THE ROLE OF MODEL COMPLEXITY IN THE EVALUATION OF STRUCTURAL EQUATION MODELS

DISSERTATION

Presented in Partial Fulfillment of the Requirements for

the Degree Doctor of Philosophy in the Graduate

School of The Ohio State University

By

Kristopher J. Preacher, M.A.

*****

The Ohio State University
2003

Dissertation Committee

Dr. Robert C. MacCallum, Advisor

Dr. Michael W. Browne

Dr. In Jae Myung

Approved by

_____

Advisor
Department of Psychology

ABSTRACT

This dissertation represents an investigation into the role of model complexity in structural equation modeling (SEM) and how traditional notions of model fit, which do not typically consider complexity, are inadequate for summarizing the success or failure of a model. Model fit is traditionally summarized in an index which communicates the agreement between a given model and a particular data set. However, demonstrating that a given model could have generated a given data set is not sufficient to show that it is a good model. Because complexity partially determines the ability of a model to fit data, model complexity should not be ignored when evaluating model fit. The importance of model complexity is examined in the SEM context by simulating correlation matrices and applying a series of models which differ in complexity. First, it is shown that models differing in the number of free parameters can fit the same random data differentially well by a simple criterion of good fit. Second, models with the same number of free parameters, but which differ in functional form, are found to fit the same data differentially well. Third, it is found that restrictions placed on parameter range have an impact on model complexity. Because traditional fit indices usually correct for model complexity due only to the number of free parameters, these findings have important implications for how models are assessed and compared in the SEM paradigm.

Dedicated to my parents.

ACKNOWLEDGMENTS

Despite the single name on the title page, dissertations are never completed in isolation. I wish to thank Woojae Kim for suggesting the MCMC approach and helping me understand how the Metropolis-Hastings algorithm works. I also want to thank Guangjian Zhang for helping me specify the multitrait-multimethod models used as examples in Chapter 4.

Many other friends and colleagues at Ohio State have helped me over the years, and they have my sincere gratitude. They include Hal Arkes, Angela Bassett, Nancy Briggs, Jinyan Fan, Andrew Hayes, Janice Kiecolt-Glaser, Cheongtag Kim, Barbara Mellers, Thomas Nygren, Derek Rucker, Yong Su, Yuri Tada, Krishna Tateneni, Philip Tetlock, Trisha Van Zandt, Michael Walker, Rebecca White, Aaron Wichman, and Shaobo Zhang. If I have inadvertently omitted anyone, please just write your name at the bottom of the page.

I particularly appreciate the guidance and wisdom of Michael Browne and In Jae Myung, who have taught me much over the last five years. Above all, I thank Bud MacCallum, whose guidance, patience, and creativity were fundamental to the completion of this work.

VITA

March 1, 1974 ..... Born – Tallahassee, Florida

1996 ..... B.A. Psychology, North Carolina State University

1998 ..... M.A. Psychology, The College of William and Mary

1998 – present ..... Graduate Teaching and Research Associate and Statistical Consultant, The Ohio State University

PUBLICATIONS

Preacher, K. J. (2003). Publishing in graduate school: Tips for new graduate students. *APS Observer*, *16*(4), pp. 33-34, 50.

Preacher, K. J., & MacCallum, R. C. (2003). Repairing Tom Swift's electric factor analysis machine. *Understanding Statistics*, *2*(1), 13-32.

MacCallum, R. C., Browne, M. W., & Preacher, K. J. (2002). Comments on the Meehl-Waller procedure for appraisal of path analysis models. *Psychological Methods*, *7*(3), 301-306.

MacCallum, R. C., Zhang, S., Preacher, K. J., & Rucker, D. D. (2002). On the practice of dichotomization of quantitative variables. *Psychological Methods*, *7*(1), 19-40.

Preacher, K. J., & MacCallum, R. C. (2002). Exploratory factor analysis in behavior genetics research: Factor recovery with small sample sizes. *Behavior Genetics*, *32*(2), 153-161.

Preacher, K. J., & Rucker, D. D. (2002). Dumbing it down: The dangers of appealing to the lowest common denominator. *Dialogue*, *17*(1), 26-27, 31.

Cunningham, W. A., Preacher, K. J., & Banaji, M. R. (2001). Implicit attitude measures: Consistency, stability, and convergent validity. *Psychological Science*, *12*(2), 163-170.

MacCallum, R. C., Widaman, K. F., Preacher, K. J., & Hong, S. (2001). Sample size in factor analysis: The role of model error. *Multivariate Behavioral Research*, *36*(4), 611-637.

FIELDS OF STUDY

Major Field: Psychology

Minor Field: Social Psychology

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 The Use of Models in Psychology

In the social sciences, models are constructed to represent theories. In accordance with the rigor required of scientific publications, models are often described mathematically. A properly constructed model, then, can serve as a surrogate for a theory for the purposes of scientific inquiry and prediction. Typically, many theories may be posited to account for observed phenomena, and many possible models could be specified to represent a given theory (Pitt & Myung, 2002). Theories and models vary with respect to quality. According to Popper (1959), a good theory is characterized by *falsifiability*. That is, it must be possible to obtain data inconsistent with the theory or a model is of no practical use. Falsifiability tends to be viewed as a dichotomy, but it is probably more accurate to think of it as a continuum. The gathering of empirical data is essentially a search for outcomes which could potentially falsify the model of interest. If no such data are obtained, the model is regarded as supported because it was not falsified in conditions under which it reasonably could be expected to fail. If a model is falsified by statistical means, the theory it represents is discarded or modified.

Falsification of competing theories (*elimination of alternatives*) was an approach advocated by Francis Bacon to lend support to a theory. The reasoning behind this strategy is that, given a set of models, all of which are wrong to some degree, one model is probably closest to the truth. After eliminating models further from the truth, the retained model can then be subjected to further testing and comparison. Combining the notion of falsifiability with the strategy of eliminating alternatives – that is, simultaneously testing predictions of several competing models against observed data to arrive at the most appropriate model – is called *strong inference* by Platt (1964). This strategy is regarded as a viable approach to determining the adequacy of theories, and has for decades governed how scientific models are evaluated. Comparisons of competing models are unfortunately rare, however; most studies testing theoretical predictions involve evaluation of a single model in isolation.

1.2 A Standard Approach to Model Evaluation and Comparison

The usual approach to model evaluation and comparison in the social sciences proceeds in steps: (1) specify a model implied by theory, (2) collect data from a representative sample, (3) fit the model to the data, and (4) evaluate the model, in light of the observed data, relative to some fixed criterion or relative to another model. Evaluation of a model relative to a fixed criterion (*model fit*) is usually addressed by examining a summary statistic or *fit index*, for example R^2 in multiple linear regression (MLR), residuals computed by comparing observed data to those implied by the model under investigation, or any of several χ^2-based indices in structural equation modeling

(SEM). If the match between the model's predictions and the observed data is deemed adequate by the chosen fit criterion, the model is said to have a *good fit* to the data, an indication that the theory represented by the model has received some degree of support.

After a model has been shown to serve as a useful approximation to reality (i.e., that it fits well to real data), it is often of interest to compare it to other models which are specified in an attempt to explain the same phenomena. Evaluation of one model relative to another model is called *model selection*, the goal of which is "to maximize the predictive accuracy of the best-fitting cases drawn from rival models" (Forster, 2001, p. 101).[1] Model selection is usually accomplished by comparing fit indices across models. Assessment of the overall quality of a model is based on some combination of parameter estimates, goodness of fit, parsimony, substantive interpretability, faithfulness to theory, explanatory adequacy, generalizability, ability to generate new research, and the model's performance relative to other models intended to account for the same phenomena (Cutting, Bruno, Brady, & Moore, 1992; Marsh & Balla, 1994; Myung & Pitt, in press).

1.3 Model Complexity

1.3.1 How Informative is Good Fit?

Myung (2000) and Rodgers and Rowe (2002) describe good model fit as a necessary but not sufficient condition for model selection. All models are able to fit data to some degree, however small, simply by virtue of having parameters that are free to vary across some range. Even models with all parameters constrained to equal particular

[1] For excellent reviews of model selection in the social sciences, see Myung, Forster, and Browne (2000) and Zellner, Keuzenkamp, and McAleer (2001).

values have the potential to fit some data sets well. The ability of a model to fit diverse data patterns, all things being equal, is termed *model complexity* or *model flexibility*, which is related to what Cutting (2000), Cutting et al. (1992), and Li, Lewandowsky, and DeBrunner (1996) call *scope*[2] and what Bamber and van Santen (2000) call *prediction range*). When complexity is not considered in the model-fitting process, researchers risk *overfitting*; that is, a complex model may fit a given data set very well because it easily fits error (noise) in addition to systematic variance (regularity), which prevents it from generalizing to other samples easily (Burnham & Anderson, 2002; Myung, 2000; Myung & Pitt, in press; Pitt & Myung, 2002; Roberts & Pashler, 2000; Schaffer, 1993; Sober, 2001). Grünwald (2000) notes,

> If you *overfit*, you think you know more than you really know. If you *underfit*, you do not know much but you know that you do not know much. In this sense, *underfitting* is relatively harmless while *overfitting* is dangerous. (p. 148)

Generalizability (what Cutting et al., 1992 call *selectivity* and Forster, 2001 calls *predictive accuracy*) is the ability of a model to successfully fit new data sets, and is one mark of a quality model. Generalizability is usually operationalized by combining goodness of fit with some measure of complexity (Pitt & Myung, 2002). Generalizability can also be thought of as a model's ability to fit regularity in data (Myung & Pitt, in

[2] *Scope* is perhaps a better word to use for this concept than *complexity*. A lay interpretation of *complex* implies that something is difficult to understand. In fact, complex models may be quite simple to understand.

press). Unfortunately, generalizability and goodness of fit, although both desirable characteristics of a model, tend to be negatively related (Myung, Balasubramanian, & Pitt, 2000). Selecting one model from among a set of competing alternatives solely on the basis of superior fit may bias unwary researchers in favor of more complex, yet less generalizable, models (Browne & Cudeck, 1993; Collyer, 1985; Cutting, 2000; Forster & Sober, 1994; Jeffreys, 1961; Pitt & Myung, 2002; Pitt, Myung, & Zhang, 2002). Thus, model selection may be affected by capitalization on chance when based solely on good fit. One suggested approach to assessing model quality in the face of the trade-off between generalizability and goodness of fit is to seek to obtain the best fit to data without overfitting (Grünwald, 2000; Leahy, 1994; Myung et al., 2000; Myung & Pitt, in press), an idea which has led to the development of many fit indices which involve penalties for model complexity.

Roberts and Pashler (2000) question the usefulness of good fit by describing several problems with traditional notions of model fit related to complexity. First, they point out that demonstrating that a model could have generated a given data set is not sufficient to show that the model is good. Demonstrating that a model is falsifiable and then failing to find falsifying data is insufficient to show that the model has value. "Without knowing how much a theory constrains possible outcomes," they say, "you cannot know how impressed to be when observation and theory are consistent" (p. 359). Put another way, "Consistency of theory and data is meaningful only if inconsistency was plausible" (Roberts & Pashler, 2002, p. 605). In other words, if a model could predict a large proportion of the *data space* (all possible observable data patterns; what Bamber &

van Santen, 2000 call the *outcome space*), it is not of much practical use (Myung, 2000) – perhaps the model could have fit other data patterns equally well or even better. Parallel distributed processing (connectionist) models, for example, are frequently criticized on the grounds that they can fit nearly any pattern of plausible results (Leahy, 1994), and thus are difficult to falsify. Unfortunately, many traditional measures of model fit do not take complexity into account, or do so only crudely.

Second, the variability of the data is commonly ignored. That is, if the data tightly constrain the range of possible model parameters (i.e., if N is relatively large), we should be more impressed with good model fit than if the parameter values are relatively unconstrained. In other words, if a number of competing models could just as easily have produced the data in hand, we should be less inclined to place our confidence in the truth of a particular model, even if it shows good fit by traditional standards; in this case it is unclear how much observed consistency between the data and the model (i.e., good fit) is worth. Variability of the data is closely related to the concept of statistical power – in some modeling contexts, large samples are required to reject false models, so small (highly variable) samples may lead to Type II error, or the retention of inadequate models. Again, most traditional fit statistics ignore the variability of the data.

A third problem with the traditional approach to evaluating model fit is that the a priori likelihood that a model will fit, regardless of the theory's truth, is usually ignored. Historically, models gain more support by correctly predicting surprising outcomes than

by correctly predicting expected outcomes.[3] However, in psychology, few outcomes are surprising or even unexpected. Psychological research is full of smooth, simple functions that make unsurprising predictions. It would not be surprising to find that most models used in psychology, even if they exhibit good fit, could fit nearly any data *considered plausible* a priori. For example, if theory predicts a linear increase in some variable, and linearly increasing data are indeed obtained, a linear regression model will fit well regardless of the degree of increase (slope). Furthermore, Roberts and Pashler point out, a line fit to half the data will very likely fit the other half just as well.

Roberts and Pashler illustrated their first two points with a schematic diagram similar to that in Figure 1.1. In Figure 1.1, the horizontal and vertical axes of each panel represent the full range of possible values of Measures A and B, respectively, describing the data space of interest. For example, the axes could represent the possible values of two correlation coefficients within a larger correlation matrix. The grey area describes a region of the data space consistent with a given theory. The dot represents a given data point (e.g., a pair of observed correlation coefficients), and the cross-hairs represent variability of the data (precision or, conversely, error in the data). Only in the upper left panel can a convincing case be made for the model under investigation: (1) the variability of the data is minimal, such that only a small set of models could have generated them, (2) the proportion of the data space consistent with theory is highly constrained, and (3) the observed data fall within this region. In the upper right panel, theory allows for a narrow range of predictions, but the variability of the data is such that it is difficult to say

[3] On the other hand, in their comment on Roberts and Pashler (2000), Rodgers and Rowe (2002) point out that a good theory is one that "had better predict a few interpretable and unsurprising results first" (p. 602).

how firmly the data agree with theory; i.e., good fit in this instance could be due to good luck, and attempts to replicate that good fit with a second sample would have a low probability of success. The lower left panel suffers from the opposite problem. That is, the variability of the data tightly constrains possible model parameters, but the model is so flexible that it could explain almost any plausible data equally as well or better (this is an example of overfitting). The bottom right panel suffers from both problems. That is, the variability of the data is such that repeated studies would likely yield data outside the range fit well by the model, and the model is so complex that it could have generated almost any plausible data set. In all but the upper left panel, good model fit by traditional standards could potentially mislead the researcher into thinking that model quality is high.

One of Roberts and Pashler's (2000) main points was simply that in order to make a fully informed judgment about the quality of a particular model, or about the difference in quality between two alternative models, it is necessary to examine their a priori abilities to fit any given data. For example, if three models (A, B, and C) are to be compared using observed data, it would be sensible to first consider whether the models are expected to fit sizeable or relatively small regions of the data space. It would also be interesting to know whether the regions of the data space fit well by the three models overlap to any degree. Perhaps Models A and C make very similar predictions in the sense that the regions of the data space they fit well overlap to a considerable degree

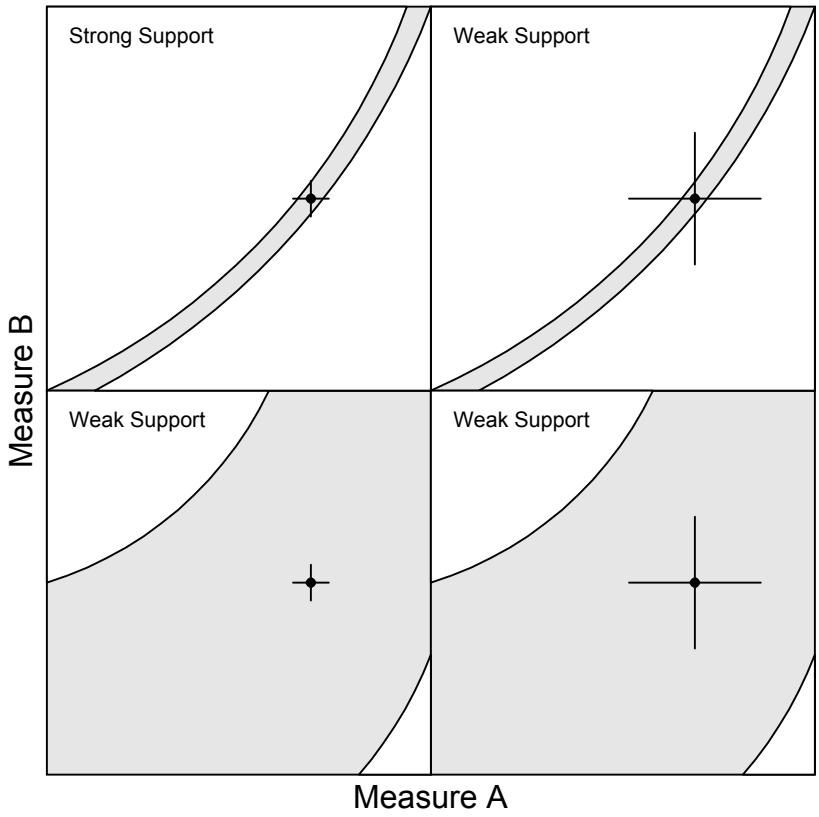


Figure 1.1: Possible relationships between theory and data (after Roberts & Pashler, 2000)

(see the Venn diagram in Figure 1.2).[4] Perhaps Model B, in turn, makes predictions so narrow and unlikely that it would be virtually impossible to find *any* data to which the model shows good fit. In this situation, empirical support would be very difficult to find for Model B (although quite impressive, should it be found), whereas it would be relatively easy to find data in support of Models A, C, or both A and C, *completely by chance*. These different a priori likelihoods of finding good fit should inform the process of model evaluation and comparison.

[4] Venn diagrams were first used in this manner as a didactic tool to explain relative model complexity (or scope) by Cutting (2000, pp. 4-5) and Myung (2000) in the context of model comparison.

In summary, because researchers continue to focus on traditional model fit indices as the sole or primary source of information about the quality of a model, three issues that should be considered important are consistently ignored: (1) what the theory predicts, (2) the variability of the data, and (3) the a priori likelihood that the theory will fit any plausible data. It is with the first and third problems noted by Roberts and Pashler (2000) – failing to determine the extent to which a model is falsifiable and failing to determine the a priori ability of a theory to account for observed data – that this dissertation is

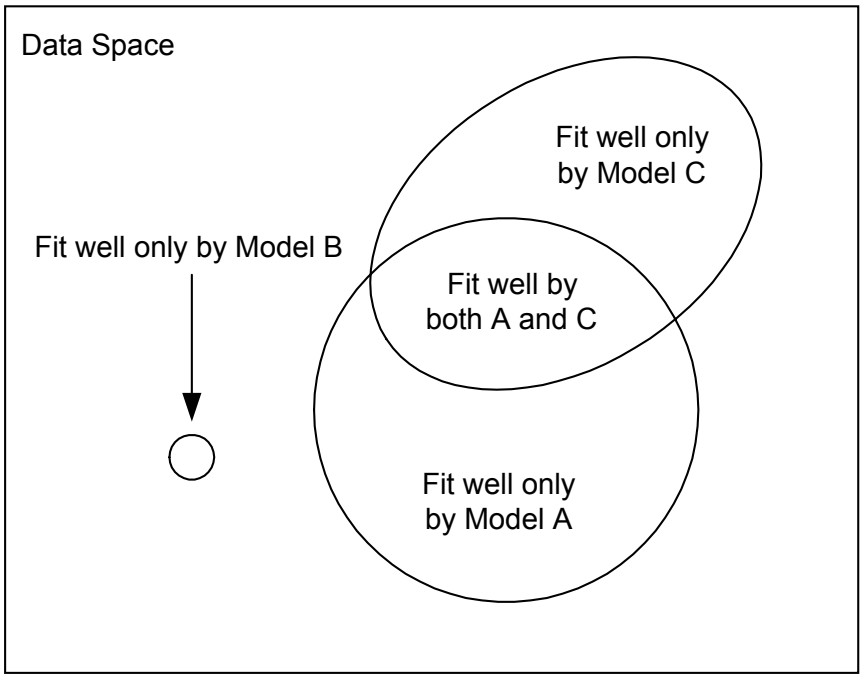


Figure 1.2: Venn diagram representing regions of the data space fit well by three alternative models. The large square in this diagram represents the data space consisting of all possible data patterns. Regions circumscribed by more than one ellipse indicate data patterns fit well by more than one model

primarily concerned. One problem with comparatively complex models is that they do not adequately constrain possible outcomes. That is, the good fit of complex models is

usually interpreted as evidence that the theory is correct, but if the model could fit a wide range of plausible data patterns equally well, then good fit is worth little.

1.3.2 The Number of Free Parameters

The problem presented by model complexity is the subject of an increasing amount of literature in mathematical psychology and allied fields (e.g., Myung, 2000). In general, models with more free parameters are better able to fit data (Hurvich, 1997). The number of free parameters (or some simple function of this number) is what Cutting et al. (1992) and Mulaik (2001) call *simplicity*, and has for years been used by many researchers as a gauge of a model's complexity (Box & Jenkins, 1970; Jeffreys, 1957, 1961). Complexity is generally regarded as an undesirable quality for a model to possess because, all else being equal, the simplest model is the most likely to be true for unobserved data (Jeffreys, 1931; Popper, 1959; Wrinch & Jeffreys, 1921). This notion is closely akin to the most familiar dictum of science, Occam's razor, the thrust of which is that scientists should not offer overly complicated explanations for phenomena when a simpler one will suffice.

Many fit indices and model selection criteria therefore correct for model complexity by incorporating some function of the number of free parameters. For example, one popular fit index in the context of SEM is the root mean square error of approximation (RMSEA or ε ; Browne & Cudeck, 1993; Steiger & Lind, 1980). In the population, RMSEA is represented as:

$$\varepsilon = \sqrt{\frac{F_0}{n}} \tag{1.1}$$

where F_0 is an index of *discrepancy of approximation*, the difference between the population covariance matrix and the implied (reproduced) covariance matrix. A point estimate of RMSEA is computed as:

$$\hat{\varepsilon} = \sqrt{\frac{\max\left\{\left(\widehat{F}_{ML} - \frac{df}{n}\right), 0\right\}}{df}} \tag{1.2}$$

where $\widehat{F}_{ML}$ is the sample maximum likelihood discrepancy function value and *df* represents degrees of freedom. The square root is taken because F_0 can be represented as a sum of squared standardized residuals (Steiger & Lind, 1980). RMSEA can be interpreted as an estimate of population discrepancy per degree of freedom, correcting for model complexity by dividing by *df*. Because models with fewer degrees of freedom have more free parameters, it can be seen that RMSEA favors models with fewer parameters (i.e., less complex models) to those with more parameters, all else being equal (Browne & Cudeck, 1993).

An example of a model selection criterion which corrects for complexity by incorporating the number of free parameters is the Akaike Information Criterion (AIC; Akaike, 1973):

$$AIC = -2\ln ML_0 + 2q \tag{1.3}$$

where ML_0 is the maximized likelihood and q is the number of free parameters. The first term $\left(-2\ln ML_0\right)$ is a measure of goodness of fit and the second term $\left(2q\right)$ is regarded as a penalty for complexity (lower values of AIC are considered better), where complexity is thought of purely in terms of the number of free model parameters. One of the attractive features of AIC is that it can be used to compare models which are not nested in terms of parameters (Forster, 2000).[5]

The root mean squared deviation (RMSD):

$$RMSD = \sqrt{\frac{SSE}{\left(n-q\right)}} = \sqrt{\frac{SSE}{df}} \tag{1.4}$$

like RMSEA, penalizes complex models by dividing by *df*. Another model selection criterion which incorporates a correction for the number of free parameters is the Bayesian information criterion (BIC; Schwarz, 1978). In addition, *parsimony indices* (James, Mulaik, & Brett, 1982; Marsh & Balla, 1994) include the ratio of *df* from the model under investigation and the *df* associated with an appropriately specified null

[5] Two models, A and B, are *nested* if one of them (A) possesses a subset of the free parameters of the other (B). In other words, if constraints are placed on B to create A, A is said to be nested within B. In SEM, the difference between χ^2 values associated with nested models is itself distributed as χ^2, with degrees of freedom equal to the number of constraints placed on B to yield A.

model[6] as a correction for model complexity (parsimony is defined as the complement to model complexity in the context of SEM).

1.3.3 Functional Form Complexity

Fit indices and model selection criteria like RMSEA, AIC, BIC, RMSD, and parsimony indices represent a trade-off between goodness of fit and complexity, with complexity defined strictly in terms of the number of free model parameters. However, simply including some function of q as a way to correct for complexity has been criticized on the grounds that model complexity is not governed completely by the number of parameters (Bozdogan, 2000; Dunn, 2000; Keuzenkamp & McAleer, 1997). This notion was introduced by Harold Jeffreys (1931, 1957, 1961; Wrinch & Jeffreys, 1921), who suggested a rough measure of complexity which took into account not only the number of model parameters, but also various aspects of the model represented in the form of a differential equation, including its polynomial order. Jeffreys' formulation was admittedly arbitrary (Keuzenkamp & McAleer, 1995), but it represented a major step toward operationalizing Occam's razor for practical use. Jeffreys' point was that *functional form* also contributes to complexity (Myung, 2000; Myung et al., 2000; Pitt & Myung, 2002; Zhang, 1999). For example, the two-parameter additive model $y = ax + b$ may be less complex than the polynomial model $y = ax^b$, even though the two models have the same number of free parameters. Because of the interplay between number of parameters and functional form in determining complexity, a simple two-parameter sine

[6] Except in special circumstances, a null (or independence) model is one in which measured variables are specified to be uncorrelated. Typically, the only parameters estimated in a null model are variances.

function may completely account for every point in a given two-dimensional scatter plot, yet a structural equation model with 50 free parameters may exhibit very poor fit to data. Therefore, attempts have been made to directly quantify model complexity so that it may be explicitly considered when models are to be evaluated.

Cutting et al. (1992) borrowed from the field of information theory to describe a new way to examine complexity in psychology: *equation length*, which is essentially the same as functional form. Short algorithms or computer programs that can be implemented to describe or generate a particular data set should be preferred over longer competing algorithms or programs because short programs are more parsimonious. To the extent that a psychological model can be compared to a descriptive or generative algorithm, this perspective is intuitively appealing. Models can be expressed as equations. Therefore, the most parsimonious theory from among a collection of competing theories may be selected on the basis of the length of the equation (model) specified to represent it. Some model selection criteria have emerged to capitalize on this idea. For example, *minimum description length* (MDL) is defined as the number of binary digits required to reproduce observed data under a particular model (Grünwald, 2000; Myung, 2000; Rissanen, 1987). In MDL, complexity is computed using both the number of free parameters and functional form, as well as other sources of complexity described below. More will be said about MDL in §1.5 and later in Chapter 5.

1.3.4 Other Sources of Complexity

Besides the number of parameters and functional form associated with a model, there are at least four other factors contributing to model complexity (Pitt et al., 2002). First, models with more restrictions on the allowable range for free parameters are comparatively less complex. Second, complexity is positively related to sample size; i.e., models increase in complexity when the size of hypothetical samples to which they could be applied increases. Third, complexity will differ with the shape of the probability distribution specified in the likelihood function. Fourth, complexity will differ depending on experimental design. For example, if experimental manipulations are sampled from only certain regions of the range of possible manipulations, the same model may differ in complexity for different choices of manipulations. This dissertation will address the influences of the number of free parameters, functional form, and parameter range on model complexity in the context of SEM.

1.4 Complexity in the Context of Structural Equation Modeling

1.4.1 Overview of Structural Equation Modeling

Before illustrating the negative consequences associated with ignoring model complexity, it is necessary to provide a brief introduction to structural equation modeling. Briefly, SEM is a modeling paradigm intended to specify, estimate, and evaluate theories of causal and associative relationships among variables. Observed *manifest variables* (MVs) are typically represented as indicators of unobserved *latent variables* (LVs). Using LISREL notation (Jöreskog & Sörbom, 1996), a structural equation model can be

represented in three matrix equations. First, the *measurement model* expresses MVs as functions of LVs:

$$\underline{x} = \Lambda_x \underline{\xi} + \underline{\delta} \tag{1.5}$$

$$\underline{y} = \Lambda_y \underline{\eta} + \underline{\varepsilon} \tag{1.6}$$

where $\underline{x}$ and $\underline{y}$ are vectors of observed scores on indicators of exogenous and endogenous LVs, respectively, Λ_x and Λ_y are matrices of regression weights (loadings) of MVs on LVs, $\underline{\xi}$ and $\underline{\eta}$ are vectors of exogenous and endogenous LVs, respectively, and $\underline{\delta}$ and $\underline{\varepsilon}$ are vectors of unique factor terms associated with indicators of exogenous and endogenous LVs, respectively. Second, the *structural model* expresses directional relationships among LVs:

$$\underline{\eta} = \mathrm{B}\underline{\eta} + \Gamma\underline{\xi} + \underline{\zeta} \tag{1.7}$$

where B is a matrix of path coefficients linking endogenous LVs to each other, Γ is a matrix of path coefficients linking endogenous LVs to exogenous LVs, and $\underline{\zeta}$ is a vector of residual terms. The *covariance structure* implied by these equations expresses the variances and covariances among MVs as functions of parameter matrices:

$$\begin{aligned}\Sigma &= \begin{bmatrix}\Sigma_{yy} & \Sigma_{yx}\\ \Sigma_{xy} & \Sigma_{xx}\end{bmatrix}\\ &= \begin{bmatrix}\Lambda_y\left(\mathrm{I}-\mathrm{B}\right)^{-1}\left(\Gamma\Phi\Gamma'+\Psi\right)\left(\mathrm{I}-\mathrm{B}'\right)^{-1}\Lambda_y{}'+\Theta_\varepsilon & \Lambda_y\Phi\Gamma'\left(\mathrm{I}-\mathrm{B}\right)^{-1}\Lambda_x{}'\\ \Lambda_x\Phi\Gamma'\left(\mathrm{I}-\mathrm{B}'\right)^{-1}\Lambda_y{}' & \Lambda_x\Phi\Lambda_x{}'+\Theta_\delta\end{bmatrix}\end{aligned} \quad (1.8)$$

where Θ_ε is the covariance matrix of unique factors associated with indicators of endogenous LVs, Θ_δ is the covariance matrix of unique factors associated with indicators of exogenous LVs, Φ is the covariance matrix of exogenous LVs, and Ψ is the covariance matrix of residual terms associated with endogenous LVs. A special case of SEM is the factor analysis model, in which there are no causal paths specified among LVs:

$$\Sigma_{xx} = \Lambda_x\Phi\Lambda_x{}' + \Theta_\delta \quad (1.9)$$

1.4.2 Model Fit in SEM

Estimates of the parameters of a structural equation model are obtained by fitting the model to observed data. This involves minimizing a loss function, usually the maximum likelihood (ML) discrepancy function, arriving at a set of parameter estimates which yield an implied covariance matrix that is as similar as possible to the observed covariance matrix of MVs. The ML discrepancy function can be represented as:

$$\widehat{F}_{ML}\left(\mathrm{S},\Sigma\right)=\log\left|\Sigma\right|-\log\left|\mathbf{S}\right|+tr\left[\mathbf{S}\Sigma^{-1}\right]-p\,, \tag{1.10}$$

where **S** is the observed covariance matrix, Σ is the reproduced covariance matrix implied by the model, $|\cdot|$ is the determinant operator, and *tr* is the trace operator. To the degree that the reproduced covariance matrix resembles the observed covariance matrix, $\widehat{F}_{ML}$ will tend to be small, and the model is said to have good fit to the data.

Another discrepancy function commonly used in SEM is ordinary least squares (OLS). The OLS discrepancy function is simply:

$$\widehat{F}_{OLS}\left(\mathbf{S},\Sigma\right)=\frac{1}{2}tr\left[\mathbf{S}-\Sigma\right]^{2}, \tag{1.11}$$

the sum of squared residuals. Despite the fact that $\widehat{F}_{OLS}$ is a more intuitive and straightforward measure of lack of fit, more is known about the distributional characteristics of $\widehat{F}_{ML}$ under the assumption of normality. Thus, many fit indices are based upon $\widehat{F}_{ML}$ rather than upon $\widehat{F}_{OLS}$ (Browne, MacCallum, Kim, Andersen, & Glaser, 2002).

It is often of interest to evaluate how well a particular structural equation model performs in isolation (absolute fit) or how well it performs relative to other models (relative fit). Most model fit indices in SEM are based on the ML likelihood ratio test statistic, a function of $\widehat{F}_{ML}$. The statistic $\left(N-1\right)\widehat{F}_{ML}$ follows an asymptotic χ^{2}

distribution if the model is correct, if the sample is sufficiently large, and if distributional assumptions are met. In general, lower χ^2 values reflect better-fitting models, with 0.0 indicating perfect fit to the data[7] (usually achievable only with a completely saturated model). However, models with more free parameters are better able to reproduce observed data than models with some of those parameters constrained, and therefore will have lower χ^2 values (barring convergence problems), making them comparatively more complex than models with constraints. This example referred to nested models, but the general principle holds to some extent even when comparing models that are not nested. The tendency for more parameters to lead to better fit could create situations in which one model is selected out of several competing models not because it is the best in any meaningful sense, but simply because it is the most complex (Collyer, 1985; Myung, 2000; Myung et al., 2000; Roberts & Pashler, 2000). A further shortcoming of the χ^2 test of fit is that as *N* increases, so does $(N-1)\hat{F}_{ML}$. Therefore, tests of fit which rely on χ^2 alone often reflect more on *N* than on a model's quality (Browne & Cudeck, 1993; Jöreskog, 1969). Because of this sensitivity to *N*, the same model could show highly discrepant fit indices when applied to different data, based solely on differences in sample size.

Even considering the sensitivity of χ^2 to *N*, no structural equation model realistically can be expected to have perfect fit – models are intended to approximate

[7] Exact fit is commonly tested in SEM by assessing the statistical significance of χ^2 with $df = \frac{1}{2}p(p+1) - q$, where *p* is the number of measured variables and *q* is the effective number of model parameters being estimated.

phenomena such that they can be understood or predicted in a general way (MacCallum, 2003). Given that no model is perfect, and that even the best of them can serve merely as an approximation to some collection of interesting phenomena, the test of the null hypothesis that the model *is* correct is an enterprise flawed at a fundamental level. Within the SEM framework, one of the basic rules researchers normally follow – obtain the largest sample possible – is inverted if the criterion of perfect fit is used. With a sample large enough, any model with at least one degree of freedom will be rejected by this criterion. In other words, if the hope is to retain the hypothesized model using the perfect fit criterion, it pays the researcher to use small samples. However, doing so yields large confidence intervals about parameter estimates and results in the retention of models which may have very low generalizability. Thus, statistical power in the SEM context is better conceptualized as the probability that a model will be rejected if it does not meet some reasonable standard of close fit (MacCallum, Browne, & Sugawara, 1996; MacCallum & Hong, 1997). However, established approaches to power analysis in SEM use confidence limits of popular fit indices to test hypotheses of close or exact fit. As sample size decreases, the confidence limits expand, meaning that if the sample is small enough, the researcher often will not be able to reject close (or even perfect) fit even for poor models. Thus, good model fit, which is traditionally interpreted to reflect a useful model, may come at the cost of precision of parameter estimates and may result in a nonsensical model which does not reflect real underlying processes.

In SEM, model fit is typically assessed by *absolute* or *incremental* fit indices. Absolute fit indices measure the fit of a model in isolation. Some indices are residual-

based, such as the root mean squared residual (RMSR; Jöreskog & Sörbom, 1986), whereas most others are based on $\widehat{F}_{ML}$, such as RMSEA (Browne & Cudeck, 1993; Steiger & Lind, 1980). Incremental fit indices are used to evaluate the fit of a model relative to a null model. Examples include the Tucker-Lewis Index (TLI; Tucker & Lewis, 1973) and the normed fit index (NFI; Bentler & Bonnett, 1980). Each of these indices has associated with it a traditional range of acceptable values, based on Monte Carlo investigations or practical experience evaluating large numbers of models and data sets. For example, values of RMSEA less than .05 or .06 might be considered to reflect close fit (Browne & Cudeck, 1993; Hu & Bentler, 1998), whereas values of TLI greater than .95 could be reckoned adequate (see Hu & Bentler, 1998 and Browne et al., 2002 for more detail on SEM fit indices). In all cases, these indices assess the fit of a given model to an observed data set.

1.5 The Inadequacy of Traditional Fit Indices

It is the generalizability of a model that researchers are ultimately interested in determining – not how well the model fits one particular data set. Because nearly all fit indices used in SEM correct for complexity in some manner,[8] it is more accurate to think of them as *generalizability indices*. The ideal generalizability index should balance two components: goodness of fit and complexity (McDonald & Marsh, 1990). The

[8] Nearly all SEM fit indices include a correction for complexity, although the correction was not always explicitly included because of its complexity-correcting effects. For example, in their initial presentation of the TLI, Tucker and Lewis (1973) include *df* in the formula as a way to render certain elements of the formula comparable to components of variance. However, *df* can also be interpreted as a correction factor for model complexity.

complexity portion should capture a model's baseline ability to fit data as precisely and accurately as possible to place competing models on equal footing. Once complexity has been accounted for, opposing models may be compared solely in terms of goodness of fit without fear that their inherent ability to fit data is biasing judgment (Cutting et al., 1992; Jeffreys, 1931). Unfortunately, fit indices which correct for complexity do so only crudely, by equating complexity with the number of free model parameters. Other contributing factors, such as functional form, are ignored, and thus traditional means of penalizing complexity do not necessarily place competing models on equal footing.

It would be very useful to be able to implement model selection criteria such as MDL (as described later in Chapter 5), which considers multiple sources of model complexity (Pitt, Kim, & Myung, in press). The consideration of model complexity has been particularly prescribed in the domain of structural equation modeling (MacCallum, 2003; Zhang, 1999). However, the problem of directly computing MDL has so far proven difficult in the SEM context given computational complexity and computer hardware limitations (Zhang, 1999). An alternative, suggested by Kim (2002), Pitt et al. (2002), and Zhang (1999), is to use an algorithm-based approach which implements MDL in principle without having to analytically derive it. In line with this suggestion, Cutting et al. (1992) strongly recommend that models be fit to random data to get some idea of their baseline ability to fit data. "Without baseline comparisons of models run on random data, we think any approach that proceeds by comparing models with matched numbers of parameters may be in jeopardy" (p. 380).

The purpose of this dissertation, therefore, is to address the problem of model complexity in structural equation modeling. It will address Roberts and Pashler's (2000) first concern – that the degree to which a model will fit any data should be considered – in the context of SEM. Following the suggestions of Pitt et al. (2002) and Cutting et al. (1992), the problem of determining model complexity will be approached by randomly generating large numbers of plausible data sets. These data sets will represent the full data space, conceptually corresponding to the union of the shaded and unshaded areas in each panel of Figure 1.1. After the data are generated, models will be fit to all of the data sets in an effort to determine the range of outcomes implied by theory, corresponding to the shaded areas in Figure 1.1. Selected models will be fit to these data to explore and illustrate the problems associated with ignoring model complexity. Thus, Roberts and Pashler's (2000) logic suggests a novel, nonanalytical approach to assessing and controlling for the effects of model complexity in the context of SEM. It is expected that the findings of this study will apply, at least in spirit, to other modeling contexts as well.

1.6 Summary

In Chapter 1, the concept of model complexity has been introduced. It was explained why model complexity should not be ignored when using one popular class of models used in psychology – structural equation models. In Chapter 2, a novel method of generating random dispersion matrices for use in Monte Carlo studies is presented. In Chapter 3, these random data will be used in conjunction with several structural equation models to illustrate the consequences of ignoring model complexity.

CHAPTER 2

DATA GENERATION

2.1 Random Correlations

Roberts and Pashler (2000) emphasized that it is insufficient to demonstrate that there exist *possible* data patterns that could falsify a model – there must be *plausible* data that could falsify a model. They define plausible data as "the union of predictions based on other plausible theories and expectations based on other data" (p. 365). In the context of structural equation models, then, to know the range of plausible outcomes implies that one must know all competing plausible theories and have a large number of other data sets on hand. Clearly this would be a difficult or impossible accomplishment for most researchers. Furthermore, in terms of covariance matrices, the distinction between "plausible" and "possible" is highly ambiguous because it relies on arbitrary levels of plausibility for all input correlation coefficients, which in turn are interdependent to a large degree. Therefore, for the sake of simplicity, it was decided to limit the conceptual data space to positive manifold correlation matrices. Positive manifold matrices are consistent with much correlational data in psychological research. For convenience, the data space was limited to correlation matrices consisting of no nonpositive elements, probably without loss of generality.

In order to investigate the implications of model complexity on fit in the context of SEM, it was necessary to generate large numbers of random correlation matrices. The idea of using random data to investigate complexity is not new. Collyer (1985) suggests *known-model analysis* as a viable approach to examine the complexity of competing models. Known-model analysis involves generating random data sets using each of the competing models, adding known amounts of error variance, and then fitting those data sets with all competing models. Complex models will tend to fit data sets generated by competitor models well, sometimes even better than the generating models themselves. Pitt et al. (2002) successfully used this approach to compare the relative complexities of two psychophysical laws, and demonstrated that Steven's law describes the relationship between perception and stimulus strength better than Fechner's law, even when Fechner's law was used to generate the data. This approach requires that a set of competing models be specified a priori, and thus can provide assessments of model complexity relative only to those competing models.

