 d0101
                        d0110 d0011 d0111 d1001 d1010 d1011 d1101 ;
array expcnt{16} c0000 c1000 c1100 c1110 c1111 c0100 c0010 c0001 c0101
                         c0110 c0011 c0111 c1001 c1010 c1011 c1101 ;
merge procextfreq (in = a) transfreq (in = b) ;
iter = &i ;
sampsize = &l ;
run ;
%end ;

%if &i > 1 %then %do ;
proc append base= pr_univ.procextend&m data = work.prextobsexp ;
run ;
%end ;

*pistar computation ;
proc nlp data = resppat tech = quanew outest = pistar_est
(drop = _iter_ _name_ _rhs_ _tech_) lis = 2 lsprecision = 0.06 noprint ;

max lik ;

parms gamma = &start_gamma ,
          theta1 = &start_theta1 ,
          theta2 = &start_theta2 ,
          theta3 = &start_theta3 ,
          theta4 = &start_theta4 ,
          theta5 = &start_theta5 ;

bounds 0 < theta1 - theta5 < 1 ,
           0 < gamma < 1 ;

nlincon  0 lt c1 le &c0000 ,
            0 lt c2 le &c1000 ,
            0 lt c3 le &c1100 ,
            0 lt c4 le &c1110 ,
```

```
        0 lt c5 le &c1111 ,
        0 lt c6 le &c0100 ,
        0 lt c7 le &c0110 ,
        0 lt c8 le &c0111 ,
        0 lt c9 le &c1001 ,
        0 lt c10 le &c0101 ,
        0 lt c11 le &c0001 ,
        0 lt c12 le &c0011 ,
        0 lt c13 le &c1010 ,
        0 lt c14 le &c1101 ,
        0 lt c15 le &c1011 ,
        0 lt c16 le &c0010 ;

c1 = &l * (theta1 * (1 - gamma) ** 4
        + theta2 * gamma * (1 - gamma) ** 3
        + theta3 * gamma **2 * (1 - gamma) ** 2
        + theta4 * gamma ** 3 * (1 - gamma)
        + theta5 * gamma ** 4 ) ;

c2 = &l * (theta1 * gamma * (1 - gamma) ** 3
        + theta2 * (1 - gamma) ** 4
        + theta3 * (1 - gamma) ** 3 * gamma
        + theta4 * (1 - gamma) ** 2 * gamma ** 2
        + theta5 * (1 - gamma) * gamma ** 3 ) ;

c3 = &l * (theta1 * gamma ** 2 * (1 - gamma) ** 2
        + theta2 * (1 - gamma) ** 3 * gamma
        + theta3 * (1 - gamma) ** 4
        + theta4 * ( 1- gamma) ** 3 * gamma
        + theta5 * (1 - gamma) ** 2 * gamma ** 2 ) ;

c4 = &l * (theta1 * gamma ** 3 * (1 - gamma)
        + theta2 * (1 - gamma) ** 2 * gamma ** 2
        + theta3 * (1 - gamma) ** 3 * gamma
        + theta4 * (1 - gamma) ** 4
        + theta5 * (1- gamma) ** 3 * gamma) ;

c5 = &l * (theta1 * gamma ** 4
        + theta2 * (1 - gamma) * gamma ** 3
        + theta3 * (1 - gamma) ** 2 * gamma ** 2
        + theta4 * (1 - gamma) ** 3 * gamma
        + theta5 * (1 - gamma) ** 4 ) ;

c6 = &l * (theta1 * (1 - gamma) ** 3 * gamma
        + theta2 * gamma ** 2 * (1 - gamma) ** 2
        + theta3 * (1 - gamma) ** 3 * gamma
```

```
          + theta4 * gamma ** 2 * (1 - gamma) ** 2
          + theta5 * gamma ** 3 * (1 - gamma)) ;

c7 = &l * (theta1 * (1 - gamma) ** 2 * gamma ** 2
          + theta2 * gamma ** 3 * (1 - gamma)
          + theta3 * gamma ** 2 * (1 - gamma) ** 2
          + theta4 * gamma * (1 - gamma) ** 3
          + theta5 * gamma ** 2 * (1 - gamma) ** 2 ) ;

c8 = &l * (theta1 * (1 - gamma) * gamma ** 3
          + theta2 * gamma ** 4
          + theta3 * gamma ** 3 * (1 - gamma)
          + theta4 * gamma ** 2 * (1 - gamma) ** 2
          + theta5 * gamma * (1 - gamma) ** 3 ) ;

c9 = &l * (theta1 * gamma ** 2 * ( 1 - gamma) ** 2
          + theta2 * (1 - gamma) ** 3 * gamma
          + theta3 * (1 - gamma) ** 2 * gamma ** 2
          + theta4 * (1 - gamma) * gamma ** 3
          + theta5 * ( 1 - gamma) ** 2 * gamma ** 2 ) ;

c10 = &l * (theta1 * (1 - gamma) ** 2 * gamma ** 2
           + theta2 * ( 1 - gamma) * gamma ** 3
           + theta3 * gamma ** 2 * (1 - gamma) ** 2
           + theta4 * gamma ** 3 * (1 - gamma)
           + theta5 * (1 - gamma) ** 2 * gamma ** 2 )  ;

c11 = &l * (theta1 * (1 - gamma) ** 3 * gamma
           + theta2 * gamma ** 2 * (1 - gamma) ** 2
           + theta3 * gamma ** 3 * (1 - gamma)
           + theta4 * gamma ** 4
           + theta5 * gamma ** 3 * (1 - gamma) )   ;

c12 = &l * (theta1 * (1 - gamma) ** 2 * gamma ** 2
           + theta2 * gamma ** 3 * (1 - gamma)
           + theta3 * gamma ** 4
           + theta4 * gamma ** 3 * (1 - gamma)
           + theta5 * (1 - gamma) ** 2 * gamma ** 2 )  ;

c13 = &l * (theta1 * gamma ** 2 * (1 - gamma) ** 2
           + theta2 * (1 - gamma) ** 3 * gamma
           + theta3 * (1 - gamma) ** 2 * gamma ** 2
           + theta4 * (1 - gamma) ** 3 * gamma
           + theta5 * gamma ** 2 * (1 - gamma)** 2 )  ;

c14 = &l * (theta1 * gamma ** 3 * (1 - gamma)
```

```
              + theta2 * (1 - gamma) ** 2 * gamma ** 2
              + theta3 * gamma * (1 - gamma) ** 3
              + theta4 * (1 - gamma) ** 2 * gamma ** 2
              + theta5 * (1 - gamma) ** 3 * gamma )  ;

c15 = &l * (theta1 * gamma ** 3 * (1 - gamma)
              + theta2 * (1 - gamma) ** 2 * gamma ** 2
              + theta3 * gamma ** 3 * (1 - gamma)
              + theta4 * (1 - gamma) ** 2 * gamma ** 2
              + theta5 * (1 - gamma) ** 3 * gamma )  ;

c16 = &l * (theta1 * (1 - gamma) ** 3 * gamma
              + theta2 * (1 - gamma) ** 2 * gamma ** 2
              + theta3 * gamma ** 3 * (1 - gamma)
              + theta4 * gamma ** 2 * (1 - gamma) ** 2
              + theta5 * gamma ** 3 * (1 - gamma))  ;

lik =  &l * (theta1 * gamma ** vara * (1 - gamma) ** (1 - vara)
                         * gamma ** varb * (1 - gamma) ** (1 - varb)
                         * gamma ** varc * (1 - gamma) ** (1 - varc)
                         * gamma ** vard * (1 - gamma) ** (1 - vard)
              + theta2 * (1 - gamma) ** vara * gamma ** (1 - vara)
                         * gamma ** varb * (1 - gamma) ** (1 - varb)
                         * gamma ** varc * (1 - gamma) ** (1 - varc)
                         * gamma ** vard * (1 - gamma) ** (1 - vard)
              + theta3 * (1 - gamma) ** vara * gamma ** (1 - vara)
                         * (1 - gamma) ** varb * gamma ** (1 - varb)
                         * gamma ** varc * (1 - gamma) ** (1 - varc)
                         * gamma ** vard * (1 - gamma) ** (1 - vard)
              + theta4 * (1 - gamma) ** vara * gamma ** (1 - vara)
                         * (1 - gamma) ** varb * gamma ** (1 - varb)
                         * (1 - gamma) ** varc * gamma ** (1 - varc)
                         * gamma ** vard * (1- gamma)** (1 - vard)
              + theta5 * (1 - gamma) ** vara * gamma ** (1 - vara)
                         * (1 - gamma) ** varb * gamma ** (1 - varb)
                         * (1 - gamma) ** varc * gamma ** (1 - varc)
                         * (1 - gamma) ** vard * gamma ** (1 - vard)) ;
run ;

*computing expected frequencies using parameter estimates from pistar model ;
data pistarfreq ;
set pistar_est ;
where _type_ = 'PARMS' ;
c0000 = &l * (theta1 * (1 - gamma) ** 4
                + theta2 * gamma * (1 - gamma) ** 3
                + theta3 * gamma **2 * (1 - gamma) ** 2
```