Botha, Shapiro, and Steiger (1988) present a *uniform index-of-fit* in response to the sensitivity of χ^2 to sample size. First they define a least-squares discrepancy function (distance):

$$d_{p,m}(\mathrm{R}) = \sqrt{\min \operatorname{tr}\left(\mathrm{R} - \Lambda\Lambda' - \Psi\right)^2} \tag{2.1}$$

A distribution of $d_{p,m}(\mathrm{R})$ is obtained by fitting the model of interest to many randomly generated correlation matrices. Then the model is fit to real data (R^*) to obtain *d**. The uniform index-of-fit is defined as:

$$i_{p,m}(\mathrm{R}^*) = \mathrm{Prob}\left\{d_{p,m}(\mathrm{R}) \geq d^*\right\}, \tag{2.2}$$

(or simply *i**); i.e., the proportion of randomly generated data patterns with worse least-squares fit than the data in hand. The uniform index-of-fit is interpreted such that larger values indicate better fit relative to random data. The *i** index can be a valuable tool when used in conjunction with other fit indices because it is independent of sample size, is distribution-free, and can give the researcher a sense of how well a model fits obtained data relative to all possible data. Furthermore, a version of *i** can be computed regardless of the estimation method used.

Cutting (2000), Cutting et al. (1992), and Dunn (2000) used a similar approach in their comparison of the fuzzy-logical model of perception (FLMP) and a competing additive model. Model complexity (or scope) was operationalized as the amount or proportion of completely random data that each model could fit well by some criterion. Like the approach of Botha et al. (1988), this approach is superior to known-model analysis because it does not require competing models in order to work. A model's complexity may be evaluated relative to all possible data patterns rather than only those data patterns derivable from an arbitrary set of competing models. In this dissertation the

general approach adopted by Botha et al. and Cutting et al. is used in preference to the known-model analysis suggested by Collyer (1985).

Defining what is meant by *random* is the same as defining the space from which the random matrices are to be selected. The criterion for randomness adopted here is:

> *Every possible square, symmetric, positive definite matrix with* 1*s on the diagonal and off-diagonal elements in the range {0, 1} has an equal probability of being selected (generated).*

The foregoing is only one possible criterion for randomness. It is possible that other criteria better capture the notion that some data patterns are more plausible than others in social sciences research; this one was chosen primarily because it does not rely on preconceived ideas about data patterns that are more or less likely to appear in practice (for similar reasoning see Dunn, 2000). Matrices generated according to this criterion can be considered uniformly distributed across the data space of interest, even if their elements (individual correlations) are not uniformly distributed. Thus, uniformity exists not at the level of individual elements, but at the level of the entire matrix. This criterion is the most intuitively appealing from among several alternatives because (1) it does not presume a particular model and (2) every possible matrix fitting the criterion has an equal chance of being selected. The requirement of uniform coverage is important because it was desirable to test models using data from every part of the plausible data space. There is no a priori reason to favor some parts of the data space over others. If one part of that

data space is represented more than another part, any findings would be biased toward (apply more to) that region of the data space.

A correlation matrix (R) is defined as a positive semidefinite matrix with unit diagonal elements and with off-diagonal elements r_{ij}, $i \neq j$, such that $\{-1.0 \leq r_{ij} \leq 1.0\}$. In correlation matrices, every off-diagonal element must lie within bounds set for it by other elements of the matrix. For example, if the correlations among variables A, B, and C are such that $r_{AB} = .95$ and $r_{AC} = .90$, then r_{BC} must lie between .711 and .991 (Stanley & Wang, 1969). Given these defining characteristics, any method which generates square, positive semidefinite matrices with unit diagonal elements, and which also involves a random component, arguably produces random correlation matrices. However, additional facts must be considered. First, even though correlation matrices may be positive semidefinite, ML estimation will break down when at least one eigenvalue of the input matrix is zero because the algorithm involves the log of the determinant of the dispersion matrix. The determinant will be zero if at least one eigenvalue is zero, and the logarithm of zero is undefined. Even when $|\mathrm{R}|$ is not exactly zero, but is near-zero, ML estimation may break down. Therefore, for present purposes correlation matrices are defined to be positive definite (i.e., all eigenvalues must be greater than zero). Second, the fact that there are several ways of defining what is meant by *random component* implies that there are many different kinds of random correlation matrices, some of which may be more appropriate for present purposes than others. Some methods result in matrices which over-represent certain regions of the data space. Others take an inordinate amount of time to generate data. Following is a brief description of some existing

methods of generating random correlation matrices. Each method will be described in terms of computational efficiency and representativeness.

2.2 Methods for Generating Random Correlation Matrices

2.2.1 The Uniform Correlation Matrix (UCM) Method

In their presentation of a new fit index for use in exploratory factor analysis, Botha et al. (1988) discuss three methods for generating random correlation matrices. The first method they refer to as the *uniform correlation matrix* (UCM) method (also called the direct acceptance-rejection method; Holmes, 1991). The method involves generating $½p(p-1)$ random uniform numbers on the interval {0, 1} and assigning them to the subdiagonal elements of a $p \times p$ matrix with unit diagonal elements. The upper triangle is set equal to the transpose of the lower triangle. The resulting matrix is tested for positive definiteness by ensuring that all eigenvalues are greater than zero before the matrix is retained. Botha et al. found that for low values of *p* this method worked well. When $p \geq 6$, however, the method slows considerably – approximately one out of every 50 matrices was positive definite at $p = 6$. This finding was replicated in the current project with a Fortran 90 program (see Appendix A). It was found that the time required to generate a positive definite matrix increased exponentially with increments in *p*. It took approximately 14 hours on a computer with a Pentium III processor to produce 1000 positive definite 7×7 matrices, and would have taken a projected 15 weeks to generate 1000 8×8 matrices.

The other two methods proposed by Botha et al., the *uniform unit sphere* (UUS) and *uniform (0,1) method* (U01), are variants on a procedure in which matrices of random unit vectors are multiplied to yield correlation matrices. These methods were programmed and their output evaluated for inclusion in the present study. However, both methods yielded matrices which tended to have unrealistically high positive off-diagonal elements.

Of the three methods suggested by Botha et al. (1988), only the first (UCM) yields matrices that could be considered random by the criterion offered above. That is, the UCM algorithm, although crude, generates matrices which meet the definition of R and which satisfy the condition of uniformity. However, although it is intuitively appealing, the UCM method is too slow to be practical past $p = 7$. Therefore, it would be desirable to find a method which yields the same sort of matrices generated by UCM, but faster. Some other methods were evaluated for how closely their output matched that of UCM.

2.2.2 Known-Spectrum Algorithms

A number of methods to generate random correlation matrices have been suggested in the statistical literature. Most of them exist either to create sample matrices from a given population matrix (e.g., Gleser, 1976; Hartley & Harris, 1963; Odell & Feiveson, 1966; Tucker, Koopman, & Linn, 1969) or generate random matrices with given spectra and randomly generated orthonormal matrices of eigenvectors (e.g., Bendel & Afifi, 1977; Bendel & Mickey, 1978; Chalmers, 1975; Chan & Li, 1983; Chu, 1995;

Davies & Higham, 2000; Johnson & Welch, 1980; Lin & Bendel, 1985; Marsaglia & Olkin, 1984; Zha & Zhang, 1995). In the latter case, eigenvalues may in turn be randomly generated, but the joint distributional characteristics of eigenvalues represent a largely unresearched area (Holmes, 1991). Analytic generation of random eigenvalues for matrices of limited order is discussed by Marasinghe and Kennedy (1982). They note that one method that should work for obtaining eigenvalues for matrices of any order is to first generate random matrices and then extract their eigenvalues, but this method presumes that a good method of generating random matrices already exists. A brief discussion of many of these generation methods can be found in Gentle (1998) and in Holmes (1991).

The method of Davies and Higham (2000), an improvement upon a method developed by Bendel and Mickey (1978), was selected as a representative known-spectrum generation algorithm (here denoted DH). However, because the method assumes a given spectrum, the algorithm was modified to incorporate random generation of eigenvalues. Eigenvalues were therefore generated from a uniform distribution and rescaled to sum to p. Then, vectors were generated to have uniformly distributed elements. These vectors were orthogonalized and unit-scaled using Gram-Schmidt orthogonalization (see, e.g., Hohn, 1966). Correlation matrices were obtained by pre- and post-multiplying a diagonal matrix of eigenvalues by eigenvector matrices $\left(R = UD_{\ell}U'\right)$ and applying a series of Givens rotations to ensure that the diagonal elements equal 1.0 while preserving the eigenvalues. Full details on the procedure can be found in Davies and Higham (2000). Even though the algorithm is very fast, involves random generation

at several points, and yields correlation matrices, the selection of an eigenvalue generation routine was ultimately arbitrary. Because the distributional properties of eigenvalues are largely unknown, it was anticipated that the DH algorithm would not yield matrices adhering to the uniformity requirement.

2.2.3 The Markov Chain Monte Carlo Method

Because of the limitations of existing methods of generating random correlation matrices, a new strategy was sought which would yield matrices with the desirable distributional characteristics of UCM, yet which would possess the speed of other existing algorithms. Therefore, Markov Chain Monte Carlo (MCMC) methodology was employed and programmed using Fortran 90. The MCMC approach employs much the same logic as the UCM method; that is, it searches the space containing all matrices with unit diagonals and off-diagonal elements in the {0, 1} range for matrices meeting the criteria and evaluates each matrix separately on an accept/reject basis. However, the MCMC algorithm narrows the search field to the region most likely to yield acceptance.

The MCMC algorithm involves choosing a starting point which conforms to the definition of a correlation matrix. For convenience, this starting matrix was defined as a square, symmetric matrix with 1s on the diagonal and .5s elsewhere. As long as the starting matrix is itself positive definite, or very near positive definite, there is no untoward lag in the algorithm. The only requirement for a successful starting matrix is that it be positive definite or nearly positive definite. Thus, starting matrices could be chosen to have .9999s, 0s, or empirically-derived correlations in every subdiagonal

element with no detrimental consequences. All of these possibilities were tested, all with successful results.

At each iteration, the algorithm takes the current matrix and perturbs its off-diagonal elements, retaining symmetry. The resulting matrix is checked for conformity to the definition of R. If it does not conform, the matrix is ignored and another is generated; the previous matrix (which always conforms) is used again as a starting point. If, on the other hand, the new matrix conforms to retention criteria, it is retained and is chosen as the starting point for the succeeding iteration, and so on.

More formally, the *Metropolis-Hastings* MCMC algorithm was used (Beichl & Sullivan, 2000; Gilks, Richardson, & Spiegelhalter, 1996). Let the $½p\left(p-1\right)\times 1$ vector X_{t+1} contain the subdiagonal elements of a matrix (*state*) at time $t+1$. State X_{t+1} is derived from the current state X_t by sampling from a *proposal distribution*, which in this case is:

$$x_{i_{(t+1)}} = x_{i_{(t)}} + j \cdot \left(\frac{u^{\frac{1}{z}}}{\|w\|} \cdot w_i \right) \tag{2.3}$$

where

$$
\begin{aligned}
&z = \frac{1}{2} p\left(p-1\right) \\
&u \sim U\left(0,1\right) \\
&w_i \sim N\left(0,1\right),\ i = 1...z \\
&\left\|w\right\| \ = \text{ length of } w \\
&j \ = \text{ jump size (order-dependent multiplier less than 1.0)}
\end{aligned}
\qquad (2.4 - 2.8)
$$

This proposal distribution function generates *j*-scaled points on the uniform hypersphere, but the proposal distribution can have virtually any form. The shape of the proposal distribution affects efficiency (the proportion of matrices meeting retention criteria), but leaves the target distribution unchanged. It might be sensible, for example, to simply replace the function above with something as simple as $x_{t+1} = x_t + u,\ u \sim U\left(0, 0.2\right)$, but ultimately the form of the function, within reasonable limits, makes little difference in terms of the ultimate result. Jump size affects the degree to which X_{t+1} differs from X_t. At the extremes, $j = 0$ would result in the same matrix being produced at every step, and a *j* that is too large will yield results practically equivalent to those of the UCM method, with the same lack of speed. The latter choice would constitute an *independence sampler*, the special case of the Metropolis-Hastings algorithm in which the candidate distribution does not depend on previous X_t s.

After a sufficient number of iterations (*burn-in*), the distribution of retained matrices approximates the target distribution of correlation matrices. However, because there is dependency between each matrix and all matrices preceding it (more dependency results from smaller jump sizes), it is not always desirable to retain every matrix. Instead,

matrices are selected and retained at regular iteration intervals. This *thinning number* should be large to eliminate any measurable dependency. In the current application, the thinning number is the same as the burn-in, and its size is order-dependent, ranging from 40 for 4×4 matrices to 6000 for 12×12 matrices. The thinning number is order-dependent because the ratio of acceptable to unacceptable matrices decreases as order increases; therefore, retained matrices can be expected to demonstrate greater sequential dependency for higher orders. The chosen thinning proportions were selected in a mainly arbitrary fashion, but seemed to serve well. Note that if the number of matrices retained is large, virtually any burn-in and thinning number will be adequate. As long as the acceptance rate is relatively high, the thinning number does not need to be large. Large thinning numbers were chosen to minimize the dependency between sequential matrices.

By not straying too far from a matrix known to fit the retention criteria at each iteration, the portion of the space through which the MCMC algorithm searches is restricted to the region most likely to yield correlation matrices. Figure 2.1 depicts a comparison of the spaces searched by each algorithm compared with the region of acceptable correlation matrices. As matrix order increases, the ratio of the number of unacceptable matrices to acceptable matrices increases exponentially, accounting for the inefficiency of UCM at orders higher than 7 or so. The MCMC algorithm is still subject to decreasing speed and efficiency with increasing order, but to a far lesser degree.

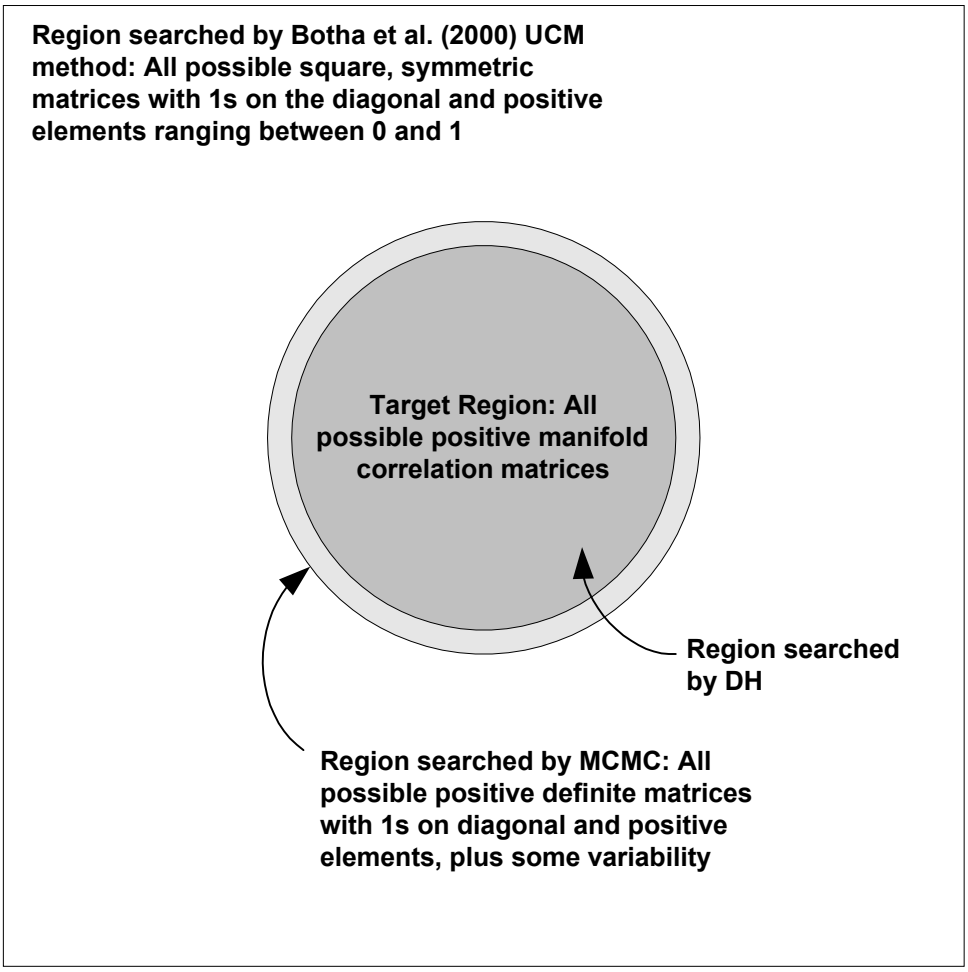


Figure 2.1: Regions of the data space searched by each algorithm

2.3 Determining the Best Generation Method

The UCM algorithm was used to generate 5000 matrices[9] of orders 4, 5, 6, and 7 (orders larger that 7 are impractical). The DH and MCMC algorithms were also used to generate matrices of orders 4 through 7. Fortran code for generating matrices from each of these methods can be found in Appendix A. Ideally, the three algorithms should yield

[9] The number 5000 was chosen not because that many matrices were necessary for modeling purposes, but because 5000 is well above the sample size typically encountered in simulation studies.

equally distributed matrices, but at different speeds. To test this prediction, six[10] elements were selected from every matrix, grouped into 20 categories based on correlation size {.00 – .05, ..., .95 – 1.00}, and used to create histograms. Comparison of histograms for matrices of corresponding orders (see Figures 2.2 – 2.5) shows highly similar results for UCM and MCMC, but noticeably discrepant results for DH.

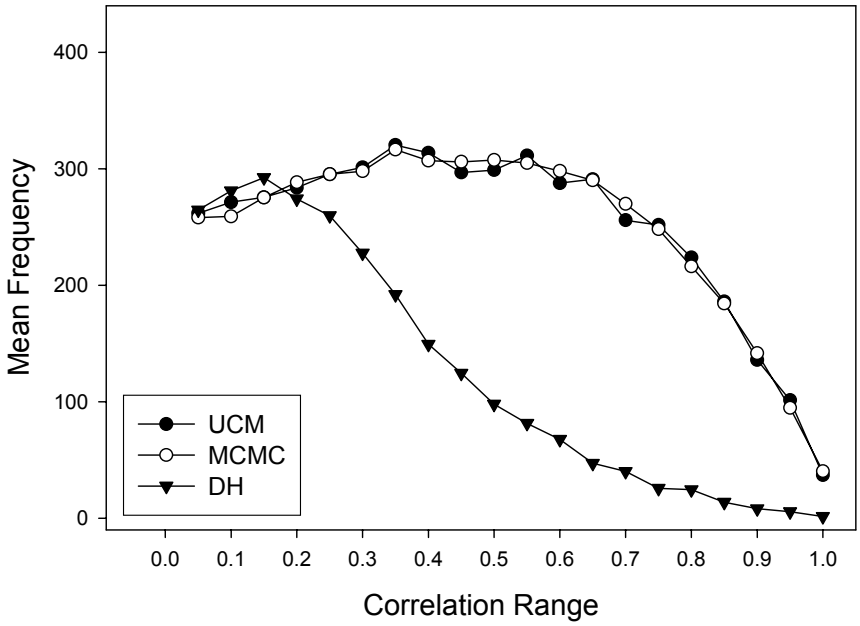


Figure 2.2: Mean correlation histograms for UCM, MCMC, and DH algorithms for 4×4 matrices

[10] Six elements were selected because 4×4 matrices contain only six subdiagonal elements. Six elements were retained for larger matrices only for the sake of comparability across orders. There is no reason to believe that certain elements will be distributed differently than others.

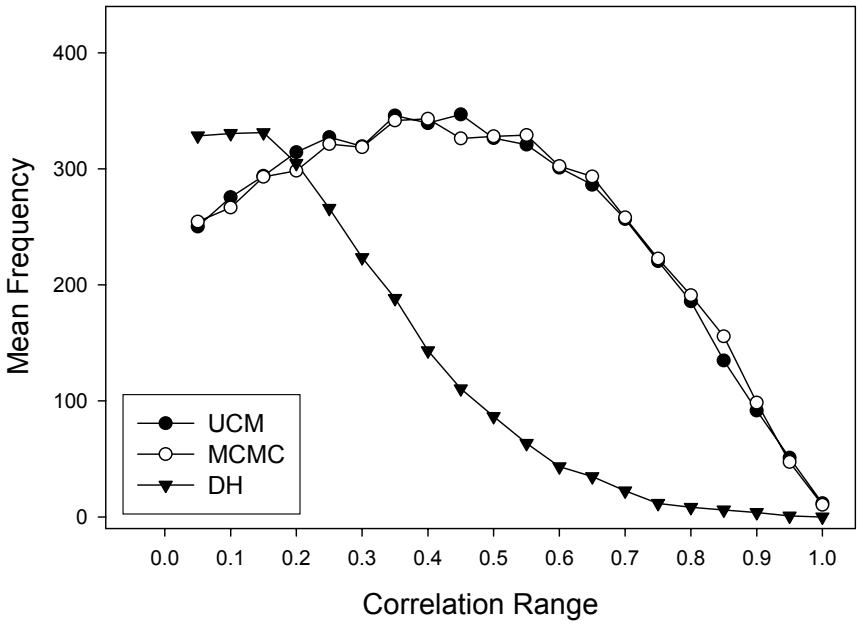


Figure 2.3: Mean correlation histograms for UCM, MCMC, and DH algorithms for 5×5 matrices

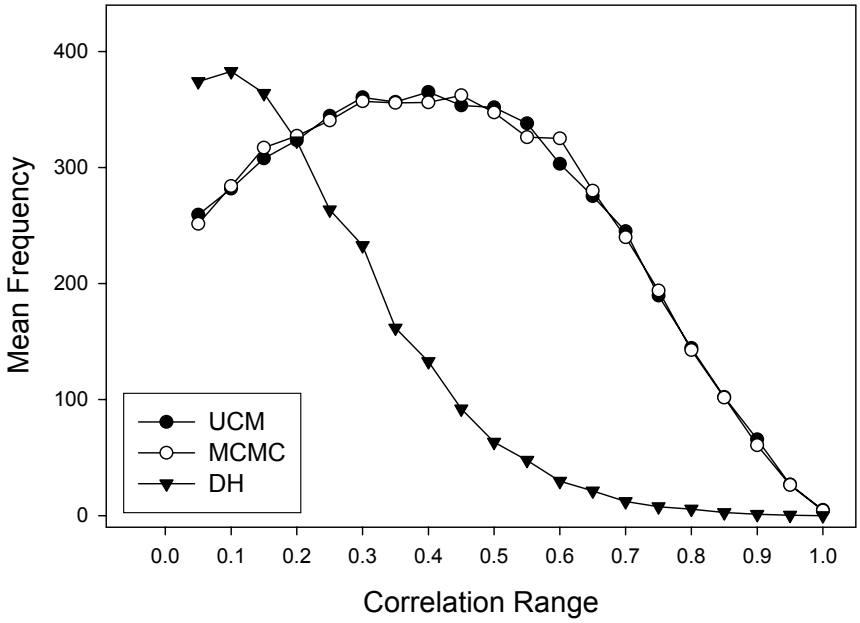


Figure 2.4: Mean correlation histograms for UCM, MCMC, and DH algorithms for 6×6 matrices

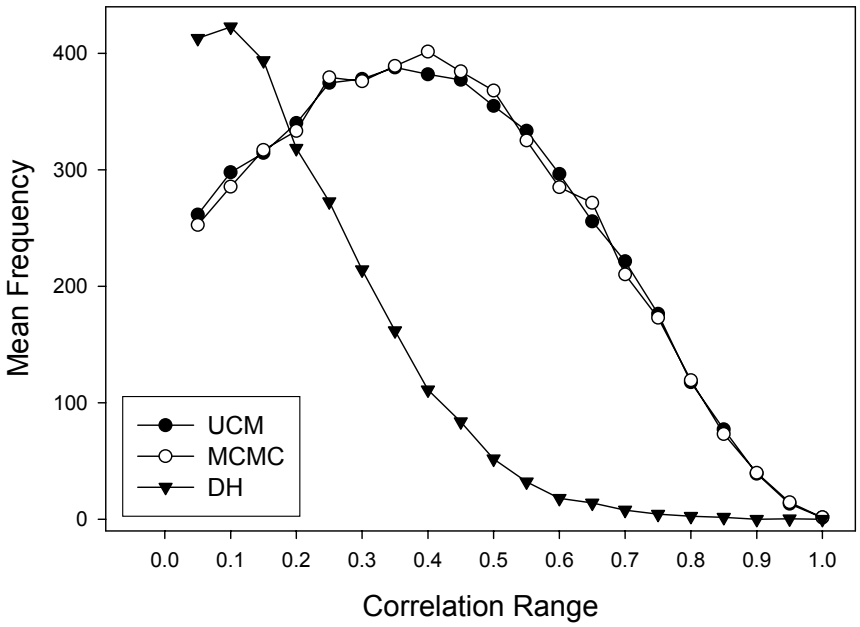
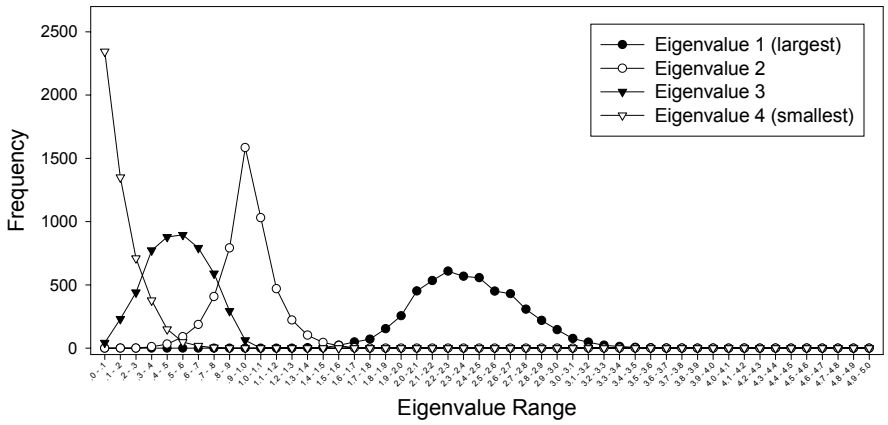


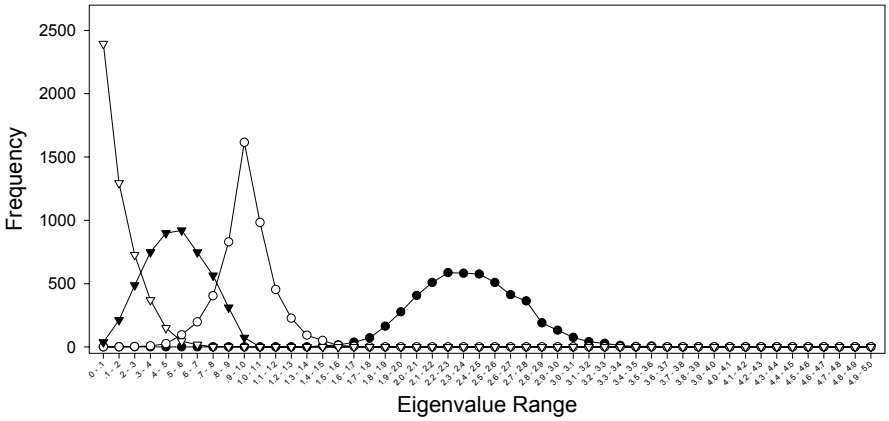
Figure 2.5: Mean correlation histograms for UCM, MCMC, and DH algorithms for 7×7 matrices

To further test the indistinguishability of matrices generated under the UCM, DH, and MCMC algorithms, eigenvalues were obtained from every matrix. Eigenvalues were grouped into 50 categories based on size and were used to create histograms. As with histograms based on the correlation coefficients themselves, histograms of eigenvalues from the UCM and MCMC algorithms appear virtually indistinguishable (see Figures 2.6 to 2.9, Panels A and B), but those from DH appear noticeably discrepant (see Figures 2.6 to 2.9, Panel C). Further evidence in favor of using MCMC to generate random matrices is presented in §3.3.

Panel A



Panel B



Panel C

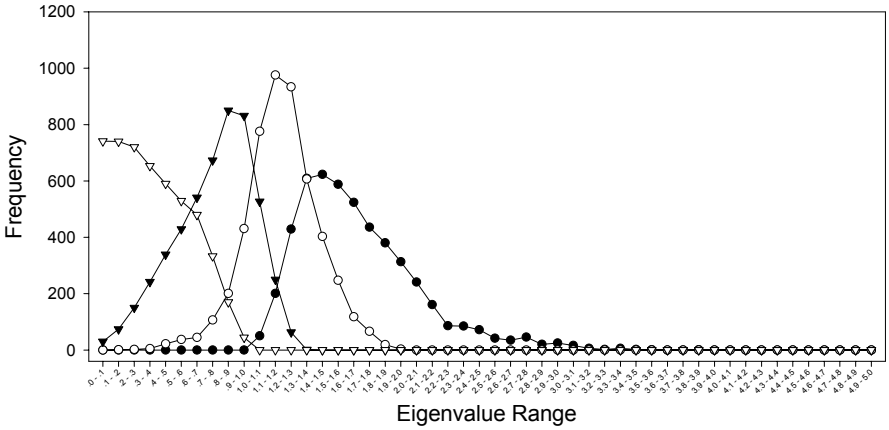
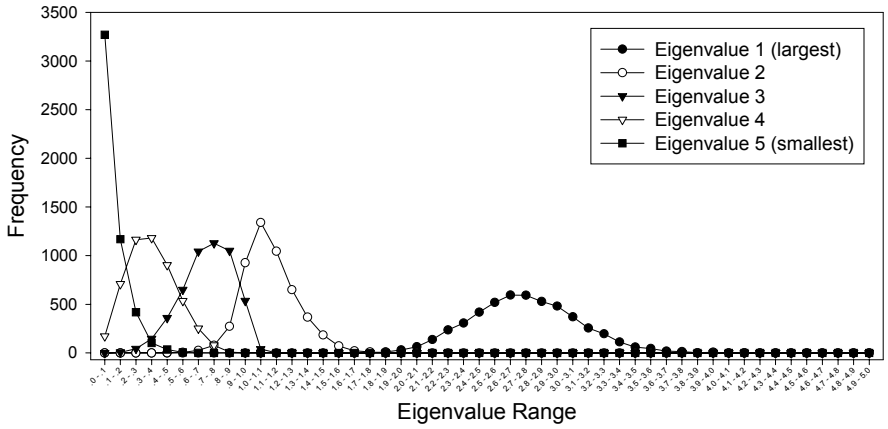
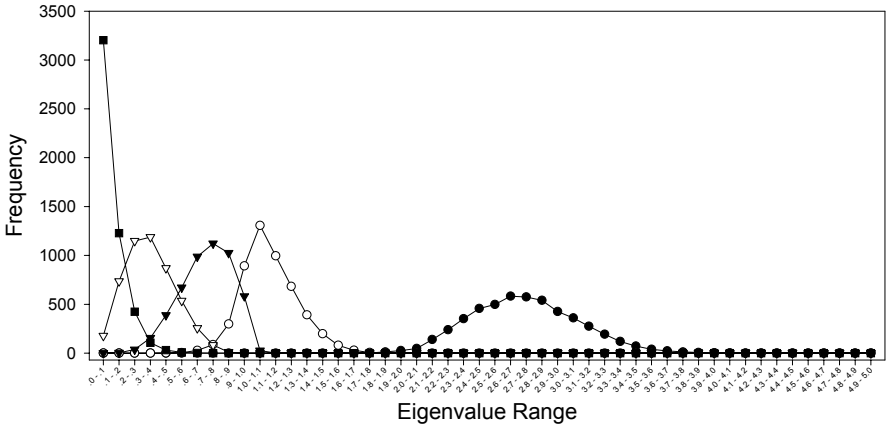


Figure 2.6: Eigenvalue distributions from UCM (Panel A), MCMC (Panel B), and DH (Panel C) algorithms for 4×4 matrices

Panel A



Panel B



Panel C

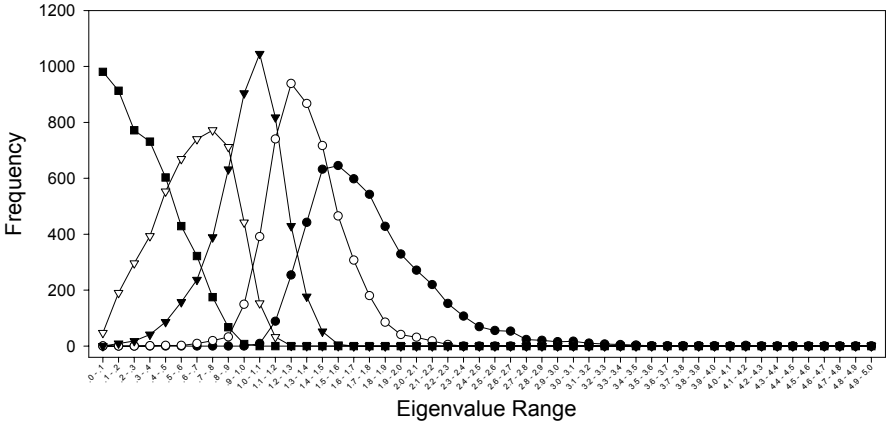
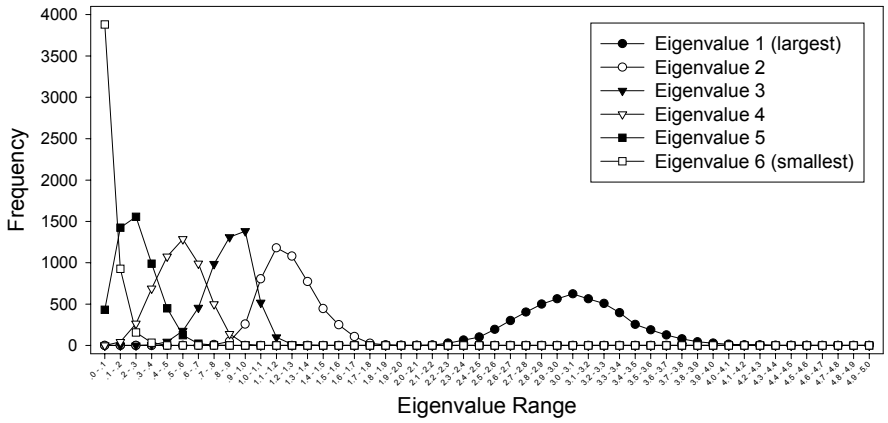
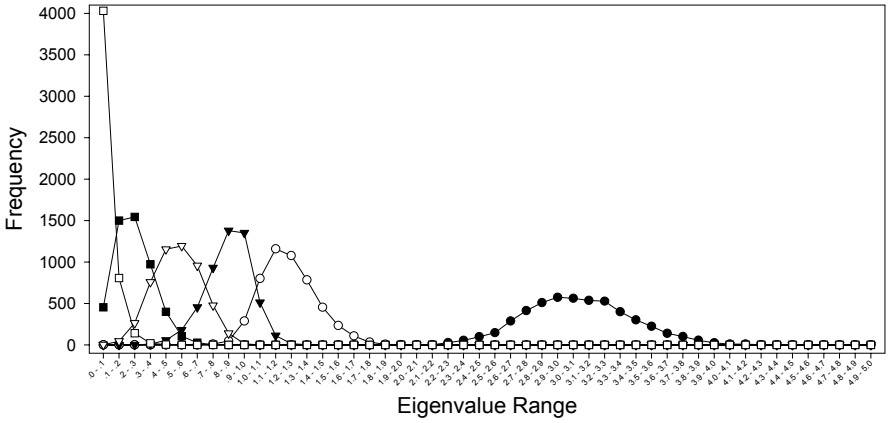


Figure 2.7: Eigenvalue distributions from UCM (Panel A), MCMC (Panel B), and DH (Panel C) algorithms for 5×5 matrices

Panel A



Panel B



Panel C

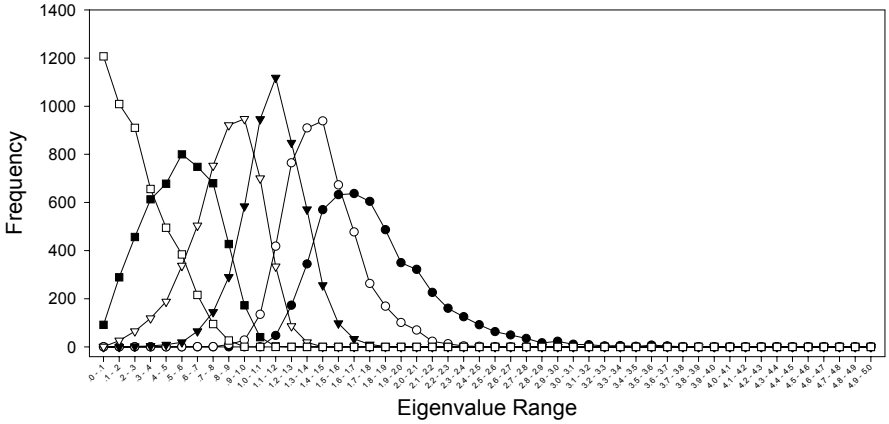
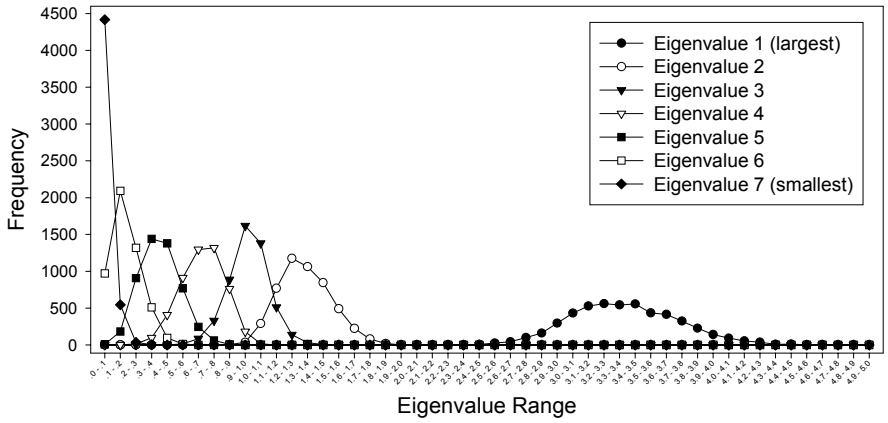
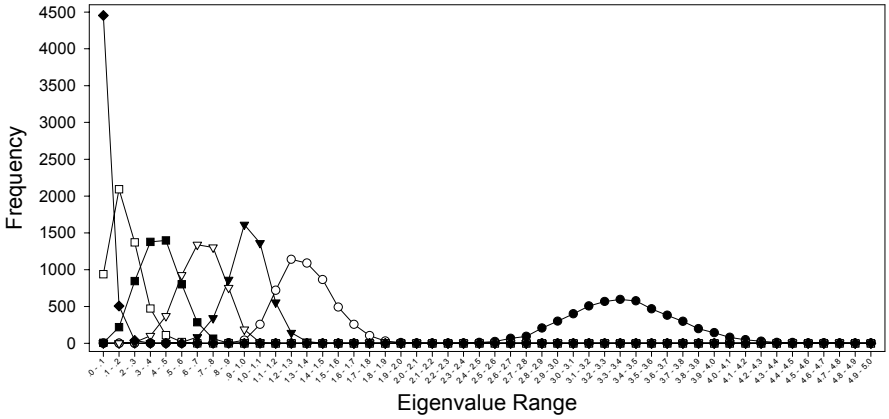


Figure 2.8: Eigenvalue distributions from UCM (Panel A), MCMC (Panel B), and DH (Panel C) algorithms for 6×6 matrices

Panel A



Panel B



Panel C

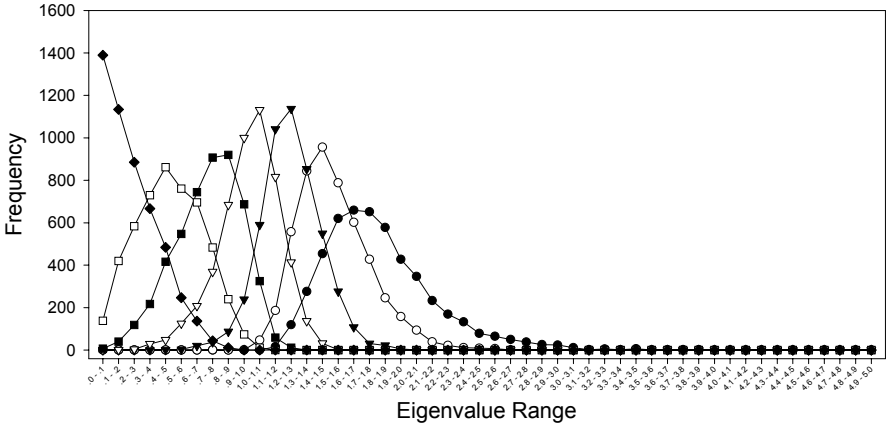


Figure 2.9: Eigenvalue distributions from UCM (Panel A), MCMC (Panel B), and DH (Panel C) algorithms for 7×7 matrices

2.4 Summary

In summary, the MCMC method was judged to provide correlation matrices indistinguishable from those generated by the UCM method, yet at a much faster rate. Exactly how much faster depends on several factors, most notably matrix order, hardware limitations, the number of matrices to be generated. What took minutes or hours to achieve with MCMC would take weeks or years with UCM. DH, although comparatively very fast because no generated matrices are rejected, yielded matrices that did not possess the desired distributional properties. In fairness, however, it should be noted that assessment of the DH algorithm by creating histograms of correlation coefficients may reflect more upon the method used to generate eigenvalues than the DH algorithm itself, which was designed with the intent of generating random correlation matrices with a given spectrum. The utility of the DH method for generating random matrices of the type required here is limited because the distributional properties of random spectra remain largely uninvestigated. Matrices generated using the MCMC method will be used in the remainder of this study.