```
                + theta4 * gamma ** 3 * (1 - gamma)
                + theta5 * gamma ** 4) ;

c1000 = &l * (theta1 * gamma * (1 - gamma) ** 3
                + theta2 * (1 - gamma) ** 4
                + theta3 * (1 - gamma) ** 3 * gamma
                + theta4 * (1 - gamma) ** 2 * gamma ** 2
                + theta5 * (1 - gamma) * gamma ** 3) ;

c1100 = &l * (theta1 * gamma ** 2 * (1 - gamma) ** 2
                + theta2 * (1 - gamma) ** 3 * gamma
                + theta3 * (1 - gamma) ** 4
                + theta4 * (1- gamma) ** 3 * gamma
                + theta5 * (1 - gamma) ** 2 * gamma ** 2) ;

c1110 = &l * (theta1 * gamma ** 3 * (1 - gamma)
                + theta2 * (1 - gamma) ** 2 * gamma ** 2
                + theta3 * (1 - gamma) ** 3 * gamma
                + theta4 * (1 - gamma) ** 4
                + theta5 * (1- gamma) ** 3 * gamma) ;

c1111 = &l * (theta1 * gamma ** 4
                + theta2 * (1 - gamma) * gamma ** 3
                + theta3 * (1 - gamma) ** 2 * gamma ** 2
                + theta4 * (1 - gamma) ** 3 * gamma
                + theta5 * (1 - gamma) ** 4) ;

c0100 = &l * (theta1 * (1 - gamma) ** 3 * gamma
                + theta2 * gamma ** 2 * (1 - gamma) ** 2
                + theta3 * (1 - gamma) ** 3 * gamma
                + theta4 * gamma ** 2 * (1 - gamma) ** 2
                + theta5 * gamma ** 3 * (1 - gamma)) ;

c0110 = &l * (theta1 * (1 - gamma) ** 2 * gamma ** 2
               + theta2 * gamma ** 3 * (1 - gamma)
               + theta3 * gamma ** 2 * (1 - gamma) ** 2
               + theta4 * gamma * (1 - gamma) ** 3
                + theta5 * gamma ** 2 * (1 - gamma) ** 2) ;

c0111 = &l * (theta1 * (1 - gamma) * gamma ** 3
                + theta2 * gamma ** 4
                + theta3 * gamma ** 3 * (1 - gamma)
                + theta4 * gamma ** 2 * (1 - gamma) ** 2
                + theta5 * gamma * (1 - gamma) ** 3) ;

c1001 = &l * (theta1 * gamma ** 2 * ( 1 - gamma) ** 2
```

```
                 + theta2 * (1 - gamma) ** 3 * gamma
                 + theta3 * (1 - gamma) ** 2 * gamma ** 2
                 + theta4 * (1 - gamma) * gamma ** 3
                 + theta5 * (1 - gamma) ** 2 * gamma ** 2)  ;

c0101 = &l * (theta1 * (1 - gamma) ** 2 * gamma ** 2
                 + theta2 * (1 - gamma) * gamma ** 3
                 + theta3 * gamma ** 2 * (1 - gamma) ** 2
                 + theta4 * gamma ** 3 * (1 - gamma)
                 + theta5 * (1 - gamma) ** 2 * gamma ** 2)  ;

c0001 = &l * (theta1 * (1 - gamma) ** 3 * gamma
                 + theta2 * gamma ** 2 * (1 - gamma) ** 2
                 + theta3 * gamma ** 3 * (1 - gamma)
                 + theta4 * gamma ** 4
                 + theta5 * gamma ** 3 * (1 - gamma)) ;

c0011 = &l * (theta1 * (1 - gamma) ** 2 * gamma ** 2
                 + theta2 * gamma ** 3 * (1 - gamma)
                 + theta3 * gamma ** 4
                 + theta4 * gamma ** 3 * (1 - gamma)
                 + theta5 * (1 - gamma) ** 2 * gamma ** 2)  ;

c1010 = &l * (theta1 * gamma ** 2 * (1 - gamma) ** 2
                 + theta2 * (1 - gamma) ** 3 * gamma
                 + theta3 * (1 - gamma) ** 2 * gamma ** 2
                 + theta4 * (1 - gamma) ** 3 * gamma
                 + theta5 * gamma ** 2 * (1 - gamma)** 2)  ;

c1101 = &l * (theta1 * gamma ** 3 * (1 - gamma)
                 + theta2 * (1 - gamma) ** 2 * gamma ** 2
                 + theta3 * gamma * (1 - gamma) ** 3
                 + theta4 * (1 - gamma) ** 2 * gamma ** 2
                 + theta5 * (1 - gamma) ** 3 * gamma) ;

c1011 = &l * (theta1 * gamma ** 3 * (1 - gamma)
                 + theta2 * (1 - gamma) ** 2 * gamma ** 2
                 + theta3 * gamma ** 3 * (1 - gamma)
                 + theta4 * (1 - gamma) ** 2 * gamma ** 2
                 + theta5 * (1 - gamma) ** 3 * gamma) ;

c0010 = &l * (theta1 * (1 - gamma) ** 3 * gamma
                 + theta2 * (1 - gamma) ** 2 * gamma ** 2
                 + theta3 * gamma ** 3 * (1 - gamma)
                 + theta4 * gamma ** 2 * (1 - gamma) ** 2
                 + theta5 * gamma ** 3 * (1 - gamma)) ;
```

```
sumfreq = c0000 + c1000 + c1100 + c1110 + c1111 + c0100 + c0110 + c0111
          + c1001 + c0101 + c0001 + c0011 + c1010 + c1101 + c1011 + c0010 ;
pistar = 1 - sumfreq / &l ;
run ;

*creating data sets for input into PROC UNIVARIATE ;
%if &i = 1 %then %do ;
data pr_univ.procpistar&m ;
merge pistarfreq (in = a) transfreq (in = b) ;
iter = &i ;
sampsize = &l ;
run ;
%end ;

%else %if &i > 1 %then %do ;
data pistarobsexp ;
merge pistarfreq (in = a) transfreq (in = b) ;
iter = &i ;
sampsize = &l ;
run ;
%end ;

%if &i > 1 %then %do ;
proc append base= pr_univ.procpistar&m  data = work.pistarobsexp ;
run ;
%end ;

proc datasets library = work ;
delete proctorsamp ;
run ;

%end ;

proc univariate data = pr_univ.procextend&m noprint ;
var theta1 theta2 theta3 theta4 theta5 theta6 gamma alpha beta delta sigma ;
output out = procextstat
mean = avgtheta1 avgtheta2 avgtheta3 avgtheta4 avgtheta5
         avgtheta6 avggamma avgalpha avgbeta avgdelta avgsigma ;
run ;

data pr_univ.reshape_procextend&m ;
set procextstat ;
parameter = 'theta1' ; mean = avgtheta1 ; output ;
parameter = 'theta2' ; mean = avgtheta2 ; output ;
parameter = 'theta3' ; mean = avgtheta3 ; output ;
parameter = 'theta4' ; mean = avgtheta4 ; output ;
```

```
parameter = 'theta5' ; mean = avgtheta5 ; output ;
parameter = 'theta6' ; mean = avgtheta6 ; output ;
parameter = 'gamma' ; mean = avggamma ; output ;
parameter = 'alpha' ; mean = avgalpha ; output ;
parameter = 'beta' ; mean = avgbeta ; output ;
parameter = 'delta' ; mean = avgdelta ; output ;
parameter = 'sigma' ; mean = avgsigma ; output ;

drop avgtheta1 avgtheta2 avgtheta3 avgtheta4 avgtheta5 avgtheta6
      avggamma avgalpha avgbeta avgdelta avgsigma ;
run ;

proc univariate data = pr_univ.procpistar&m noprint ;
var theta1 theta2 theta3 theta4 theta5 pistar gamma ;
output out = procpistat
mean = avgtheta1 avgtheta2 avgtheta3 avgtheta4 avgtheta5
          avgpistar avggamma ;
run ;

data pr_univ.reshape_pistar&m ;
set procpistat ;
parameter = 'theta1' ; mean = avgtheta1 ; output ;
parameter = 'theta2' ; mean = avgtheta2 ; output ;
parameter = 'theta3' ; mean = avgtheta3 ; output ;
parameter = 'theta4' ; mean = avgtheta4 ; output ;
parameter = 'theta5' ; mean = avgtheta5 ; output ;
parameter = 'pistar' ; mean = avgpistar ; output ;
parameter = 'gamma' ; mean = avggamma ; output ;

drop avgtheta1 avgtheta2 avgtheta3 avgtheta4 avgtheta5
      avgpistar avggamma ;
run ;

%end ;

%mend proctorsim ;

%proctorsim
```

```
*SAS program for extended IO and IO pistar models ;
libname pr_univ 'c:\nlp programs\io4\g5_iuc25_u_u' ;
filename pr_log 'c:\nlp programs\io4\g5_iuc25_u_u\proclog.txt' ;
filename pr_out 'c:\nlp programs\io4\g5_iuc25_u_u\procout.txt' ;
proc printto log = pr_log print = pr_out ;
run ;

*response patterns used as input into PROC NLP ;
data resppat ;
input vara varb varc vard pattern ;
cards ;
1  1  1  1  1
1  1  1  0  2
1  1  0  1  6
1  1  0  0  3
1  0  1  1  6
1  0  1  0  6
1  0  0  1  6
1  0  0  0  4
0  1  1  1  6
0  1  1  0  6
0  1  0  1  6
0  1  0  0  6
0  0  1  1  6
0  0  1  0  6
0  0  0  1  6
0  0  0  0  5
;
run ;

data iodata ;
pi_omission = 0.05 ;
pi_intrusion = 0.20 ;
 do j = 1 to 100000 ;
   latnum = ranuni(0) ;
    if 0 lt latnum le 0.10 then do ;
      latclass = 1 ;
      varanum = ranuni(0) ;
      varbnum = ranuni(0) ;
      varcnum = ranuni(0) ;
      vardnum = ranuni(0) ;
        if 0 lt varanum le pi_intrusion then vara = 1 ;
          else if pi_intrusion lt varanum le 1.00000 then vara = 0 ;
        if 0 lt varbnum le pi_intrusion then varb = 1 ;
          else if pi_intrusion lt varbnum le 1.00000 then varb = 0 ;
        if 0 lt varcnum le pi_intrusion then varc = 1 ;
```

```
      else if pi_intrusion lt varcnum le 1.00000 then varc = 0 ;
    if 0 lt vardnum le pi_intrusion then vard = 1 ;
      else if pi_intrusion lt vardnum le 1.00000 then vard = 0 ;
end ;

else if 0.10 lt latnum le 0.20 then do ;
  latclass = 2 ;
  varanum = ranuni(0) ;
  varbnum = ranuni(0) ;
  varcnum = ranuni(0) ;
  vardnum = ranuni(0) ;
    if 0 lt varanum le (1 - pi_omission) then vara = 1 ;
      else if (1 - pi_omission) lt varanum le 1.00000 then vara = 0 ;
    if 0 lt varbnum le pi_intrusion then varb = 1 ;
      else if pi_intrusion lt varbnum le 1.00000 then varb = 0 ;
    if 0 lt varcnum le pi_intrusion then varc = 1 ;
      else if pi_intrusion lt varcnum le 1.00000 then varc = 0 ;
    if 0 lt vardnum le pi_intrusion then vard = 1 ;
      else if pi_intrusion lt vardnum le 1.00000 then vard = 0 ;
end ;

else if 0.20 lt latnum le 0.35 then do ;
  latclass = 3 ;
  varanum = ranuni(0) ;
  varbnum = ranuni(0) ;
  varcnum = ranuni(0) ;
  vardnum = ranuni(0) ;
    if 0 lt varanum le (1 - pi_omission) then vara = 1 ;
      else if (1 - pi_omission) lt varanum le 1.00000 then vara = 0 ;
    if 0 lt varbnum le (1 - pi_omission)  then varb = 1 ;
      else if (1 - pi_omission) lt varbnum le 1.00000 then varb = 0 ;
    if 0 lt varcnum le pi_intrusion then varc = 1 ;
      else if pi_intrusion lt varcnum le 1.00000 then varc = 0 ;
    if 0 lt vardnum le pi_intrusion then vard = 1 ;
      else if pi_intrusion lt vardnum le 1.00000 then vard = 0 ;
end ;

else if 0.35 lt latnum le 0.55 then do ;
  latclass = 4 ;
  varanum = ranuni(0) ;
  varbnum = ranuni(0) ;
  varcnum = ranuni(0) ;
  vardnum = ranuni(0) ;
    if 0 lt varanum le (1 - pi_omission) then vara = 1 ;
      else if (1 - pi_omission) lt varanum le 1.00000 then vara = 0 ;
    if 0 lt varbnum le (1 - pi_omission) then varb = 1 ;
```

```
      else if (1 - pi_omission) lt varbnum le 1.00000 then varb = 0 ;
    if 0 lt varcnum le (1 - pi_omission) then varc = 1 ;
      else if (1 - pi_omission) lt varcnum le 1.00000 then varc = 0 ;
    if 0 lt vardnum le pi_intrusion then vard = 1 ;
      else if pi_intrusion lt vardnum le 1.00000 then vard = 0 ;
  end ;

  else if 0.55 lt latnum le 0.75 then do ;
    latclass = 5 ;
    varanum = ranuni(0) ;
    varbnum = ranuni(0) ;
    varcnum = ranuni(0) ;
    vardnum = ranuni(0) ;
      if 0 lt varanum le (1 - pi_omission) then vara = 1 ;
        else if (1 - pi_omission) lt varanum le 1.00000 then vara = 0 ;
      if 0 lt varbnum le (1 - pi_omission) then varb = 1 ;
        else if (1 - pi_omission) lt varbnum le 1.00000 then varb = 0 ;
      if 0 lt varcnum le (1 - pi_omission) then varc = 1 ;
         else if (1 - pi_omission) lt varcnum le 1.00000 then varc = 0 ;
      if 0 lt vardnum le (1 - pi_omission) then vard = 1 ;
         else if (1 - pi_omission) lt vardnum le 1.00000 then vard = 0 ;
    end ;

  else if 0.75 lt latnum le 1.0000 then do ;
    latclass = 6 ;
    varanum = ranuni(0) ;
    varbnum = ranuni(0) ;
    varcnum = ranuni(0) ;
    vardnum = ranuni(0) ;
    if varanum le 0.20 then vara = 1 ;
      else if varanum gt 0.20 then vara = 0 ;
    if varbnum le 0.10 then varb = 1 ;
      else if varbnum gt 0.10 then varb = 0 ;
    if varcnum le 0.15 then varc = 1 ;
      else if varcnum gt 0.15 then varc = 0 ;
    if vardnum le 0.30 then vard = 1 ;
      else if vardnum gt 0.30 then vard = 0 ;
  end ;
  output ;
 end ;
run ;

*options symbolgen mlogic mprint ;
*macro for sampling, estimation, and summarization of replicate statistics ;
%macro iosim ;
%do m = 1 %to 3 ;
```

```
%let l = %eval(120 * 2 ** &m) ;
%do i = 1 %to 1000 ;

proc surveyselect data = iodata method = urs n = &l out = iosamp outhits noprint ;
run ;

proc freq data = iosamp noprint ;
  tables vara * varb * varc * vard /sparse out = iodata (drop = percent) ;
run ;

data transfreq ;
array obsfreq{16} d0000 d0001 d0010 d0011 d0100 d0101 d0110 d0111
                  d1000 d1001 d1010 d1011 d1100 d1101 d1110 d1111 ;
do m = 1 to 16 ;
  set iodata ;
    if count > 0 then obsfreq{m} = count ;
      else if count = 0 then obsfreq{m} = 1 ;
      *in case of sampling zeros, Pan (2006) suggests using 1 as a flattening constant  ;
end ;
drop vara varb varc vard m count ;
run ;

data _null_ ;
set iodata ;
if count > 0 then countr = count ;
  else countr = 1 ;
convara = left(put(vara,1.)) ;
convarb = left(put(varb,1.)) ;
convarc = left(put(varc,1.)) ;
convard = left(put(vard,1.)) ;
call symput('c'||convara||convarb||convarc||convard, trim(left(put(countr,3.)))) ;
run ;

data procabridge ;
set iodata ;
if count > 0 then countr = count ;
  else countr = 1 ;
run ;

proc nlp data = procabridge tech = dbldog outest = io_est
(drop = _iter_ _name_ _rhs_ _tech_) noprint ;

max loglik ;
parms beta_o = 0.05 ,
      beta_i = 0.20 ,
      theta1 = 0.10 ,
```

```
          theta2 = 0.10 ,
          theta3 = 0.15 ,
          theta4 = 0.20 ,
          theta5 = 0.20 ,
          theta6 = 0.25 ,
          alpha = 0.20 ,
          beta = 0.10 ,
          delta = 0.15 ,
          sigma = 0.30 ;

bounds 0 < theta1 - theta6 < 1 ,
         0 < beta_i < 1 ,
         0 < beta_o < 1 ,
         0 < alpha < 1 ,
         0 < beta < 1 ,
         0 < delta < 1 ,
         0 < sigma < 1 ;

lincon theta1 + theta2 + theta3 + theta4 + theta5 + theta6 = 1 ;

loglik = countr * log(theta6 * alpha ** vara * (1 - alpha) ** (1 - vara)
                             * beta ** varb * (1 - beta) ** (1 - varb)
                             * delta ** varc * (1 - delta) ** (1 - varc)
                             * sigma ** vard * (1 - sigma) ** (1 - vard)
                    + theta1 * beta_i ** vara * (1 - beta_i) ** (1 - vara)
                             * beta_i ** varb * (1 - beta_i) ** (1 - varb)
                             * beta_i ** varc * (1 - beta_i) ** (1 - varc)
                             * beta_i ** vard * (1 - beta_i) ** (1 - vard)
                    + theta2 * (1 - beta_o) ** vara * beta_o ** (1 - vara)
                             * beta_i ** varb * (1 - beta_i) ** (1 - varb)
                             * beta_i ** varc * (1 - beta_i) ** (1 - varc)
                             * beta_i ** vard * (1 - beta_i) ** (1 - vard)
                    + theta3 * (1 - beta_o) ** vara *    beta_o ** (1 - vara)
                             * (1 - beta_o) ** varb * beta_o ** (1 - varb)
                             * beta_i ** varc * (1 - beta_i) ** (1 - varc)
                             * beta_i ** vard * (1 - beta_i) ** (1 - vard)
                    + theta4 * (1 - beta_o) ** vara * beta_o ** (1 - vara)
                             * (1 - beta_o) ** varb * beta_o ** (1 - varb)
                             * (1 - beta_o) ** varc * beta_o ** (1 - varc)
                             * beta_i ** vard * (1 - beta_i) ** (1 - vard)
                    + theta5 * (1 - beta_o) ** vara * beta_o ** (1 - vara)
                             * (1 - beta_o) ** varb * beta_o ** (1 - varb)
                             * (1 - beta_o) ** varc * beta_o ** (1 - varc)
                             * (1 - beta_o) ** vard * beta_o ** (1 - vard)) ;
run ;
data procextfreq ;
```