In Chapter 2 a novel method of generating random correlation matrices using an MCMC algorithm was proposed and executed. Chapter 3 will illustrate the negative consequences associated with ignoring model complexity in the SEM context by applying a series of models to random data generated using the MCMC method.

CHAPTER 3

INVESTIGATING MODEL COMPLEXITY IN STRUCTURAL EQUATION MODELING

3.1 Overview of Model Fitting Strategy

Chapter 2 introduced a novel approach to generating random dispersion matrices which are uniformly representative of the data space relevant to SEM. In this chapter, structural equation models will be fit to the generated data in order to (1) assess the degree of complexity of particular models and (2) gain an understanding of the kinds of problems that could be encountered in fitting models to randomly generated data. One problem observed in preliminary modeling trials was a high rate of nonconvergence associated with ML estimation for some models. The pattern of constraints characterizing some models may be very unreasonable given some potentially observable data patterns. As long as there is at least one degree of freedom in a particular model, there will be some data patterns for which the model is simply inappropriate (grossly misspecified). Gross misspecification is often reflected in nonconvergence in addition to poor fit when ML estimation is used. Thus, convergence problems can be expected to occur with some frequency when models are applied to random data patterns using ML estimation. Because of the nonconvergence problem associated with ML estimation, the unweighted

(ordinary) least squares (ULS; OLS) discrepancy function was chosen instead. OLS estimation almost always converges within a reasonable number of iterations and always results in a fit statistic.

The goal of the model-fitting phase will be to compare alternative or competing models in terms of the proportion of the data space which the models fit well by some criterion. Specifically, each model will be fit to 5000 randomly generated data matrices, and cumulative distributions of fit indices will be constructed for each model. To accomplish these tasks, a Fortran program (see Appendix B) was written to construct LISREL syntax files (see Appendix C). After using LISREL to fit candidate models to random data in large batch runs, another Fortran program (see Appendix D) was used to extract information about fit and model convergence from LISREL output files. The results are summarized in plots.

Several model testing scenarios are proposed. Because the purpose of this chapter will be to illustrate the consequences of ignoring complexity during model evaluation, it is first necessary to illustrate that models which predict the same data patterns are equally complex. To this end, equivalent models will be fit to the same randomly generated data. It is expected that equivalent models will be equally complex because equivalent models are simply reparameterizations of each other – they should fit the same data equally well.

Second, it is important to demonstrate that models with different numbers of free parameters differ in the number of random data patterns they can fit well by some criterion. Thus, it would be of interest to specify models nested in terms of parameters and fit them to the same randomly generated data. Specifically, 0-, 1-, 2-, and 3-factor

unrestricted factor analysis models will be specified and fit to data. In addition, because an equality constraint placed on a pair of parameters functionally represents the loss of one free parameter (and thus the gain of one degree of freedom), the effect on model complexity of imposing equality constraints will be examined. It is expected that, in agreement with established literature on model complexity, models with more free parameters (those with more factors and fewer equality constraints) will fit a larger proportion of the data space by some criterion of good fit than those with fewer free parameters.

Third, and most importantly, it is crucial to demonstrate that functional form contributes to model complexity. Thus, it would be of interest to specify non-nested models with the same number of free parameters and fit them to the same random data. If these models are found to fit different proportions of the data space well by the same criterion, that finding would constitute a different illustration of how comparatively more complex models can fit more data, and how complexity does not rely solely upon the number of free parameters. The effect of placing equality constraints on different sets of parameters will be presented as a special case of altering functional form.

Fourth, it would be of interest to specify models with the same structure and the same number of parameters and degrees of freedom, but in which the allowable parameter range differs. For example, an inequality constraint placed on an otherwise free parameter will not alter *df* or the number of parameters (Rindskopf, 1984), but should reduce model complexity in some cases, depending on the kind of inequality constraint imposed.

Finally, Chapter 4 will explore the consequences of ignoring model complexity in practice by evaluating theoretically derived models which have been applied to real data in the past. It is likely that there exist many examples in which a theoretically derived (but rejected) model was almost certain to be falsified on the basis of low complexity. Similarly, there likely exist many examples of empirically supported models which have the a priori ability to fit almost any given data set well. Furthermore, identifying a situation in which one model was chosen as superior to another on the basis of superior fit, and then demonstrating that selection of that model was probably due in part to differences in inherent model complexity, could have significant consequences for practice.

3.2 Demonstration of Equal Complexity: Equivalent Models

Before proceeding to fitting models expected to differ in complexity, it would be helpful to provide an example in which competing models are *not* expected to differ in complexity. Models expected to be equally complex are *equivalent models* – that is, models with seemingly different structures which nevertheless can be shown to fit the same data sets equally well. To this end a Monte Carlo investigation was undertaken using Fortran 90 programs and LISREL 8.53 software for fitting structural equation models. LISREL was chosen for its ability to quickly estimate large numbers of models sequentially. Two equivalent models (depicted in Figures 3.1 and 3.2) were specified and

fit to 5000 randomly generated correlation matrices. Sample size (*N*) was set to 1000 for this analysis and all subsequent analyses.[11] Most models converged to solutions.[12]

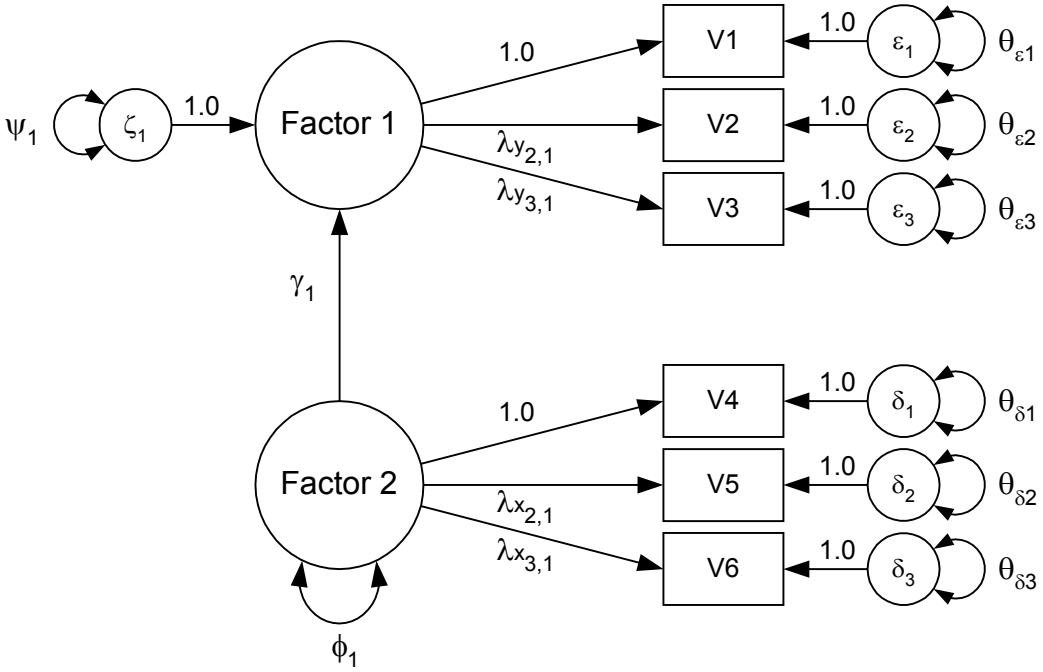


Figure 3.1: Model A – Path model with Factor 1 regressed on Factor 2

[11] Sample size should have no effect on results. *N* affects the precision of parameter estimates in OLS, but does not alter the point estimates themselves.

[12] In Monte Carlo studies such as this, it is equally legitimate to include improper solutions, exclude them, or constrain solutions to be proper (Gerbing & Anderson, 1993). The output command AD=OFF turns off LISREL's automatic admissibility check. Because there was no a priori reason to exclude improper solutions, AD=OFF was specified for every model. In this and all subsequent analyses, the maximum number of iterations was set to a high number (1000) to allow a reasonable amount of time for slow-converging models to reach convergence. Although not all models converged, fit indices were always produced which indicated the best fit obtainable after 1000 iterations.

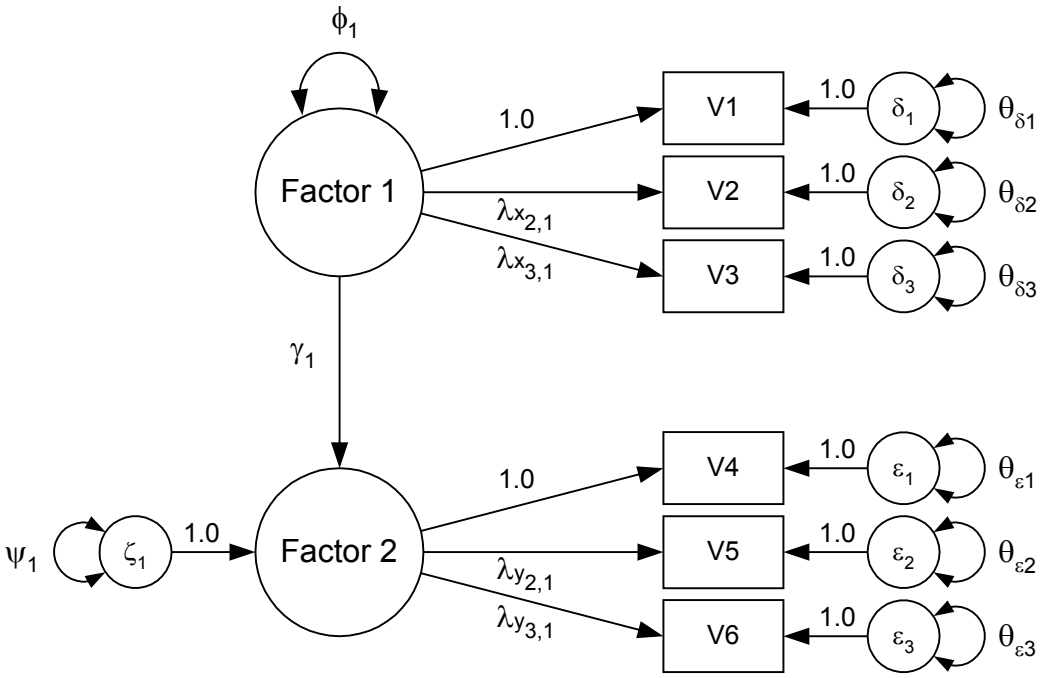


Figure 3.2: Model B – An equivalent model with Factor 2 regressed on Factor 1

Cumulative probability distribution functions (CDFs) of the root mean squared residual (RMSR)[13], a simple measure of goodness of fit uncomplicated by penalties for model complexity, are plotted in Figure 3.3. For simplicity, and because it provides an intuitive common metric across models fit to matrices of different orders, RMSR will be used as the sole fit index for model comparison for the remainder of this dissertation. The RMSR CDFs in Figure 3.3 exactly overlap, confirming that Models A and B are, as expected, equally complex in the sense that they would fit the same data patterns equally well.

[13] RMSR (also abbreviated RMR) is here defined as the square root of the mean squared non-duplicated elements of the reproduced correlation matrix. Hence, $\text{RMSR} = \sqrt{2\widehat{F}_{OLS} / \left(p\left(p+1 \right) \right)}$, where $\widehat{F}_{OLS}$ is the minimum OLS discrepancy function value (the sum of squared nonduplicated off-diagonal residuals). For correlation matrices, RMSR equals the more generally applicable standardized root mean squared residual (SRMR; Bentler, 1995).

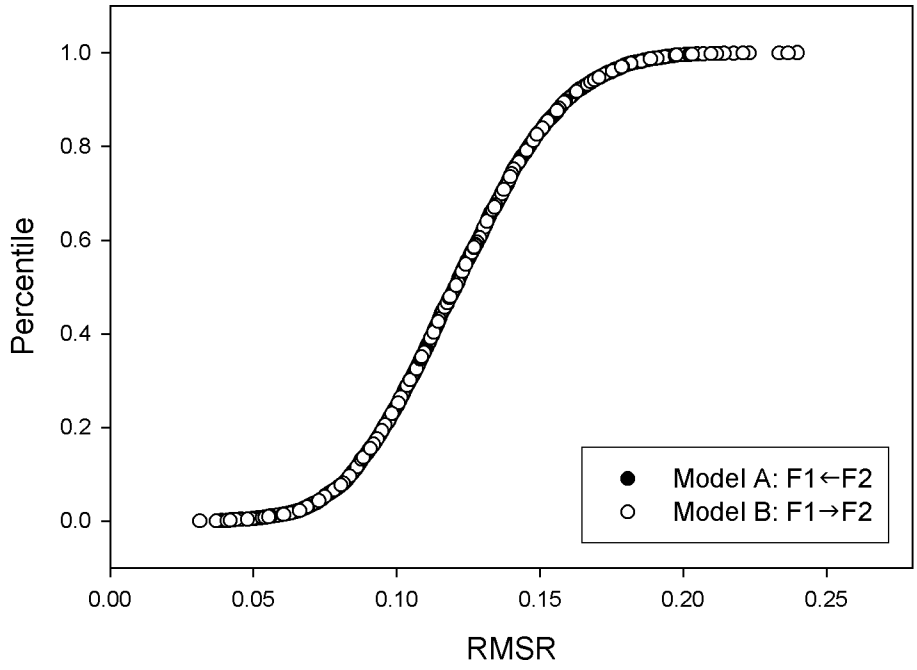


Figure 3.3: Cumulative frequency distributions of RMSR for two equivalent structural equation models

From cumulative frequency plots of RMSR (see Figure 3.3), the percentage of data patterns fit well by some criterion of good fit may be ascertained. In their Monte Carlo investigation of cutoff criteria for common ML-based fit indices, Hu and Bentler (1999) recommend a two-index strategy, such that the ML-based standardized RMSR is combined with some other ML-based fit index to determine whether or not a model should be retained. Hu and Bentler found that, in general, an RMSR cutoff of at least .09 is preferable for sample sizes less than 500 and a cutoff of at least .06 is preferable for sample sizes greater than 1000 when used in conjunction with the TLI. Cutoff values of RMSR greater than .09 are preferable when used in conjunction with BL89 (Bollen,

1989), RNI (McDonald & Marsh, 1990), CFI (Bentler, 1995), Gamma Hat (Steiger, 1989), Mc (McDonald, 1989), and RMSEA. They recommend that a cutoff value close to .08 be chosen when RMSR is used in isolation, although a value closer to .09 seems to be more appropriate when used in conjunction with other fit indices. Byrne (1989) offers RMSR = .05 as a criterion for good fit, but in light of the findings of Hu and Bentler (1999) and experience fitting models to random data, this criterion was judged far too conservative to be of use. To the extent that cutoff criteria established for the ML-based RMSR can be extended to the OLS-based RMSR, the criterion of .08 is used in this dissertation for convenience. This (ultimately arbitrary) criterion is used merely as a basis for comparison, not as a statement that a cutoff of .08 should always be used with RMSR. Using a cutoff value of .08 for RMSR, the percentage of solutions meeting the criterion of good fit for both equivalent models is 7.38%. Using a Venn diagram similar to that in Figure 1.2, the abilities of Models A and B to fit random data are illustrated in Figure 3.4.

This section confirmed the expectation that formally equivalent models are equally complex in the sense that they fit identical proportions of the data space well by any criterion. This result is consistent with the prediction that reparameterized versions of the same model should be equally complex (Pitt et al., 2002). This was shown by demonstrating that the RMSR CDFs overlapped perfectly. Of greater interest for present purposes, however, are situations in which these CDFs do not necessarily overlap.

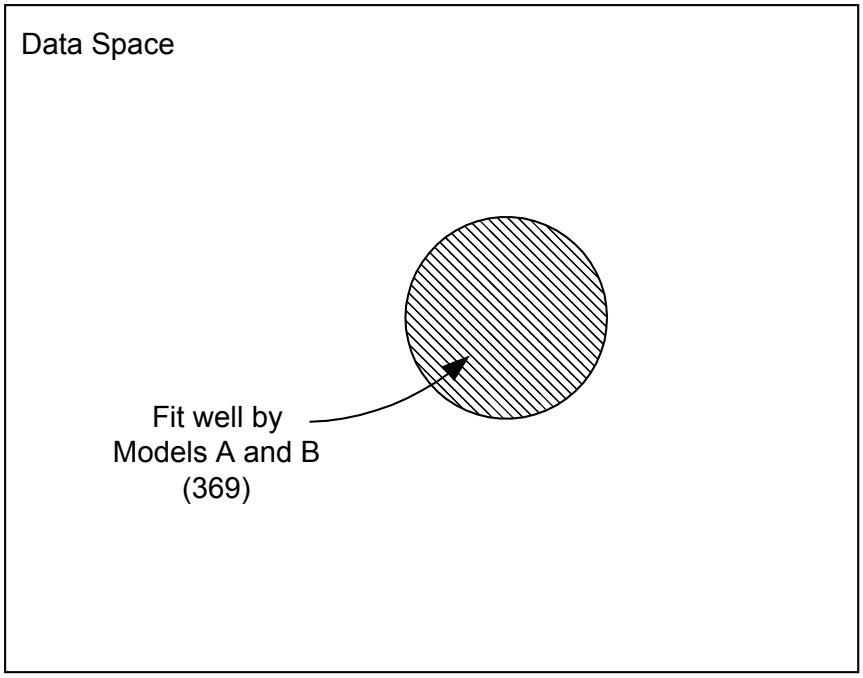


Figure 3.4: Venn diagram representing regions of the data space fit well by equivalent models A and B (drawn to scale)

3.3 Complexity Due to Number of Free Parameters: Nested Models

In the second step of the model-fitting strategy, four unrestricted factor models (0, 1, 2, and 3 factors) were fit to 6×6, 9×9, and 12×12 matrices randomly generated using the MCMC algorithm already described, and to 6×6 matrices randomly generated using the UCM algorithm for comparison purposes. Cumulative distribution functions of RMSR are plotted in Figures 3.5 – 3.8. Even though RMSR represents a common metric for comparison across matrix orders, the pattern of results differs across plots. These discrepancies can be attributed to differing number of parameters, degrees of freedom, and distributional characteristics of the correlations to be fit (see the histograms in Figures 2.2 – 2.5).

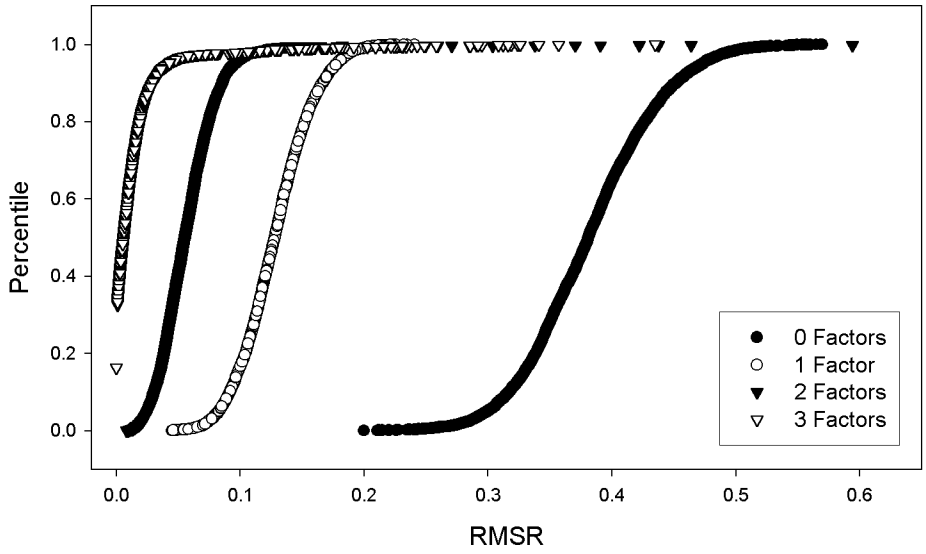


Figure 3.5: Cumulative frequency distributions of RMSR from UCM algorithm for 6×6 matrices

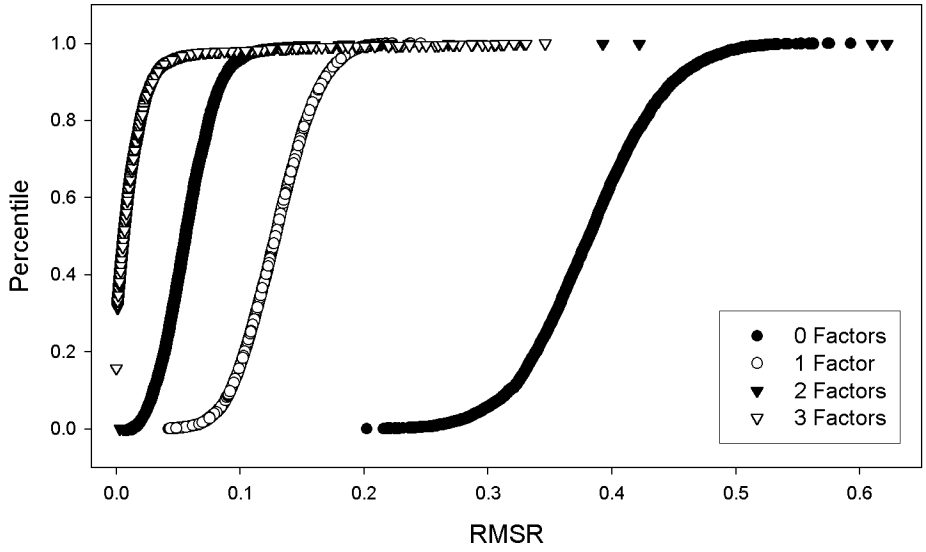


Figure 3.6: Cumulative frequency distributions of RMSR from MCMC algorithm for 6×6 matrices

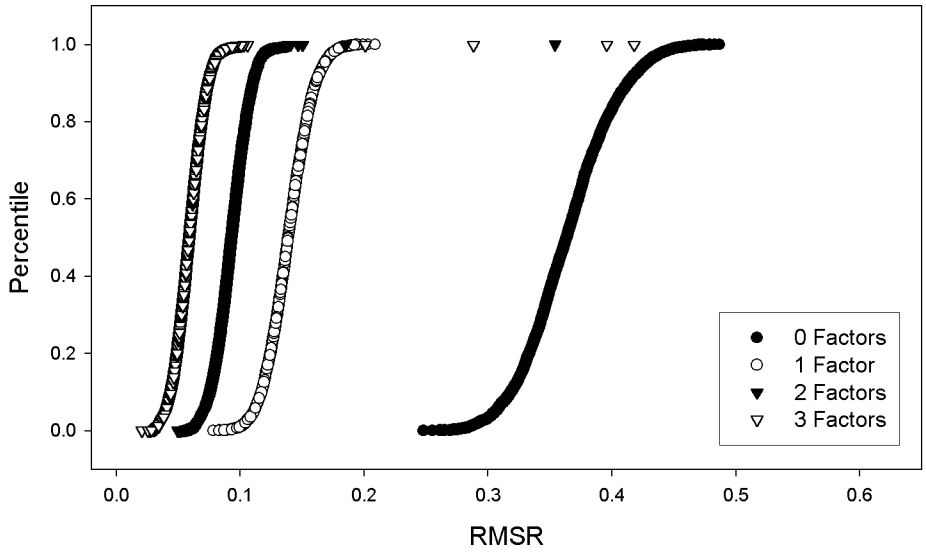


Figure 3.7: Cumulative frequency distributions of RMSR from MCMC algorithm for 9×9 matrices

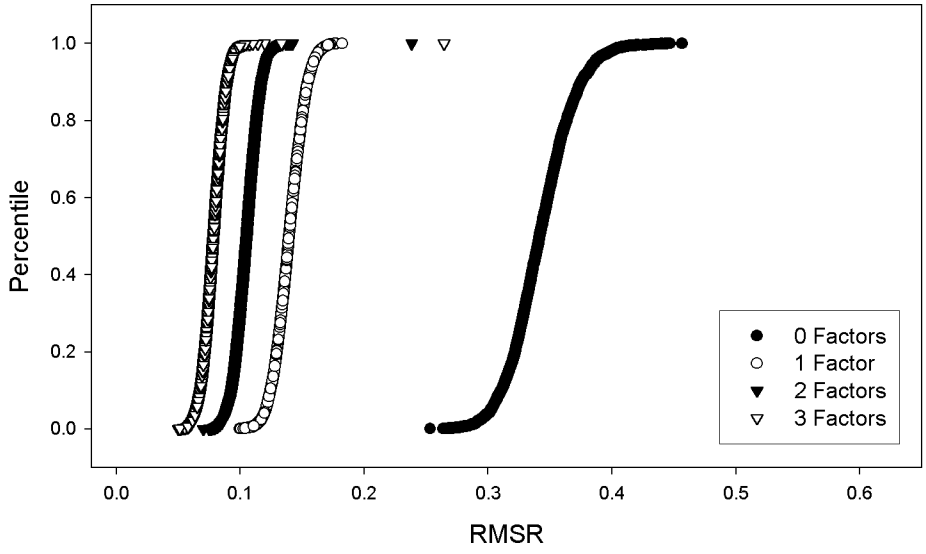


Figure 3.8: Cumulative frequency distributions of RMSR from MCMC algorithm for 12×12 matrices

Figure 3.9 depicts mean plots of RMSR using all 5000 solutions from every cell of the design. Figure 3.10 contains a plot similar to that in Figure 3.9, but using only those solutions with no convergence problems. See Tables 3.1 and 3.2 for means and sample sizes associated with Figures 3.9 and 3.10. Even though the sample size was sometimes substantially reduced by removing solutions with convergence problems (see Table 3.2), the mean plots for RMSR in Figures 3.9 and 3.10 appear virtually identical. In agreement with the hypothesis, models with more factors yielded, on average, smaller values of RMSR.

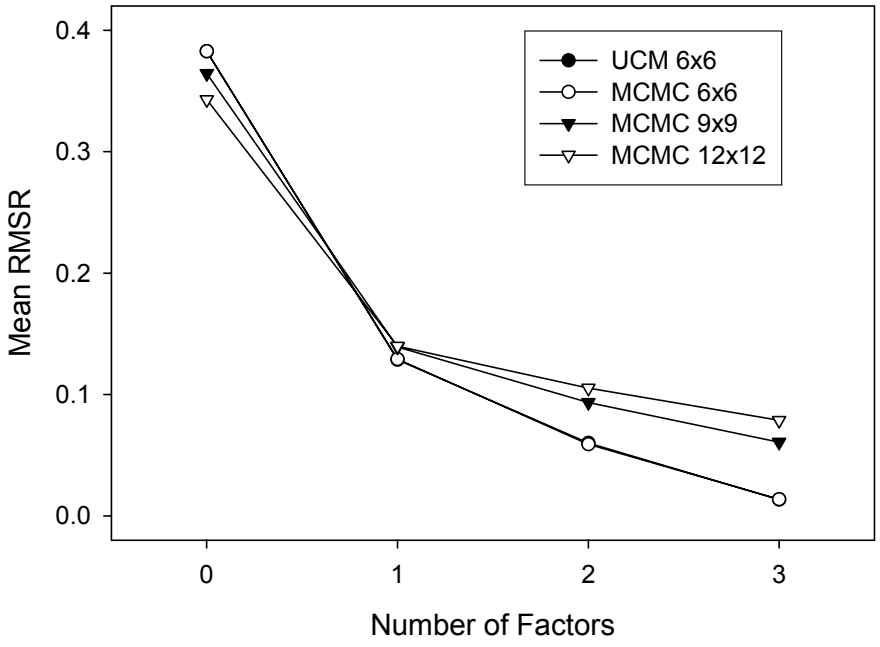


Figure 3.9: Mean plots of RMSR using all solutions

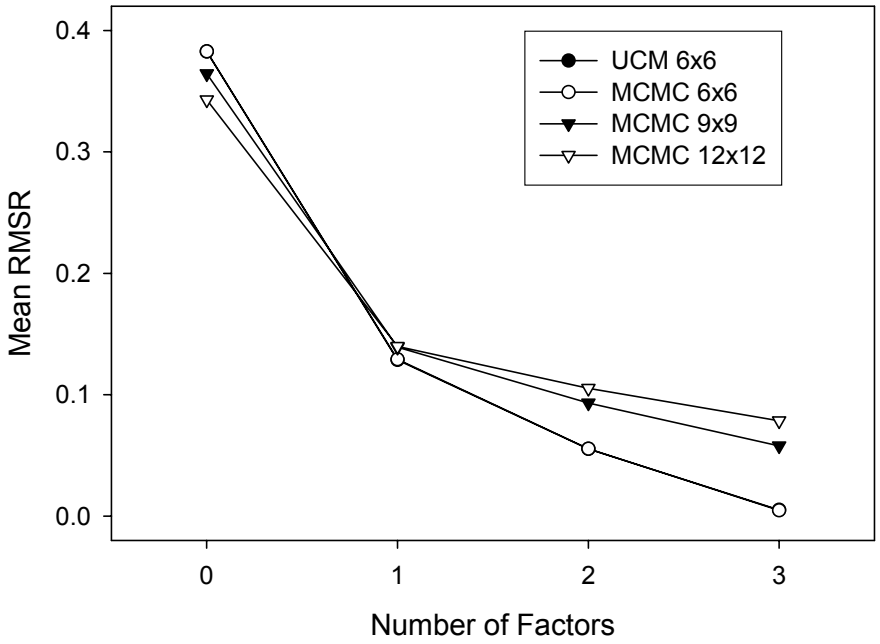


Figure 3.10: Mean plots of RMSR using only those solutions with no convergence problems

Note that the CDFs for UCM 6×6 and MCMC 6×6 (Figures 3.5 and 3.6) appear virtually identical, and that the lines for UCM 6×6 and MCMC 6×6 completely overlap in Figures 3.9 and 3.10. The high degree of similarity in RMSR distributions for these two sets of matrices further supports the decision to use an MCMC algorithm for generating random correlation matrices.

Using a cutoff value of .08 for RMSR, the percentages of solutions meeting the good fit criterion for all factor models are reported in Table 3.3. According to Table 3.3, virtually all data sets were fit well by this criterion using the 3-factor model, especially for lower-order matrices. This should come as no surprise in the case of 6×6 matrices,

Algorithm	*p*	*m*	*n*	mean RMSR
UCM	6	0	5000	.3825
UCM	6	1	5000	.1286
UCM	6	2	5000	.0601
UCM	6	3	5000	.0135
MCMC	6	0	5000	.3827
MCMC	6	1	5000	.1291
MCMC	6	2	5000	.0590
MCMC	6	3	5000	.0138
MCMC	9	0	5000	.3644
MCMC	9	1	5000	.1391
MCMC	9	2	5000	.0935
MCMC	9	3	5000	.0608
MCMC	12	0	5000	.3429
MCMC	12	1	5000	.1398
MCMC	12	2	5000	.1053
MCMC	12	3	5000	.0788

Note. *n* refers to the number of samples included in the simulation. Sample size (*N*) for each solution was set to 1000.

Table 3.1: Mean RMSR values for simulations involving nested factor models, all solutions

because these models were just-identified (there were exactly as many model parameters as there were nonduplicated correlations, therefore $df = 0$). Barring convergence and estimation problems, the fit of these models should have been perfect, such that RMSR = 0. In fact, for 3-factor models fit to 6×6 matrices, RMSR = 0 for 32.62% and 31.42% of sample matrices generated with the UCM and MCMC algorithms, respectively. At the

Algorithm	*p*	*m*	*n*	mean RMSR
UCM	6	0	5000	.3825
UCM	6	1	5000	.1286
UCM	6	2	3492	.0554
UCM	6	3	1911	.0051
MCMC	6	0	5000	.3827
MCMC	6	1	5000	.1291
MCMC	6	2	3478	.0555
MCMC	6	3	1813	.0047
MCMC	9	0	5000	.3644
MCMC	9	1	5000	.1391
MCMC	9	2	4519	.0932
MCMC	9	3	3406	.0579
MCMC	12	0	5000	.3429
MCMC	12	1	5000	.1398
MCMC	12	2	4753	.1052
MCMC	12	3	4368	.0786

Table 3.2: Mean RMSR values for simulations involving nested factor models, solutions with convergence problems removed

other extreme, the 0-factor model fit no sample matrices well by the .08 criterion. Very small percentages of sample matrices were fit well according to the .08 criterion when 1-factor models were fit to 6×6 and 9×9 matrices and when 2-factor models were fit to 9×9 and 12×12 matrices. If theory posited that a 2-factor model should explain the correlations among a set of 12 measured variables, an RMSR of .08 would be very impressive in light of the fact that such a model would fit less than one-half of one percent of random data matrices at least that well. On the other hand, if theory posited

that a 3-factor model should explain the correlations, an RMSR of .08 would not be so impressive because over half of the random 12×12 matrices were fit at least that well.

Algorithm	*p*	0-factor	1-factor	2-factor	3-factor
UCM	6	0.00	3.94	85.02	97.68
MCMC	6	0.00	4.04	84.98	97.82
MCMC	9	0.00	0.02	16.14	95.62
MCMC	12	0.00	0.00	0.48	56.34

Table 3.3: Percentage of factor models meeting the RMSR < .08 criterion for good fit

Table 3.4 contains a comparison of frequencies of random data sets fit well by a given model relative to those fit well by competing models for 9×9 matrices. Each entry in the right column represents the number of data sets fit well (by the RMSR = .08 criterion) by the model represented in the left column. Thus, 218 data sets were not fit well by any of the factor models, and no data sets were fit well by all of the models. The 3-factor model fit 4781 (3975 + 805 + 1) data sets well. Of these, 806 (805 + 1) were also fit well by the 2-factor model. Of these, only one data set was also fit well by the 1-factor model. The 0-factor model fit no data sets well. These results are close to what one would expect to see with nested models. Theoretically, the regions of the data space fit well by models with more factors should subsume regions fit well by more restricted models because 0-, 1-, 2-, and 3-factor models are hierarchically nested in terms of parameters.

However, due to the occasional occurrence of estimation problems, Table 3.4 contains a violation of this general principle; one data set was fit well by the 2-factor model but not by the 3-factor model. In addition to this violation, there were 11 cases in which the fit of one model was better than that of at least one model with more factors. Except for the one violation reported in Table 3.4, these results are also presented in Venn diagram form in Figure 3.11.

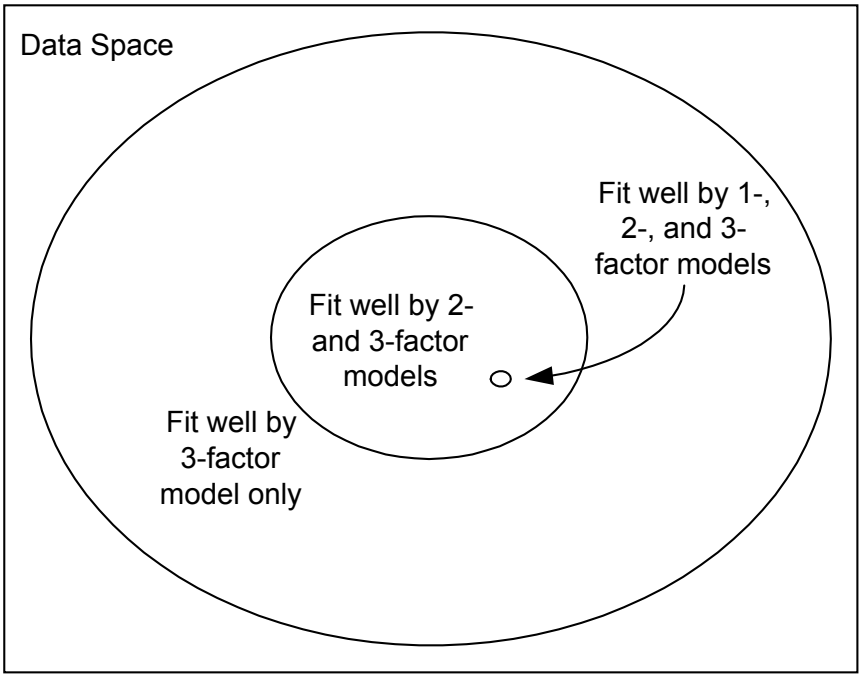


Figure 3.11: Venn diagram representing regions of the data space fit well by factor models (not drawn to scale)

Model	Frequency
No model	218
0 only	0
1 only	0
2 only	1
3 only	3975
0 and 1 only	0
0 and 2 only	0
0 and 3 only	0
1 and 2 only	0
1 and 3 only	0
2 and 3 only	805
0, 1, and 2 only	0
0, 1, and 3 only	0
0, 2, and 3 only	0
1, 2, and 3 only	1
0, 1, 2, and 3	0

Note. Numbers in the first column refer to the number of factors.

Table 3.4: Frequencies of data sets fit well by combinations of factor models

These first attempts to fit models differing in complexity to random correlation matrices were encouraging, and conformed entirely to expectations. As more factors are included to account for correlations among a given number of MVs, more parameters are introduced into the model in the form of factor loadings and factor correlations. Because factor models with increasing numbers of factors are nested in terms of parameters, models with more factors should always fit better than models with fewer parameters when applied to the same data, barring convergence problems and improper solutions. This expectation was borne out in the present example, as is illustrated in Figures 3.5 –

3.10, in which models with fewer factors (and therefore fewer parameters) are characterized by higher values of RMSR.

3.4 Complexity Due to Number of Free Parameters: Equality Constraints

Often theory dictates that two model parameters will be equal in the population, as in the case in which two predictor variables are posited to have an equal influence on some dependent variable. Thus, another constraint commonly imposed in SEM is an *equality constraint*. Equality constraints represent the requirement that two or more otherwise free parameters must be equal to each other. Equality-constrained parameters can best be understood as single parameters which perform two functions. Hypotheses about the equality of two parameters are easily testable within the SEM framework. Every equality constraint adds one degree of freedom to the model. The model with the imposed equality constraint(s) is nested in a model without those constraints, so the legitimacy of imposing the constraints (or, equivalently, freeing them) can be tested by means of the χ^2 difference test with degrees of freedom equal to the number of equality constraints imposed.

To investigate the effects of imposing equality constraints on model complexity, two models were specified. In Model A, an exogenous latent variable (ξ_1) is hypothesized to exert an effect on an endogenous latent variable (η_1), which in turn exerts an effect on a second endogenous LV (η_2). In other words, η_1 is hypothesized to completely mediate the effect of ξ_1 on η_2 (see Figure 3.12). Model B is identical to Model A, save for the fact that all unique variances associated with indicators of a given LV are constrained to

equality. Thus, the set of six equality constraints $\{\theta_{\delta 1} = \theta_{\delta 2} = \theta_{\delta 3};\ \theta_{\varepsilon 1} = \theta_{\varepsilon 2} = \theta_{\varepsilon 3};$ $\theta_{\varepsilon 4} = \theta_{\varepsilon 5} = \theta_{\varepsilon 6}\}$ is imposed.

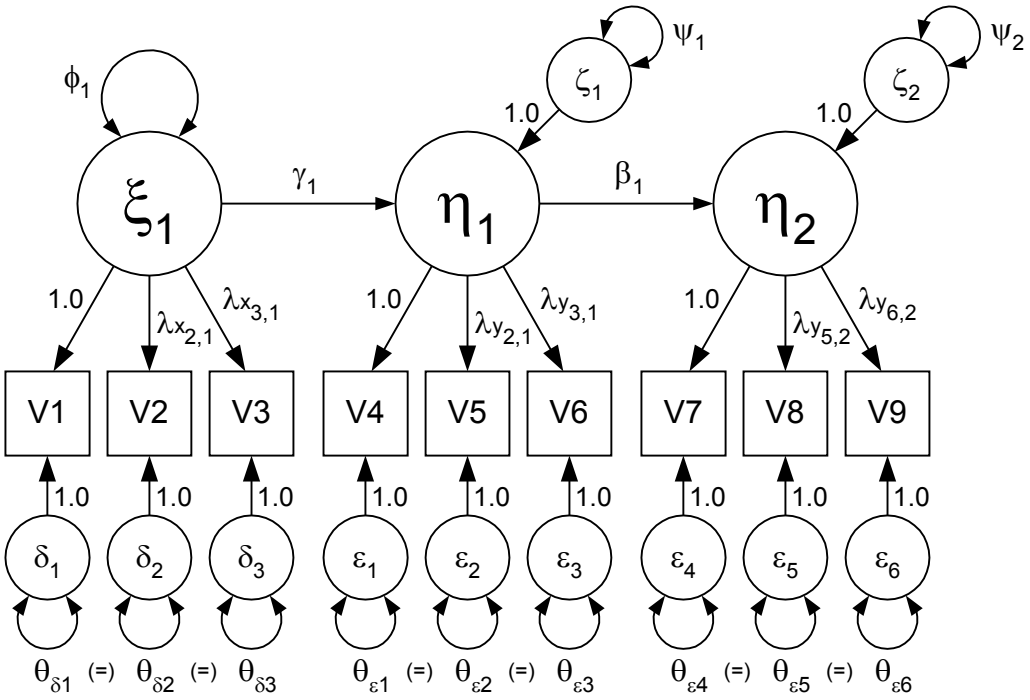


Figure 3.12: A simple structural equation model. Model A has no constraints on unique variances. Model B constrains within-factor unique variances (θs) to equality (i.e., Model B adds six constraints)

Figure 3.13 shows CDFs of RMSR for the two models. As expected, Model B (the version with equality constraints) is less complex than Model A, in the sense that it fits a smaller proportion of the data space well than does Model A. However, despite the apparent superiority of Model A relative to Model B in terms of fit, neither of these models fits random data particularly well. Only three solutions in the unconstrained condition (Model A) and no solutions in the constrained condition (Model B) yielded an RMSR less than .08.

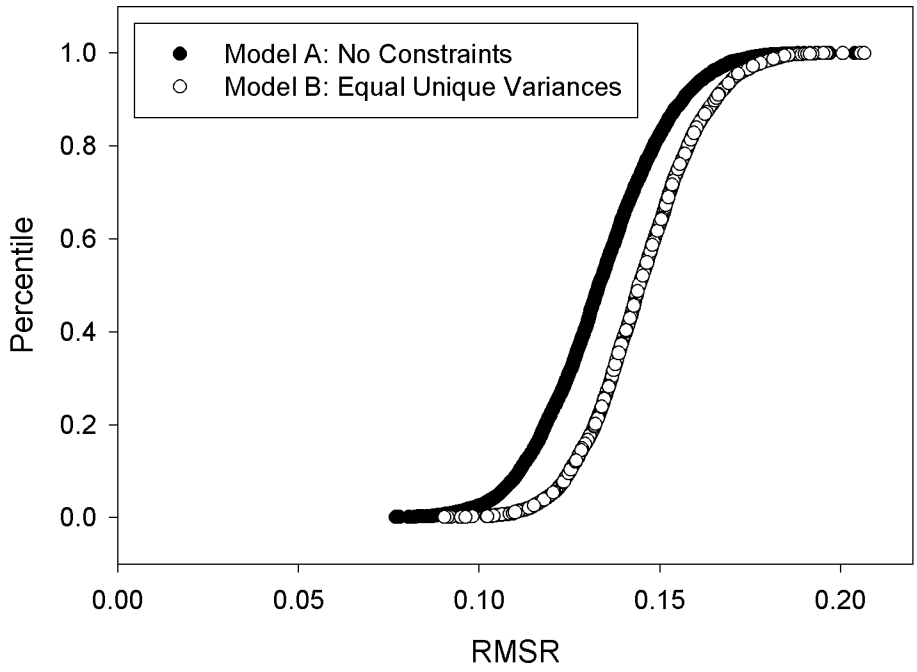


Figure 3.13: Cumulative frequency distributions of RMSR illustrating the effect of adding equality constraints

3.5 Complexity Due to Number of Free Parameters: Different Constraint Types

In the preceding sections it was shown how imposing fixed-value constraints (§3.3) and equality constraints (§3.4) yields models with lower complexity. It is reasonable to predict that a model with an equality constraint will possess different complexity than a model in which one of those parameters is constrained to equal a particular value. Despite the unit increase in *df* per constraint, equality-constrained parameters still have the potential to vary over a range of possible values to achieve a good fit to data, whereas a parameter constrained to equal a particular value does not. A parameter not subject to constraints has the potential to assume any value in its allowable

range. Which of the two constraint types will lead to a more complex model probably depends upon many model characteristics.

To illustrate how these different kinds of constraints can have drastically different effects on model complexity, two models, A and B, were specified and fit to 5000 random 6×6 correlation matrices. Both Models A and B were two-factor CFA models with three indicators loading on each factor, as in Figure 3.14. Model A involved constraining one loading $\left(\lambda_{x_{1,1}}\right)$ to equal zero. Model B involved imposing an equality

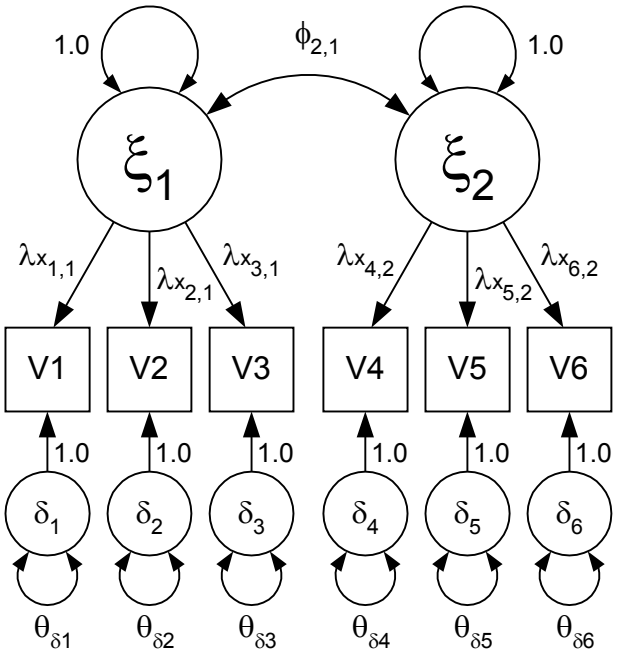


Figure 3.14: A simple CFA. Model A constrains $\lambda_{x_{1,1}}$ to equal zero. Model B constrains $\lambda_{x_{1,1}}$ to equal $\lambda_{x_{4,2}}$

constraint on two loadings $\left(\lambda_{x_{1,1}} = \lambda_{x_{4,2}}\right)$. Fitting these models to random data resulted in the highly discrepant RMSR distributions seen in Figure 3.15. Model A (with the fixed loading) yielded only one RMSR less than .08, whereas Model B (with the equality

constraint) yielded 143 RMSR values less than .08. Thus, despite the fact that the models share the same degrees of freedom, one is clearly more complex than the other.

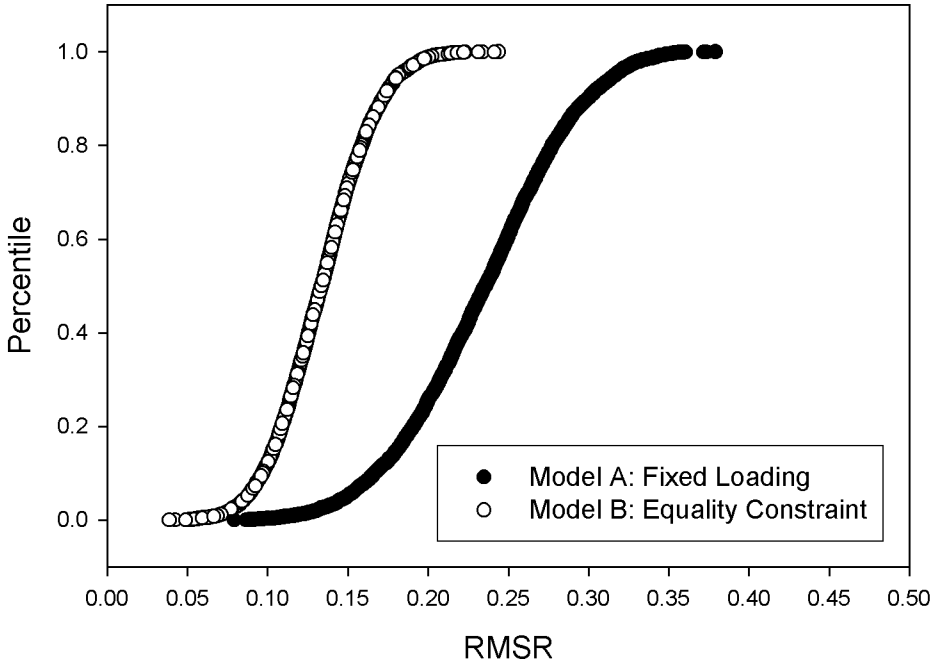


Figure 3.15: Cumulative frequency distributions of RMSR for two 2-factor CFA models

Thus far in the model-fitting strategy, there have been few surprising results. It is well known that reducing the number of free parameters in a model will adversely affect model fit. It follows that adding constraints to a model will reduce a model's complexity. Section 3.5 illustrated that the type of constraint imposed can have an additional bearing on model complexity. In fact, imposing different constraints amounts to fitting models with different functional forms. Section 3.6 will further illustrate how complexity may be influenced by functional form.

3.6 Complexity Due to Functional Form: Implied Zero Correlations

The next step in the modeling strategy was to fit models with the same number of free parameters, yet which differ in complexity due to functional form. This step is very important because it addresses the central concern of this dissertation; that is, it addresses the issue of models possessing the ability to fit data differentially well depending solely on differences in inherent complexity. Because traditional fit indices in the SEM paradigm typically correct for model complexity by adding some function of the number of free parameters, it is important to demonstrate how complexity does not depend solely on the number of free parameters. This issue was already addressed in §3.5 by showing how two models with the same *df* can differ in complexity.

To further address this issue, two models were specified, each a special case of a more general two-factor model (see Figures 3.16 and 3.17). Model A, depicted in Figure 3.16, specifies 12 MVs as functions of two uncorrelated factors, with one variable (V1) a function of both factors. Model B, depicted in Figure 3.17, specifies the same 12 MVs as functions of two correlated factors, but with no dual loadings for any MV. Thus, both Models A and B possess the same number of LVs, MVs, degrees of freedom, and free parameters. However, they differ in functional form complexity. To understand why, consider that in Model A, because Factor 1 and Factor 2 are specified to be orthogonal (uncorrelated), there will be 30 implied zeroes in the reproduced correlation matrix. For example, there is no way for the model to fit correlations involving variables from the set {V2 – V5} with variables from the set {V7 – V12}. Model B, however, has no implied zeroes because the factors are allowed to correlate; every observed correlation has the

potential to be fit to some extent. On the basis of the implied zeroes in Model A, it is expected that Model B will be more complex; i.e., that it will fit more of the data space by some fit criterion than will Model A.

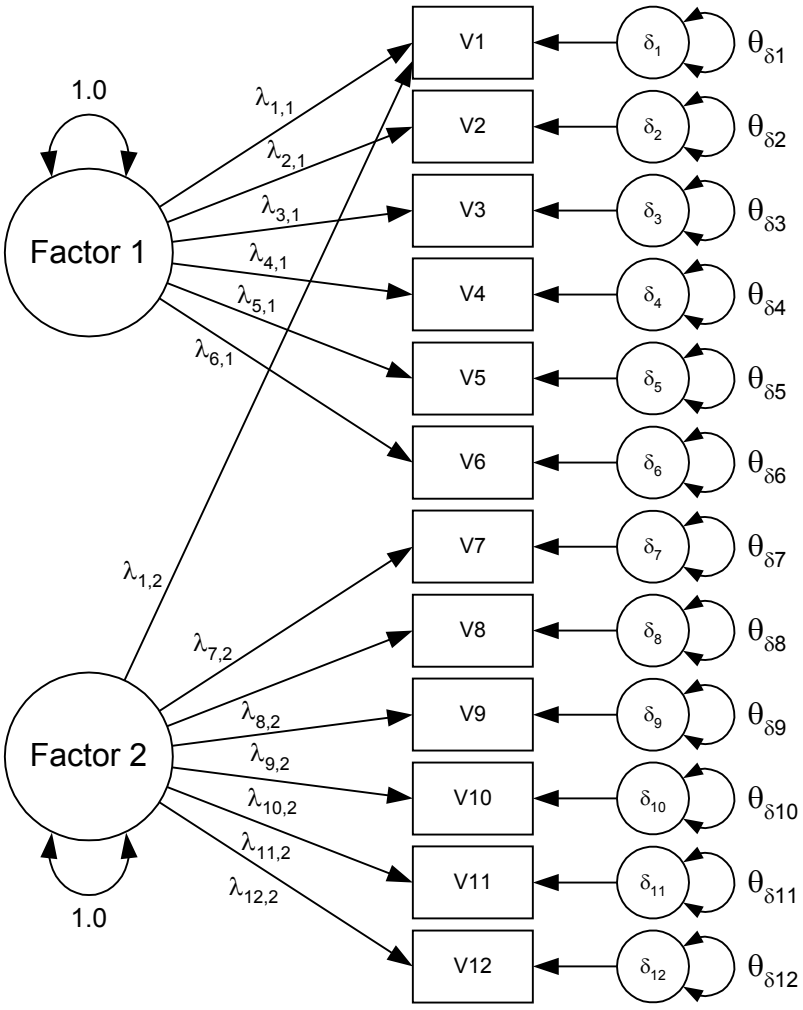


Figure 3.16: Model A – Orthogonal factors, extra factor loading of V1 on Factor 2

This expectation was tested by fitting both models to the same data, a set of random 12×12 correlation matrices. Using the same procedure described for earlier stages of model testing, Models A and B were fit to 5000 12×12 matrices generated with the MCMC algorithm. Cumulative distribution functions of RMSR are plotted in Figure 3.18. In agreement with expectations, the model considered more complex a priori (Model B) was superior to the less complex model (Model A) in terms of fit, even though

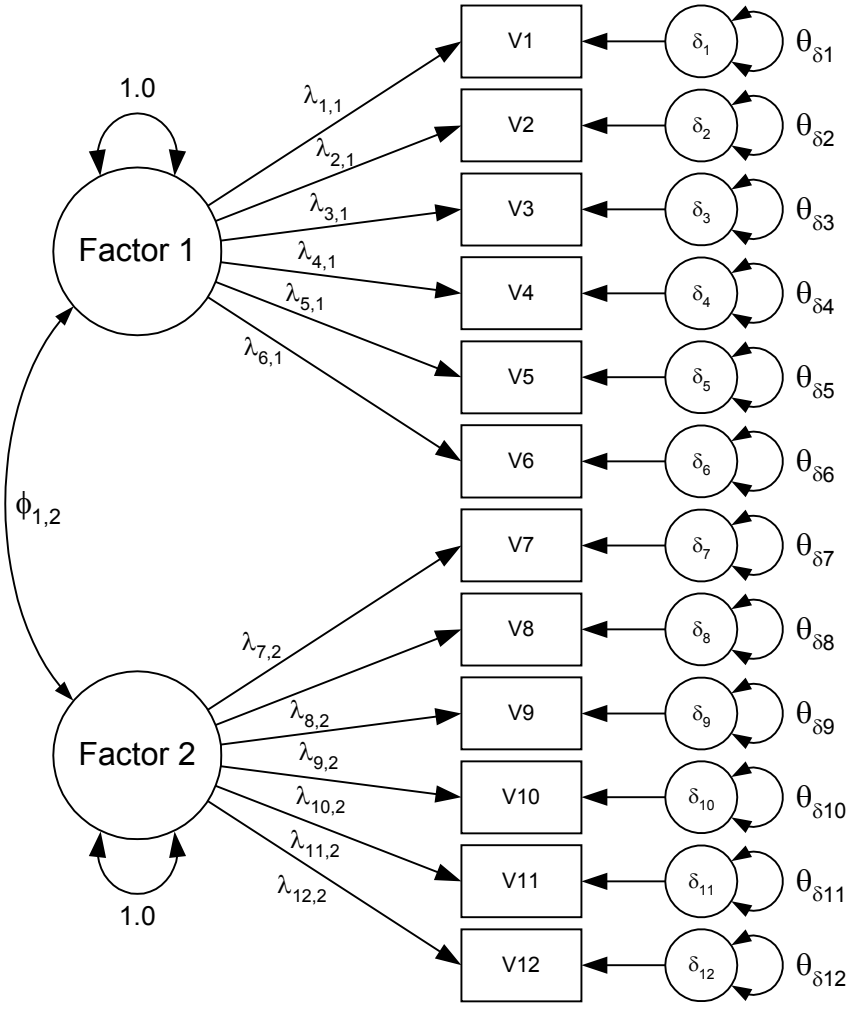


Figure 3.17: Model B – Correlated factors

the models share the same degrees of freedom and number of parameters and were fit to the same data. Admittedly, however, neither model showed particularly good fit to random data. Even the better fitting Model A yielded no RMSRs below .08.

To further illustrate the idea already presented in this section, a 4-factor CFA model was defined such that two factors (Factors 1 and 2) were each represented by four MVs, and the other two factors (Factors 3 and 4) were each represented by two MVs.

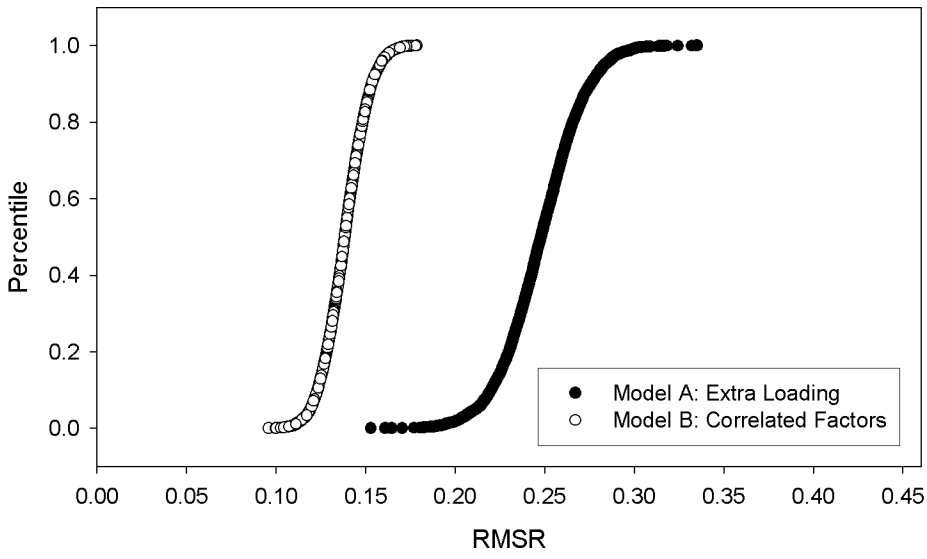


Figure 3.18: Cumulative frequency distributions of RMSR for non-nested models

This model is represented in Figure 3.19. Using this model as a baseline, three models were defined such that one parameter $\left(\phi_{1,2}, \phi_{2,3}, \text{or } \phi_{3,4}\right)$ was removed in each case. It was expected that removal of $\phi_{1,2}$ (i.e., constraining it to equal zero) would result in the most pronounced decrease in model fit because this constraint would result in 16 implied zero correlations. Removal of $\phi_{2,3}$ was expected to result in a smaller decrease in fit

because it implies only eight zero correlations. Removal of $\phi_{1,2}$ was expected to result in the smallest decrease in fit because it implies only four zero correlations.

As is illustrated by the cumulative distribution functions of RMSR in Figure 3.20, these expectations were confirmed. As the number of implied zeroes increases, model fit

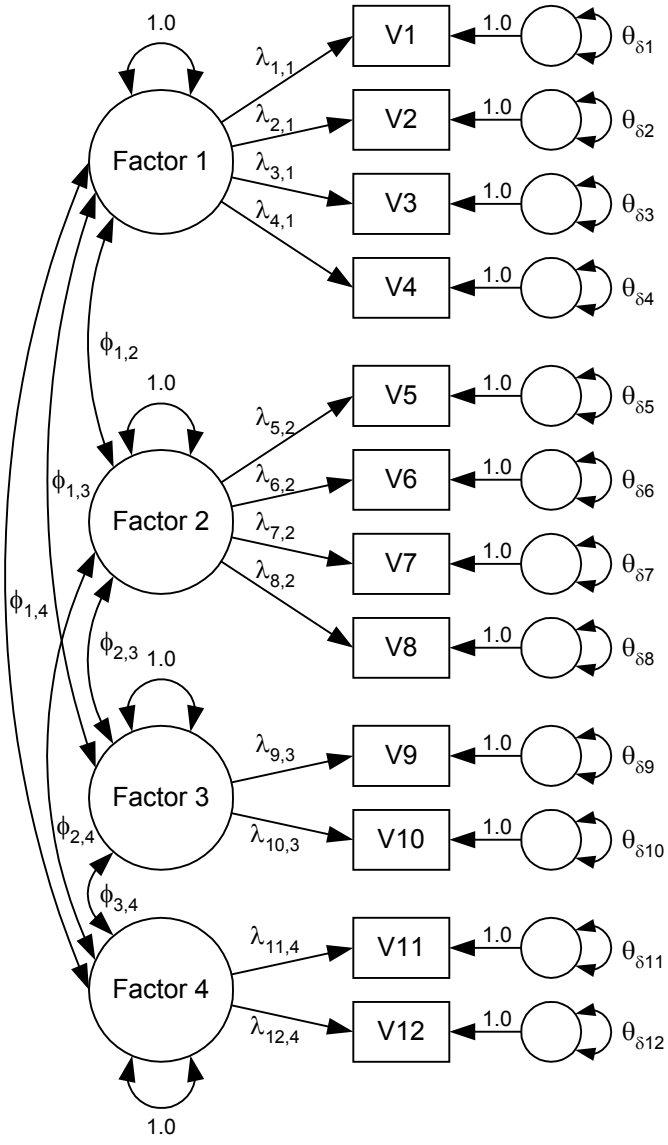


Figure 3.19: Baseline CFA model. Model A omits parameter $\phi_{1,2}$, Model B omits $\phi_{2,3}$, and Model C omits $\phi_{3,4}$

decreases, despite no change in degrees of freedom, the number of parameters, or the data to which the models are fit. None of the data sets fit by any of the three confirmatory factor models fit the data well by the RMSR = .08 criterion.

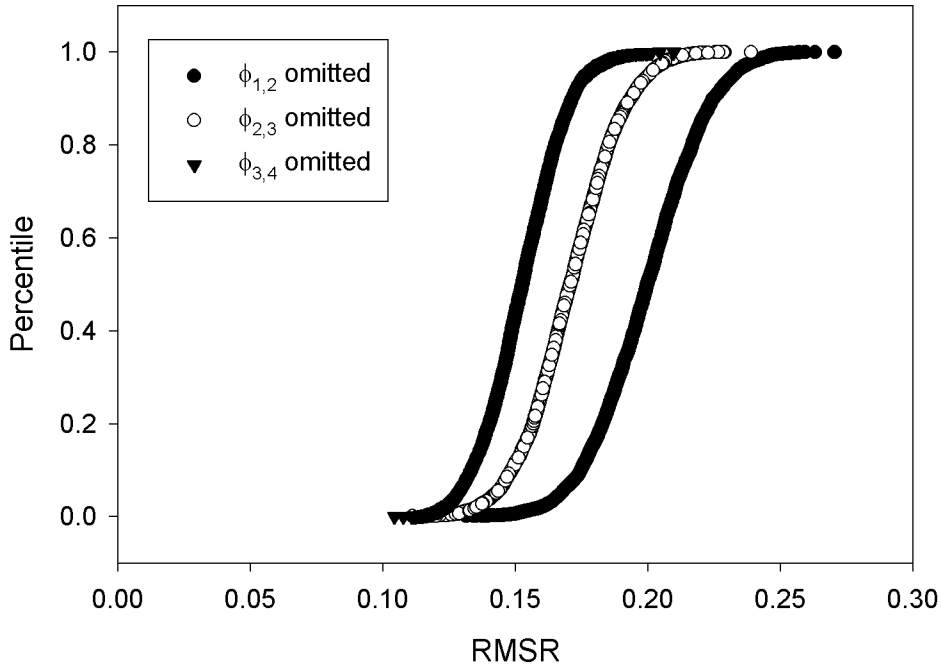


Figure 3.20: Cumulative frequency distributions of RMSR for three 4-factor models

3.7 Complexity Due to Functional Form: Different Regions of Good Fit

The results from fitting the models in §3.6 suggest that if two models fit different-sized regions of a correlation matrix well, then complexity should differ. The two examples in §3.6 represent extreme cases involving implied zero correlations, but the same principal should apply in general. For example, if two structurally different models imply no zeroes in the reproduced correlation matrix, yet fit certain subsets of

correlations differentially well, it is to be expected that there should be an observable effect on model complexity.

To illustrate this idea, consider the models depicted in Figures 3.21 and 3.22 (Models A and B, respectively). Model A is an unrestricted simplex model, generally applied in situations in which a band-diagonal correlation matrix is expected, as with repeated measurements of a single variable. Model B is a contrived 1-factor model, with two loadings constrained to equality so that Models A and B will have the same degrees of freedom (*df* = 10). Model B is expected to be more complex than Model A, because Model A is designed to account well only for correlation matrices conforming to a band-diagonal pattern. The 1-factor Model B can potentially fit many more correlation patterns. Thus, this pair of models is expected to provide a particularly good example of how two structurally different models, which nevertheless have the same degrees of freedom, may fit the same data differentially well without implied zero correlations.

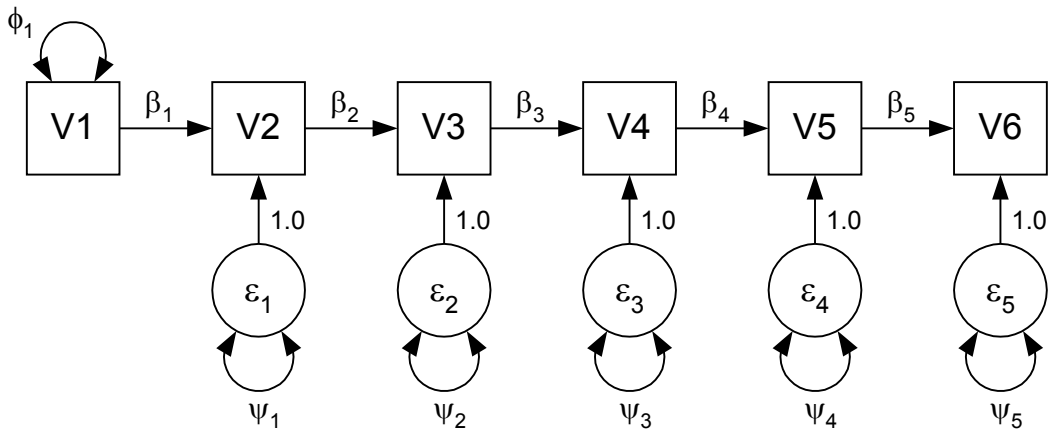


Figure 3.21: Model A – An unrestricted simplex model

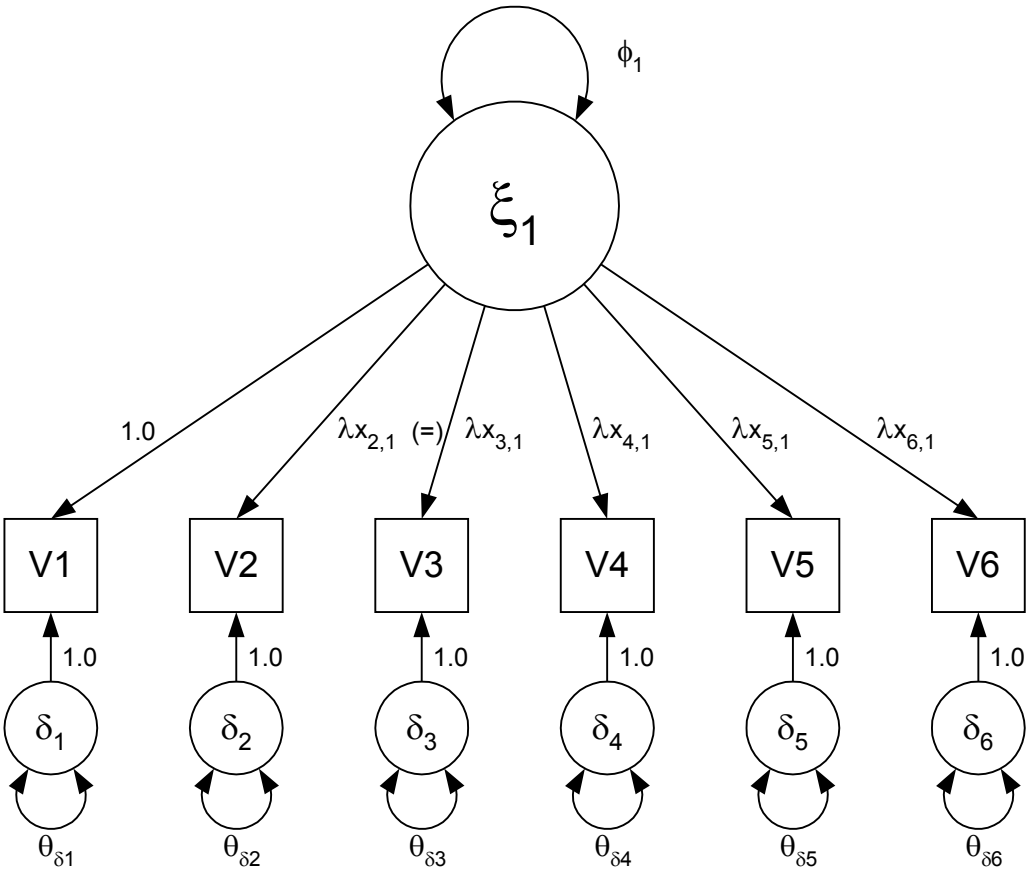


Figure 3.22: Model B – A 1-factor model with one equality constraint

Models A and B were each fit to 5000 randomly generated 6×6 matrices, with the resulting RMSR CDFs depicted in Figure 3.23. The factor model fit substantially better than the simplex model. Fifteen random data sets fit by Model A yielded RMSR values less than .08, whereas 67 data sets fit by Model B yielded RMSRs lower than .08. The large difference between Models A and B in model complexity can be attributed to differences in functional form. The simplex model fails to fit correlations between MVs which are distally connected, yet fits correlations between adjacent MVs very well. The factor model, on the other hand, can fit many more correlation patterns. Thus, it would

seem that, even when there are no implied zero correlations and when the same number of free parameters is involved, models differing in functional form can be quite different in terms of complexity.

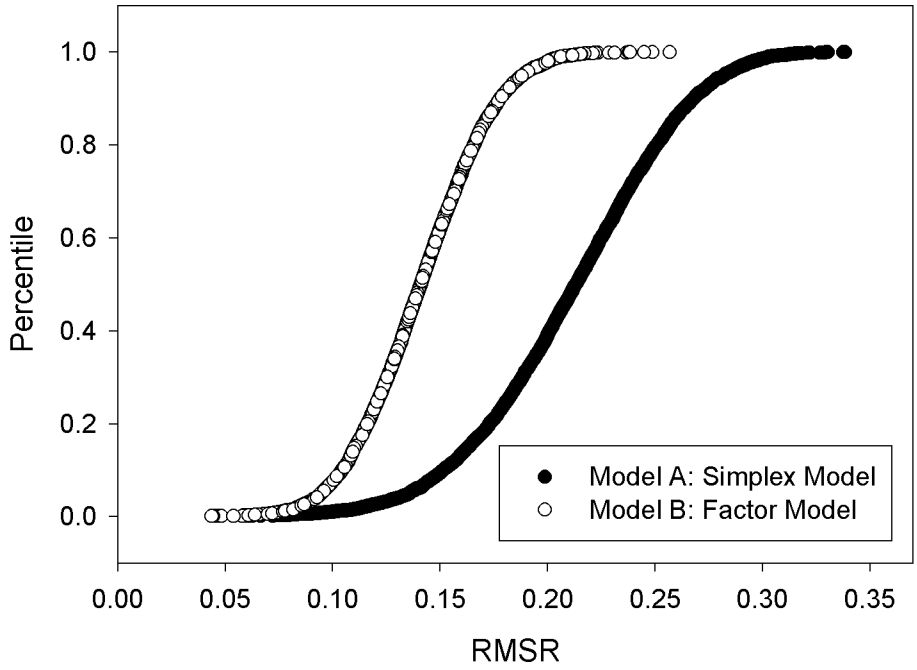


Figure 3.23: Cumulative frequency distributions of RMSR for simplex and 1-factor models

In many of the preceding sections, competing models were shown to fit certain regions of the data space differentially well, illustrating the effects of model complexity on the practice of model comparison. However, this example shows that the regions of the data space fit well by competing models do not necessarily need to overlap. It could easily happen that one model fits a certain region of the data space well while a competing model fits a completely nonoverlapping region of the data space well. A given

observed data pattern is more likely, all things considered, to fall within the region described well by the more complex of two competing models. It is this fact which makes it necessary to account for model complexity in the process of model comparison.

Regarding the simplex and factor models compared in this section, it would be of interest to explore the regions of the data space fit well by each model. Table 3.5 shows the number of random data sets, out of 5000, fit well and poorly by the two models jointly and in isolation.

	Model B	
Model A	RMSR < .08	RMSR ≥ .08
RMSR < .08	4	11
RMSR ≥ .08	63	4922

Table 3.5: Frequencies of data sets fit well and poorly by Models A (simplex) and B (factor)

According to Table 3.5, even though Model B fit 67 data sets well by the .08 criterion, Model A fit 11 data sets well that Model B fit poorly. Similarly, even though Model A fit 15 data sets well, Model B fit 63 data sets well that Model A fit poorly. This point will be raised later in §3.9, and again in Chapter 4 in the context of models drawn from published research.

3.8 The Relative Importance of Free Parameters and Functional Form

It could be claimed that the impact of functional form on model complexity is negligible relative to the influence of the number of free parameters. If there is any truth to that assertion, then correcting for the number of free parameters may be a sufficient penalty for model complexity, and further consideration of functional form complexity would supply little additional information. In reality, functional form potentially can determine a model's complexity to a far greater extent than can the number of free parameters alone.

To illustrate this point, two models were specified. Model A is a simple structural equation model in which the unique variances of indicators for particular LVs have been constrained to equality (see Figure 3.24). Model B is a simple orthogonal CFA model (see Figure 3.25). Note that Model B has more free parameters (q = 12) than Model A (q = 9), and thus would ordinarily be expected to be more complex. However, the functional form of Model B is such that there will be nine implied zero correlations, which are predicted to seriously curtail Model B's ability to fit data.

In fact, as can be seen from the RMSR CDFs in Figure 3.26, the model with more free parameters (Model B) fit substantially worse than the model with fewer free parameters (Model A). Neither model fit particularly well overall (only 20 data sets fit by Model A and none fit by Model B yielded RMSRs lower than .08), but the large difference in the proportion of data patterns fit by the two models at any criterion value for RMSR is telling. It appears that the number of free parameters is not always the most important determinant of model complexity.

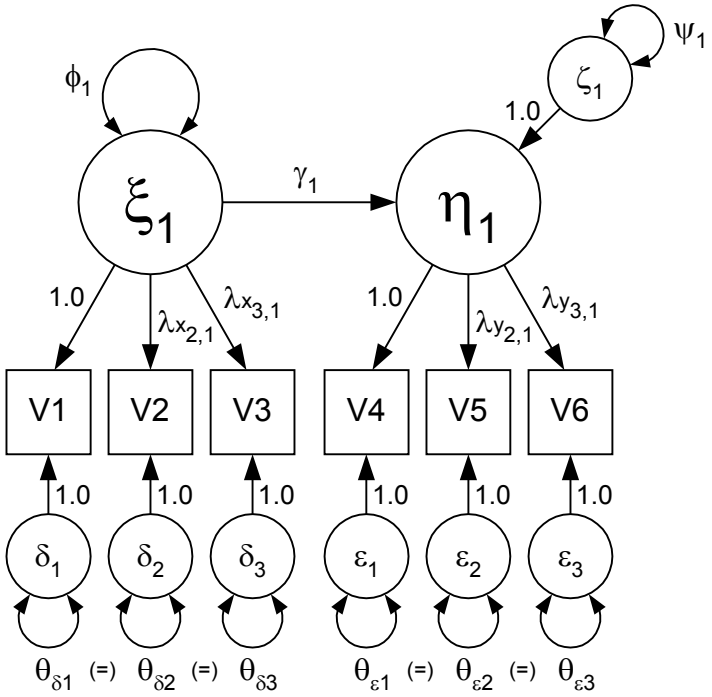


Figure 3.24: Model A – Simple structural equation model with within-factor unique variances constrained to equality

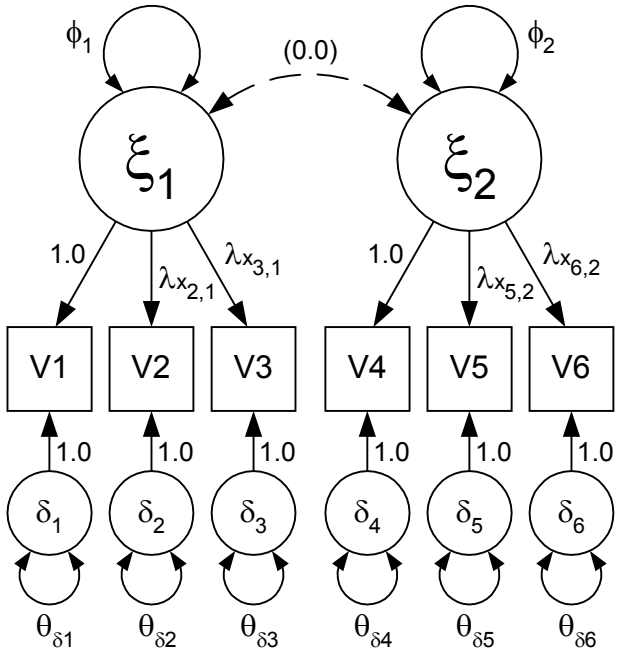


Figure 3.25: Model B – CFA model with the factor correlation constrained to equal zero

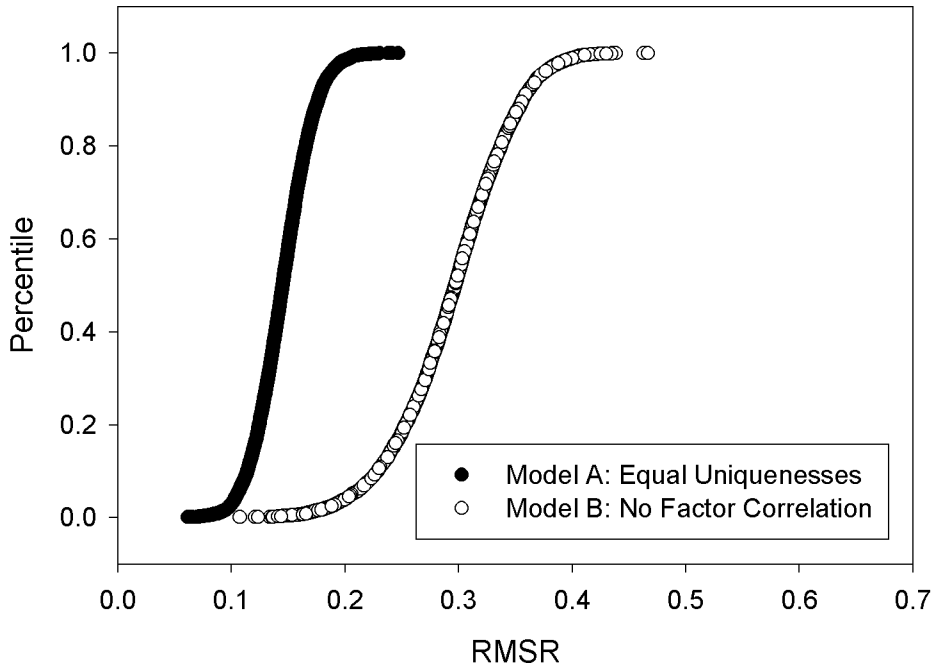


Figure 3.26: Cumulative frequency distributions of RMSR for an equal unique variance model (Model A) and an orthogonal factor CFA model (Model B)

3.9 Complexity Due to Parameter Range: Inequality Constraints

In §3.3 to §3.5 it was demonstrated that the number of parameters can influence model fit, and that these differences in fit influence model complexity. In §3.6 and §3.7 it was shown how functional form can also influence model complexity. In §3.8 it was shown that functional form may actually override the number of free parameters to become a more important determinant of model complexity. This section will address an additional factor involved in determining model complexity: parameter range.

Pitt et al. (2002) present *extension of the parameter space* as a factor contributing to model complexity, separate from functional form. In the context of SEM, the

parameter space consists of as many orthogonal dimensions as there are free parameters. Every degree of freedom, then, represents a dimension in which the model could be falsified. However, if the axes representing the ranges of one or more model parameters are restricted in some way, the result is a less complex model – the model is more falsifiable even though the dimensionality of the parameter space has not been increased and functional form has not been altered.[14] Whereas the dimensionality of the parameter space represents the contribution of the number of free parameters to model complexity, the relative *sizes* of these dimensions should also be considered.

The third step in the model fitting strategy, then, was to demonstrate that changing the range over which some parameters may vary, without altering the number or identity of those parameters, can have an effect on model complexity. In the SEM context, altering parameter range is conceptually distinct from altering parameter identity because it entails no change in the structural hypothesis. Rather, placing constraints on parameter range represents theory-driven hypotheses about the roles of individual parameters. For example, it might be theoretically important to restrict a certain parameter to be positive, or to be smaller than .20. Restricting an estimated covariance parameter (ϕ) to lie below .7, for example, should have an impact on model complexity because data sets for which the optimal value of ϕ lies above .7 will not be fit as well as

[14] Rindskopf (1984) remarks upon the dilemma faced by researchers who use range restrictions in their models. A range-restricted model can be thought of as nested within a comparable unrestricted model, but because degrees of freedom do not change with the imposition of constraints on parameter range, such models are not formally comparable via the standard χ^2 difference test, and thus the legitimacy of imposing a range restriction is difficult to evaluate.

they could be without such a restriction. Such restrictions are termed *inequality constraints* in the SEM context.

This goal of this section was accomplished by fitting models differing only in the allowable range for parameter estimates to the same random data. In other words, inequality constraints were placed on otherwise free parameters. In such models, if the estimate for a range-restricted parameter is the same regardless of whether a range restriction is specified, then the fit of the corresponding unrestricted and restricted models will be identical. However, if the fit-maximizing value of a parameter does not fall within the bounds set for that parameter, fit will suffer as a consequence. Thus, models with relatively fewer or no such restrictions should be at least as complex as those with restrictions. Models with range restrictions are especially interesting in light of the complexity issue because (1) corresponding restricted and unrestricted models share the same *df* and number of free parameters and (2) parameter range is a component of model complexity that is different from, yet related to, functional form and the number of free parameters (Pitt et al., 2002).