```
set io_est ;
where _type_ = 'PARMS' ;
c0000 = &l * (theta1 * (1 - beta_i) ** 4
            + theta2 * beta_o * (1 - beta_i) ** 3
            + theta3 * beta_o **2 * (1 - beta_i) ** 2
            + theta4 * beta_o ** 3 * (1 - beta_i)
            + theta5 * beta_o ** 4
            + theta6 * (1 - alpha) * (1 - beta) * (1 - delta) * (1 - sigma)) ;

c1000 = &l * (theta1 * beta_i * (1 - beta_i) ** 3
            + theta2 * (1 - beta_o) * (1 - beta_i) ** 3
            + theta3 * (1 - beta_o) * beta_o * (1 - beta_i) ** 2
            + theta4 * (1 - beta_o) * beta_o ** 2 * ( 1 - beta_i)
            + theta5 * (1 - beta_o) * beta_o ** 3
            + theta6 * alpha * (1 - beta) * (1 - delta) * (1 - sigma)) ;

c1100 = &l * (theta1 * beta_i ** 2 * (1 - beta_i) ** 2
            + theta2 * (1 - beta_o) * beta_i * (1 - beta_i) ** 2
            + theta3 * (1 - beta_o) ** 2 * (1 - beta_i) ** 2
            + theta4 * beta_o * (1 - beta_o) ** 2 * (1 - beta_i)
            + theta5 * (1 - beta_o) ** 2 * beta_o ** 2
            + theta6 * alpha * beta * (1 - delta) * (1 - sigma) ) ;

c1110 = &l * (theta1 * beta_i ** 3 * (1 - beta_i)
            + theta2 * (1 - beta_o) * beta_i ** 2 * (1 - beta_i)
            + theta3 * (1 - beta_o) ** 2 * beta_i * (1 - beta_i)
            + theta4 * (1 - beta_o) ** 3 * (1 - beta_i)
            + theta5 * (1 - beta_o) ** 3 * beta_o
            + theta6 * alpha * beta * delta * (1 - sigma)) ;

c1111 = &l * (theta1 * beta_i ** 4
            + theta2 * (1 - beta_o) * beta_i ** 3
            + theta3 * (1 - beta_o) ** 2 * beta_i ** 2
            + theta4 * (1 - beta_o) ** 3 * beta_i
            + theta5 * (1 - beta_o) ** 4
            + theta6 * alpha * beta * delta * sigma) ;

c0100 = &l * (theta1 * (1 - beta_i) ** 3 * beta_i
            + theta2 * beta_o * beta_i * (1 - beta_i) ** 2
            + theta3 * beta_o * (1 - beta_o) * (1 - beta_i) ** 2
            + theta4 * beta_o ** 2 * (1 - beta_o) * (1 - beta_i)
            + theta5 * beta_o ** 3 * (1 - beta_o)
            + theta6 * (1 - alpha) * beta * (1 - delta) * (1 - sigma)) ;

c0110 = &l * (theta1 * (1 - beta_i) ** 2 * beta_i ** 2
            + theta2 * beta_o * beta_i ** 2 * (1 - beta_i)
```

```
                + theta3 * beta_o * (1 - beta_o) * beta_i * (1 - beta_i)
                + theta4 * beta_o * (1 - beta_o) ** 2 * (1 - beta_i)
                + theta5 * beta_o ** 2 * (1 - beta_o) ** 2
                + theta6 * (1 - alpha) * beta * delta * (1 - sigma)) ;

c0111 = &l * (theta1 * (1 - beta_i) * beta_i ** 3
                + theta2 * beta_o * beta_i ** 3
                + theta3 * beta_o * (1 - beta_o) * beta_i ** 2
                + theta4 * beta_o * (1 - beta_o) ** 2 * beta_i
                + theta5 * (1 - beta_o) ** 3 * beta_o
                + theta6 * (1 - alpha) * beta * delta * sigma ) ;

c1001 = &l * (theta1 * beta_i ** 2 * ( 1 - beta_i) ** 2
                + theta2 * (1 - beta_o) * (1 - beta_i) ** 2 * beta_i
                + theta3 * (1 - beta_o) * beta_o * (1 - beta_i) * beta_i
                + theta4 * (1 - beta_o) * beta_o ** 2 * beta_i
                + theta5 * (1 - beta_o) ** 2 * beta_o ** 2
                + theta6 * alpha * (1 - beta) * (1 - delta) * sigma ) ;

c0101 = &l * (theta1 * beta_i ** 2 * (1 - beta_i) ** 2
                + theta2 * beta_o * beta_i ** 2 * (1 - beta_i)
                + theta3 * beta_o * (1 - beta_o) * (1 - beta_i) * beta_i
                + theta4 * beta_o ** 2 * (1 - beta_o) * beta_i
                + theta5 * beta_o ** 2 * (1 - beta_o) ** 2
                + theta6 * (1 - alpha) * beta * (1 - delta) * sigma ) ;

c0001 = &l * (theta1 * (1 - beta_i) ** 3 * beta_i
                + theta2 * beta_o * (1 - beta_i) ** 2 * beta_i
                + theta3 * beta_o ** 2 * (1 - beta_i)* beta_i
                + theta4 * beta_o ** 3 * beta_i
                + theta5 * beta_o ** 3 * (1 - beta_o)
                + theta6 * (1 - alpha) * (1 - beta) * (1 - delta) * sigma ) ;

c0011 = &l * (theta1 * (1 - beta_i) ** 2 * beta_i ** 2
                + theta2 * beta_o * (1 - beta_i) * beta_i ** 2
                + theta3 * beta_o ** 2 * beta_i ** 2
                + theta4 * beta_o ** 2 * (1 - beta_o) * beta_i
                + theta5 * beta_o ** 2 * (1 - beta_o) ** 2
                + theta6 * (1 - alpha) * (1 - beta) * delta * sigma ) ;

c1010 = &l * (theta1 * beta_i ** 2 * (1 - beta_i) ** 2
                + theta2 * (1 - beta_o) * (1 - beta_i) ** 2 * beta_i
                + theta3 * (1 - beta_o) * beta_o * beta_i * (1 - beta_i)
                + theta4 * (1 - beta_o) ** 2 * beta_o * (1 - beta_i)
                + theta5 * (1 - beta_o) ** 2 * beta_o ** 2
                + theta6 * alpha * (1 - beta) * delta * (1 - sigma) ) ;
```

```
c1101 = &l * (theta1 * beta_i ** 3 * (1 - beta_i)
                + theta2 * (1 - beta_o) * beta_i ** 2 * (1 - beta_i)
                + theta3 * (1 - beta_o) ** 2 * beta_i * (1 - beta_i)
                + theta4 * (1 - beta_o) ** 2 * beta_o * beta_i
                + theta5 * (1 - beta_o) ** 3 * beta_o
                + theta6 * alpha * beta * (1 - delta) * sigma ) ;

c1011 = &l * (theta1 * beta_i ** 3 * (1 - beta_i)
                + theta2 * (1 - beta_o) * (1 - beta_i) * beta_i ** 2
                + theta3 * (1 - beta_o) * beta_o * beta_i ** 2
                + theta4 * (1 - beta_o) ** 2 * beta_o * beta_i
                + theta5 * (1 - beta_o) ** 3 * beta_o
                + theta6 * alpha * (1 - beta) * delta * sigma ) ;

c0010 = &l * (theta1 * (1 - beta_i) ** 3 * beta_i
                + theta2 * (1 - beta_i) ** 2 * beta_i * beta_o
                + theta3 * beta_o ** 2 * (1 - beta_i)* beta_i
                + theta4 * beta_o ** 2 * (1 - beta_i) * (1 - beta_o)
                + theta5 * beta_o ** 3 * (1 - beta_o)
                + theta6 * (1 - alpha) * (1 - beta) * delta * (1 - sigma) ) ;
run ;

*creating macro variables to serve as starting values for pistar solution ;
proc sql noprint ;
select theta1, theta2, theta3, theta4, theta5, beta_i, beta_o
into :start_theta1, :start_theta2, :start_theta3, :start_theta4 ,
:start_theta5, :start_beta_i, :start_beta_o
from work.procextfreq ;
quit ;

*creating data sets for input into PROC UNIVARIATE to compute replicate statistics ;
%if &i = 1 %then %do ;
data pr_univ.procextend&m ;
array obscnt{16} d0000 d1000 d1100 d1110 d1111 d0100 d0010 d0001 d0101
                 d0110 d0011 d0111 d1001 d1010 d1011 d1101 ;
array expcnt{16} c0000 c1000 c1100 c1110 c1111 c0100 c0010 c0001 c0101
                 c0110 c0011 c0111 c1001 c1010 c1011 c1101 ;
merge ioextfreq (in = a) transfreq (in = b) ;
iter = &i ;
sampsize = &l ;
run ;
%end ;

%else %if &i > 1 %then %do ;
data prextobsexp ;
```

```
array obscnt{16} d0000 d1000 d1100 d1110 d1111 d0100 d0010 d0001 d0101
                    d0110 d0011 d0111 d1001 d1010 d1011 d1101 ;
array expcnt{16} c0000 c1000 c1100 c1110 c1111 c0100 c0010 c0001 c0101
                    c0110 c0011 c0111 c1001 c1010 c1011 c1101 ;
merge procextfreq (in = a) transfreq (in = b) ;
iter = &i ;
sampsize = &l ;
run ;
%end ;

%if &i > 1 %then %do ;
proc append base= pr_univ.procextend&m data = work.prextobsexp ;
run ;
%end ;

*pistar computation ;
proc nlp data = resppat tech = quanew outest = pistar_est
(drop = _iter_ _name_ _rhs_ _tech_) lis = 2 lsprecision = 0.06 noprint ;

max lik ;

parms beta_o = &start_beta_o ,
        beta_i = &start_beta_i  ,
        theta1 = &start_theta1  ,
        theta2 = &start_theta2  ,
        theta3 = &start_theta3  ,
        theta4 = &start_theta4  ,
        theta5 = &start_theta5  ;

bounds 0 < theta1 - theta5 < 1 ,
        0 < beta_i beta_o < 1 ;

nlincon  0 lt c_0000 le &c0000 ,
         0 lt c_1000 le &c1000 ,
         0 lt c_1100 le &c1100 ,
         0 lt c_1110 le &c1110 ,
         0 lt c_1111 le &c1111 ,
         0 lt c_0100 le &c0100 ,
         0 lt c_0110 le &c0110 ,
         0 lt c_0111 le &c0111 ,
         0 lt c_1001 le &c1001 ,
         0 lt c_0101 le &c0101 ,
         0 lt c_0001 le &c0001 ,
         0 lt c_0011 le &c0011 ,
         0 lt c_1010 le &c1010 ,
         0 lt c_1101 le &c1101 ,
```

```
            0 lt c_1011 le &c1011 ,
            0 lt c_0010 le &c0010 ;

c_0000 = &l * (theta1 * (1 - beta_i) ** 4
                + theta2 * beta_o * (1 - beta_i) ** 3
                + theta3 * beta_o **2 * (1 - beta_i) ** 2
                + theta4 * beta_o ** 3 * (1 - beta_i)
                + theta5 * beta_o ** 4) ;

c_1000 = &l * (theta1 * beta_i * (1 - beta_i) ** 3
                + theta2 * (1 - beta_o) * (1 - beta_i) ** 3
                + theta3 * (1 - beta_o) * beta_o * (1 - beta_i) ** 2
                + theta4 * (1 - beta_o) * beta_o ** 2 * ( 1 - beta_i)
                + theta5 * (1 - beta_o) * beta_o ** 3) ;

c_1100 = &l * (theta1 * beta_i ** 2 * (1 - beta_i) ** 2
                + theta2 * (1 - beta_o) * beta_i * (1 - beta_i) ** 2
                + theta3 * (1 - beta_o) ** 2 * (1 - beta_i) ** 2
                + theta4 * beta_o * (1 - beta_o) ** 2 * (1 - beta_i)
                + theta5 * (1 - beta_o) ** 2 * beta_o ** 2) ;

c_1110 = &l * (theta1 * beta_i ** 3 * (1 - beta_i)
                + theta2 * (1 - beta_o) * beta_i ** 2 * (1 - beta_i)
                + theta3 * (1 - beta_o) ** 2 * beta_i * (1 - beta_i)
                + theta4 * (1 - beta_o) ** 3 * (1 - beta_i)
                + theta5 * (1 - beta_o) ** 3 * beta_o) ;

c_1111 = &l * (theta1 * beta_i ** 4
                + theta2 * (1 - beta_o) * beta_i ** 3
                + theta3 * (1 - beta_o) ** 2 * beta_i ** 2
                + theta4 * (1 - beta_o) ** 3 * beta_i
                + theta5 * (1 - beta_o) ** 4) ;

c_0100 = &l * (theta1 * (1 - beta_i) ** 3 * beta_i
                + theta2 * beta_o * beta_i * (1 - beta_i) ** 2
                + theta3 * beta_o * (1 - beta_o) * (1 - beta_i) ** 2
                + theta4 * beta_o ** 2 * (1 - beta_o) * (1 - beta_i)
                + theta5 * beta_o ** 3 * (1 - beta_o)) ;

c_0110 = &l * (theta1 * (1 - beta_i) ** 2 * beta_i ** 2
                + theta2 * beta_o * beta_i ** 2 * (1 - beta_i)
                + theta3 * beta_o * (1 - beta_o) * beta_i * (1 - beta_i)
                + theta4 * beta_o * (1 - beta_o) ** 2 * (1 - beta_i)
                + theta5 * beta_o ** 2 * (1 - beta_o) ** 2 ) ;

c_0111 = &l * (theta1 * (1 - beta_i) * beta_i ** 3
```

```
                + theta2 * beta_o * beta_i ** 3
                + theta3 * beta_o * (1 - beta_o) * beta_i ** 2
                + theta4 * beta_o * (1 - beta_o) ** 2 * beta_i
                + theta5 * (1 - beta_o) ** 3 * beta_o) ;

c_1001 = &l * (theta1 * beta_i ** 2 * ( 1 - beta_i) ** 2
                + theta2 * (1 - beta_o) * (1 - beta_i) ** 2 * beta_i
                + theta3 * (1 - beta_o) * beta_o * (1 - beta_i) * beta_i
                + theta4 * (1 - beta_o) * beta_o ** 2 * beta_i
                + theta5 * (1 - beta_o) ** 2 * beta_o ** 2 ) ;

c_0101 = &l * (theta1 * beta_i ** 2 * (1 - beta_i) ** 2
                + theta2 * beta_o * beta_i ** 2 * (1 - beta_i)
                + theta3 * beta_o * (1 - beta_o) * (1 - beta_i) * beta_i
                + theta4 * beta_o ** 2 * (1 - beta_o) * beta_i
                + theta5 * beta_o ** 2 * (1 - beta_o) ** 2 )  ;

c_0001 = &l * (theta1 * (1 - beta_i) ** 3 * beta_i
                + theta2 * beta_o * (1 - beta_i) ** 2 * beta_i
                + theta3 * beta_o ** 2 * (1 - beta_i)* beta_i
                + theta4 * beta_o ** 3 * beta_i
                + theta5 * beta_o ** 3 * (1 - beta_o) )   ;

c_0011 = &l * (theta1 * (1 - beta_i) ** 2 * beta_i ** 2
                + theta2 * beta_o * (1 - beta_i) * beta_i ** 2
                + theta3 * beta_o ** 2 * beta_i ** 2
                + theta4 * beta_o ** 2 * (1 - beta_o) * beta_i
                + theta5 * beta_o ** 2 * (1 - beta_o) ** 2 )  ;

c_1010 = &l * (theta1 * beta_i ** 2 * (1 - beta_i) ** 2
                + theta2 * (1 - beta_o) * (1 - beta_i) ** 2 * beta_i
                + theta3 * (1 - beta_o) * beta_o * beta_i * (1 - beta_i)
                + theta4 * (1 - beta_o) ** 2 * beta_o * (1 - beta_i)
                + theta5 * (1 - beta_o) ** 2 * beta_o ** 2 )  ;

c_1101 = &l * (theta1 * beta_i ** 3 * (1 - beta_i)
                + theta2 * (1 - beta_o) * beta_i ** 2 * (1 - beta_i)
                + theta3 * (1 - beta_o) ** 2 * beta_i * (1 - beta_i)
                + theta4 * (1 - beta_o) ** 2 * beta_o * beta_i
                + theta5 * (1 - beta_o) ** 3 * beta_o )  ;

c_1011 = &l * (theta1 * beta_i ** 3 * (1 - beta_i)
                + theta2 * (1 - beta_o) * (1 - beta_i) * beta_i ** 2
                + theta3 * (1 - beta_o) * beta_o * beta_i ** 2
                + theta4 * (1 - beta_o) ** 2 * beta_o * beta_i
                + theta5 * (1 - beta_o) ** 3 * beta_o )  ;
```

```
c_0010 = &l * (theta1 * (1 - beta_i) ** 3 * beta_i
              + theta2 * (1 - beta_i) ** 2 * beta_i * beta_o
              + theta3 * beta_o ** 2 * (1 - beta_i)* beta_i
              + theta4 * beta_o ** 2 * (1 - beta_i) * (1 - beta_o)
              + theta5 * beta_o ** 3 * (1 - beta_o))  ;

lik = &l* (theta1 * beta_i ** vara * (1 - beta_i) ** (1 - vara)
                  * beta_i ** varb * (1 - beta_i) ** (1 - varb)
                  * beta_i ** varc * (1 - beta_i) ** (1 - varc)
                  * beta_i ** vard * (1 - beta_i) ** (1 - vard)
          + theta2 * (1 - beta_o) ** vara * beta_o ** (1 - vara)
                   * beta_i ** varb * (1 - beta_i) ** (1 - varb)
                   * beta_i ** varc * (1 - beta_i) ** (1 - varc)
                   * beta_i ** vard * (1 - beta_i) ** (1 - vard)
          + theta3 * (1 - beta_o) ** vara * beta_o ** (1 - vara)
                   * (1 - beta_o) ** varb * beta_o ** (1 - varb)
                   * beta_i ** varc * (1 - beta_i) ** (1 - varc)
                   * beta_i ** vard * (1 - beta_i) ** (1 - vard)
          + theta4 * (1 - beta_o) ** vara * beta_o ** (1 - vara)
                   * (1 - beta_o) ** varb * beta_o ** (1 - varb)
                   * (1 - beta_o) ** varc * beta_o ** (1 - varc)
                   * beta_i ** vard * (1 - beta_i) ** (1 - vard)
          + theta5 * (1 - beta_o) ** vara * beta_o ** (1 - vara)
                   * (1 - beta_o) ** varb * beta_o ** (1 - varb)
                   * (1 - beta_o) ** varc * beta_o ** (1 - varc)
                   * (1 - beta_o) ** vard * beta_o ** (1 - vard)) ;
run ;