Specifically, two versions of a 2-factor CFA model (see Figure 3.27) were fit to 5000 random 6×6 correlation matrices. In one version (Model A), unique variances remained unconstrained. In another (Model B), one unique variance $(\theta_{\delta 1})$ was restricted to lie below 0.1. Due to the inevitable misspecification problems associated with fitting factor models to random data, 2918 of the 5000 constrained models (Model B) did not converge to a final solution within 1000 iterations. On the other hand, only two of the

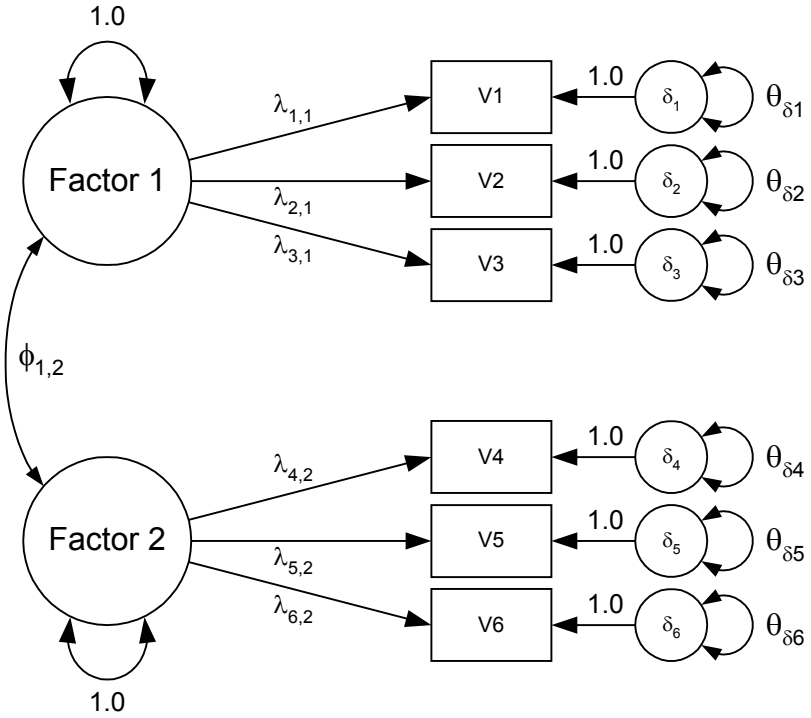


Figure 3.27: CFA model

unconstrained models failed to converge. The occurrence of such convergence problems made little difference, however. Figure 3.28 shows cumulative frequency distributions of RMSR for the models with and without the range restriction for all 5000 solutions. Figure 3.29 also shows distributions of RMSR for both models, but using only those solutions which converged. Both sets of CDFs demonstrate the point that models with a range restriction imposed on one parameter are less complex than an identical model without such a restriction, in the sense that they fit a smaller proportion of the data space by some arbitrary criterion. Of the models with unrestricted parameter range, 7.38% fit random data sets with RMSR values under .08. However, only 2.26% of models with the range restriction (and only 1.06% of those which converged to a solution) met the .08 criterion.

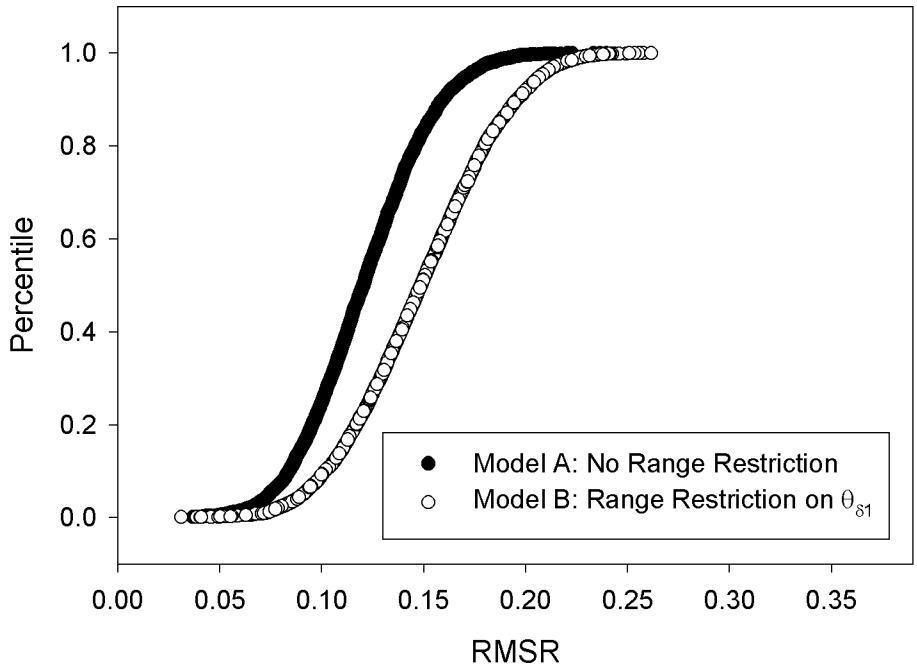


Figure 3.28: Cumulative frequency distributions of RMSR for two 2-factor models. All solutions were used, regardless of convergence

Models with and without range restrictions can be thought of as nested in the sense that one model is a more constrained version of the other, even though they share the same *df*. Therefore, it would be reasonable to expect that the region of the data space fit well by the restricted model should be nested within the region of the data space fit well by the less restricted model (allowing for violations due to estimation errors). Table 3.6 contains frequencies of data sets fit well by both Models A and B, either Model A or Model B, and neither Model A nor Model B. Table 3.6 implies that the region of the data space fit well by Model B (with the range restriction) is indeed a subset of the region fit well by Model A, as illustrated in Venn diagram form in Figure 3.30.

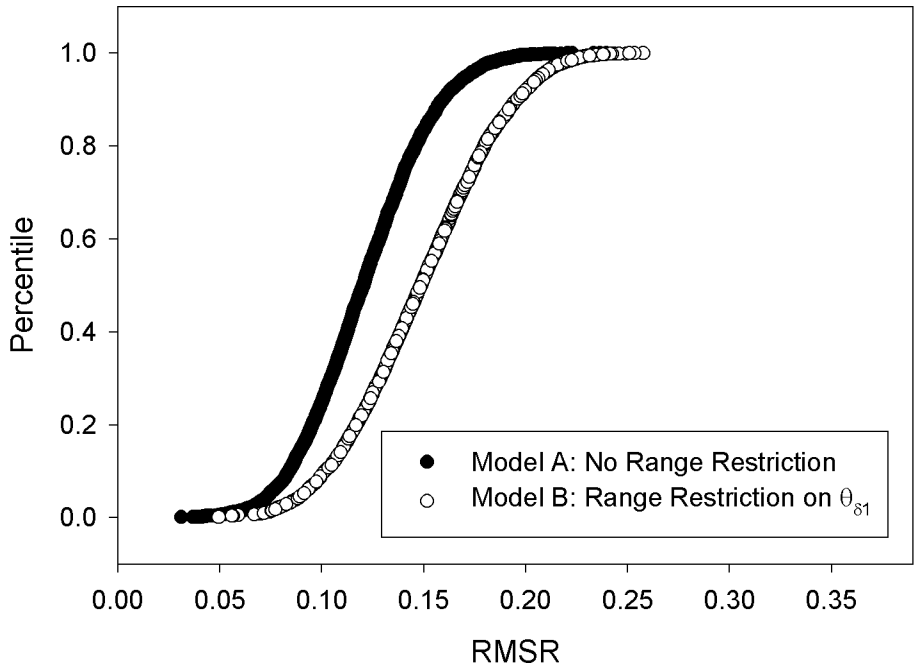


Figure 3.29: Cumulative frequency distributions of RMSR for two 2-factor models. Only those solutions which converged are included

	Model B	
Model A	RMSR < .08	RMSR ≥ .08
RMSR < .08	113	256
RMSR ≥ .08	0	4631

Table 3.6: Frequencies of data sets fit well and poorly by Models A (no restriction) and B (range restriction)

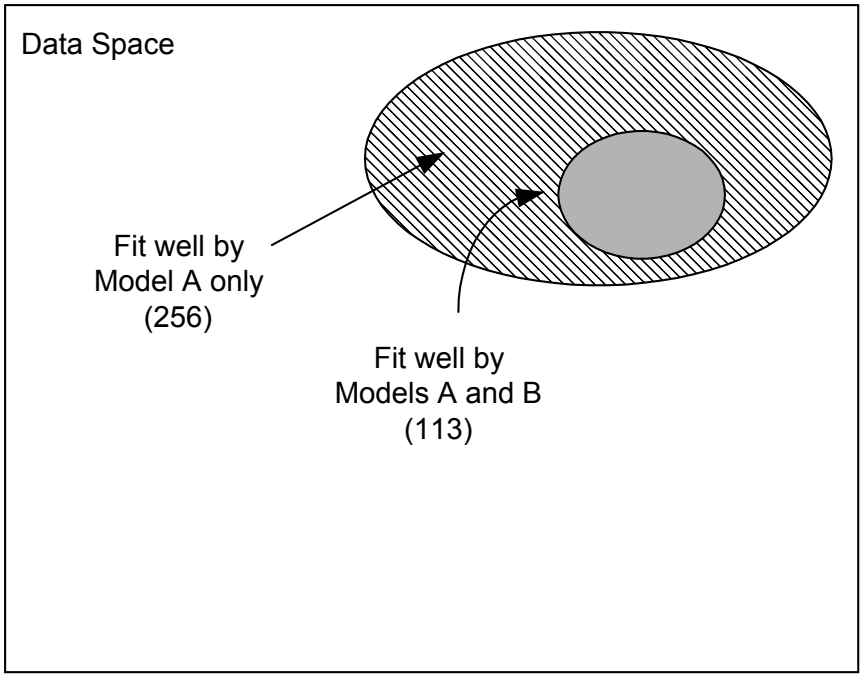


Figure 3.30: Venn diagram representing regions of the data space fit well by models with and without a range restriction (not drawn to scale)

The influence of parameter range on model complexity can best be understood by analogy. Consider a three-parameter model, and the parameter space defined by the allowable ranges of those three parameters as a three-dimensional box. Thus, the parameter vector $\hat{\theta}_x$ represents the coordinates $\left\{\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3\right\}$ of one instantiation of the model in the parameter space, given data set *x*. Restricting the allowable range of $\hat{\theta}_3$ to half its original size can be understood as chopping the box in half. If the box is thus chopped, fewer potential data patterns can fit into it. To take the analogy to an extreme, we can continue chopping the box in half until the allowable range for $\hat{\theta}_3$ is infinitesimal. The astute modeler would at some point be obliged to question the utility of continuing to think of the box as three-dimensional. Correcting for model complexity solely in terms of

the number of free parameters can be thought of as considering every dimension of the box as equally important, even if one of them is paper-thin. Noting that restricting parameter range can affect complexity would be comparable to considering the box two-dimensional for all practical purposes.

3.10 A Note on Comparing the Complexity of Alternative Models

Using the graphical method introduced in this chapter, it was repeatedly suggested that comparison of the complexity of one model relative to another should be conducted with respect to a particular criterion of good fit. Here, the chosen criterion was RMSR = .08. However, there are (probably rare) cases in which one model may appear to be more complex than another at the chosen criterion for good fit, yet the order may reverse when some other criterion is chosen. For example, consider the path model depicted in Figure 3.31, which represents a theory presented by Coffman, Levitt, Deets, and Quigley (1991) linking social support, relationship quality, and mothers' attitudes toward newborns. This study will be discussed in more detail in Chapter 4, but is used here for illustrative purposes. Consider also the simple two-factor model depicted in Figure 3.32, in which the unique variances associated with indicators of Factor 1 have been constrained to equality. Note that this model comparison example is highly contrived – it is very doubtful that models such as these would ever be compared in practice, but they can be used to form an interesting example.

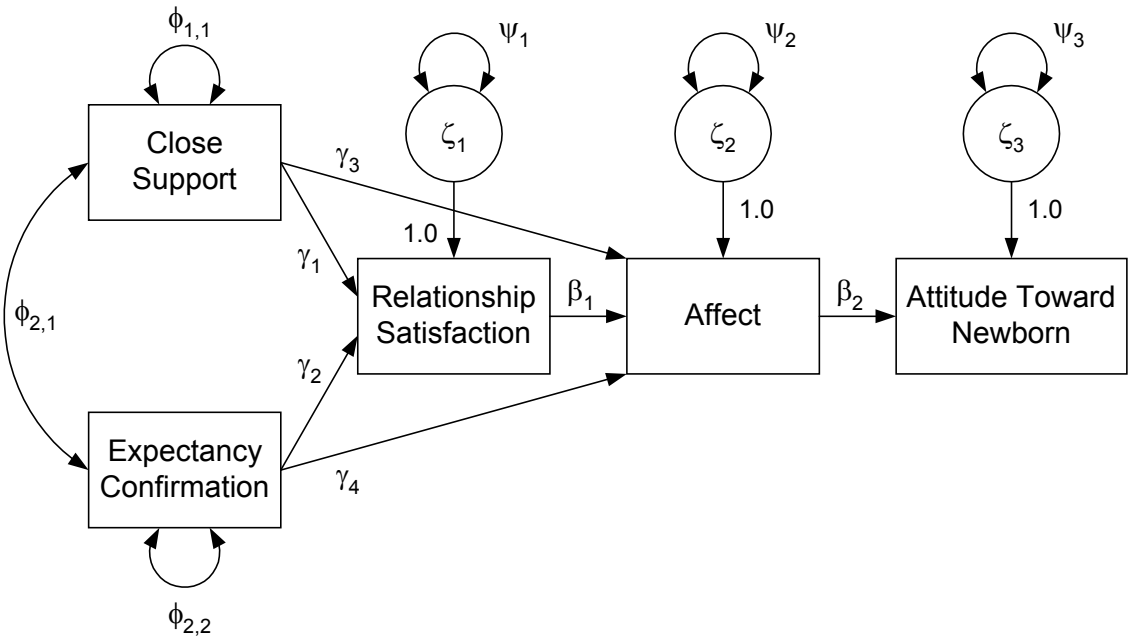


Figure 3.31: Path diagram representing Coffman et al.'s (1991) hypothesized model

When applied to the same 5000 random data patterns, these two models produce the RMSR CDFs presented in Figure 3.33. The key difference between the CDFs in Figure 3.33 and those presented earlier in this chapter is that they overlap. If a researcher were to choose RMSR = .08 as the criterion for good fit, then the Coffman et al. (1991) path model clearly emerges as more complex. However, if a researcher were to choose .14 as the criterion, the two models would appear to be equally complex. Finally, if .17 were chosen as the criterion, then the two-factor model appears to be more complex. The fact that the CDFs overlap is a clue that the regions of the data space fit well by each model, although of approximately equal area, probably have different shapes.

In situations exhibiting a pattern of fit to random data such as that in Figure 3.33, it is probably more sensible to compare the complexity of alternative models not by contrasting their RMSR CDFs at particular values of RMSR, but rather by comparing the

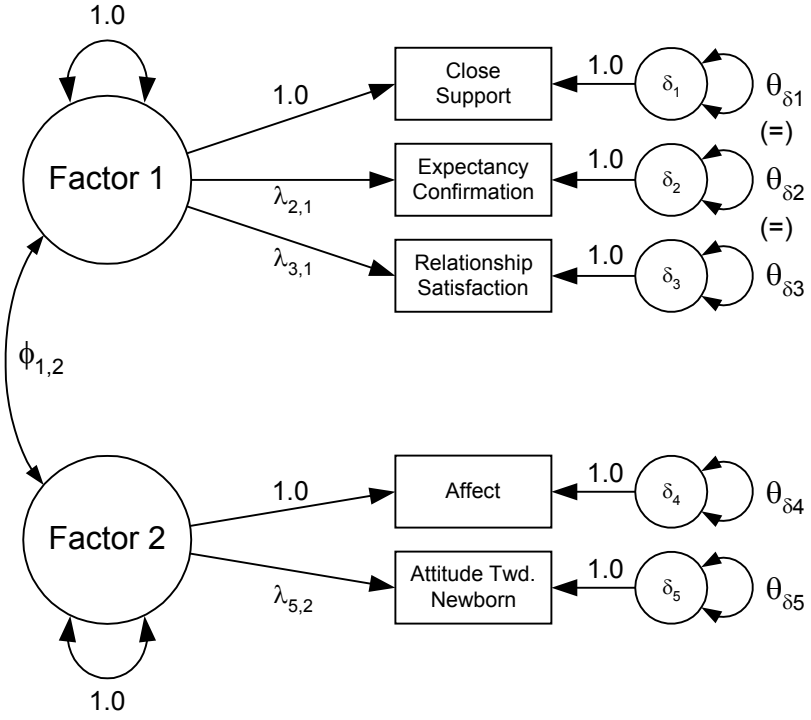


Figure 3.32: Path diagram representing a contrived two-factor model

two models in terms of *mean* RMSR, without regard to any particular criterion value. The mean value of RMSR for the Coffman et al. (1991) model was RMSR = .117, whereas the mean value of RMSR for the two-factor model was .125, indicating that the Coffman et al. model fit random data about as well as the two-factor model, but a little better; there was simply less variability in RMSR values for the two-factor model. This example might suggest that it would always be more appropriate to compare the complexity of alternative models by examining differences in mean fit indices when fit to random data.

Using this approach, one gains an appreciation for the appropriate ranking of several competing models in terms of overall complexity, but loses information about how well the model fits random data in general.

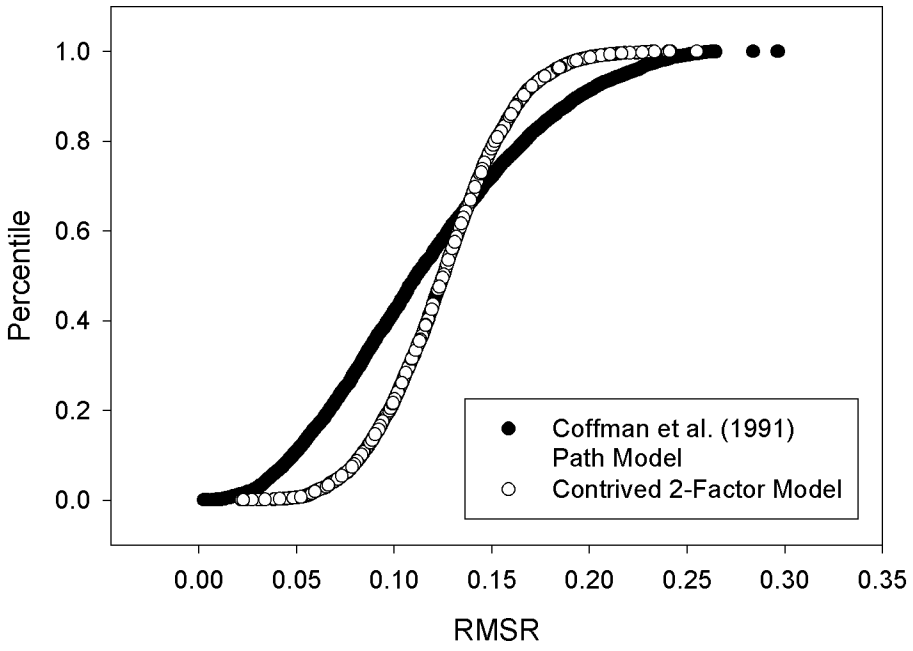


Figure 3.33: Path diagram representing a contrived two-factor model

3.11 Summary

In Chapter 3 it has been illustrated, by means of fitting several example structural equation models, that (1) model complexity should not be ignored in the process of evaluating the quality or generalizability of models and (2) complexity should not be operationalized as merely the number of free model parameters. This chapter introduced a novel method of examining model complexity in the form of cumulative distribution plots of RMSR, a simple residual-based fit index which includes no correction for model

complexity. Showing that two models differ in terms of their RMSR CDFs is sufficient to demonstrate that they are differentially complex, where complexity is defined as the proportion of the data space a model fits well by some criterion. One recommended criterion for good fit is that RMSR should be less than .08, but other criteria can be chosen. Indeed, discrepancy functions other than OLS and fit indices other than RMSR could have been chosen for purposes of demonstration.

Using this CDF approach, in §3.2 it was demonstrated that equivalent models fit the same data equally well, and therefore have equivalent complexities. In §3.3, §3.4, and §3.5 it was demonstrated that the number of free model parameters affects model complexity by restricting the proportions of the data space a model fits well according to some arbitrary criterion. In §3.6 and §3.7 it was shown that functional form – how model parameters combine to fit observed data – can affect model complexity. Section 3.8 illustrated how the effect of functional form complexity can outweigh the contribution of the number of free parameters in some instances. Section 3.9 illustrated the effect on complexity of placing restrictions on permissible parameter range. An analogy was offered to clarify the role of parameter range in determining model complexity. Finally, in §3.10 it was demonstrated that the graphical method may not always yield a clear decision about which of several competing models is more complex. In such cases, examination of mean values of RMSR (or some other fit index which similarly does not correct for complexity) can help.

Because popular SEM fit indices either do not correct for model complexity or do so crudely by considering only the number of free parameters, it is suggested that these

indices are inadequate for the purposes of model evaluation and model selection. In Chapter 4 it will be demonstrated that the phenomena illustrated in Chapter 3 are not limited to contrived examples, but rather have implications for models presented in published research.

CHAPTER 4

EXAMPLES FROM SUBSTANTIVE RESEARCH

4.1 Introduction

The results presented in Chapter 3 lead to the conclusion that failing to properly consider model complexity – whether due to the number of free parameters, functional form, or parameter range – can have negative consequences on model assessment. These consequences can include drawing incorrect conclusions about model performance and generalizability and drawing invalid comparisons between competing models. It was not unduly difficult to devise examples for use in illustrating the importance of complexity in Chapter 3. However, it remains to be demonstrated that improperly accounting for complexity can lead to problems in actual practice. Theoretically, failing to properly account for model complexity can lead to the unwarranted retention of overly complex models, the unwarranted rejection of models with relatively low complexity, and invalid comparisons among competing models, quite apart from considerations involving model plausibility or observed data.

Therefore, examples were sought in published psychological research and methodological literature which would illustrate some of the points set forth in Chapter 3. Specifically, examples were sought to illustrate that there are real consequences

associated with ignoring complexity, including the possibility of improper model retention, rejection, and comparison.

4.2 Model Evaluation: Determinants of Current-Events Knowledge in Adults

The first example is drawn from a recent study examining mediators of the relationship between Age and Current-Events Knowledge (CEK; Beier & Ackerman, 2001). Specifically, crystallized intelligence (Gc) was hypothesized to fully mediate the relationship between Age and CEK. The effect of Age on Gc, in turn, was specified to be partially mediated by fluid intelligence (Gf). Several personality characteristics were specified to partially mediate the effect of Gf on Gc, including Science/Math Self-Concept, Femininity as assessed with the Bem Sex Role Inventory (Bem, 1974), Typical Intellectual Engagement (TIE; Goff & Ackerman, 1992), Openness to Experience as assessed with the NEO-FFI (Costa & McCrae, 1992), and Traditionalism as assessed with the MPQ (Tellegen, 1982). In addition, the effect of Gc on CEK was thought to be partially mediated by self-estimates of knowledge and verbal ability. Such mediation models are usually regarded as particularly illuminating, as they are intended to model *how* or *by what means* changes in an independent variable lead to changes in some outcome variable. The full hypothesized model is depicted in Figure 4.1, patterned after a similar figure in Beier and Ackerman (2001).

In the model represented in Figure 4.1, all variables are measured rather than latent. Residual terms are omitted from the diagram to reduce clutter, but were included in the model. Several residual covariances were estimated on the presumption that some

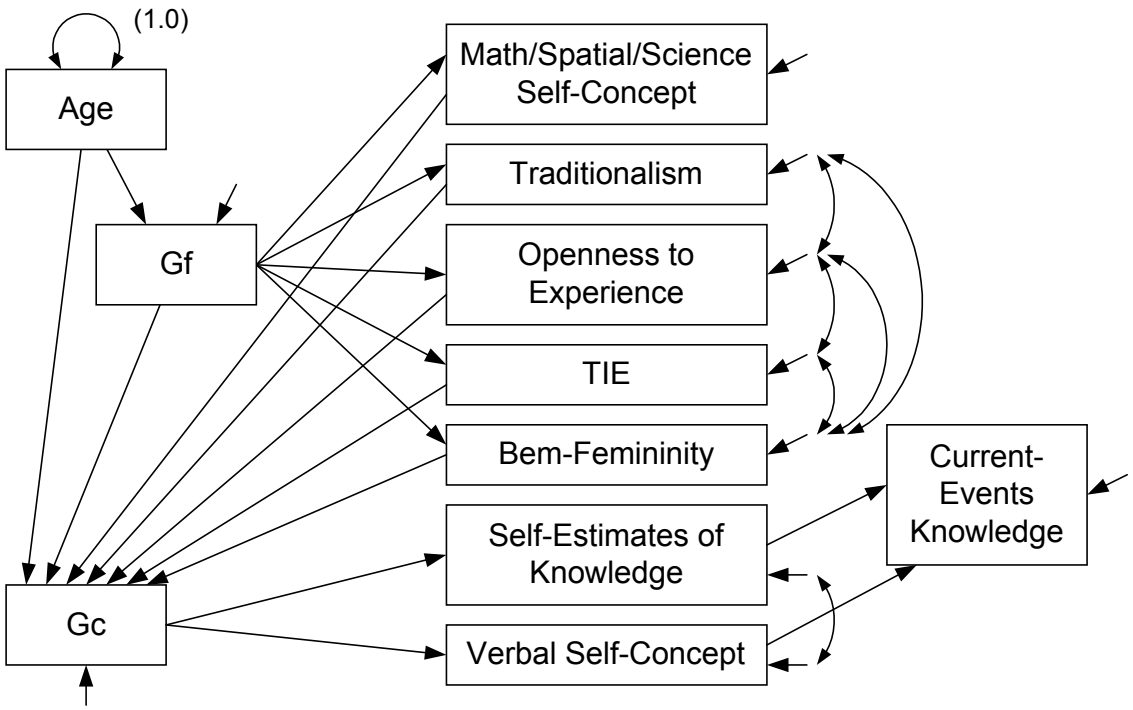


Figure 4.1: Path diagram representing Beier and Ackerman's (2001) hypothesized mediation model

personality characteristics are related for reasons not modeled. Beier and Ackerman (2001) used LISREL to fit the model to observed data collected from 153 individuals. Although not stated in their article, the variance of Age was constrained equal to 1.0 (M. Beier, personal communication, March 26, 2003), yielding a model with 31 degrees of freedom. Model fit was poor by all examined fit criteria ($\chi^2(32) = 140.73$, $p < .001$; RMSEA = .15, CFI = .84). On the basis of poor initial fit, the model was subsequently modified to include a direct path connecting Age to CEK, with some improvement in fit ($\chi^2(31) = 93.64$, $p < .001$; RMSEA = .12, CFI = .91), although the point estimate of RMSEA would be considered unacceptably high by most researchers (Browne & Cudeck, 1993; Hu & Bentler, 1998). The authors attributed this marked lack of fit to the

possibility that some of the items used to form the Gf composite likely also loaded on a Gc factor, and vice versa. However, this explanation is unlikely to be relevant, given that a path connecting Gc to Gf was included in the hypothesized model, and poor scale construction would more likely result in smaller parameter estimates than in lack of fit. The number of model parameters constrained to zero suggests an alternative explanation for the observed lack of fit, namely that the hypothesized model had an extremely small chance of fitting *any* data set well, even one conceptually in close agreement with the theory represented by the hypothesized model.

To address this possibility, the model in Figure 4.1 was fit to 5000 randomly generated 11×11 correlation matrices using LISREL. The resulting values of RMSR were extracted from the output and plotted in Figure 4.2. Examination of Figure 4.2 makes it clear that one of the reasons contributing to the observed poor fit was likely model complexity. Of the 5000 values of RMSR obtained by fitting the model to random data, none were lower than .08 (the lowest was .0866). In fact, a value of .08 would have been over 3.5 standard deviations below the mean RMSR, once outliers were removed.

This demonstration serves to illustrate that some theoretically derived models in published research can possess little baseline ability to fit any given data, and so are at a severe disadvantage even before data are collected. Such models allow for a narrow range of predictions, which is generally considered a desirable property for models to possess (Roberts & Pashler, 2000), but if the range of predictions is severely restricted, very few data patterns can be found that will agree closely with the model. In other words, theories like that represented by the model in Figure 4.1 may be *too* falsifiable. That is, even data

that are in close agreement with theory may falsify the model chosen to represent that theory because the model is inherently unable to fit data.

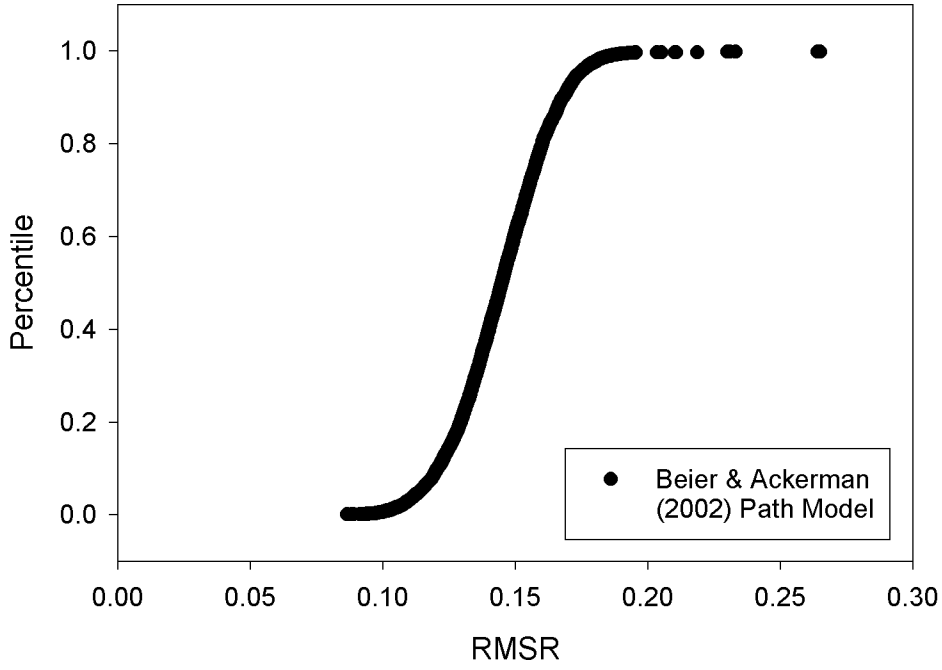


Figure 4.2: Cumulative frequency distribution of RMSR for the Beier and Ackerman (2002) path model. Eight extraordinarily high values were omitted from this plot (.81, 3.22, 3.81, 4.05, 4.54, 6.08, 6.34, and 8.68)

4.3 Model Evaluation: Close Relationships in Mothers of Distressed Newborns

At the other end of the complexity spectrum are models that are so complex that they are able to fit almost any given data pattern. Such models represent a form of capitalization on chance because they are able to fit a large proportion of the data space, and thus are unable to constrain possible outcomes to a reasonably small range. Under such circumstances, according to Roberts and Pashler (2000), we "cannot know how impressed to be when observation and theory are consistent" (p. 359).

An example of an overly flexible model was presented and tested by Coffman et al. (1991). The purpose of their study was to investigate the role of relationship quality in mediating the effects of social support and support expectancy on maternal affect, and similarly the role of affect in mediating the link between support and relationship quality on mothers' attitudes toward their newborns. A five-variable path analysis model, represented in Figure 4.3, was specified to examine these hypotheses. In a sample comprised of 36 mothers of healthy newborns and 47 mothers of distressed newborns, the model was found to fit well ($\chi^2(3) = 0.24$, $p = .97$; AGFI = .99). The same model was subsequently tested in distressed and healthy subsamples, with similar good fit (distressed: $\chi^2(3) = 3.02$, $p = .39$; AGFI = .88; healthy: $\chi^2(3) = 6.88$, $p = .08$; AGFI = .67).

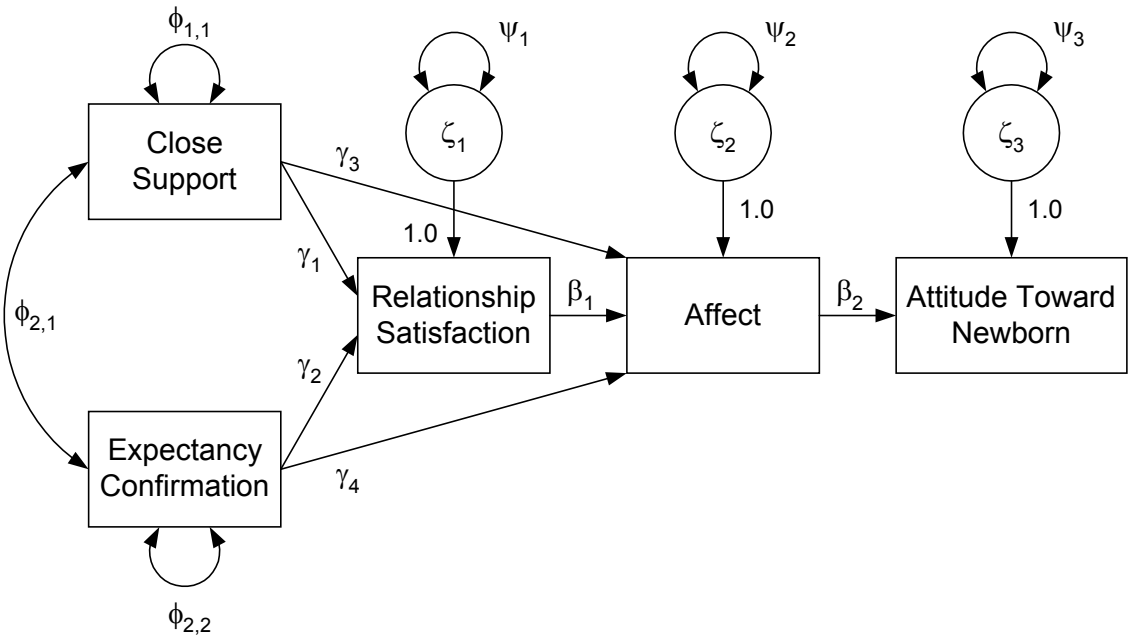


Figure 4.3: Path diagram representing Coffman et al.'s (1991) hypothesized mediation model

The observed good fit of the hypothesized path model was used in conjunction with interpretation of individual parameter estimates as a basis for drawing substantive conclusions about the relationships among observed variables. However, model complexity may represent a possible alternative explanation for the observed good fit. To investigate this possibility, an identical path model was fit to 5000 randomly generated 5×5 correlation matrices. The CDF of RMSR values is plotted in Figure 4.4. Fully 28.5% of random data sets were fit well according to the .08 criterion of good fit. In other words, the proposed path model could have fit over a quarter of all possible data sets as well or better by this criterion, so it should come as no surprise that the model displayed good fit when applied to observed data. Although this development does not disprove the theoretical ideas driving the model developed by Coffman et al. (1991), it may cast doubt on the model's usefulness. On the other hand, the uniform index-of-fit, i^*, for the Coffman et al. data is .998, an extraordinarily high value. Even though many random data patterns were fit well by the RMSR = .08 criterion, the particular data set obtained by Coffman et al. was fit particularly well by the hypothesized model.

In §4.2 and §4.3 it was shown how a model's complexity can (1) render it virtually unable to fit any data or (2) allow it to fit a large proportion of plausible data sets. However, it is difficult to separate the contributing effects of the number of free parameters and functional form when models are evaluated in isolation. This task is more easily accomplished in the context of model comparison. In §4.4 and §4.5, examples will be presented which show how giving inadequate attention to model complexity can lead

to problems in model comparison. Section 4.5 in particular will address the effect of functional form.

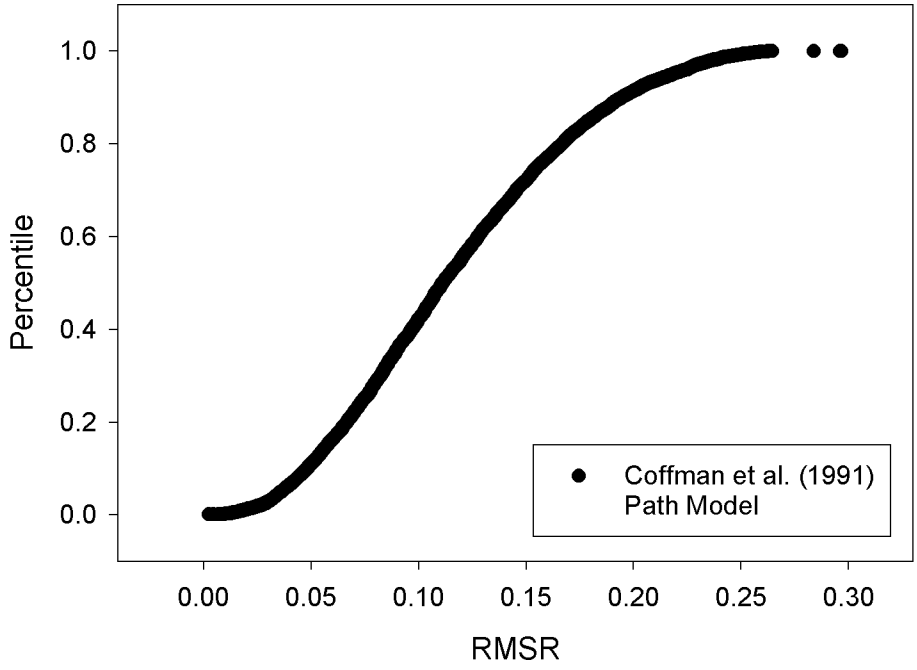


Figure 4.4: Cumulative frequency distribution of RMSR for the Coffman et al. (1991) path model

4.4 Model Comparison: Functional Status, Quality of Life, and Social Support

Newsom and Schulz (1996) develop and test a model in which three kinds of social support – belonging support (BS), appraisal support (AS), and tangible support (TS) – are hypothesized to mediate the relationship between physical impairment (PI) and quality of life (QOL), as assessed by life satisfaction (LS) and depression (DP). In other words, PI is expected to lead to changes in QOL indirectly by causing changes in social

support structures, which in turn have implications for QOL. Their hypothesized model (Model H) is represented schematically in Figure 4.5, omitting measured indicators.

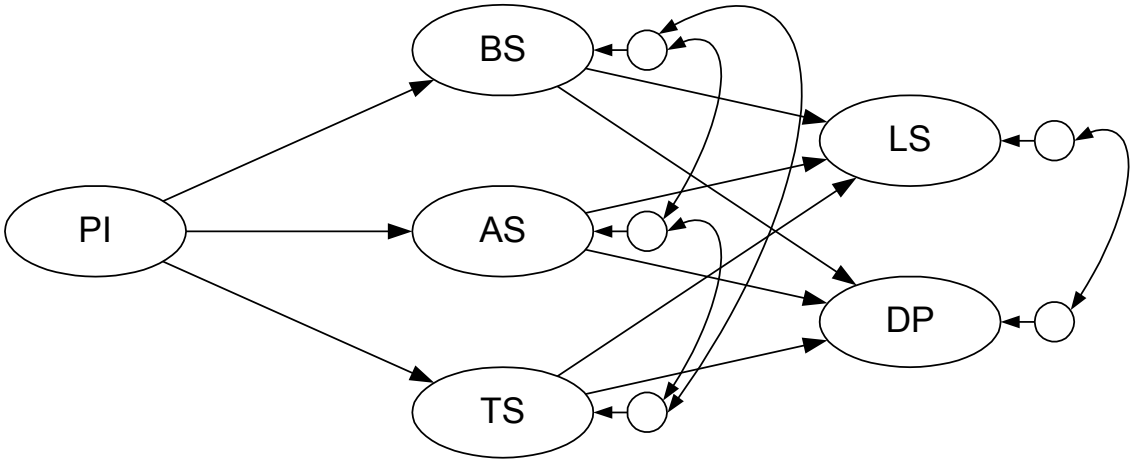


Figure 4.5: Model H – Schematic representing Newsom and Schulz's (1996) hypothesized structural equation model

After settling on a satisfactory measurement model (a confirmatory factor model linking MVs to their respective LVs), Newsom and Schulz (1996) fit their hypothesized structural model to a data set comprised of scores from 4263 individuals. Their model failed the exact fit test, $\chi^2(172) = 1135.62$, which is not surprising, considering the large sample size. However, Model H was judged to be a good match to the data using other fit indices (GFI = .975, TLI = .926, IFI = .939). The authors then fit a series of plausible alternative models – Models 1, 2, 3, and 4 – to the data in order to examine the relative plausibility of Model H. Models 1 and 4 are presented in Figures 4.6 and 4.7, respectively. Models 1, 2, and 3 were structurally equivalent and therefore yielded the same fit indices, so Models 2 and 3 are omitted from present consideration. Model fitting results are reported in Table 4.1. On the basis of these indices, Model H was judged to be

superior to Model 1, but indistinguishable (in practical terms) from Model 4. Models H and 4 were thus retained.