*computing expected frequencies using parameter estimates from pistar model ;
data pistarfreq ;
set pistar_est ;
where _type_ = 'PARMS' ;
c0000 = &l * (theta1 * (1 - beta_i) ** 4
             + theta2 * beta_o * (1 - beta_i) ** 3
             + theta3 * beta_o **2 * (1 - beta_i) ** 2
             + theta4 * beta_o ** 3 * (1 - beta_i)
             + theta5 * beta_o ** 4) ;

c1000 = &l * (theta1 * beta_i * (1 - beta_i) ** 3
             + theta2 * (1 - beta_o) * (1 - beta_i) ** 3
             + theta3 * (1 - beta_o) * beta_o * (1 - beta_i) ** 2
             + theta4 * ( 1 - beta_o) * beta_o ** 2 * ( 1 - beta_i)
             + theta5 * (1 - beta_o) * beta_o ** 3) ;

c1100 = &l * (theta1 * beta_i ** 2 * (1 - beta_i) ** 2
```

```
                  + theta2 * (1 - beta_o) * beta_i * (1 - beta_i) ** 2
                  + theta3 * (1 - beta_o) ** 2 * (1 - beta_i) ** 2
                  + theta4 * beta_o * (1 - beta_o) ** 2 * (1 - beta_i)
                  + theta5 * (1 - beta_o) ** 2 * beta_o ** 2) ;

c1110 = &l * (theta1 * beta_i ** 3 * (1 - beta_i)
                  + theta2 * (1 - beta_o) * beta_i ** 2 * (1 - beta_i)
                  + theta3 * (1 - beta_o) ** 2 * beta_i * (1 - beta_i)
                  + theta4 * (1 - beta_o) ** 3 * (1 - beta_i)
                  + theta5 * (1 - beta_o) ** 3 * beta_o) ;

c1111 = &l * (theta1 * beta_i ** 4
                  + theta2 * (1 - beta_o) * beta_i ** 3
                  + theta3 * (1 - beta_o) ** 2 * beta_i ** 2
                  + theta4 * (1 - beta_o) ** 3 * beta_i
                  + theta5 * (1 - beta_o) ** 4) ;

c0100 = &l * (theta1 * (1 - beta_i) ** 3 * beta_i
                  + theta2 * beta_o * beta_i * (1 - beta_i) ** 2
                  + theta3 * beta_o * (1 - beta_o) * (1 - beta_i) ** 2
                  + theta4 * beta_o ** 2 * (1 - beta_o) * (1 - beta_i)
                  + theta5 * beta_o ** 3 * (1 - beta_o)) ;

c0110 = &l * (theta1 * (1 - beta_i) ** 2 * beta_i ** 2
                  + theta2 * beta_o * beta_i ** 2 * (1 - beta_i)
                  + theta3 * beta_o * (1 - beta_o) * beta_i * (1 - beta_i)
                  + theta4 * beta_o * (1 - beta_o) ** 2 * (1 - beta_i)
                  + theta5 * beta_o ** 2 * (1 - beta_o) ** 2 ) ;

c0111 = &l * (theta1 * (1 - beta_i) * beta_i ** 3
                  + theta2 * beta_o * beta_i ** 3
                  + theta3 * beta_o * (1 - beta_o) * beta_i ** 2
                  + theta4 * beta_o * (1 - beta_o) ** 2 * beta_i
                  + theta5 * (1 - beta_o) ** 3 * beta_o) ;

c1001 = &l * (theta1 * beta_i ** 2 * ( 1 - beta_i) ** 2
                  + theta2 * (1 - beta_o) * (1 - beta_i) ** 2 * beta_i
                  + theta3 * (1 - beta_o) * beta_o * (1 - beta_i) * beta_i
                  + theta4 * (1 - beta_o) * beta_o ** 2 * beta_i
                  + theta5 * ( 1 - beta_o) ** 2 * beta_o ** 2 ) ;

c0101 = &l * (theta1 * beta_i ** 2 * (1 - beta_i) ** 2
                  + theta2 * beta_o * beta_i ** 2 * (1 - beta_i)
                  + theta3 * beta_o * (1 - beta_o) * (1 - beta_i) * beta_i
                  + theta4 * beta_o ** 2 * (1 - beta_o) * beta_i
                  + theta5 * beta_o ** 2 * (1 - beta_o) ** 2 )  ;
```

```
c0001 = &l * (theta1 * (1 - beta_i) ** 3 * beta_i
            + theta2 * beta_o * (1 - beta_i) ** 2 * beta_i
            + theta3 * beta_o ** 2 * (1 - beta_i)* beta_i
            + theta4 * beta_o ** 3 * beta_i
            + theta5 * beta_o ** 3 * (1 - beta_o) )  ;

c0011 = &l * (theta1 * (1 - beta_i) ** 2 * beta_i ** 2
            + theta2 * beta_o * (1 - beta_i) * beta_i ** 2
            + theta3 * beta_o ** 2 * beta_i ** 2
            + theta4 * beta_o ** 2 * (1 - beta_o) * beta_i
            + theta5 * beta_o ** 2 * (1 - beta_o) ** 2 )  ;

c1010 = &l * (theta1 * beta_i ** 2 * (1 - beta_i) ** 2
            + theta2 * (1 - beta_o) * (1 - beta_i) ** 2 * beta_i
            + theta3 * (1 - beta_o) * beta_o * beta_i * (1 - beta_i)
            + theta4 * (1 - beta_o) ** 2 * beta_o * (1 - beta_i)
            + theta5 * (1 - beta_o) ** 2 * beta_o ** 2 )  ;

c1101 = &l * (theta1 * beta_i ** 3 * (1 - beta_i)
            + theta2 * (1 - beta_o) * beta_i ** 2 * (1 - beta_i)
            + theta3 * (1 - beta_o) ** 2 * beta_i * (1 - beta_i)
            + theta4 * (1 - beta_o) ** 2 * beta_o * beta_i
            + theta5 * (1 - beta_o) ** 3 * beta_o )  ;

c1011 = &l * (theta1 * beta_i ** 3 * (1 - beta_i)
            + theta2 * (1 - beta_o) * (1 - beta_i) * beta_i ** 2
            + theta3 * (1 - beta_o) * beta_o * beta_i ** 2
            + theta4 * (1 - beta_o) ** 2 * beta_o * beta_i
            + theta5 * (1 - beta_o) ** 3 * beta_o )  ;

c0010 = &l * (theta1 * (1 - beta_i) ** 3 * beta_i
            + theta2 * (1 - beta_i) ** 2 * beta_i * beta_o
            + theta3 * beta_o ** 2 * (1 - beta_i)* beta_i
            + theta4 * beta_o ** 2 * (1 - beta_i) * (1 - beta_o)
            + theta5 * beta_o ** 3 * (1 - beta_o))  ;

sumfreq = c0000 + c1000 + c1100 + c1110 + c1111 + c0100 + c0110 + c0111
        + c1001 + c0101 + c0001 + c0011 + c1010 + c1101 + c1011 + c0010 ;
pistar = 1 - sumfreq / &l ;
run ;

*creating data sets for input into PROC UNIVARIATE ;
 %if &i = 1 %then %do ;
 data pr_univ.procpistar&m ;
 merge pistarfreq (in = a) transfreq (in = b) ;
```

```
 iter = &i ;
 sampsize = &l ;
 run ;
 %end ;

 %else %if &i > 1 %then %do ;
 data pistarobsexp ;
 merge pistarfreq (in = a) transfreq (in = b) ;
 iter = &i ;
 sampsize = &l ;
 run ;
 %end ;

 %if &i > 1 %then %do ;
 proc append base= pr_univ.procpistar&m  data = work.pistarobsexp ;
 run ;
 %end ;

 proc datasets library = work ;
 delete iosamp ;
 run ;

 %end ;

proc univariate data = pr_univ.procextend&m noprint ;
var theta1 theta2 theta3 theta4 theta5 theta6 beta_i beta_o alpha beta delta sigma  ;
output out = procextstat
mean = avgtheta1 avgtheta2 avgtheta3 avgtheta4 avgtheta5
        avgtheta6 avgbeta_i avgbeta_o avgalpha avgbeta avgdelta avgsigma ;
run ;

data pr_univ.reshape_procextend&m ;
set procextstat ;
parameter = 'theta1' ; mean = avgtheta1 ; output ;
parameter = 'theta2' ; mean = avgtheta2 ; output ;
parameter = 'theta3' ; mean = avgtheta3 ; output ;
parameter = 'theta4' ; mean = avgtheta4 ; output ;
parameter = 'theta5' ; mean = avgtheta5 ; output ;
parameter = 'theta6' ; mean = avgtheta6 ; output ;
parameter = 'intruserr' ; mean = avgbeta_i ; output ;
parameter = 'omiterr' ; mean = avgbeta_o ; output ;
parameter = 'alpha' ; mean = avgalpha ; output ;
parameter = 'beta' ; mean = avgbeta ; output ;
parameter = 'delta' ; mean = avgdelta ; output ;
parameter = 'sigma' ; mean = avgsigma ; output ;
drop avgtheta1 avgtheta2 avgtheta3 avgtheta4 avgtheta5 avgtheta6
```

```
        avgbeta_i avgbeta_o avgalpha avgbeta avgdelta avgsigma ;
run ;

proc univariate data = pr_univ.procpistar&m noprint ;
var theta1 theta2 theta3 theta4 theta5 pistar beta_i beta_o ;
output out = procpistat
mean = avgtheta1 avgtheta2 avgtheta3 avgtheta4 avgtheta5
             avgpistar avgbeta_i avgbeta_o ;
run ;

data pr_univ.reshape_pistar&m ;
set procpistat ;
parameter = 'theta1' ; mean = avgtheta1 ; output ;
parameter = 'theta2' ; mean = avgtheta2 ; output ;
parameter = 'theta3' ; mean = avgtheta3 ; output ;
parameter = 'theta4' ; mean = avgtheta4 ; output ;
parameter = 'theta5' ; mean = avgtheta5 ; output ;
parameter = 'intruserr' ; mean = avgbeta_i  ; output ;
parameter = 'omiterr' ; mean = avgbeta_o ; output ;
parameter = 'pistar' ; mean = avgpistar ; output ;

drop avgtheta1 avgtheta2 avgtheta3 avgtheta4 avgtheta5
        avgpistar avgbeta_i avgbeta_o ;
run ;

%end ;

%mend iosim ;

%iosim
```

```
*SAS program for computing Goodman model estimates for Proctor-based universe ;
libname gduniv 'c:\nlp programs\proctor4\g20_iuc40_e_e' ;
filename pr_log 'c:\nlp programs\proctor4\g20_iuc40_e_e\proclog.txt' ;
filename pr_out 'c:\nlp programs\proctor4\g20_iuc40_e_e\procout.txt' ;

proc printto log = pr_log print = pr_out ;
run ;

data good4set1 ;
pi_error = 0.20 ;
 do j = 1 to 100000 ;
  latnum = ranuni(0) ;
   if 0 lt latnum le 0.12 then do ;
    latclass = 1 ;
    varanum = ranuni(0) ;
    varbnum = ranuni(0) ;
    varcnum = ranuni(0) ;
    vardnum = ranuni(0) ;
     if 0 lt varanum le pi_error then vara = 1 ;
      else if pi_error lt varanum le 1.00000 then vara = 0 ;
     if 0 lt varbnum le pi_error then varb = 1 ;
      else if pi_error lt varbnum le 1.00000 then varb = 0 ;
     if 0 lt varcnum le pi_error then varc = 1 ;
      else if pi_error lt varcnum le 1.00000 then varc = 0 ;
     if 0 lt vardnum le pi_error then vard = 1 ;
      else if pi_error lt vardnum le 1.00000 then vard = 0 ;
   end ;

   else if 0.12 lt latnum le 0.24 then do ;
    latclass = 2 ;
    varanum = ranuni(0) ;
    varbnum = ranuni(0) ;
    varcnum = ranuni(0) ;
    vardnum = ranuni(0) ;
     if 0 lt varanum le (1 - pi_error) then vara = 1 ;
      else if (1 - pi_error) lt varanum le 1.00000 then vara = 0 ;
     if 0 lt varbnum le pi_error then varb = 1 ;
      else if pi_error lt varbnum le 1.00000 then varb = 0 ;
     if 0 lt varcnum le pi_error then varc = 1 ;
      else if pi_error lt varcnum le 1.00000 then varc = 0 ;
     if 0 lt vardnum le pi_error then vard = 1 ;
      else if pi_error lt vardnum le 1.00000 then vard = 0 ;
   end ;

   else if 0.24 lt latnum le 0.36 then do ;
    latclass = 3 ;
```

```
  varanum = ranuni(0) ;
  varbnum = ranuni(0) ;
  varcnum = ranuni(0) ;
  vardnum = ranuni(0) ;
    if 0 lt varanum le (1 - pi_error) then vara = 1 ;
      else if (1 - pi_error) lt varanum le 1.00000 then vara = 0 ;
    if 0 lt varbnum le (1 - pi_error)  then varb = 1 ;
      else if (1 - pi_error) lt varbnum le 1.00000 then varb = 0 ;
    if 0 lt varcnum le pi_error then varc = 1 ;
      else if pi_error lt varcnum le 1.00000 then varc = 0 ;
    if 0 lt vardnum le pi_error then vard = 1 ;
      else if pi_error lt vardnum le 1.00000 then vard = 0 ;
end ;

else if 0.36 lt latnum le 0.48 then do ;
  latclass = 4 ;
  varanum = ranuni(0) ;
  varbnum = ranuni(0) ;
  varcnum = ranuni(0) ;
  vardnum = ranuni(0) ;
    if 0 lt varanum le (1 - pi_error) then vara = 1 ;
      else if (1 - pi_error) lt varanum le 1.00000 then vara = 0 ;
    if 0 lt varbnum le (1 - pi_error) then varb = 1 ;
      else if (1 - pi_error) lt varbnum le 1.00000 then varb = 0 ;
    if 0 lt varcnum le (1 - pi_error) then varc = 1 ;
      else if (1 - pi_error) lt varcnum le 1.00000 then varc = 0 ;
    if 0 lt vardnum le pi_error then vard = 1 ;
      else if pi_error lt vardnum le 1.00000 then vard = 0 ;
end ;

else if 0.48 lt latnum le 0.60 then do ;
  latclass = 5 ;
  varanum = ranuni(0) ;
  varbnum = ranuni(0) ;
  varcnum = ranuni(0) ;
  vardnum = ranuni(0) ;
    if 0 lt varanum le (1 - pi_error) then vara = 1 ;
      else if (1 - pi_error) lt varanum le 1.00000 then vara = 0 ;
    if 0 lt varbnum le (1 - pi_error) then varb = 1 ;
      else if (1 - pi_error) lt varbnum le 1.00000 then varb = 0 ;
    if 0 lt varcnum le (1 - pi_error) then varc = 1 ;
      else if (1 - pi_error) lt varcnum le 1.00000 then varc = 0 ;
    if 0 lt vardnum le (1 - pi_error) then vard = 1 ;
      else if (1 - pi_error) lt vardnum le 1.00000 then vard = 0 ;
end ;
```

```
    else if 0.60 lt latnum le 1.0000 then do ;
     latclass = 6 ;
     varanum = ranuni(0) ;
     varbnum = ranuni(0) ;
     varcnum = ranuni(0) ;
     vardnum = ranuni(0) ;
     if varanum le 0.30 then vara = 1 ;
      else if varanum gt 0.30 then vara = 0 ;
     if varbnum le 0.30 then varb = 1 ;
      else if varbnum gt 0.30 then varb = 0 ;
     if varcnum le 0.30 then varc = 1 ;
      else if varcnum gt 0.30 then varc = 0 ;
     if vardnum le 0.30 then vard = 1 ;
      else if vardnum gt 0.30 then vard = 0 ;
    end ;
   output ;
  end ;
run ;

data resppat  ;
input vara varb varc vard pattern  ;
cards ;
1  1  1  1  1   1
1  1  1  0  2   2
1  1  0  1  6   6
1  1  0  0  3   3
1  0  1  1  6   7
1  0  1  0  6   8
1  0  0  1  6   9
1  0  0  0  4   4
0  1  1  1  6  10
0  1  1  0  6  11
0  1  0  1  6  12
0  1  0  0  6  13
0  0  1  1  6  14
0  0  1  0  6  15
0  0  0  1  6  16
0  0  0  0  5   5   ;
run ;

%macro gdmnrep ;

%do m = 1 %to 3 ;
%let l = %eval(120 * 2 ** &m) ;

%do j = 1 %to 1000 ;
```

```
proc surveyselect data = good4set1 method = urs n = &l out = gdmansamp  outhits ;
run ;

proc freq data = gdmansamp noprint  ;
  tables vara * varb * varc * vard / sparse out = goodmancnt (drop = percent)  ;
run ;

proc sort data = goodmancnt ;  by vara varb varc vard ;
run ;

proc sort data = resppat ;  by vara varb varc vard ;
run ;

data mergpattern ;
merge goodmancnt (in = a) resppat (in = b) ;  by vara varb varc vard ;
run ;

*create a one-record data set containing the observed frequencies ;
data transfreq ;
array obsfreq{16} d0000 d0001 d0010 d0011 d0100  d0101 d0110 d0111
                  d1000 d1001 d1010 d1011 d1100 d1101 d1110 d1111 ;
  do m = 1 to 16 ;
  set mergpattern ;
    if count > 0 then obsfreq{m} = count ;
      else if count = 0 then obsfreq{m} = 1 ;
      *in case of sampling zeros, Pan (2006) suggests using 1 as a flattening constant  ;
  end ;
  drop vara varb varc vard m count pattern ;
run ;

*determine maximum likelihood estimates ;
proc nlp data = mergpattern tech = quanew outest = goodmanest ;
where pattern not in (1,2,3,4,5) ;
max loglik ;
parms mu = 2.2 ,
      lambda1 = 0.10 ,
      lambda2 = -0.10 ,
      beta1 = -0.10 ,
      beta2 = 0.10 ,
      gamma1 = -0.20 ,
      gamma2 = 0.20 ,
      delta1 = -0.15 ,
      delta2 = 0.15 ;

lincon lambda1 + lambda2 = 0 ,
      beta1 + beta2 = 0 ,
```

```
        gamma1 + gamma2 = 0 ,
        delta1 + delta2 = 0 ;

loglik = count * (mu + lambda1*vara + lambda2*(1-vara)
                        + beta1*varb + beta2*(1-varb)
                        + gamma1*varc + gamma2*(1-varc)
                        + delta1*vard + delta2*(1-vard))
             - exp(mu + lambda1*vara + lambda2*(1-vara)
                        + beta1*varb + beta2*(1-varb)
                        + gamma1*varc + gamma2*(1-varc)
                        + delta1*vard + delta2*(1-vard)) ;
run;

*calculate conditional probabilities for unscalable class ;
data cond_probs ;
set goodmanest ;
where _TYPE_ = 'PARMS' ;
condprob_a = exp(lambda1) / (exp(lambda1) + exp(lambda2)) ;
condprob_b = exp(beta1) / (exp(beta1) + exp(beta2)) ;
condprob_c = exp(gamma1) / (exp(gamma1) + exp(gamma2)) ;
condprob_d = exp(delta1) / (exp(delta1) + exp(delta2)) ;

unscalable_condprod1 = condprob_a * condprob_b * condprob_c * condprob_d ;
unscalable_condprod2 = condprob_a * condprob_b * condprob_c * (1-condprob_d) ;
unscalable_condprod3 = condprob_a * condprob_b * (1-condprob_c) * (1-condprob_d) ;
unscalable_condprod4 = condprob_a * (1-condprob_b) * (1-condprob_c)
                           * (1-condprob_d) ;
unscalable_condprod5 = (1-condprob_a) * (1-condprob_b) * (1-condprob_c)
                           * (1-condprob_d) ;

unscalable_condprod_sum = unscalable_condprod1 + unscalable_condprod2
                              + unscalable_condprod3 + unscalable_condprod4
                              + unscalable_condprod5 ;
run ;