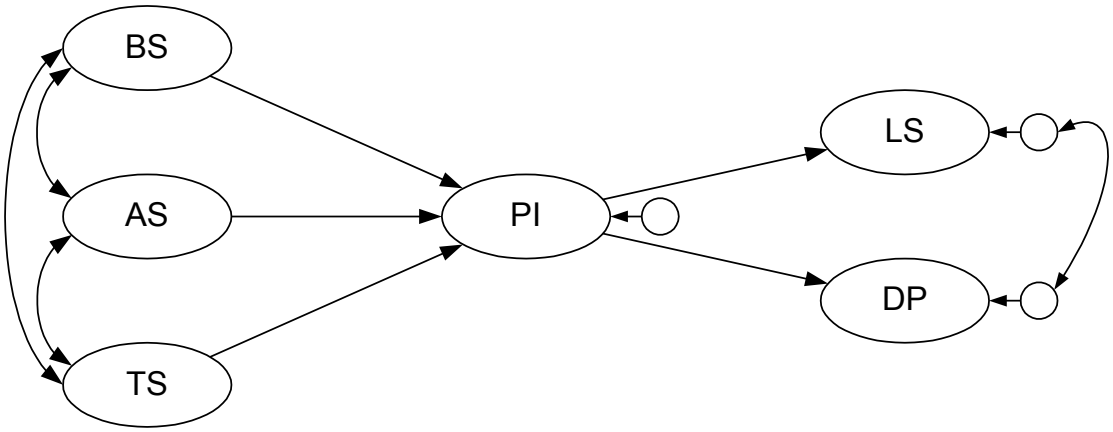


Figure 4.6: Model 1 – Schematic representing an alternative model with PI as mediator

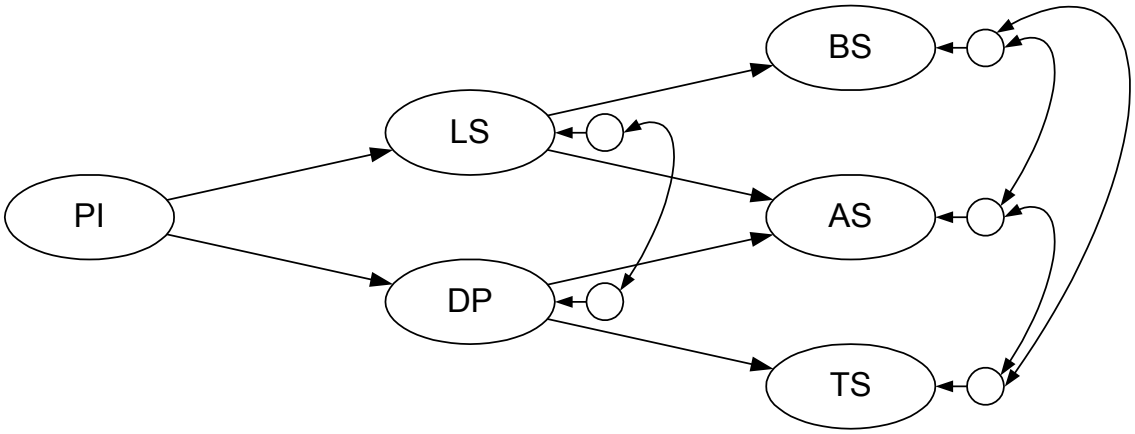


Figure 4.7: Model 4 – Schematic representing an alternative model with LS and DP as mediators

Because the three plausible alternative models proposed by Newsom and Schulz (1996) were not nested, but were compared on the basis of fit indices, the possibility exists that differences in model complexity gave some models an unfair advantage over

Model	*q*	*df*	χ^2	GFI	TLI	IFI
H	57	174	1135.62	.975	.926	.939
1	53	178	1527.14	.966	.899	.914
4	56	175	1041.74	.977	.934	.945

Table 4.1: Fit indices obtained by Newsom and Schulz (1996)

others. First, the three models differ in their number of free parameters. Model H had 57 free parameters, whereas Model 1 had only 53 and Model 4 had 56. Alone, this would suggest that Models H and 4 had a significant a priori advantage over Model 1 simply because Model 1 had fewer free parameters. However, the three models also differed in functional form. Therefore, differences in observed model fit could have been due not only to differences in theoretical plausibility, but also in part to differences in the number of free model parameters and functional form. In short, it may not have been entirely fair to compare Models H, 1, and 4 on the basis of model fit. In this section a more in-depth treatment of these models is undertaken by fitting simplified versions to random data.

In Newsom and Schulz (1996), each latent variable in Model H was assessed by at least two measured indicators, for a total of 21 MVs. This number of MVs is large even for the MCMC algorithm, so the models were reframed as path models with only six

MVs and no LVs. Because Models H, 1, and 4 all share the same measurement model, this simplification was judged to be acceptable. The revised Models H, 1, and 4 are presented in Figures 4.8, 4.9, and 4.10, respectively.

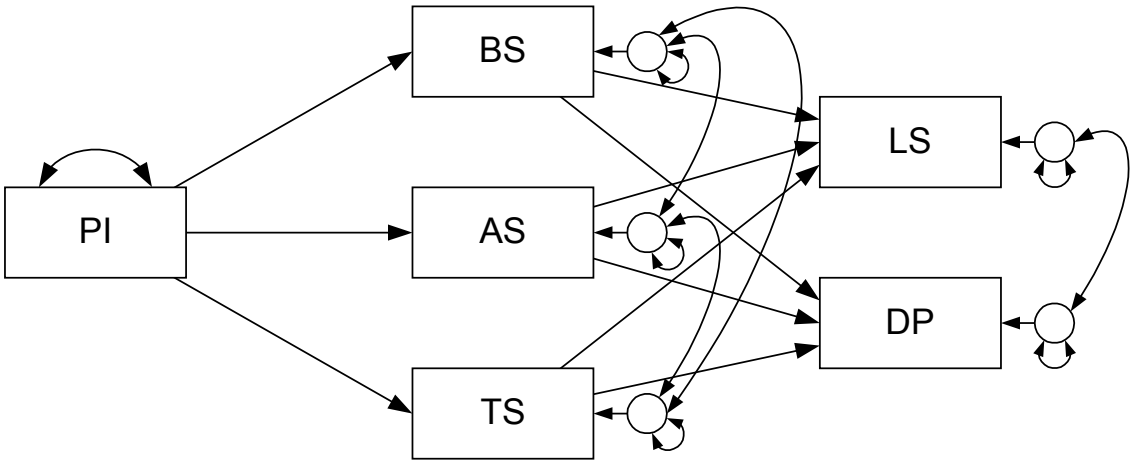


Figure 4.8: Revised Model H – Path model representing Newsom and Schulz's (1996) hypothesized mediation model

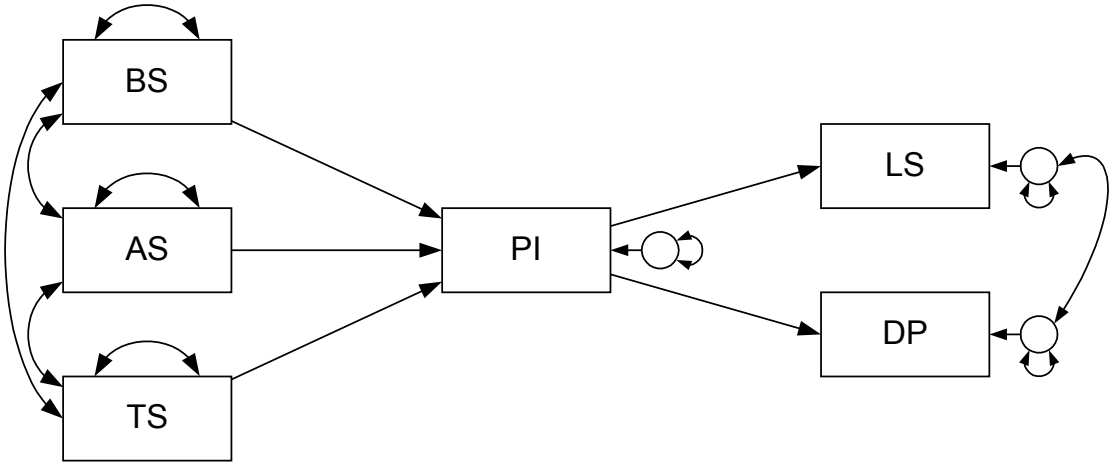


Figure 4.9: Revised Model 1 – Path model representing an alternative model with PI as a mediator

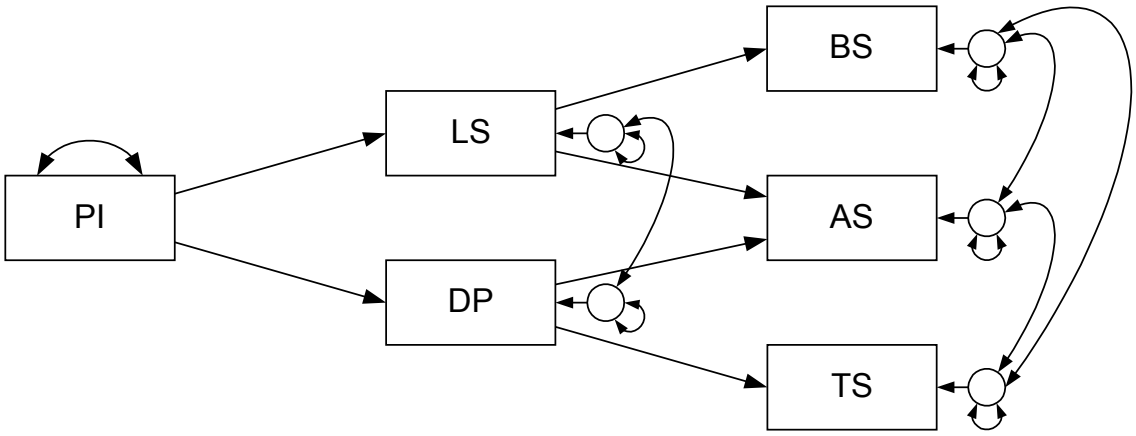


Figure 4.10: Revised Model 4 – Path model representing an alternative model with LS and DP as mediators

The revised Models H, 1, and 4 were fit to 5000 random 6×6 correlation matrices, with the resulting CDFs of RMSR plotted in Figure 4.11. Fully 95.12% of random data sets were fit well by Model H using the RMSR = .08 criterion, whereas only 10.38% of data sets were fit well by Model 1, indicating that the complexity of Model H far exceeds that of Model 1. Model 4 was also quite complex with respect to Model 1, fitting 69.30% of random correlation matrices well. Given these findings, and assuming that it is justified to generalize from these simplified six-variable models to their more complicated counterparts in Newsom and Schulz (1996), it is not at all surprising that the authors found the pattern of results they did. It is noteworthy that Model 4 performed slightly better than Model H in their study, yet when fit to random data, Model 4 fit noticeably (25.82%) less well than Model H. Had the authors known that Model 4 was at such an initial disadvantage with respect to Model H, perhaps they would have concluded that Model 4 had received significantly more support than Model H rather than

approximately equal support. To be fair, however, it is acknowledged that direct comparisons can be drawn between the present treatment and the analyses in Newsom and Schulz (1996) only to the extent that their LVs can be replaced with MVs.

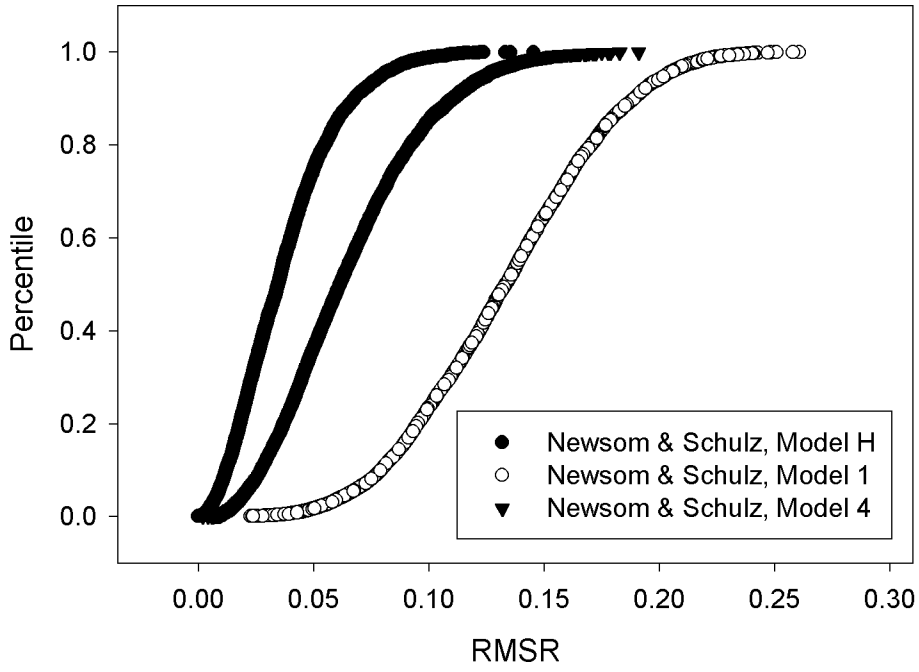


Figure 4.11: Cumulative frequency distributions of RMSR for three alternative mediation models

In §3.7 it was shown that the regions of the data space fit well by competing models need not be nested – they may overlap only partially or not at all. The data patterns fit well by one model need not all be fit well by the other models. Newsom and Schulz (1996) found that Models H and 4 fared better than Model 1, but it remains to be seen if the data patterns fit well by Model 1 are a subset of those fit well by Models H and 4. Table 4.2 presents results similar to those in Table 3.4 for the Newsom and Schulz

models. Figure 4.12 depicts a highly simplified representation of the situation confronted by the authors. Circles represent regions of the data space (not drawn to scale) fit well by each model using the RMSR = .08 criterion for good fit. Figure 4.12 helps explain why it was not entirely surprising to find that observed data were consistent with Models H and 4, as the region of the data space fit well by these two models (and not Model 1) covered 60.54% of the entire data space. Thus, there was a large a priori likelihood that any observed data pattern would lead to the obtained results.

Model	Frequency
No model	243
H only	1210
1 only	0
4 only	1
H and 1 only	82
H and 4 only	3027
1 and 4 only	0
H, 1, and 4	437

Table 4.2: Frequencies of data sets fit well by combinations of mediation models

Figure 4.12 also makes it clear that pure support for Model 4 would be almost impossible to obtain. Only one random matrix out of 5000 fell in the region fit well by Model 4 and not fit well by any other model under investigation. It can also be seen that there were no data patterns fit well by Model 1 that were not also fit well by Model H. On

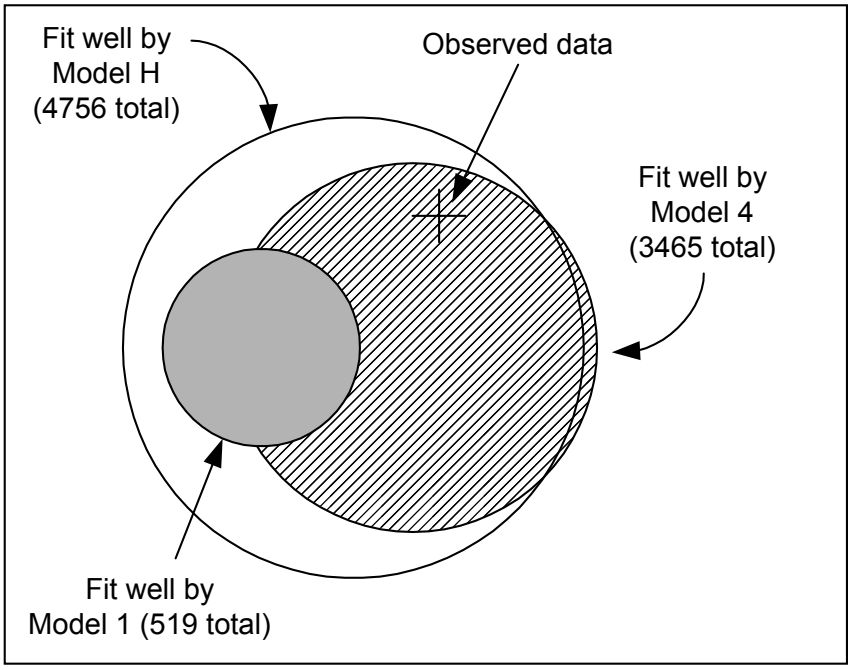


Figure 4.12: Venn diagram representing regions of the data space fit well by Newsom and Schulz's (1996) alternative Models H, 1, and 4 (not drawn to scale)

the other hand, 437 data patterns were fit well by all three models, so there was a non-negligible a priori probability that Model 1 would receive support in addition to Models H and 4. However, Model 1 did not receive support – it was falsified. In other words, the obtained results may be used more to argue against Model 1 than to argue in favor of Models H and 4.

4.5 Model Comparison: Two Models for Multitrait-Multimethod Data

Campbell and Fiske (1959) proposed construction of a *multitrait-multimethod* (MTMM) matrix for simultaneously assessing both convergent and discriminant validity. A MTMM matrix is a matrix of correlations among *t* traits $(t \geq 2)$ measured by means of

m methods $(m \geq 2)$. The resulting $mt \times mt$ matrix possesses several characteristics that can aid in determining the degree of convergent and discriminant validity. Over the years a number of models have been proposed to account for the pattern of correlations commonly observed in MTMM data. For example, several variations exist on a confirmatory factor analytic (CFA) approach. These models usually specify two or more factors to account for trait intercorrelations and two or more factors to account for method intercorrelations. While intuitively appealing, these CFA models are often characterized by improper solutions and uninterpretable parameter estimates.

The *components of covariance* model (or *covariance components analysis*; CCA) discussed by Wothke (1996) resembles the CFA approach to a large degree, but owes much to the analysis of variance (ANOVA) paradigm. CCA differs from the usual CFA approach in two principal respects: (1) a general factor is included to account for common variance that cannot be assigned uniquely to traits or methods and (2) the identification conditions necessary to avoid indeterminacy take the form of (t – 1) trait contrasts and (m – 1) method contrasts. Thus, the CCA model allows for the comparison of nested subsets of individual method or trait factors (contrasts specified a priori).

In the CCA model, the *mt* variables serve as indicators for (m – 1) method *contrast factors* and (t – 1) trait *contrast factors*, as well as a *general factor* uncorrelated with either method or trait contrast factors. The general factor is specified to account for the fact that common variance shared by all variables cannot be assigned reliably either to traits or to methods, something other CFA-based models fail to consider. The CCA covariance model (Wothke, 1996) is:

$$\Sigma_x = \mathrm{D}_x \mathrm{K} \Phi^* \mathrm{K} \mathrm{D}_x + \Theta \tag{4.1}$$

where K is a $tm \times (t + m - 1)$ columnwise (usually orthonormal) matrix of contrast coefficients, Φ^* is the covariance matrix of latent contrasts, Θ is the diagonal matrix of unique variances, and D_x is a diagonal matrix of scale parameters which permits the analysis of scale-free correlation matrices. The first column of K represents fixed loadings on a general factor. The next $(t - 1)$ columns represent orthogonal contrast coefficients for traits. The last $(m - 1)$ columns represent contrasts for methods. For example, the K matrix in Equation 4.2 represents contrasts of trait 1 vs. traits 2 and 3, trait 2 vs. trait 3, method 1 vs. methods 2 and 3, and method 2 vs. method 3 (Wothke, 1996, p. 20):

$$\mathrm{K} = \left[\begin{array}{c|cc|cc} 1/3 & \sqrt{2}/3 & 0 & \sqrt{2}/3 & 0 \\ 1/3 & -1/\sqrt{18} & 1/\sqrt{6} & \sqrt{2}/3 & 0 \\ 1/3 & -1/\sqrt{18} & -1/\sqrt{6} & \sqrt{2}/3 & 0 \\ \hline 1/3 & \sqrt{2}/3 & 0 & -1/\sqrt{18} & 1/\sqrt{6} \\ 1/3 & -1/\sqrt{18} & 1/\sqrt{6} & -1/\sqrt{18} & 1/\sqrt{6} \\ 1/3 & -1/\sqrt{18} & -1/\sqrt{6} & -1/\sqrt{18} & 1/\sqrt{6} \\ \hline 1/3 & \sqrt{2}/3 & 0 & -1/\sqrt{18} & -1/\sqrt{6} \\ 1/3 & -1/\sqrt{18} & 1/\sqrt{6} & -1/\sqrt{18} & -1/\sqrt{6} \\ 1/3 & -1/\sqrt{18} & -1/\sqrt{6} & -1/\sqrt{18} & -1/\sqrt{6} \end{array}\right] \tag{4.2}$$

The fitted covariance matrix of latent contrasts, $\hat{\Phi}^*$, contains the results of the specified contrasts. For example, if the variance of the second trait contrast is near zero,

the implication is that traits 2 and 3 are empirically indistinguishable. Wothke (1996) points out that if K is chosen to be orthonormal (i.e., so that K'K = I), then $\hat{\Phi}^*$ is a correlation matrix, and direct comparisons can be made between contrasts.

Wothke (1996) describes several types of CCA models, defined by restrictions placed on $\hat{\Phi}^*$: (1) the *fully correlated* $\hat{\Phi}^*$ *model*, in which all contrasts are allowed to correlate with each other and with the general factor; (2) the *independent-common-variation model*, in which all contrasts are allowed to correlate with each other, but not with the general factor; (3) the *block-diagonal* $\hat{\Phi}^*$ (BD) *model*, in which trait contrasts are allowed to intercorrelate and method contrasts are allowed to intercorrelate, but trait and method contrasts are not allowed to correlate with each other or with the general factor, and (4) the *diagonal* $\hat{\Phi}^*$ *model*, in which all contrast correlations are fixed to zero. A path diagram illustrating the (BD) CCA model is presented in Figure 4.13.

Building on Swain's (1975) multiplicative model, Browne (1984) proposed the *direct product model* (DPM) for MTMM data, in which the observed covariance matrix is regarded as the Kronecker (right direct) product of a *methods covariance component matrix* and a *traits covariance component matrix*:

$$\Sigma_x = \Sigma_M \otimes \Sigma_T \tag{4.3}$$

Σ_x may be regarded equivalently as a covariance or correlation matrix. Whereas observed covariances are regarded as the sum of trait and method components in additive CFA approaches, covariances are represented as products in the DPM.

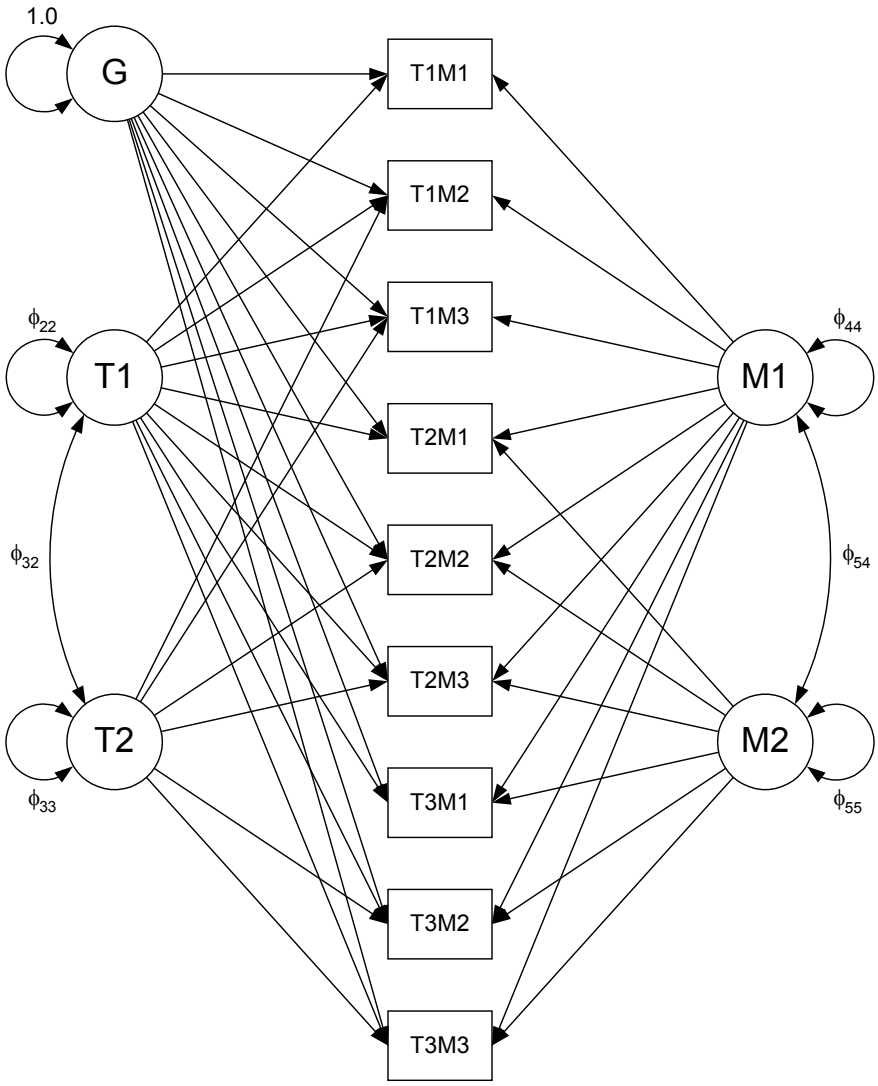


Figure 4.13: The block-diagonal components of covariation (CCA) model for MTMM data. Unlabeled loadings are constrained to equal values reported in matrix K in Equation 4.2

Browne also describes the *composite direct product model* (CDPM), in which correlations corrected for attenuation possess the direct product structure described above. Some deficiencies associated with the DPM are eliminated when the CDPM is employed (Browne, 1984, 1985; Cudeck, 1988). The CDPM is specified as:

$$\Sigma_x = \Delta\left(\mathrm{P}_M \otimes \mathrm{P}_T + D_v^2\right)\Delta \tag{4.4}$$

where Δ is a diagonal matrix consisting of common score standard deviations, P_M and P_T are common score correlation matrices of methods and traits, respectively, and D_v^2 is a diagonal matrix of unique variances. Under the CDPM, common scores (rather than observed scores) are assumed to possess a multiplicative structure.

Wothke and Browne (1990) provide a reparameterization of the CDPM:

$$\Sigma = \Delta\left(\left(C_M \otimes I_T\right)\left(I_M \otimes \Sigma_T\right)\left(C_M \otimes I_T\right)' + D_\eta^2\right)\Delta \tag{4.5}$$

where C_M is an $m \times m$ lower triangular matrix of method components, $\left(C_M\right)\left(C_M\right)' = \Sigma_M$, and I_T and I_M are identity matrices of order *t* and *m*, respectively (Wothke, 1996). Wothke provides a worked example and LISREL code for the reparameterized model. A path diagram illustrating the CDPM is presented in Figure 4.14.

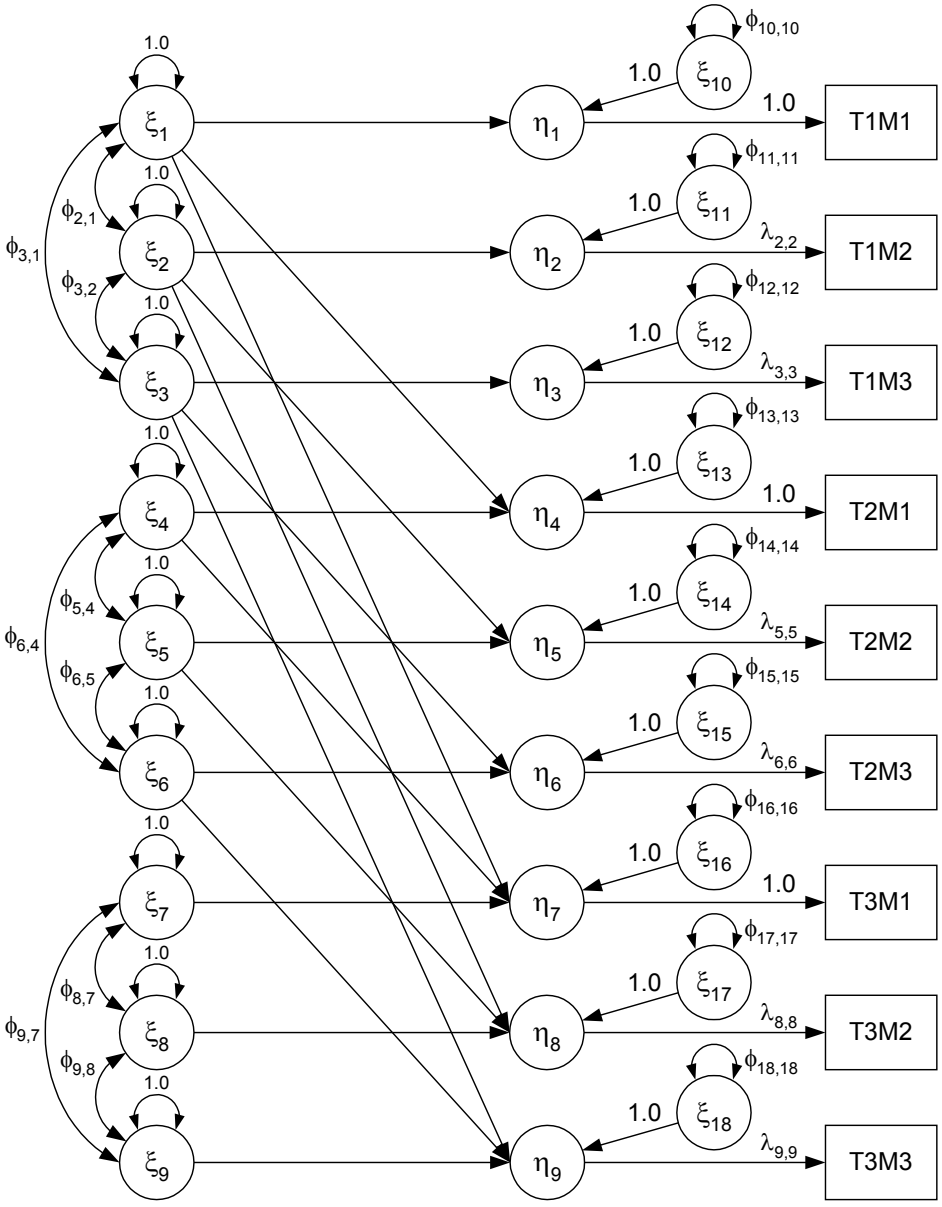


Figure 4.14: The composite direct product model (CDPM) for MTMM data. Unlabeled path coefficients are constrained to the equality constraints described in Equation 4.9

With suitable constraints, the CDPM can be reparameterized as a second-order factor model with heteroscedastic error terms (Wothke, 1996):

$$\Sigma = \Lambda\Gamma\Phi\Gamma'\Lambda' . \tag{4.6}$$

The matrix Γ is defined as:

$$\Gamma = \left(I_M \otimes C_T \mid I_M \otimes E_T^{1/2} \right), \tag{4.7}$$

where E_T is an error covariance matrix corresponding to traits. The matrix Φ is defined as:

$$\Phi = \left(\begin{array}{c|c} \Sigma_M \otimes I_T & 0 \\ \hline 0 & E_M \otimes I_T \end{array} \right). \tag{4.8}$$

The unlabeled paths in Figure 4.14 correspond to elements of the Γ matrix. For the case involving three traits and three methods, Γ is specified as:

$$\Gamma = \left[\begin{array}{ccccccccc|ccccccccc} a & & & & & & & & & 1 & & & & & & & & \\ & a & & & & & & & & & 1 & & & & & & & \\ & & a & & & & & & & & & 1 & & & & & & \\ b & & & d & & & & & & & & & 1 & & & & & \\ & b & & & d & & & & & & & & & 1 & & & & \\ & & b & & & d & & & & & & & & & 1 & & & \\ c & & & e & & & f & & & & & & & & & 1 & & \\ & c & & & e & & & f & & & & & & & & & 1 & \\ & & c & & & e & & & f & & & & & & & & & 1 \end{array} \right], \tag{4.9}$$

where elements sharing lower-case letters are constrained to equality and empty cells are assigned values of zero. Φ is specified as:

$$\Phi = \left[\begin{array}{ccc|ccccccccc} 1 & g & h & & & & & & & & \\ g & 1 & i & & & & & & & & \\ h & i & 1 & & & & & & & & \\ & & & 1 & g & h & & & & & \\ & & & g & 1 & i & & & & & \\ & & & h & i & 1 & & & & & \\ & & & & & & 1 & g & h & & \\ & & & & & & g & 1 & i & & \\ & & & & & & h & i & 1 & & \\ \hline & & & & & & & & & j & \\ & & & & & & & & & & k \\ & & & & & & & & & & & l \\ & & & & & & & & & & & & m \\ & & & & & & & & & & & & & n \\ & & & & & & & & & & & & & & o \\ & & & & & & & & & & & & & & & p \\ & & & & & & & & & & & & & & & & q \\ & & & & & & & & & & & & & & & & & r \end{array}\right], \quad (4.10)$$

where, again, elements sharing lower-case letters are constrained to equality and empty cells are constrained to zero.

Browne (1993), Chan (1992), and Wothke (1996) note that CCA (the BD version) and the CDPM have the same degrees of freedom, and thus their fit statistics may be directly comparable. Indeed, the two models have been found to closely correspond (Browne, 1993; Wothke, 1996). Both CCA and the CDPM can be viewed as

generalizations of a trait-only CFA model, but they are not directly comparable because (1) they are not nested models and (2) they reflect different underlying philosophies of how methods interact with traits. Thus, because these two theoretically-derived models were developed to account for the same general patterns of correlation commonly observed in MTMM data, it would be of interest to see if they differ in baseline ability to fit *any given* data. The fact that they have the same degrees of freedom is coincidental (Wothke, 1996), but convenient from the perspective of examining model complexity. In other words, because they share the same number of free model parameters, these two models may be compared purely in terms of functional form complexity using the graphical method described in Chapter 3. If the two models differ in complexity, then caution must be taken when comparing fit indices from the two models.

The baseline ability of the CCA and CDPM models to fit random data has been examined in the past. Chan (1992) compared the performance of four models designed to account for MTMM data, including CCA, CDPM, a CFA model, and an *equal correlation* model to serve as an example of poor model specification. Root mean square error (RMSE)[15] was used as a measure of badness of fit. The primary difference between Chan's procedure and the present one is that Chan's random correlation matrices were generated using Botha et al.'s (1988) UUS and U01 methods, which were found to have unrealistically high and uniform elements, and a random permutation method. In addition, Chan fit models to only 100 random matrices, whereas the present treatment uses 5000 random matrices.

---

[15] Chan defined RMSE as $\sqrt{d*/df} = \sqrt{\sqrt{2F_{OLS}}/df}$, which corrects for complexity by dividing by *df*.

The CCA model and CDPM (Models A and B, respectively) were fit to 5000 random correlation matrices. Despite the differences in procedure, the results using MCMC matrices are strikingly similar to Chan's (1992) results. CCA seems to have a slight edge in terms of model fit, fitting random data better than CDPM by a very small margin (see Figure 4.15). Out of 5000 solutions, 33 data sets fit by the CCA model had RMSR values less than .08, whereas only 23 data sets fit by the CDPM had RMSR values less than .08. Model fit is not used as the sole determinant of model quality with the CCA model and CDPM, but it is one factor. Researchers should consider this fact when deciding which of the two models is more appropriate in their own research.

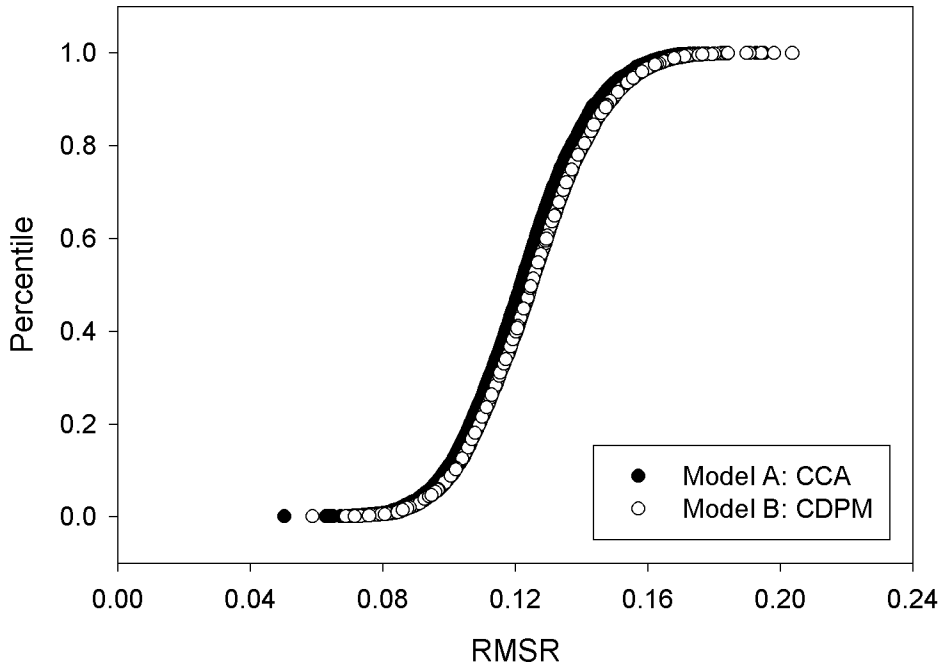


Figure 4.15: Cumulative frequency distributions of RMSR for two MTMM models

4.6 Summary

To summarize, in §4.2 an example was presented which illustrates that some models may not be complex enough to be useful. Because only an extremely restricted portion of the data space would be fit well by the Beier and Ackerman (2001) model, it would be very impressive indeed to show close fit between their model and an observed data set. In §4.3 an example was presented which demonstrates that some models used in applied research are so complex that they have high inherent ability to fit almost any given data. The Coffman et al. (1991) model was able to fit 28.5% of randomly generated data sets, so it is likely that the observed good fit can be attributed in part to its baseline ability to fit data rather than to its approximation to the truth. Section 4.4 contained an example of model comparison (Newsom & Schulz, 1996) in which two models were judged to be superior to an alternative model on the basis of model fit, despite large discrepancies in model complexity due to the joint effect of the number of free parameters and functional form. Section 4.5 contained an example of model comparison drawn from the methodological literature on the analysis of multitrait-multimethod correlation matrices.

The findings reported in this chapter are not intended to reflect poorly on the theories being tested, but rather on the methods used in testing the models constructed to represent those theories. In addition, nowhere is it claimed that the conclusions drawn by the authors cited in this chapter are incorrect. However, the possibility of drawing incorrect conclusions about their models was probably higher than the authors had assumed, given that the fit indices they used did not fully account for model complexity

in a way that permitted fair comparisons on the basis of fit. Furthermore, it should be emphasized that models are typically not evaluated solely on the basis of fit – other factors also determine a model's quality, such as the interpretability of observed parameter estimates and the theoretical plausibility of the entire model. The models in §4.2 to §4.4 were also evaluated on the basis of these factors by the authors, so no blanket statement can be made regarding the overall quality or generalizability of these models. However, it should be emphasized that complexity plays an important role in the process of model evaluation and comparison, and therefore can have a potentially major impact on substantive conclusions.

CHAPTER 5

CONCLUSIONS AND DISCUSSION

5.1 Summary

This dissertation has introduced an extended concept of model complexity to the field of structural equation modeling, and makes the claim that the continued use of traditional fit indices, which correct for model complexity either crudely or not at all, can only hinder the scientific enterprise. Chapter 1 introduced the concept of model complexity and explained why good fit is not always as good as it may seem. Chapter 2 introduced a new method of generating random correlation matrices for use in later demonstrations illustrating the ideas presented in Chapter 1. Chapter 3 used the random data generated by methods described in Chapter 2 to demonstrate that different models may possess different inherent abilities to fit any given data, and that this inherent difference in ability (complexity) does not rely solely upon differences in the number of free model parameters. Chapter 4 demonstrated that the ideas developed in Chapter 1 and illustrated in Chapter 3 also apply in practice, using real, theoretically derived models culled from published research.

5.2 Primary Findings

Several specific findings have important consequences for future approaches to assessing model fit in the SEM context. In Chapter 3, several illustrations were presented which demonstrate the importance of considering model complexity in the context of evaluating and comparing structural equation models. As expected, the number of free parameters was found to have a large impact on model complexity. In addition, functional form (the structural configuration of a model) can have a potentially major influence on a model's ability to fit data. Imposing an inequality constraint on a parameter restricts the range of possible estimated values without altering degrees of freedom or functional form. Imposing an equality constraint on two parameters also leaves functional form unchanged, yet alters degrees of freedom without necessarily restricting the allowable parameter range (in the sense that theoretically no value is ruled out a priori). Constraining a model parameter to a fixed value alters the degrees of freedom, functional form, and parameter range (in the sense that the allowable range for the parameter is reduced to a single value). All of these actions act to reduce a model's complexity, or a priori ability to fit data. However, in fit indices traditionally used in SEM, only the number of free parameters is considered; parameter range and functional form are ignored entirely. Some consequences associated with incompletely accounting for model complexity were illustrated using published examples in Chapter 4.

5.3 Implications and Recommendations for Practice

Researchers should keep in mind that joint determination of model complexity and goodness of fit is only one tool in the researcher's modeling tool-chest. Other requirements for quality models are that they make sound theoretical sense, yield interpretable parameter estimates, generalize to new samples, and lead to the formation of new ideas and lines of research. The assumptions necessary for modeling (e.g., linear relationships, multivariate normality, etc.) should be met to within a reasonable degree. In addition, the data must be of high quality, such that the chosen variables are reliable and valid. It should now be clear that model complexity is also an important factor in assessing model quality, and that it should not be ignored in the process of model evaluation and comparison.

Requirements for quality models aside, good modeling requires good modelers. The process of selecting a model from among competing models, or evaluating models in isolation, should never be accomplished in a fully mechanical manner – "...judgment cannot be avoided in model selection" (McDonald & Marsh, 1990, p. 252). The researcher should bring to bear his or her knowledge of the content domain producing the model in order to properly evaluate the model's appropriateness and the interpretability of parameter estimates (Browne, 2000). Without meeting these requirements, even models which meet the criteria of parsimony and good fit must be called into question.

Confirmation bias has been described as "the inappropriate bolstering of hypotheses or beliefs whose truth is in question" (Nickerson, 1998, p. 175). The continued tendency for scientists to specify and investigate relatively complex models

can be seen as a sort of confirmation bias, in which scientists propose theory-derived models with small potential for falsification. Complex models also represent a form of capitalization on chance. Perhaps what most casual practitioners of SEM do not realize is that every free parameter represents a part of the model about which the researcher is uncertain. In most cases, theory dictates the *constraints* placed on a model, not the model's free parameters. It is the fixed values, equality constraints, and range restrictions that are jointly tested and that are reflected in lack of fit – not the free parameters[16] (Mulaik, James, Van Alstine, Bennett, Lind, & Stilwell, 1989). This notion can be recast in terms of falsifiability. Every parameter that is free to vary across some range represents one fewer dimension along which the whole model could be falsified (Mulaik, 2001). To the extent to which the potential for falsification is a desirable characteristic of a model, it would behoove researchers to specify models with many overidentifying constraints rather than models with many free parameters. The danger, of course, is that so many theory-implied constraints could be placed on a model that complexity becomes too low to yield good fit with *any* data set, even one mostly in agreement with theoretical predictions. In other words, care must be taken to ensure that a model is neither too complex nor too constrained – balance must be sought. It is unclear how such balance might be obtained in an objective manner.

[16] The significance, direction, and magnitude of individual parameters may also be of theoretical interest, but hypotheses regarding these model characteristics are typically tested by means of individual Wald tests.

5.4 Limitations

5.4.1 Random Dispersion Matrices

Although the inferences drawn from the illustrations presented in this dissertation are believed to be valid and generally applicable, there were nevertheless some limitations to the strategy which might limit generalizability. For example, the present investigation involved generation of correlation matrices in which all elements were positive. This was done in an attempt to address Roberts and Pashler's (2000) emphasis on fitting models to *plausible* data rather than to *possible* data. It is unlikely, for example, that self-esteem will ever be found to correlate negatively with life satisfaction, making certain assumptions about adequate sample size and valid instruments. It was assumed that in most applications, variables can be meaningfully rescaled such that all correlations are expected to be in the positive direction. Thus, it was considered acceptable to restrict all population correlations to lie above zero. However, this restriction is acknowledged to be unrealistic in some situations, and may have consequences in situations where negative correlations are a possibility. In fact, there are situations in which it is necessary to specify models with negatively correlated MVs, as in some models containing formative composites whose indicators are to be additively combined (Bollen, 1984; MacCallum & Browne, 1993) or models containing derivatives and second derivatives as indicators (Boker & Nesselroade, in press).

5.4.2 Focus on OLS Estimation

In this dissertation, preference was given to OLS estimation to the exclusion of other common estimation methods such as ML or generalized least squares (GLS). OLS was chosen as a matter of convenience, and because it minimizes a simple, residual-based goodness of fit index. In addition, OLS is more computationally robust than ML, and is less prone to breaking down when confronted with singular or near-singular dispersion matrices. OLS is also immune to a problem that can occur from time to time with some data sets when ML estimation is used. When unique variances are small, the reproduced correlation matrix can have one or more very small eigenvalues, which tends to grossly inflate the χ^2 statistic associated with ML (Browne et al., 2002). Barring estimation problems, it is potentially important to examine issues of complexity using discrepancy functions other than OLS. Because distributional assumptions are another component of model complexity (Pitt et al., 2002), there may exist situations in which one model (Model A) is found to be more complex than another (Model B) by the OLS criterion, but Model B is found to be more complex than Model A when some other criterion is used.

The use of OLS estimation naturally entailed use of the RMSR fit index. RMSR was used to the exclusion of other fit indices because of its relative simplicity, the intuitive relationship between residuals and model fit, and the fact that it contains no correction for complexity and thus reflects a pure measure of goodness of fit. In addition, although RMSR can be easily computed when other discrepancy functions are used, it is minimized in OLS estimation. It is acknowledged that using RMSR = .08 as a universal criterion of good fit may not always be appropriate. When possible, it is recommended

that RMSR be supplemented in practice with the i^* uniform index-of-fit of Botha et al. (1988).

5.5 Directions for Future Research

5.5.1 Sampling Issues

The findings presented in this dissertation illuminate many exciting avenues for future research. For example, future research should examine the issue of complexity in light of sampling. That is, future studies should address the extent to which model complexity influences the interpretability of results when samples of size *N* are drawn from populations with known characteristics. In this dissertation, all matrices were generated without regard to whether they should be considered population or sample matrices. However, Roberts and Pashler (2000) point out that variability of the data should not be ignored in the model evaluation process. The variability of the data to which a model could potentially be fit is proportional to $n^{-1/2}$, so sample size represents an important component of model complexity.

5.5.2 Random Covariances

The MCMC method developed for generation of random correlation matrices in Chapter 2 has many potential applications. In addition to its demonstrated usefulness for investigating baseline rates of model fit, MCMC can be used to generate matrices for studies investigating convergence rates or improper solutions for models applied to correlational data. Simple methods exist for producing raw data from correlation

matrices, so the MCMC method could theoretically be used to generate random correlation matrices for the purpose of generating raw data for various modeling applications. For example, Monte Carlo studies investigating the consequences of missing data for statistical analyses could benefit greatly from this method of data generation. Perhaps the most obvious use for random correlation matrices is in the study of the distributional properties of uniform dispersion matrices. Even though elements and eigenvalues of uniform random matrices can be generated numerically (as plotted in Figures 2.2 to 2.9), their distributional form has yet to be defined analytically.

Future research should also address the question of generation of random covariance matrices rather than limiting conclusions to those which can be based on correlation matrices alone. Random covariance matrices would allow the examination, in terms of model complexity, of multi-sample structural equation models and latent growth curve models, two popular applications of SEM. However, in general, it may not be reasonable to expect that random correlation matrices rescaled to covariance matrices (or vice versa) will share the desirable property of representativeness. The implications of this observation should be explored for any study employing Monte Carlo generation of dispersion matrices. Generation of covariance matrices using the MCMC approach should theoretically involve only minor extensions to the methods presented here. For example, if the researcher sets upper and lower bounds for the allowable population standard deviation of each MV, the diagonal elements of the generated dispersion matrices may be perturbed along with the off-diagonal elements. Of course, a better approach to generating random correlation or covariance matrices would be to develop an

analytic generation mechanism which always results in positive definite matrices; i.e., one that does not involve the inevitably inefficient accept/reject stage of the MCMC algorithm. However, until the distributional properties of eigenvalues for random gramian matrices are developed, it is doubtful that an analytic method can be developed. Future research should be devoted to the investigation of these distributional properties.

5.5.3 Monte Carlo Investigation of Model Complexity

If the MCMC approach to matrix generation is to be further developed and refined, efforts should be undertaken to find more efficient ways to sample from the posterior distribution of correlation matrices. Even though MCMC is far more efficient than the UCM method described in Chapter 2, the proportion of acceptable to unacceptable matrices becomes very small as matrix order increases beyond 12. Consequently, even the MCMC algorithm has trouble locating acceptable matrices. As computers become faster, this problem may disappear within a few years. Alternatively, a possible solution exists in that multiple Markov chains may be used instead of a single chain. It may also be possible to find optimum values of generation parameters such as jump size, burn-in, and thinning number. In addition, it could be the case that different proposal distributions can be used to create more efficient versions of the algorithm.

5.5.4 Model Complexity and Power

Roberts and Pashler (2000) note that the variability of the data is often ignored in modeling. In the SEM context, variable data means small sample sizes, and it is well

known that smaller sample sizes reduce parameter precision and lower the power to reject inadequate models. However, it is also likely that there is a strong link between complexity and the statistical power to reject a false or poor model, such that models with high complexity may be difficult to reject. Complex models are characterized by the ability to fit a broad range of possible data patterns. The more data patterns (i.e., the greater the proportion of the data space) a model can fit well by some criterion, the more difficult it will be to find data that can falsify it. In other words, complex models are by definition difficult to falsify. All things being equal, larger samples are needed to reject models that are more difficult to falsify.

Current approaches to assessing power in the context of structural equation modeling (e.g., MacCallum et al., 1996; MacCallum & Hong, 1997) use fit indices which correct for complexity in terms of the number of parameters, but not for other aspects of model complexity (Zhang, 1999). Inadequate treatment of model complexity in the context of power determination has the potential to yield misleading conclusions. For example, MacCallum et al. (1996) show that the power to reject the null hypothesis of close fit (using the RMSEA = .05 criterion) involves *N*, *df*, and effect size (operationally, two values of RMSEA representing *null* and *alternative* levels of fit). Hence, the determination of the minimum sample size *N* necessary to achieve a given level of power depends on *df*, effect size, and the desired level of power. In principle, model complexity is already incorporated into this process in the form of *df*, but in earlier sections it has been shown that *df* does not contain all pertinent information about a model's complexity. For example, it is quite possible to find two models (A and B) with equal degrees of

freedom and more or less equal fit to a given data set using RMSEA, and to conclude on the basis of this good fit that the models are of equal quality (or have comparable generalizability). However, closer examination of model complexity may reveal that the functional form of Model A gives it a substantial advantage over Model B in its ability to fit any given data. This difference in baseline ability to fit data is not reflected in *df*, and thus will not be taken into account in power analysis. The continued use of RMSEA in power analysis, while consistent within the framework established by MacCallum et al. (1996), may be suboptimal with respect to the importance of model complexity. If a new approach to power analysis and determination of minimum *N* were based on a fit index which more completely accounted for model complexity, it might be the case that Model A is found to require a larger *N* to reject the hypothesis of close fit than is Model B. Complexity could perhaps be incorporated into the MacCallum et al. power analysis framework by using knowledge of complexity to guide the choice of the alternative RMSEA. Future research should be undertaken to fully explore the relationship between model complexity and the power to reject poor models.

5.5.5 The Future of Fit Assessment in SEM

Ideally, in the context of model comparison, we would like to see models with equal fit but different complexities or models with equal complexities but different fits. In the former situation, complexity would be the obvious deciding factor (Forster & Sober, 1994; Quine, 1966); in the latter situation, the better fitting model would emerge as the winner (Dunn, 2000). Unfortunately, researchers are rarely presented with so convenient

an outcome – usually, the models being compared vary along dimensions of both fit and complexity, and the case is here made that traditional SEM fit indices do not adequately deal with this conflation of fit and complexity. Several times it was implied or stated that most SEM fit indices correct for model complexity in a crude, impractical fashion, yet no practical alternative was offered. The method used in this dissertation involved generation of representative data and fitting candidate models to these data to gauge relative complexity. That does not imply that simulation is the only, or even the best, method of quantifying parsimony.

A promising alternative to traditional fit indices has come to light in recent years in the information theoretic literature. The principal of *minimum description length* (MDL), mentioned briefly in Chapter 1, may present the ideal alternative to traditional fit indices in the SEM paradigm. MDL is a formalization of *Kolmogorov complexity* (Grünwald, 2000; Rissanen, 1996), the shortest code length necessary to fully represent a data sequence. *Stochastic complexity*, which is used in the practical application of MDL, is analogous to Kolmogorov complexity, using a class of models rather than an encoding language as a basis for expressing complexity. One way to represent MDL is:

$$MDL_{1996} = -\ln(L) + \frac{k}{2}\ln\left(\frac{n}{2\pi}\right) + \ln \int_{\theta \in \mathfrak{M}_k} \sqrt{\left|I\left(\hat{\theta}\right)\right|} d\theta \qquad (5.1)$$

where k is the number of free model parameters, $\hat{\theta}$ is the collection of maximum likelihood parameter estimates for a particular model drawn from model class $\mathfrak{M}$ with k free parameters, L is the maximum likelihood loss function, and $\left|I\left(\hat{\theta}\right)\right|$ is the determinant

of the Fisher information matrix (Grünwald, 2000, Rissanen, 1996, 2001a). The first term can be considered a badness of fit measure and the next two terms together can be considered a measure of model complexity. The second term involves the familiar number of free model parameters (k), but also recognizes that model complexity increases logarithmically with sample size (n).[17] The third term in Equation 5.1 embodies the notion of functional form complexity (Myung & Pitt, 1997). If most models in $\mathfrak{M}$ are highly similar, then $\left|I\left(\hat{\theta}\right)\right|$ will be small. If, however, $\mathfrak{M}$ contains many distinguishable models, $\left|I\left(\hat{\theta}\right)\right|$ will be large. MDL thus contains, in its second and third terms, a formal realization of Jeffreys' (1931, 1957, 1961) intuitive operationalization of complexity which takes into account both the number of free parameters and functional form (Keuzenkamp, McAleer, & Zellner, 2001). In addition, the first and third terms reflect the particular estimation method used in the derivation of MDL (ML) and, further, because $\sqrt{\left|I\left(\hat{\theta}\right)\right|}$ is integrated over the parameter space $\theta \in \mathfrak{M}$, parameter range is also considered. MDL thus goes beyond the counting of free parameters and considers other information relevant to complexity, including functional form, parameter range, sample size, and estimation method. If two competing models are equally stochastically complex, the use of MDL will reduce to a simple comparison in terms of goodness of fit (Su, Myung, & Pitt, in press).

A newer version of MDL, which represents a further refinement of MDL_{1996}, is:

[17] With MDL, the contribution of functional form to complexity approaches zero as n approaches infinity.

$$MDL_{2001} = -\ln\left(\frac{f\left(x_{obs};\hat{\theta}\left(x_{obs}\right)\right)}{\int f\left(x;\hat{\theta}\left(x\right)\right)dx}\right), \tag{5.2}$$

where *f* is the maximum likelihood function. MDL_{2001} frames the complexity component of MDL as a normalized sum of all maximum likelihood best fits (Rissanen, 2001b). MDL_{1996} is not exactly equal to the more complete MDL_{2001}, but provides a very close approximation (Su et al., in press). The integration necessary for MDL_{2001} occurs over the space consisting of all possible data patterns (i.e., the data space) rather than all possible parameter values (as in MDL_{1996}). The extent of the data space that a model could potentially fit well has, in fact, been described as an intuitive metric for complexity (Dunn, 2000; Myung et al., 2000; Myung & Pitt, 1997; Pitt et al., 2002).[18] The relationship between the MDL criterion and the graphical method presented in this dissertation is worth noting. One way of conceptualizing the complexity portion of MDL is as the sum of best ML fits (i.e., *L*s) to all possible data. In other words, best-fitting ML values are plotted against data for all possible data patterns, and integration is used to find the area under the resulting surface. However, if a researcher wanted simply to rank several models in increasing order of complexity, finding the area is unnecessary; all that is required is the expected mean likelihood for each model. The graphical method developed in the present study does exactly that, and can be viewed as an empirical analog to finding the expected best-fit discrepancy function value, using OLS rather than

[18] For empirical evaluations of MDL compared to other model selection criteria, see Myung (2000) and Myung and Pitt (in press).

ML. Theoretically, using ML instead of OLS, the mean minimum-fit log likelihood will be an estimate of the complexity component of MDL_{2001}. It is likely that a rank ordering of models using MDL and another using mean RMSR would be similar or identical in most modeling situations.

Unfortunately, both MDL_{1996} and MDL_{2001} are notoriously difficult to compute. The primary hurdle preventing implementation of MDL_{1996} is that computation of $\left|I\left(\hat{\theta}\right)\right|$ is virtually intractable for highly parameterized models (Kim, 2002; Su et al., in press; Zhang, 1999), such as those typically encountered in SEM. MDL_{2001} contains a similarly complicated integral. Fortunately, Su et al. (in press) provide a reparameterization of $I\left(\hat{\theta}\right)$ for use in MDL_{1996} which involves far less computation, and numerical integration approaches also look promising (Kim, 2002; Zhang, 1999).

MDL holds great potential for application in the SEM paradigm. In current practice, researchers are obliged to ensure that competing models are nested in order to enable statistical comparison. With generalizability indices like MDL, nesting is no longer necessary. Facilitating the use of MDL in SEM software packages would allow comparison of any number of non-nested models (Rissanen, 2001a) or evaluation and comparison of nonlinear models (Myung & Pitt, in press). MDL is therefore a much more widely applicable model selection criterion than, e.g., AIC, which corrects only for the number of parameters, and any of a number of SEM model fit indices which prohibit comparison of non-nested models. Finally, the ability to apply MDL would permit the

comparison of models in which parameter range has been restricted (an unsolved problem noted by Rindskopf, 1984).

On the other hand, MDL cannot be used to compare models with different numbers of MVs. In addition, no formal procedures have yet been developed for power analysis or the statistical comparison of models using MDL.[19] Furthermore, MDL is currently defined only in reference to the maximum likelihood discrepancy function. For example, there is no obvious value in OLS estimation corresponding to the sample size component of MDL; sample size is simply not an issue in obtaining point estimates and model fit indices using OLS estimation, whereas it is a very important component of ML estimation. Reframing MDL for use with OLS estimation (or other methods) is thus not as simple as replacing the $-\ln(L)$ term in MDL_{1996}, but would be useful for situations in which ML yields improper solutions or otherwise uninterpretable results.

It is strongly recommended that MDL be investigated in light of its applicability to model evaluation and comparison in the context of SEM. It would be particularly beneficial to the field if MDL_{1996} were routinely included in output from SEM software packages so that it could be used in conjunction with traditional fit indices. MDL includes information not only about goodness of fit and complexity due to the number of free parameters, but also complexity due to functional form, parameter range, sample size, and estimation method. It is recommended that, at the very least, fit indices should

[19] Indeed, it is questionable whether it would be wise to develop such procedures (Burnham & Anderson, 2002), as there are different motivations behind hypothesis testing and model selection. On the other hand, if several competing models are ranked in order of increasing MDL, and it is found that the two best models are nearly indistinguishable, it would be useful to have the ability to perform a statistical test of the null hypothesis that the two models are equally generalizable. See Golden (2000) for additional material.

be interpreted in conjunction with a measure of model complexity, or perhaps the *i** uniform index-of-fit proposed by Botha et al. (1988). In particular, it is recommended that future research be devoted to the application of MDL in the context of structural equation modeling in particular and other sophisticated analytic techniques used in the social sciences in general.

5.6 Conclusion

Throughout this dissertation it has been assumed that it is preferable to select the simplest model from a set of alternative models, all things being equal. Comparatively less complex models are generally easier to understand, and tend to generalize better than more complex models when sample size is small. The world is a very complex place, however, so it may well be argued that in many cases the less complex of two competing models should *not* be favored. There are likely many situations in which the more complex model should be preferred because it in fact is closer to the truth (Mulaik et al., 1989; Quine, 1966). Collyer (1985) states, "the better fit of weaker [more complex] models neither validates nor invalidates the underlying ideas" (p. 178). Copernicus' heliocentric model of planetary motion was ultimately proven false, but in its time it served as a useful approximation to the truth for purposes of prediction, and led to many scientific advances. Newtonian physics is now thought to be fundamentally false in light of the mounting evidence in favor of Einstein's theory of relativity, but that does not

prevent it from being used daily as a useful approximation to the truth.[20] The point is that, when researchers form theories and models as part of their search for the truth, pragmatic considerations must enter into the process at some point. All theories are false, and all models are at least as false as the theories they represent, so the best we can hope for from a model is a useful approximation to the truth. Selecting the more parsimonious of two otherwise equally good models achieves that goal.

The purpose of this dissertation was to introduce to the SEM literature an expanded conceptualization of model complexity, and to demonstrate the consequences of ignoring it. It is hoped that this exploration will provide significant steps toward the development of a new family of SEM fit indices which correct for model complexity in a more complete fashion than by simply adding a function of the number of free parameters. If psychology is to advance as a science, researchers need to recognize that continuing to use crude methodology to assess and compare models can have dire consequences. The central message of this dissertation is that good fit of a hypothesized model to observed data, although desirable, can arise from all sorts of reasons, and being close to the truth is only one of those reasons. Good fit can also result from a model's inherent ability to predict data patterns. Cherished models may have to be abandoned if their past successes can be ascribed more to complexity than to their ability to reflect true underlying processes. To avoid this pitfall, future applications of SEM should involve

[20] Keuzenkamp and McAleer (1995) note rhetorically, "If ... Einstein's relativity theory is superior to classical mechanics in describing red shifts in the spectra emitted by stars, should it be used to predict the position at time *t* of a football that has been set in motion at *t* – 1?" (p. 18).

either explicit consideration of model complexity (such as with generalizability indices like MDL) or convincing justifications for ignoring it.

LIST OF REFERENCES

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In B. N. Petrov & F. Csaki (Eds.), *Second international symposium on information theory*, p. 267. Akademiai Kiado, Budapest, Hungary.

Bamber, D., & van Santen, J. P. H. (2000). How to assess a model's testability and identifiability. *Journal of Mathematical Psychology*, *44*, 20-40.

Beichl, I., & Sullivan, F. (2000). The Metropolis Algorithm. *Computing in Science & Engineering*, *2*(1), 65-69.

Beier, M. E., & Ackerman, P. L. (2001). Current-events knowledge in adults: An investigation of age, intelligence, and nonability determinants. *Psychology and Aging*, *16*(4), 615-628.

Bem, S. L. (1974). The measurement of psychological androgyny. *Journal of Consulting and Clinical Psychology*, *46*, 155-162.

Bendel, R. B., & Afifi, A. A. (1977). Comparison of stopping rules in forward stepwise regression. *Journal of the American Statistical Association*, *72*, 46-53.

Bendel, R. B., & Mickey, M. R. (1978). Population correlation matrices for sampling experiments. *Communications in Statistics - Simulation and Computation*, *B7*, 163-182.

Bentler, P. M. (1995). *EQS structural equations program manual*. Encino, CA: Multivariate Software.

Bentler, P. M., & Bonnett, D. G. (1980). Significance tests and goodness of fit in the analysis of covariance structures. *Psychological Bulletin*, *88*(3), 588-606.

Boker, S. M., & Nesselroade, J. R. (in press). A method for modeling the intrinsic dynamics of intraindividual variability: Recovering the parameters of simulated oscillators in multi-wave panel data. *Multivariate Behavioral Research*.

Bollen, K. A. (1984). Multiple indicators: Internal consistency or no necessary relationship? *Quality and Quantity*, *18*, 377-385.

Bollen, K. A. (1989). A new incremental fit index for general structural equation models. *Sociological Research and Methods*, *17*, 303-316.

Botha, J. D., Shapiro, A., & Steiger, J. H. (1988). Uniform indices-of-fit for factor analysis models. *Multivariate Behavioral Research*, *23*, 443-450.

Box, G. E. P., & Jenkins, G. M. (1970). *Time series analysis: Forecasting and control*. London: Holden-Day.

Bozdogan, H. (2000). Akaike's information criterion and recent developments in information complexity. *Journal of Mathematical Psychology*, *44*, 62-91.

Browne, M. W. (1984). The decomposition of multitrait-multimethod matrices. *British Journal of Mathematical and Statistical Psychology*, *37*, 1-21.

Browne, M. W. (1985). Multitrait-multimethod matrices. In S. Kotz & N. G. Johnson (Eds.), *Encyclopedia of statistical sciences* (Vol. 5). New York: Wiley.

Browne, M. W. (1993). Models for multitrait-multimethod matrices. In R. Steyer, K. F. Wender, & K. F. Widaman (Eds.), *Psychometric methodology. Proceedings of the 7th European meeting of the Psychometric Society in Trier* (pp. 61-73). Stuttgart and New York: Gustav Fischer Verlag.

Browne, M. W. (2000). Cross-validation methods. *Journal of Mathematical Psychology*, *44*, 108-132.

Browne, M. W., & Cudeck, R. (1993). Alternative ways of assessing model fit. In K. A. Bollen & J. S. Long (Eds.), *Testing structural equation models*. Newbury Park, CA: Sage Publications.

Browne, M. W., MacCallum, R. C., Kim, C., Andersen, B. L., & Glaser, R. (2002). When fit indices and residuals are incompatible. *Psychological Methods*, *7*(4), 403-421.

Burnham, K. P., & Anderson, D. R. (2002). *Model selection and multimodel inference* (2nd ed.). New York: Springer-Verlag.

Byrne, B. M. (1989). *A primer of LISREL: Basic applications and programming for confirmatory factor analytic models*. New York: Springer-Verlag.

Campbell, D. T., & Fiske, D. W. (1959). Convergent and discriminant validation by the multitrait-multimethod matrix. *Psychological Bulletin*, *56*(2), 81-105.

Chalmers, C. P. (1975). Generation of correlation matrices with a given eigen-structure. *The Journal of Statistical Computation and Simulation*, *4*, 133-139.

Chan, N. N., & Li, K. H. (1983). Diagonal elements and eigenvalues of a real symmetric matrix. *Journal of Mathematical Analysis and Applications*, *91*, 562-566.

Chan, S.-F. F. (1992). *A comparison of multiplicative and additive models for multitrait-multimethod matrices based on the uniform index-of-fit and RMSE*. Unpublished master's thesis, The Ohio State University, Columbus, Ohio.

Chu, M. T. (1995). Constructing a Hermitian matrix from its diagonal entries and eigenvalues. *SIAM Journal on Matrix Analysis and Applications*, *16*(1), 207-217.

Coffman, S., Levitt, M. J., Deets, C., & Quigley, K. L. (1991). Close relationships in mothers of distressed and normal newborns: Support, expectancy confirmation, and maternal well-being. *Journal of Family Psychology*, *5*(1), 93-107.

Collyer, C. E. (1985). Comparing strong and weak models by fitting them to computer-generated data. *Perception and Psychophysics*, *38*, 476-481.

Costa, P. T., & McCrae, R. R. (1992). *Revised NEO Personality Inventory and Five-Factor Inventory professional manual*. Odessa, FL: Psychological Assessment Resources.

Cudeck, R. (1988). Multiplicative models and MTMM matrices. *Journal of Educational Statistics*, *13*(2), 131-147.

Cutting, J. E. (2000). Accuracy, scope, and flexibility of models. *Journal of Mathematical Psychology*, *44*, 3-19.

Cutting, J. E., Bruno, N., Brady, N. P., & Moore, C. (1992). Selectivity, scope, and simplicity of models: A lesson from fitting judgments of perceived depth. *Journal of Experimental Psychology: General*, *121*(3), 364-381.

Davies, P. I., & Higham, N. J. (2000). Numerically stable generation of correlation matrices and their factors. *BIT*, *40*, 640-651.

Dunn, J. C. (2000). Model complexity: The fit to random data reconsidered. *Psychological Research*, *63*, 174-182.

Forster, M. R. (2000). Key concepts in model selection: Performance and generalizability. *Journal of Mathematical Psychology*, *44*, 205-231.

Forster, M. R. (2001). The new science of simplicity. In A. Zellner, H. A. Keuzenkamp, & M. McAleer (Eds.), *Simplicity, inference, and modelling* (pp. 83-119). Cambridge: Cambridge University Press.

Forster, M. R., & Sober, E. (1994). How to tell when simpler, more unified, or less *ad hoc* theories will provide more accurate predictions. *British Journal for the Philosophy of Science*, *45*, 1-35.

Gentle, J. E. (1998). *Random number generation and Monte Carlo methods*. New York: Springer.

Gerbing, D. W., & Anderson, J. C. (1993). Monte Carlo evaluations of goodness-of-fit indices for structural equation models. In K. A. Bollen & J. S. Long (Eds.), *Testing structural equation models*. Newbury Park, CA: Sage Publications.

Gilks, W. R., Richardson, S., & Spiegelhalter, D. J. (1996). *Markov chain Monte Carlo in practice*. London: Chapman & Hall.

Gleser, L. J. (1976). A canonical representation for the noncentral Wishart distribution useful for simulation. *Journal of the American Statistical Association*, *71*, 690-695.

Goff, M., & Ackerman, P. L. (1992). Personality-intelligence relations: Assessment of typical intellectual engagement. *Journal of Educational Psychology*, *84*, 537-552.

Golden, R. M. (2000). Statistical tests for comparing possibly misspecified and nonnested models. *Journal of Mathematical Psychology*, *44*, 153-170.

Grünwald, P. (2000). Model selection based on minimum description length. *Journal of Mathematical Psychology*, *44*, 133-152.

Hartley, H. O., & Harris, D. L. (1963). Monte Carlo computations in normal correlation problems. *Journal of the Association for Computing Machinery*, *10*, 302-306.

Hohn, F. E. (1966). *Elementary matrix algebra*. New York: MacMillan.

Holmes, R. B. (1991). On random correlation matrices. *SIAM Journal on Matrix Analysis and Applications*, *12*(2), 239-272.

Hu, L.-T., & Bentler, P. M. (1998). Fit indices in covariance structure modeling: Sensitivity to underparameterized model misspecification. *Psychological Methods*, *3*, 424-453.

Hurvich, C. M. (1997). Mean square over degrees of freedom: New perspectives on a model selection treasure. In D. R. Brillinger, L. T. Fernholz, & S. Morgenthaler (Eds.), *The practice of data analysis: Essays in honor of John W. Tukey* (pp. 203-215). Princeton, NJ: Princeton University Press.

James, L. R., Mulaik, S. A., & Brett, J. M. (1982). *Causal analysis: Assumptions, models, and data*. Beverly Hills, CA: Sage.

Jeffreys, H. (1931). *Scientific inference*. Cambridge: Cambridge University Press.

Jeffreys, H. (1957). *Scientific inference* (2nd ed.). Cambridge: Cambridge University Press.

Jeffreys, H. (1961). *Theory of probability* (2nd ed.). London: Oxford University Press.

Johnson, D. G., & Welch, W. J. (1980). The generation of pseudo-random correlation matrices. *Journal of Statistical Computation and Simulation*, *11*, 55-69.

Jöreskog, K. G. (1969). A general approach to confirmatory maximum likelihood factor analysis. *Psychometrika*, *34*(2), 183-202.

Jöreskog, K. G., & Sörbom, D. (1996). *LISREL 8 user's reference guide*. Uppsala: Scientific Software International.

Keuzenkamp, H. A., & McAleer, M. (1995). Simplicity, scientific inference and econometric modelling. *The Economic Journal*, *105*, 1-21.

Keuzenkamp, H. A., & McAleer, M. (1997). The complexity of simplicity. *Mathematics and Computers in Simulation*, *43*, 553-561.

Keuzenkamp, H. A., McAleer, M., & Zellner, A. (2001). The enigma of simplicity. In A. Zellner, H. A. Keuzenkamp, & M. McAleer (Eds.), *Simplicity, inference, and modelling* (pp. 1-10). Cambridge: Cambridge University Press.

Kim, W. (2002). *Applications of Markov Chain Monte Carlo in MDL-based model selection*. Unpublished masters thesis, The Ohio State University, Columbus, Ohio.

Leahy, K. (1994). The overfitting problem in perspective. *AI Expert*, *9*(10), 35-36.

Li, S.-C., Lewandowsky, S., & DeBrunner, V. E. (1996). Using parameter sensitivity and interdependence to predict model scope and falsifiability. *Journal of Experimental Psychology: General*, *125*(4), 360-369.

Lin, S. P., & Bendel, R. B. (1985). Algorithm AS 213: Generation of population correlation matrices with specified eigenvalues. *Applied Statistics*, *34*(2), 193-198.

MacCallum, R. C. (2003). Working with imperfect models. *Multivariate Behavioral Research*, *38*(1), 113-139.

MacCallum, R. C., & Browne, M. W. (1993). The use of causal indicators in covariance structure models: Some practical issues. *Psychological Bulletin*, *114*(3), 533-541.

MacCallum, R. C., Browne, M. W., & Sugawara, H. M. (1996). Power analysis and determination of sample size for covariance structure modeling. *Psychological Methods*, *1*, 130-149.

MacCallum, R. C., & Hong, S. (1997). Power analysis in covariance structure modeling using GFI and AGFI. *Multivariate Behavioral Research*, *32*, 193-210.

Marasinghe, M. G., & Kennedy, W. J., Jr. (1982). Direct methods for generating extreme characteristic roots of certain random matrices. *Communications in Statistics - Simulation and Computation*, *11*, 527-542.

Marsaglia, G., & Olkin, I. (1984). Generating correlation matrices. *SIAM Journal on Scientific and Statistical Computing*, *5*, 470-475.

Marsh, H. W., & Balla, J. (1994). Goodness of fit in confirmatory factor analysis: The effects of sample size and model parsimony. *Quality & Quantity*, *28*, 185-217.

McDonald, R. P. (1989). An index of goodness-of-fit based on noncentrality. *Journal of Classification*, *6*, 97-103.

McDonald, R. P., & Marsh, H. W. (1990). Choosing a multivariate model: Noncentrality and goodness of fit. *Psychological Bulletin*, *107*, 247-255.

Mulaik, S. A. (2001). The curve-fitting problem: An objectivist view. *Philosophy of Science*, *68*, 218-241.

Mulaik, S. A., James, L. R., Van Alstine, J., Bennett, N., Lind, S., & Stilwell, C. D. (1989). Evaluation of goodness-of-fit indices for structural equation models. *Psychological Bulletin*, *105*(3), 430-445.

Myung, I. J. (2000). The importance of complexity in model selection. *Journal of Mathematical Psychology*, *44*, 190-204.

Myung, I. J., Balasubramanian, V., & Pitt, M. A. (2000). Counting probability distributions: Differential geometry and model selection. *Proceedings of the National Academy of Sciences*, *97*(21), 11170-11175.

Myung, I. J., Forster, M. R., & Browne, M. W. (2000). Model selection [Special issue]. *Journal of Mathematical Psychology*, *44*.

Myung, I. J., & Pitt, M. A. (1997). Applying Occam's razor in modeling cognition: A Bayesian approach. *Psychonomic Bulletin & Review*, *4*, 79-95.

Myung, I. J., & Pitt, M. A. (in press). Model comparison methods. In L. Brand & M. L. Johnson (Eds.), *Methods in enzymology: Numerical computer methods, Part D*.

Newsom, J. T., & Schulz, R. (1996). Social support as a mediator in the relation between functional status and quality of life in older adults. *Psychology and Aging*, *11*, 34-44.

Nickerson, R. S. (1998). Confirmation bias: A ubiquitous phenomenon in many guises. *Review of General Psychology*, *2*(2), 175-220.

Odell, P. L., & Feiveson, A. H. (1966). A numerical procedure to generate a sample covariance matrix. *Journal of the American Statistical Association*, *61*, 199-203.

Pitt, M. A., Kim, W., & Myung, I. J. (2003). *Complexity versus generalizability in model selection*. *Psychonomic Bulletin and Review*, *10*, 29-44.

Pitt, M. A., & Myung, I. J. (2002). When a good fit can be bad. *Trends in Cognitive Sciences*, *6*(10), 421-425.

Pitt, M. A., Myung, I. J., & Zhang, S. (2002). Toward a method of selecting among computational models of cognition. *Psychological Review*, *109*(3), 472-491.

Platt, J. R. (1964, October 16). Strong inference. *Science*, *146*(3642), 347-353.

Popper, K. R. (1959). *The logic of scientific discovery*. London: Hutchinson.

Quine, W. V. (1966). *The ways of paradox and other essays*. New York: Random House.

Rindskopf, D. (1984). Using phantom and imaginary latent variables to parameterize constraints in linear structural models. *Psychometrika*, *49*(1), 37-47.

Rissanen, J. (1987). Stochastic complexity and the MDL principle. *Econometric Reviews*, *6*, 85-102.

Rissanen, J. (1996). Fisher information and stochastic complexity. *IEEE Transactions on Information Theory*, *IT-42*(1): 40-47.

Rissanen, J. (2001a). Simplicity and statistical inference. In A. Zellner, H. A. Keuzenkamp, & M. McAleer (Eds.), *Simplicity, inference, and modelling* (pp. 156-164). Cambridge: Cambridge University Press.

Rissanen, J. (2001b). Strong optimality of the normalized ML models as universal codes and information in data. *IEEE Transactions on Information Theory*, *47*(5), 1712-1717.

Roberts, S., & Pashler, H. (2000). How persuasive is a good fit? A comment on theory testing. *Psychological Review*, *107*(2), 358-367.

Roberts, S., & Pashler, H. (2002). Reply to Rodgers and Rowe (2002). *Psychological Review*, *109*(3), 605-607.

Rodgers, J. L., & Rowe, D. C. (2002). Theory development should begin (but not end) with good empirical fits: A comment on Roberts and Pashler (2000). *Psychological Review*, *109*(3), 599-604.

Schaffer, C. (1993). Overfitting avoidance as bias. *Machine Learning*, *10*, 153-178.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*, 461-464.

Sober, E. (2001). What is the problem of simplicity? In A. Zellner, H. A. Keuzenkamp, & M. McAleer (Eds.), *Simplicity, inference, and modelling* (pp. 13-31). Cambridge: Cambridge University Press.

Stanley, J. C., & Wang, M. D. (1969). Restrictions on the possible values of r_{12}, given r_{13} and r_{23}. *Educational and Psychological Measurement*, *29*, 579-581.

Steiger, J. H. (1989). *EzPATH: A supplementary module for SYSTAT and SYGRAPH*. Evanston, IL: SYSTAT.

Steiger, J. H., & Lind, J. C. (1980). *Statistically based tests for the number of common factors*. Paper presented at the annual meeting of the Psychometric Society, Iowa City, IA.

Su, Y., Myung, I. J., & Pitt, M. A. (in press). Minimum description length and cognitive modeling. In P. Grünwald, I. J. Myung, & M. A. Pitt (Eds.), *Advances in minimum description length: Theory and applications*. MIT Press.

Swain, A. J. (1975). *Analysis of parametric structures for variance matrices*. Unpublished doctoral dissertation, University of Adelaide.

Tellegen, A. (1982). *Brief manual for the Multidimensional Personality Questionnaire (MPQ)*. University of Minnesota, Twin Cities, Minneapolis: Author.

Tucker, L. R, Koopman, R. F., & Linn, R. L. (1969). Evaluation of factor analytic research procedures by means of simulated correlation matrices. *Psychometrika*, *34*, 421-459.

Tucker, L. R, & Lewis, C. (1973). A reliability coefficient for maximum likelihood factor analysis. *Psychometrika*, *38*(1), 1-10.

Wothke, W. (1996). Models for multitrait-multimethod matrix analysis. In G. A. Marcoulides & R. E. Schumacher (Eds.) *Advanced structural equation modeling*. Mahwah, NJ: Erlbaum.

Wothke, W., & Browne, M. W. (1990). The direct product model for the MTMM matrix parameterized as a second order factor analysis model. *Psychometrika*, *55*(2), 255-262.

Wrinch, D., & Jeffreys, H. (1921). On certain fundamental principles of scientific inquiry. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *42*, 369-390.

Zellner, A., Keuzenkamp, H. A., & McAleer, M. (Eds.) (2001). *Simplicity, inference, and modelling*. Cambridge: Cambridge University Press.

Zha, H., & Zhang, Z. (1995). A note on constructing a symmetric matrix with specified diagonal entries and eigenvalues. *BIT*, *35*, 448-452.

Zhang, S. (1999). *Applications of geometric complexity and the minimum description length principle in mathematical modeling of cognition*. Unpublished doctoral dissertation, The Ohio State University, Columbus, Ohio.