%if &j = 1 %then %do ;

data gduniv.goodmanstat&m  ;
array obscnt{16} d0000 d0001 d0010 d0011 d0100  d0101 d0110 d0111
                 d1000 d1001 d1010 d1011 d1100 d1101 d1110 d1111 ;
array expcnt{16} c0000 c0001 c0010 c0011 c0100  c0101 c0110 c0111
                 c1000 c1001 c1010 c1011 c1100 c1101 c1110 c1111 ;
merge cond_probs (in = a)  transfreq (in = b)  ;
iter = &j ;
obs_sum = d0000 + d0001 + d0010 + d0011 + d0100 + d0101 + d0110 + d0111 +
            d1000 + d1001 + d1010 + d1011 + d1100 + d1101 +  d1110 + d1111 ;
```

```
obs_prop1 = d1111 / obs_sum ;
obs_prop2 = d1110 / obs_sum ;
obs_prop3 = d1100 / obs_sum ;
obs_prop4 = d1000 / obs_sum ;
obs_prop5 = d0000 / obs_sum ;
Guttman_prop_sum = obs_prop1 + obs_prop2 + obs_prop3 + obs_prop4 + obs_prop5  ;

mix_prop0 = (1 - Guttman_prop_sum) / (1 - unscalable_condprod_sum) ;
mix_prop1 = obs_prop1 - mix_prop0 * unscalable_condprod1 ;
mix_prop2 = obs_prop2 - mix_prop0 * unscalable_condprod2 ;
mix_prop3 = obs_prop3 - mix_prop0 * unscalable_condprod3 ;
mix_prop4 = obs_prop4 - mix_prop0 * unscalable_condprod4 ;
mix_prop5 = obs_prop5 - mix_prop0 * unscalable_condprod5 ;

c1101  = exp(mu + lambda1 + beta1 + gamma2 + delta1) ;
c1011 = exp(mu + lambda1 + beta2 + gamma1 + delta1) ;
c1010 = exp(mu + lambda1 + beta2 + gamma1 + delta2) ;
c1001 = exp(mu + lambda1 + beta2 + gamma2 + delta1) ;
c0111 = exp(mu + lambda2 + beta1 + gamma1 + delta1) ;
c0110 = exp(mu + lambda2 + beta1 + gamma1 + delta2) ;
c0101 = exp(mu + lambda2 + beta1 + gamma2 + delta1) ;
c0100 = exp(mu + lambda2 + beta1 + gamma2 + delta2) ;
c0011 = exp(mu + lambda2 + beta2 + gamma1 + delta1) ;
c0010 = exp(mu + lambda2 + beta2 + gamma1 + delta2) ;
c0001 = exp(mu + lambda2 + beta2 + gamma2 + delta1) ;
c1111 = d1111 ;
c1110 = d1110 ;
c1100 = d1100 ;
c1000 = d1000 ;
c0000 = d0000 ;
run ;
%end ;

%else %if &j > 1 %then %do ;
data goodmanparms  ;
array obscnt{16} d0000 d0001 d0010 d0011 d0100  d0101 d0110 d0111
                 d1000 d1001 d1010 d1011 d1100 d1101  d1110 d1111 ;
array expcnt{16} c0000 c0001 c0010 c0011 c0100  c0101 c0110 c0111
                 c1000 c1001 c1010 c1011 c1100 c1101  c1110 c1111 ;
merge cond_probs (in = a)  transfreq (in = b)  ;
iter = &j ;
obs_sum = d0000 + d0001 + d0010 + d0011 + d0100 + d0101 + d0110 + d0111 +
          d1000 + d1001 + d1010 + d1011 + d1100 + d1101 +  d1110 + d1111 ;

obs_prop1 = d0000 / obs_sum ;
obs_prop2 = d1000 / obs_sum ;
```

```
obs_prop3 = d1100 / obs_sum ;
obs_prop4 = d1110 / obs_sum ;
obs_prop5 = d1111 / obs_sum ;
Guttman_prop_sum = obs_prop1 + obs_prop2 + obs_prop3 + obs_prop4 + obs_prop5  ;

mix_prop0 = (1 - Guttman_prop_sum) / (1 - unscalable_condprod_sum) ;
mix_prop1 = obs_prop1 - mix_prop0 * unscalable_condprod1 ;
mix_prop2 = obs_prop2 - mix_prop0 * unscalable_condprod2 ;
mix_prop3 = obs_prop3 - mix_prop0 * unscalable_condprod3 ;
mix_prop4 = obs_prop4 - mix_prop0 * unscalable_condprod4 ;
mix_prop5 = obs_prop5 - mix_prop0 * unscalable_condprod5 ;

c1101 = exp(mu + lambda1 + beta1 + gamma2 + delta1) ;
c1011 = exp(mu + lambda1 + beta2 + gamma1 + delta1) ;
c1010 = exp(mu + lambda1 + beta2 + gamma1 + delta2) ;
c1001 = exp(mu + lambda1 + beta2 + gamma2 + delta1) ;
c0111 = exp(mu + lambda2 + beta1 + gamma1 + delta1) ;
c0110 = exp(mu + lambda2 + beta1 + gamma1 + delta2) ;
c0101 = exp(mu + lambda2 + beta1 + gamma2 + delta1) ;
c0100 = exp(mu + lambda2 + beta1 + gamma2 + delta2) ;
c0011 = exp(mu + lambda2 + beta2 + gamma1 + delta1) ;
c0010 = exp(mu + lambda2 + beta2 + gamma1 + delta2) ;
c0001 = exp(mu + lambda2 + beta2 + gamma2 + delta1) ;
c1111 = d1111 ;
c1110 = d1110 ;
c1100 = d1100 ;
c1000 = d1000 ;
c0000 = d0000 ;
run ;

proc append base = gduniv.goodmanstat&m data = goodmanparms ;
run ;

%end ;

proc datasets library = work  ; delete goodmnsmp ;
quit ;

%end ;

%end ;

%mend gdmnrep ;

%gdmnrep
```

REFERENCES

Agresti, A. (1990). *Categorical data analysis*. New York, NY: John Wiley & Sons.

Clogg, C. C., & Sawyer, D. O. (1981). *A comparison of alternative models for analyzing the scalability of response patterns*. In S. Leinhardt (ed.). Sociological Methodology. San Francisco: Josey-Bass.

Dayton, C. M. (1998). *Latent class scaling analysis*. Thousand Oaks, CA: Sage Publications, Inc.

Dayton, C. M. (2003). Applications and computational strategies for the two-point mixture index of fit. *British Journal of Mathematical and Statistical Psychology, 56,* 1-13.

Dayton, C. M., & Macready, G. B. (1976). A probabilistic model for validation of behavorial hierarchies. *Psychometrika, 41*, 189-204.

Dayton, C. M., & Macready, G. B. (1980). A scaling model with response errors and intrinsically unscalable respondents. *Psychometrika, 45,* 343-356.

Dayton, C. M., & Macready, G. B. (2007). *Latent class analysis in psychometrics*. Handbook of Statistics, 26, 421-446.

de Menezes, L. M. (1999). On fitting latent class models for binary data: The estimation of standard errors. *British Journal of Mathematical and Statistical Psychology, 52,* 149-168.

Elley, W. B. (1992). *How in the world do students read?* IEA Study of Reading Literacy. The Hague: IEA.

Formann, A. K. (2000). Rater agreement and the generalized Rudas-Clogg-Lindsay index of fit. *Statistics in Medicine, 19*, 1881-1888.

Formann, A. K. (2006). Testing the Rasch model by means of the mixture fit index. *British Journal of Mathematical and Statistical Psychology, 59,* 89-95.

Goodman, L. A. (1975). A new model for scaling response patterns: An application of the quasi-independence concept. *Journal of the American Statistical Association, 70,* 755-768.

Guttman, L. (1947). On Festinger's evaluation of scale analysis. *Psychological Bulletin, 44,* 451-465.

Hayek, L. C. (1978). Properties of maximum likelihood estimators for a class of probabilistic models. Unpublished doctoral dissertation, University of Maryland.

Holt, J. A., & Macready, G. B. (1989). A simulation study of the difference chi-square statistic for comparing latent class models under violation of regularity conditions. *Applied Psychological Measurement, 13,* 221-231.

Pan, X. (2006). Factors influencing the mixture index of model fit in contingency tables showing independence. Unpublished doctoral dissertation, University of Maryland.

Proctor, C. H. (1970). A probabilistic formulation and statistical analysis of Guttman scaling. *Psychometrika, 35,* 73-78.

Read, T. R. C., & Cressie, N. A. C. (1988). *Goodness of fit statistics for discrete multivariate analysis.* New York: Springer.

Rudas, T. (1999). The mixture index of fit and minimax regression. *Metrika, 50,* 163-172.

Rudas, T. (2002). *A latent class approach to measuring the fit of a statistical model.* Invited chapter in Hagenaars, J. A., & McCutcheon, A. L. (eds.). Applied Latent Class Analysis. Cambridge, United Kingdom: Cambridge University Press.

Rudas, T., Clogg, C. C., & Lindsay, B. G. (1994). A new index of fit based on mixture methods for the analysis of contingency tables. *Journal of the Royal Statistical Society, Series B, 56,* 623-639.

Rudas, T., & Zwick, R. (1997). Estimating the importance of differential variable functioning. *Journal of Educational and Behavioral Sciences, 22,* 31-45.

SAS Institute Inc. (2004). SAS/OR User's Guide, Version 9. Cary, NC: SAS Institute Inc.

Vermunt, J. K. (1997). The LEM User Manual. WORC Paper. Tilburg University, The Netherlands.

Verdes, E., & Rudas, T. (2002). The π^* index as a new alternative for assessing goodness of fit of logistic regression.

Xi, L. (1994). The mixture index of fit for the independence model in contingency tables. Master of Arts paper, Department of Statistics, Pennsylvania State University.

Xi, L., & Lindsay, B. G. (1996). A note on calculating the π^* index of fit for the analysis of contingency tables. *Sociological Methods & Research, 25,* 248-259.