APPENDIX A

FORTRAN CODE FOR GENERATING RANDOM CORRELATION MATRICES

The Fortran programs in Appendices A, B, and D were written and compiled using Microsoft Fortran Powerstation 4.0, a compiler for the Fortran 90 programming language. These programs are intended to be used directly from the compiler rather than in executable form. This strategy allows easy alteration of generation parameters such as matrix order, file names, etc. without the need to create a user interface.

```
! This program was designed to generate random positive population correlation
! matrices by 3 methods:
!   1. Botha, Shapiro, & Steiger's (1988) random uniform numbers method
!   2. Woojae Kim's MCMC method (2002, personal communication)
!   3. Davies & Higham's (2000) random eigenstructure method

program randomr5
  use portlib     ! external code library
  use msimsl      ! external code library
  implicit none

  integer :: ios,nmat,nr,i,j,k,m,o,array1(20,6),array2(20,6),array3(20,6),numneg,reject
  integer, allocatable, dimension(:,:)  :: eigarray1,eigarray2,eigarray3
  integer :: it,nynon0,nznon0,vecmeth,itr,subdiag,t,xflag,yflag,m1,m2,m3
  real*8  :: xtemp(1),vec(6),x,eig,jmpsize,thin
  real*8  :: minx,sumx,eps,randii(1),randjj(1),randkk(1),qtemp(1),xx,betax
  real*8  :: alpha,ii,jj,kk,s,c,t1,t2,msign,norm,vecdot,dot,ss
  real*8, allocatable, dimension(:)     :: freq1,univec,aa,y,z,tvec1,tvec2
  real*8, allocatable, dimension(:)     :: w,zeros,xvec,xcand,ycand,tmp,xtmp
  real*8, allocatable, dimension(:,:)   :: freq2,A,Q,Q2,B1,B2,cs
  real*8, allocatable, dimension(:,:)   :: B1b,B2b,d,u,xvect,xvecyy,yy
  real*4, allocatable, dimension(:,:)   :: r
  complex, allocatable, dimension(:)    :: eigens
  character(2)                          :: ordervar
  character(8)                          :: clocktime
  character(24)                         :: mxs_m1,mxs_m2,mxs_m3,eig_m1,eig_m2,eig_m3
  character(24) :: frq_m1,frq_m2,frq_m3,hst_m1,hst_m2,hst_m3,efq_m1,efq_m2,efq_m3
```

```
interface
  function dot(DX,DY,N)
    use msimsl
    implicit none
    real*8, dimension(:)  :: DX,DY
    integer               :: N,INCX=1,INCY=1
  end function
end interface

do o=7,7                 ! number of rows/cols per R matrix

  m1=0                   ! which method(s) should I use this time?
  m2=1                   ! 1=on; 0=off
  m3=0
  reject=0               ! tally used to compute rejection
  nmat=5000              ! number of matrices of each type
  nr=1                   ! number of random numbers to generate
  subdiag=int(o*(o-1)/2) ! number of subdiagonal elements
  allocate (r(o,o),eigens(o),freq1(6*nmat),freq2((6*nmat),2))
  allocate (xcand(subdiag),ycand(subdiag),tmp(subdiag))
  allocate (A(o,o),Q(o,o),aa(o),univec(o),y(o),z(o))
  allocate (B1(o,2),B2(2,o),tvec1(o),tvec2(o),xtmp(subdiag))
  allocate (cs(2,2),B1b(o,2),B2b(2,o),d(o,o),w(o),u(o,o),zeros(o))
  allocate (yy(1,o),eigarray1(50,o),eigarray2(50,o),eigarray3(50,o))
  if (o<10) ordervar='0'//char(o+48)
  if (o>9) ordervar=char(int(floor(real(o/10)))+48)//char(int(mod(o,10))+48)
  mxs_m1='c:\randomr\mxs_m1_'//ordervar//'.out'
  mxs_m2='c:\randomr\mxs_m2_'//ordervar//'.out'
  mxs_m3='c:\randomr\mxs_m3_'//ordervar//'.out'
  eig_m1='c:\randomr\eig_m1_'//ordervar//'.out'
  eig_m2='c:\randomr\eig_m2_'//ordervar//'.out'
  eig_m3='c:\randomr\eig_m3_'//ordervar//'.out'
  frq_m1='c:\randomr\frq_m1_'//ordervar//'.out'
  frq_m2='c:\randomr\frq_m2_'//ordervar//'.out'
  frq_m3='c:\randomr\frq_m3_'//ordervar//'.out'
  hst_m1='c:\randomr\hst_m1_'//ordervar//'.out'
  hst_m2='c:\randomr\hst_m2_'//ordervar//'.out'
  hst_m3='c:\randomr\hst_m3_'//ordervar//'.out'
  efq_m1='c:\randomr\efq_m1_'//ordervar//'.out'
  efq_m2='c:\randomr\efq_m2_'//ordervar//'.out'
  efq_m3='c:\randomr\efq_m3_'//ordervar//'.out'
  do i=1,20
    do j=1,6
      array1(i,j)=0
      array2(i,j)=0
      array3(i,j)=0
    enddo
  enddo
  do i=1,50
    do j=1,o
      eigarray1(i,j)=0
      eigarray2(i,j)=0
      eigarray3(i,j)=0
    enddo
  enddo

  !~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~!
  ! Method 1: Random Uniform Numbers  !
  !~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~!

  if (m1==1) then
    open(unit=1,file=mxs_m1,position="rewind",iostat=ios)  ! matrix file, method 1
    open(unit=4,file=eig_m1,position="rewind",iostat=ios)  ! eigenvalues for method 1
    open(unit=9,file=frq_m1,position="rewind",iostat=ios)  ! corr freqs for method 1
    open(unit=12,file=hst_m1,position="rewind",iostat=ios) ! histogram for method 1
    open(unit=17,file=efq_m1,position="rewind",iostat=ios) ! eigenvalue hist, method 1
```

```
 i=0
 do
   i=i+1
   if (i==nmat+1) exit
   do j=1,o ! fill r with random uniform numbers, 1s on the diagonal.
     do k=1,j
       call drnun(nr,xtemp)
       r(j,k)=xtemp(1)
       r(k,j)=r(j,k)
       r(j,j)=1
     enddo
   enddo
   CALL EVLRG (o,r,o,eigens) ! see if positive definite; if not, cycle.
   numneg=0
   do j=1,o
     eig=real(eigens(j))
     if (eig<=0) then
       numneg=numneg+1
     endif
   enddo
   if (numneg>0) then
     i=i-1
     cycle
   endif
   clocktime=CLOCK()
   write (6,1) 'Just did method 1, matrix ',i,o,' x ',o,clocktime
   1 format (1X,A32,I4,2X,I2,A2,I2,2X,A8)
   write (4,4) (real(eigens(j)),j=1,o) ! output eigenvalues for plots later.
   4 format (1X,(20F12.8,1X))
   do j=1,o
if (real(eigens(j))>=0.0.and.real(eigens(j))<0.1) eigarray1(1,j)=eigarray1(1,j)+1
if (real(eigens(j))>=0.1.and.real(eigens(j))<0.2) eigarray1(2,j)=eigarray1(2,j)+1
if (real(eigens(j))>=0.2.and.real(eigens(j))<0.3) eigarray1(3,j)=eigarray1(3,j)+1
if (real(eigens(j))>=0.3.and.real(eigens(j))<0.4) eigarray1(4,j)=eigarray1(4,j)+1
if (real(eigens(j))>=0.4.and.real(eigens(j))<0.5) eigarray1(5,j)=eigarray1(5,j)+1
if (real(eigens(j))>=0.5.and.real(eigens(j))<0.6) eigarray1(6,j)=eigarray1(6,j)+1
if (real(eigens(j))>=0.6.and.real(eigens(j))<0.7) eigarray1(7,j)=eigarray1(7,j)+1
if (real(eigens(j))>=0.7.and.real(eigens(j))<0.8) eigarray1(8,j)=eigarray1(8,j)+1
if (real(eigens(j))>=0.8.and.real(eigens(j))<0.9) eigarray1(9,j)=eigarray1(9,j)+1
if (real(eigens(j))>=0.9.and.real(eigens(j))<1.0) eigarray1(10,j)=eigarray1(10,j)+1
if (real(eigens(j))>=1.0.and.real(eigens(j))<1.1) eigarray1(11,j)=eigarray1(11,j)+1
if (real(eigens(j))>=1.1.and.real(eigens(j))<1.2) eigarray1(12,j)=eigarray1(12,j)+1
if (real(eigens(j))>=1.2.and.real(eigens(j))<1.3) eigarray1(13,j)=eigarray1(13,j)+1
if (real(eigens(j))>=1.3.and.real(eigens(j))<1.4) eigarray1(14,j)=eigarray1(14,j)+1
if (real(eigens(j))>=1.4.and.real(eigens(j))<1.5) eigarray1(15,j)=eigarray1(15,j)+1
if (real(eigens(j))>=1.5.and.real(eigens(j))<1.6) eigarray1(16,j)=eigarray1(16,j)+1
if (real(eigens(j))>=1.6.and.real(eigens(j))<1.7) eigarray1(17,j)=eigarray1(17,j)+1
if (real(eigens(j))>=1.7.and.real(eigens(j))<1.8) eigarray1(18,j)=eigarray1(18,j)+1
if (real(eigens(j))>=1.8.and.real(eigens(j))<1.9) eigarray1(19,j)=eigarray1(19,j)+1
if (real(eigens(j))>=1.9.and.real(eigens(j))<2.0) eigarray1(20,j)=eigarray1(20,j)+1
if (real(eigens(j))>=2.0.and.real(eigens(j))<2.1) eigarray1(21,j)=eigarray1(21,j)+1
if (real(eigens(j))>=2.1.and.real(eigens(j))<2.2) eigarray1(22,j)=eigarray1(22,j)+1
if (real(eigens(j))>=2.2.and.real(eigens(j))<2.3) eigarray1(23,j)=eigarray1(23,j)+1
if (real(eigens(j))>=2.3.and.real(eigens(j))<2.4) eigarray1(24,j)=eigarray1(24,j)+1
if (real(eigens(j))>=2.4.and.real(eigens(j))<2.5) eigarray1(25,j)=eigarray1(25,j)+1
if (real(eigens(j))>=2.5.and.real(eigens(j))<2.6) eigarray1(26,j)=eigarray1(26,j)+1
if (real(eigens(j))>=2.6.and.real(eigens(j))<2.7) eigarray1(27,j)=eigarray1(27,j)+1
if (real(eigens(j))>=2.7.and.real(eigens(j))<2.8) eigarray1(28,j)=eigarray1(28,j)+1
if (real(eigens(j))>=2.8.and.real(eigens(j))<2.9) eigarray1(29,j)=eigarray1(29,j)+1
if (real(eigens(j))>=2.9.and.real(eigens(j))<3.0) eigarray1(30,j)=eigarray1(30,j)+1
if (real(eigens(j))>=3.0.and.real(eigens(j))<3.1) eigarray1(31,j)=eigarray1(31,j)+1
if (real(eigens(j))>=3.1.and.real(eigens(j))<3.2) eigarray1(32,j)=eigarray1(32,j)+1
if (real(eigens(j))>=3.2.and.real(eigens(j))<3.3) eigarray1(33,j)=eigarray1(33,j)+1
if (real(eigens(j))>=3.3.and.real(eigens(j))<3.4) eigarray1(34,j)=eigarray1(34,j)+1
if (real(eigens(j))>=3.4.and.real(eigens(j))<3.5) eigarray1(35,j)=eigarray1(35,j)+1
if (real(eigens(j))>=3.5.and.real(eigens(j))<3.6) eigarray1(36,j)=eigarray1(36,j)+1
if (real(eigens(j))>=3.6.and.real(eigens(j))<3.7) eigarray1(37,j)=eigarray1(37,j)+1
```

```
if (real(eigens(j))>=3.7.and.real(eigens(j))<3.8) eigarray1(38,j)=eigarray1(38,j)+1
if (real(eigens(j))>=3.8.and.real(eigens(j))<3.9) eigarray1(39,j)=eigarray1(39,j)+1
if (real(eigens(j))>=3.9.and.real(eigens(j))<4.0) eigarray1(40,j)=eigarray1(40,j)+1
if (real(eigens(j))>=4.0.and.real(eigens(j))<4.1) eigarray1(41,j)=eigarray1(41,j)+1
if (real(eigens(j))>=4.1.and.real(eigens(j))<4.2) eigarray1(42,j)=eigarray1(42,j)+1
if (real(eigens(j))>=4.2.and.real(eigens(j))<4.3) eigarray1(43,j)=eigarray1(43,j)+1
if (real(eigens(j))>=4.3.and.real(eigens(j))<4.4) eigarray1(44,j)=eigarray1(44,j)+1
if (real(eigens(j))>=4.4.and.real(eigens(j))<4.5) eigarray1(45,j)=eigarray1(45,j)+1
if (real(eigens(j))>=4.5.and.real(eigens(j))<4.6) eigarray1(46,j)=eigarray1(46,j)+1
if (real(eigens(j))>=4.6.and.real(eigens(j))<4.7) eigarray1(47,j)=eigarray1(47,j)+1
if (real(eigens(j))>=4.7.and.real(eigens(j))<4.8) eigarray1(48,j)=eigarray1(48,j)+1
if (real(eigens(j))>=4.8.and.real(eigens(j))<4.9) eigarray1(49,j)=eigarray1(49,j)+1
if (real(eigens(j))>=4.9.and.real(eigens(j))<=5.0) eigarray1(50,j)=eigarray1(50,j)+1
   enddo
   vec(1)=r(2,1) ! tally correlations in certain ranges for histograms later.
   vec(2)=r(3,1)
   vec(3)=r(4,1)
   vec(4)=r(o,1)
   vec(5)=r(o,2)
   vec(6)=r(o,3)
   do j=1,6
     freq1(j-6+(i*6))=vec(j)
     x=vec(j)
     if (x>=0.00.and.x<0.05) array1(1,j)=array1(1,j)+1
     if (x>=0.05.and.x<0.10) array1(2,j)=array1(2,j)+1
     if (x>=0.10.and.x<0.15) array1(3,j)=array1(3,j)+1
     if (x>=0.15.and.x<0.20) array1(4,j)=array1(4,j)+1
     if (x>=0.20.and.x<0.25) array1(5,j)=array1(5,j)+1
     if (x>=0.25.and.x<0.30) array1(6,j)=array1(6,j)+1
     if (x>=0.30.and.x<0.35) array1(7,j)=array1(7,j)+1
     if (x>=0.35.and.x<0.40) array1(8,j)=array1(8,j)+1
     if (x>=0.40.and.x<0.45) array1(9,j)=array1(9,j)+1
     if (x>=0.45.and.x<0.50) array1(10,j)=array1(10,j)+1
     if (x>=0.50.and.x<0.55) array1(11,j)=array1(11,j)+1
     if (x>=0.55.and.x<0.60) array1(12,j)=array1(12,j)+1
     if (x>=0.60.and.x<0.65) array1(13,j)=array1(13,j)+1
     if (x>=0.65.and.x<0.70) array1(14,j)=array1(14,j)+1
     if (x>=0.70.and.x<0.75) array1(15,j)=array1(15,j)+1
     if (x>=0.75.and.x<0.80) array1(16,j)=array1(16,j)+1
     if (x>=0.80.and.x<0.85) array1(17,j)=array1(17,j)+1
     if (x>=0.85.and.x<0.90) array1(18,j)=array1(18,j)+1
     if (x>=0.90.and.x<0.95) array1(19,j)=array1(19,j)+1
     if (x>=0.95.and.x<=1.00) array1(20,j)=array1(20,j)+1
   enddo
   do j=1,o ! write matrix to output to file.
     write (1,2) (r(j,k),k=1,o)
     2 format (30F10.6)
   enddo
   write (1,3)
   3 format ()
 enddo
 call dsvrgn((6*nmat),freq1,freq1) ! sort the freq1 array to get cumul. percentiles.
 do i=1,(6*nmat)
   freq2(i,1)=freq1(i)
   freq2(i,2)=real(i)/6000
   write(9,5) (freq2(i,j),j=1,2)
   5 format (2F10.3)
 enddo
 do i=1,20 ! write tallies to output file for histograms.
   write(12,6) (array1(i,j),j=1,6)
   6 format (6I12)
 enddo
 do i=1,50
   write(17,8) (eigarray1(i,j),j=1,o)
 enddo
 close (1)
 close (4)
```

```
  close (9)
  close (12)
  close (17)
endif

!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~!
! Method 2: Kim's MCMC Method       !
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~!

if (m2==1) then
  open(unit=2,file=mxs_m2,position="rewind",iostat=ios)  ! matrix file, method 2
  open(unit=7,file=eig_m2,position="rewind",iostat=ios)  ! eigenvalues for method 2
  open(unit=10,file=frq_m2,position="rewind",iostat=ios) ! corr freqs for method 2
  open(unit=13,file=hst_m2,position="rewind",iostat=ios) ! histogram for method 2
  open(unit=16,file=efq_m2,position="rewind",iostat=ios) ! eigenvalue hist, method 2
  select case(o)
    case(4)
      jmpsize=.48
      itr=200000
    case(5)
      jmpsize=.4
      itr=500000
    case(6)
      jmpsize=.34
      itr=500000
    case(7)
      jmpsize=.3
      itr=1000000
    case(8)
      jmpsize=.26
      itr=1000000
    case(9)
      jmpsize=.23
      itr=500000
    case(10)
      jmpsize=.21
      itr=5000000
    case(11)
      jmpsize=.19
      itr=8000000
    case(12)
      jmpsize=.175
      itr=30000000
  end select
  thin=itr/nmat
  nr=int(o*(o-1)/2)
  do i=1,nr
    xcand(i)=.5
    xtmp(i)=0
  enddo
  t=0
  do
    t=t+1
    if (t==itr+1) exit
    call drnnoa(nr,tmp)
    call drnun(1,xtemp)
    do i=1,nr
      xtmp(i)=((xtemp(1))**(1/nr))*tmp(i)/sqrt(dot(tmp,tmp,nr))
    enddo
    do i=1,nr
      ycand(i)=xcand(i)+jmpsize*xtmp(i)
    enddo
    yflag=0
    do i=1,nr
      if (ycand(i)<0.or.ycand(i)>1) yflag=1 ! all values must be {0..1}
    enddo
    if (yflag==1) reject=reject+1
```

```
    if (yflag/=1) then
      i=1 ! put contents of y in l.t. of r (columnwise), check eigenvalues.
      do k=1,o-1
        do j=k+1,o
          r(j,k)=ycand(i)
          r(k,j)=ycand(i)
          i=i+1
        enddo
      enddo
      do k=1,o
        r(k,k)=1
      enddo
      CALL EVLRG(o,r,o,eigens) ! see if positive definite.
      numneg=0
      do j=1,o
        eig=real(eigens(j))
        if (eig<=0) numneg=numneg+1
      enddo
      if (numneg==0) then
        xcand=ycand ! if it checks out, retain for next loop.
      else
        reject=reject+1
      endif
    endif
    i=1 ! put contents of x in l.t. of r (columnwise), write eigenvalues.
    do k=1,o-1
      do j=k+1,o
        r(j,k)=xcand(i)
        r(k,j)=xcand(i)
        i=i+1
      enddo
    enddo
    do k=1,o
      r(k,k)=1
    enddo
    CALL EVLRG(o,r,o,eigens) ! see if positive definite.
    xflag=0
    do j=1,o
      eig=real(eigens(j))
      if (eig<=0) xflag=1
    enddo
    if (mod(real(t),thin)==0.and.xflag/=1) then ! if t is a mult. of thin, record.
      clocktime=CLOCK()
      write (6,1) 'Just did method 2, matrix ',int(t/thin),o,' x ',o,clocktime
      write (7,4) (real(eigens(j)),j=1,o) ! output eigenvalues for plots later.
      do j=1,o
if (real(eigens(j))>=0.0.and.real(eigens(j))<0.1) eigarray2(1,j)=eigarray2(1,j)+1
if (real(eigens(j))>=0.1.and.real(eigens(j))<0.2) eigarray2(2,j)=eigarray2(2,j)+1
if (real(eigens(j))>=0.2.and.real(eigens(j))<0.3) eigarray2(3,j)=eigarray2(3,j)+1
if (real(eigens(j))>=0.3.and.real(eigens(j))<0.4) eigarray2(4,j)=eigarray2(4,j)+1
if (real(eigens(j))>=0.4.and.real(eigens(j))<0.5) eigarray2(5,j)=eigarray2(5,j)+1
if (real(eigens(j))>=0.5.and.real(eigens(j))<0.6) eigarray2(6,j)=eigarray2(6,j)+1
if (real(eigens(j))>=0.6.and.real(eigens(j))<0.7) eigarray2(7,j)=eigarray2(7,j)+1
if (real(eigens(j))>=0.7.and.real(eigens(j))<0.8) eigarray2(8,j)=eigarray2(8,j)+1
if (real(eigens(j))>=0.8.and.real(eigens(j))<0.9) eigarray2(9,j)=eigarray2(9,j)+1
if (real(eigens(j))>=0.9.and.real(eigens(j))<1.0) eigarray2(10,j)=eigarray2(10,j)+1
if (real(eigens(j))>=1.0.and.real(eigens(j))<1.1) eigarray2(11,j)=eigarray2(11,j)+1
if (real(eigens(j))>=1.1.and.real(eigens(j))<1.2) eigarray2(12,j)=eigarray2(12,j)+1
if (real(eigens(j))>=1.2.and.real(eigens(j))<1.3) eigarray2(13,j)=eigarray2(13,j)+1
if (real(eigens(j))>=1.3.and.real(eigens(j))<1.4) eigarray2(14,j)=eigarray2(14,j)+1
if (real(eigens(j))>=1.4.and.real(eigens(j))<1.5) eigarray2(15,j)=eigarray2(15,j)+1
if (real(eigens(j))>=1.5.and.real(eigens(j))<1.6) eigarray2(16,j)=eigarray2(16,j)+1
if (real(eigens(j))>=1.6.and.real(eigens(j))<1.7) eigarray2(17,j)=eigarray2(17,j)+1
if (real(eigens(j))>=1.7.and.real(eigens(j))<1.8) eigarray2(18,j)=eigarray2(18,j)+1
if (real(eigens(j))>=1.8.and.real(eigens(j))<1.9) eigarray2(19,j)=eigarray2(19,j)+1
if (real(eigens(j))>=1.9.and.real(eigens(j))<2.0) eigarray2(20,j)=eigarray2(20,j)+1
if (real(eigens(j))>=2.0.and.real(eigens(j))<2.1) eigarray2(21,j)=eigarray2(21,j)+1
```

```
if (real(eigens(j))>=2.1.and.real(eigens(j))<2.2) eigarray2(22,j)=eigarray2(22,j)+1
if (real(eigens(j))>=2.2.and.real(eigens(j))<2.3) eigarray2(23,j)=eigarray2(23,j)+1
if (real(eigens(j))>=2.3.and.real(eigens(j))<2.4) eigarray2(24,j)=eigarray2(24,j)+1
if (real(eigens(j))>=2.4.and.real(eigens(j))<2.5) eigarray2(25,j)=eigarray2(25,j)+1
if (real(eigens(j))>=2.5.and.real(eigens(j))<2.6) eigarray2(26,j)=eigarray2(26,j)+1
if (real(eigens(j))>=2.6.and.real(eigens(j))<2.7) eigarray2(27,j)=eigarray2(27,j)+1
if (real(eigens(j))>=2.7.and.real(eigens(j))<2.8) eigarray2(28,j)=eigarray2(28,j)+1
if (real(eigens(j))>=2.8.and.real(eigens(j))<2.9) eigarray2(29,j)=eigarray2(29,j)+1
if (real(eigens(j))>=2.9.and.real(eigens(j))<3.0) eigarray2(30,j)=eigarray2(30,j)+1
if (real(eigens(j))>=3.0.and.real(eigens(j))<3.1) eigarray2(31,j)=eigarray2(31,j)+1
if (real(eigens(j))>=3.1.and.real(eigens(j))<3.2) eigarray2(32,j)=eigarray2(32,j)+1
if (real(eigens(j))>=3.2.and.real(eigens(j))<3.3) eigarray2(33,j)=eigarray2(33,j)+1
if (real(eigens(j))>=3.3.and.real(eigens(j))<3.4) eigarray2(34,j)=eigarray2(34,j)+1
if (real(eigens(j))>=3.4.and.real(eigens(j))<3.5) eigarray2(35,j)=eigarray2(35,j)+1
if (real(eigens(j))>=3.5.and.real(eigens(j))<3.6) eigarray2(36,j)=eigarray2(36,j)+1
if (real(eigens(j))>=3.6.and.real(eigens(j))<3.7) eigarray2(37,j)=eigarray2(37,j)+1
if (real(eigens(j))>=3.7.and.real(eigens(j))<3.8) eigarray2(38,j)=eigarray2(38,j)+1
if (real(eigens(j))>=3.8.and.real(eigens(j))<3.9) eigarray2(39,j)=eigarray2(39,j)+1
if (real(eigens(j))>=3.9.and.real(eigens(j))<4.0) eigarray2(40,j)=eigarray2(40,j)+1
if (real(eigens(j))>=4.0.and.real(eigens(j))<4.1) eigarray2(41,j)=eigarray2(41,j)+1
if (real(eigens(j))>=4.1.and.real(eigens(j))<4.2) eigarray2(42,j)=eigarray2(42,j)+1
if (real(eigens(j))>=4.2.and.real(eigens(j))<4.3) eigarray2(43,j)=eigarray2(43,j)+1
if (real(eigens(j))>=4.3.and.real(eigens(j))<4.4) eigarray2(44,j)=eigarray2(44,j)+1
if (real(eigens(j))>=4.4.and.real(eigens(j))<4.5) eigarray2(45,j)=eigarray2(45,j)+1
if (real(eigens(j))>=4.5.and.real(eigens(j))<4.6) eigarray2(46,j)=eigarray2(46,j)+1
if (real(eigens(j))>=4.6.and.real(eigens(j))<4.7) eigarray2(47,j)=eigarray2(47,j)+1
if (real(eigens(j))>=4.7.and.real(eigens(j))<4.8) eigarray2(48,j)=eigarray2(48,j)+1
if (real(eigens(j))>=4.8.and.real(eigens(j))<4.9) eigarray2(49,j)=eigarray2(49,j)+1
if (real(eigens(j))>=4.9.and.real(eigens(j))<=5.0) eigarray2(50,j)=eigarray2(50,j)+1
     enddo
     vec(1)=r(2,1) ! tally correlations in certain ranges for histograms later.
     vec(2)=r(3,1)
     vec(3)=r(4,1)
     vec(4)=r(o,1)
     vec(5)=r(o,2)
     vec(6)=r(o,3)
     do j=1,6
       freq1(j-6+(i*6))=vec(j)
       x=vec(j)
       if (x>=0.00.and.x<0.05) array2(1,j)=array2(1,j)+1
       if (x>=0.05.and.x<0.10) array2(2,j)=array2(2,j)+1
       if (x>=0.10.and.x<0.15) array2(3,j)=array2(3,j)+1
       if (x>=0.15.and.x<0.20) array2(4,j)=array2(4,j)+1
       if (x>=0.20.and.x<0.25) array2(5,j)=array2(5,j)+1
       if (x>=0.25.and.x<0.30) array2(6,j)=array2(6,j)+1
       if (x>=0.30.and.x<0.35) array2(7,j)=array2(7,j)+1
       if (x>=0.35.and.x<0.40) array2(8,j)=array2(8,j)+1
       if (x>=0.40.and.x<0.45) array2(9,j)=array2(9,j)+1
       if (x>=0.45.and.x<0.50) array2(10,j)=array2(10,j)+1
       if (x>=0.50.and.x<0.55) array2(11,j)=array2(11,j)+1
       if (x>=0.55.and.x<0.60) array2(12,j)=array2(12,j)+1
       if (x>=0.60.and.x<0.65) array2(13,j)=array2(13,j)+1
       if (x>=0.65.and.x<0.70) array2(14,j)=array2(14,j)+1
       if (x>=0.70.and.x<0.75) array2(15,j)=array2(15,j)+1
       if (x>=0.75.and.x<0.80) array2(16,j)=array2(16,j)+1
       if (x>=0.80.and.x<0.85) array2(17,j)=array2(17,j)+1
       if (x>=0.85.and.x<0.90) array2(18,j)=array2(18,j)+1
       if (x>=0.90.and.x<0.95) array2(19,j)=array2(19,j)+1
       if (x>=0.95.and.x<=1.00) array2(20,j)=array2(20,j)+1
     enddo
     do j=1,o ! write matrix to output to file.
       write (2,2) (r(j,k),k=1,o)
     enddo
     write (2,3)
   endif
 enddo
 call dsvrgn(6000,freq1,freq1) ! sort the freq1 array to get cumulative percentiles.
```

```
      do i=1,6000
        freq2(i,1)=freq1(i)
        freq2(i,2)=real(i)/6000
        write(10,5) (freq2(i,j),j=1,2)
      enddo
      do i=1,20 ! write tallies to output file for histograms.
        write(13,6) (array2(i,j),j=1,6)
      enddo
      do i=1,50
        write(16,8) (eigarray2(i,j),j=1,o)
        8 format (7I12)
      enddo
      close (2)
      close (7)
      close (10)
      close (13)
      close (16)
    endif
!   write (6,10) 1.-(real(reject)/itr)
!   10 format (1X,F12.3)

    !~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~!
    ! Method 3: Davies & Higham (2000)  !
    !~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~!

    if (m3==1) then
      open(unit=3,file=mxs_m3,position="rewind",iostat=ios)  ! matrix file, method 3
      open(unit=8,file=eig_m3,position="rewind",iostat=ios)  ! eigenvalues for method 3
      open(unit=11,file=frq_m3,position="rewind",iostat=ios) ! corr freqs for method 3
      open(unit=14,file=hst_m3,position="rewind",iostat=ios) ! histogram for method 3
      open(unit=18,file=efq_m3,position="rewind",iostat=ios) ! eigenvalue hist, method 3
      !Generation Parameters
      vecmeth=1   ! 1 = Gram-Schmidt orthogonalized vectors;
                  ! 2 = Haar-distributed eigenvectors
      eps=.00001  ! tolerance epsilon
      i=0
      outer: do
        i=i+1
        if (i==nmat+1) exit
        sumx=0 ! generate random eigenvalues as o-scaled uniform numbers.
        do j=1,o
          call drnun(1,randii)
          univec(j)=randii(1)
          sumx=sumx+univec(j)
        enddo
        do j=1,o
          univec(j)=(univec(j)/sumx)*o
        enddo
        sumx=0
        minx=0
        do j=1,o
          sumx=sumx+univec(j)
          if (univec(j)<minx) minx=univec(j)
          do k=1,o
            A(j,k)=0
          enddo
          A(j,j)=univec(j)
        enddo
        if ((abs(sumx-o)/o)>(100*eps).or.minx<0) then
          write (6,7) 'Elements of univec must be nonnegative and sum to ',o
          7 format (1X,A45,I4)
          pause
          stop
        endif
        if (vecmeth==1) then ! calculate Q, the orthogonal matrix of initial eigenvectors
          do j=1,o ! Gram-Schmidt orthogonalization
            do k=1,o
```

```
      call drnun(1,qtemp)
      u(j,k)=qtemp(1)
      d(j,k)=0
    enddo
  enddo
  do j=1,o   ! use Gram-Schmidt orth. to yield orthonormal equivalent of u (Q).
    w(j)=u(j,1) ! first, norm. all columns of u, place in w. Norms are in norm.
  enddo         ! The first column of Q is simply the normed first vector.
  norm=sqrt(dot(w,w,o))
  do j=1,o
    Q(j,1)=w(j)/norm
  enddo
  do k=2,o ! perform G-S orthogonalization on the other columns.
    do m=1,o
      Q(m,k)=u(m,k)
    enddo
    do j=1,k-1
      do m=1,o ! compute projection of column k.
        tvec1(m)=u(m,k)
        tvec2(m)=Q(m,j)
      enddo
      vecdot=dot(tvec1,tvec2,o)
      do m=1,o
        Q(m,k)=Q(m,k)-(vecdot*tvec2(m))
      enddo
      do m=1,o
        w(m)=Q(m,k)
      enddo
      norm=sqrt(dot(w,w,o))
    enddo
    do m=1,o
      Q(m,k)=w(m)/norm
    enddo
  enddo
else
  do j=1,o ! Sampling from Haar Distribution.
    zeros(j)=0
    do k=1,o
      Q(j,k)=0
      Q(j,k)=1
    enddo
  enddo
  do j=o-1,1,-1
    allocate(xvec(o-j+1),xvect(1,o-j+1),Q2(o-j+1,o),xvecyy(o-j+1,o))
    do k=j,o
      do m=1,o
        Q2(k-j+1,m)=Q(k,m)
      enddo
    enddo
    do k=1,o-j+1
      call drnnoa(1,randii)
      xvec(k)=randii(1)
    enddo
    ss=sqrt(dot(xvec,xvec,o-j+1))
    msign=1
    if (ss<0) msign=-1
    ss=msign*ss
    zeros(j)=-msign
    xvec(1)=xvec(1)+ss
    betax=ss*xvec(1)
    do k=1,o-j+1
      xvect(1,k)=xvec(k)
    enddo
    yy=matmul(xvect,Q2)
    do k=1,o
      yy(1,k)=yy(1,k)/betax
    enddo
```

```
    do k=1,o-j+1
      do m=1,o
        xvecyy(k,m)=xvec(k)*yy(1,m)
      enddo
    enddo
    Q2=Q2-xvecyy
    do k=j,o
      do m=1,o
        Q(k,m)=Q2(k-j+1,m)
      enddo
    enddo
    deallocate(xvec,xvect,Q2,xvecyy)
  enddo
  do j=1,o-1
    do k=1,o
      Q(j,k)=zeros(j)*Q(j,k)
    enddo
  enddo
  call drnnoa(1,randii)
  msign=1
  if (randii(1)<0) msign=-1
  do j=1,o
    Q(o,j)=Q(o,j)*msign
  enddo
endif
A=matmul(Q,matmul(A,transpose(Q))) ! find QAQ'.
do j=1,o ! aa=diag(A)
  aa(j)=A(j,j)
enddo
do j=1,o ! find elements of aa < 1 and elements of aa > 1.
  y(j)=0
  z(j)=0
enddo
j=0
k=0
do m=1,o
  if (aa(m)<1) then
    j=j+1
    y(j)=m
  endif
  if (aa(m)>1) then
    k=k+1
    z(k)=m
  endif
enddo
it=1
do
  nynon0=0
  nznon0=0
  do m=1,o
    if (y(m)/=0) nynon0=nynon0+1
    if (z(m)/=0) nznon0=nznon0+1
  enddo
  if (nynon0==0.or.nznon0==0) exit
  call drnun(1,randii)
  ii=ceiling(randii(1)*nynon0)
  if (ii==0) ii=1
  ii=y(ii)
  call drnun(1,randjj)
  jj=ceiling(randjj(1)*nznon0)
  if (jj==0) jj=1
  jj=z(jj)
  kk=0
  if (ii>jj) then
    kk=jj
    jj=ii
    ii=kk
```

```
  endif
  alpha=sqrt((A(ii,jj)**2)-((aa(ii)-1)*(aa(jj)-1)))
  msign=1
  if (A(ii,jj)<0) msign=-1
  t1=(A(ii,jj)+(msign*alpha))/(aa(jj)-1)
  t2=(aa(ii)-1)/((aa(jj)-1)*t1)
  call drnun(1,randkk)
  if (randkk(1)<=0.5) then
    c=1/sqrt(1+(t1**2))
    s=t1*c
  else
    c=1/sqrt(1+(t2**2))
    s=t2*c
  endif
  cs(1,1)=c
  cs(1,2)=s
  cs(2,1)=-s
  cs(2,2)=c
  do m=1,o
    do j=1,2
      B1(m,j)=0
      B2(j,m)=0
    enddo
  enddo
  do k=1,o ! multiply the ii,jj columns of A by the cs matrix.
    B1(k,1)=A(k,ii)
    B1(k,2)=A(k,jj)
  enddo
  B1b=matmul(B1,cs)
  do k=1,o
    A(k,ii)=B1b(k,1)
    A(k,jj)=B1b(k,2)
  enddo
  do k=1,o ! multiply the ii,jj rows of the new A by the cs matrix.
    B2(1,k)=A(ii,k)
    B2(2,k)=A(jj,k)
  enddo
  B2b=matmul(transpose(cs),B2)
  do k=1,o
    A(ii,k)=B2b(1,k)
    A(jj,k)=B2b(2,k)
  enddo
  A(ii,ii)=1 ! ensure (ii,ii) element is exactly 1.
  do m=1,o   ! aa=diag(A)
    aa(m)=A(m,m)
  enddo
  do m=1,o ! find elements of aa < 1 and elements of aa > 1.
    y(m)=0
    z(m)=0
  enddo
  j=0
  k=0
  do m=1,o
    if (aa(m)<1) then
      j=j+1
      y(j)=m
    endif
    if (aa(m)>1) then
      k=k+1
      z(k)=m
    endif
  enddo
  it=it+1
enddo
do m=1,o ! set last diagonal element to 1.
  A(m,m)=1
enddo
```

```
   A=A+transpose(A) ! force symmetry.
   do m=1,o
     do j=1,o
       A(m,j)=A(m,j)/2
     enddo
   enddo
   do m=1,o
     do j=1,o
       r(m,j)=A(m,j)
     enddo
   enddo
   CALL EVLRG (o,r,o,eigens) ! see if positive definite; if not, cycle.
   numneg=0
   do j=1,o
     eig=real(eigens(j))
     if (eig<=0) then
       numneg=numneg+1
     endif
   enddo
   if (numneg>0) then
     i=i-1
     cycle outer
   endif
   clocktime=CLOCK()
   write (6,1) 'Just did method 3, matrix ',i,o,' x ',o,clocktime
   write (8,4) (real(eigens(j)),j=1,o) ! output eigenvalues for plots later.
   do j=1,o
if (real(eigens(j))>=0.0.and.real(eigens(j))<0.1) eigarray3(1,j)=eigarray3(1,j)+1
if (real(eigens(j))>=0.1.and.real(eigens(j))<0.2) eigarray3(2,j)=eigarray3(2,j)+1
if (real(eigens(j))>=0.2.and.real(eigens(j))<0.3) eigarray3(3,j)=eigarray3(3,j)+1
if (real(eigens(j))>=0.3.and.real(eigens(j))<0.4) eigarray3(4,j)=eigarray3(4,j)+1
if (real(eigens(j))>=0.4.and.real(eigens(j))<0.5) eigarray3(5,j)=eigarray3(5,j)+1
if (real(eigens(j))>=0.5.and.real(eigens(j))<0.6) eigarray3(6,j)=eigarray3(6,j)+1
if (real(eigens(j))>=0.6.and.real(eigens(j))<0.7) eigarray3(7,j)=eigarray3(7,j)+1
if (real(eigens(j))>=0.7.and.real(eigens(j))<0.8) eigarray3(8,j)=eigarray3(8,j)+1
if (real(eigens(j))>=0.8.and.real(eigens(j))<0.9) eigarray3(9,j)=eigarray3(9,j)+1
if (real(eigens(j))>=0.9.and.real(eigens(j))<1.0) eigarray3(10,j)=eigarray3(10,j)+1
if (real(eigens(j))>=1.0.and.real(eigens(j))<1.1) eigarray3(11,j)=eigarray3(11,j)+1
if (real(eigens(j))>=1.1.and.real(eigens(j))<1.2) eigarray3(12,j)=eigarray3(12,j)+1
if (real(eigens(j))>=1.2.and.real(eigens(j))<1.3) eigarray3(13,j)=eigarray3(13,j)+1
if (real(eigens(j))>=1.3.and.real(eigens(j))<1.4) eigarray3(14,j)=eigarray3(14,j)+1
if (real(eigens(j))>=1.4.and.real(eigens(j))<1.5) eigarray3(15,j)=eigarray3(15,j)+1
if (real(eigens(j))>=1.5.and.real(eigens(j))<1.6) eigarray3(16,j)=eigarray3(16,j)+1
if (real(eigens(j))>=1.6.and.real(eigens(j))<1.7) eigarray3(17,j)=eigarray3(17,j)+1
if (real(eigens(j))>=1.7.and.real(eigens(j))<1.8) eigarray3(18,j)=eigarray3(18,j)+1
if (real(eigens(j))>=1.8.and.real(eigens(j))<1.9) eigarray3(19,j)=eigarray3(19,j)+1
if (real(eigens(j))>=1.9.and.real(eigens(j))<2.0) eigarray3(20,j)=eigarray3(20,j)+1
if (real(eigens(j))>=2.0.and.real(eigens(j))<2.1) eigarray3(21,j)=eigarray3(21,j)+1
if (real(eigens(j))>=2.1.and.real(eigens(j))<2.2) eigarray3(22,j)=eigarray3(22,j)+1
if (real(eigens(j))>=2.2.and.real(eigens(j))<2.3) eigarray3(23,j)=eigarray3(23,j)+1
if (real(eigens(j))>=2.3.and.real(eigens(j))<2.4) eigarray3(24,j)=eigarray3(24,j)+1
if (real(eigens(j))>=2.4.and.real(eigens(j))<2.5) eigarray3(25,j)=eigarray3(25,j)+1
if (real(eigens(j))>=2.5.and.real(eigens(j))<2.6) eigarray3(26,j)=eigarray3(26,j)+1
if (real(eigens(j))>=2.6.and.real(eigens(j))<2.7) eigarray3(27,j)=eigarray3(27,j)+1
if (real(eigens(j))>=2.7.and.real(eigens(j))<2.8) eigarray3(28,j)=eigarray3(28,j)+1
if (real(eigens(j))>=2.8.and.real(eigens(j))<2.9) eigarray3(29,j)=eigarray3(29,j)+1
if (real(eigens(j))>=2.9.and.real(eigens(j))<3.0) eigarray3(30,j)=eigarray3(30,j)+1
if (real(eigens(j))>=3.0.and.real(eigens(j))<3.1) eigarray3(31,j)=eigarray3(31,j)+1
if (real(eigens(j))>=3.1.and.real(eigens(j))<3.2) eigarray3(32,j)=eigarray3(32,j)+1
if (real(eigens(j))>=3.2.and.real(eigens(j))<3.3) eigarray3(33,j)=eigarray3(33,j)+1
if (real(eigens(j))>=3.3.and.real(eigens(j))<3.4) eigarray3(34,j)=eigarray3(34,j)+1
if (real(eigens(j))>=3.4.and.real(eigens(j))<3.5) eigarray3(35,j)=eigarray3(35,j)+1
if (real(eigens(j))>=3.5.and.real(eigens(j))<3.6) eigarray3(36,j)=eigarray3(36,j)+1
if (real(eigens(j))>=3.6.and.real(eigens(j))<3.7) eigarray3(37,j)=eigarray3(37,j)+1
if (real(eigens(j))>=3.7.and.real(eigens(j))<3.8) eigarray3(38,j)=eigarray3(38,j)+1
if (real(eigens(j))>=3.8.and.real(eigens(j))<3.9) eigarray3(39,j)=eigarray3(39,j)+1
if (real(eigens(j))>=3.9.and.real(eigens(j))<4.0) eigarray3(40,j)=eigarray3(40,j)+1
```

```
   if (real(eigens(j))>=4.0.and.real(eigens(j))<4.1) eigarray3(41,j)=eigarray3(41,j)+1
   if (real(eigens(j))>=4.1.and.real(eigens(j))<4.2) eigarray3(42,j)=eigarray3(42,j)+1
   if (real(eigens(j))>=4.2.and.real(eigens(j))<4.3) eigarray3(43,j)=eigarray3(43,j)+1
   if (real(eigens(j))>=4.3.and.real(eigens(j))<4.4) eigarray3(44,j)=eigarray3(44,j)+1
   if (real(eigens(j))>=4.4.and.real(eigens(j))<4.5) eigarray3(45,j)=eigarray3(45,j)+1
   if (real(eigens(j))>=4.5.and.real(eigens(j))<4.6) eigarray3(46,j)=eigarray3(46,j)+1
   if (real(eigens(j))>=4.6.and.real(eigens(j))<4.7) eigarray3(47,j)=eigarray3(47,j)+1
   if (real(eigens(j))>=4.7.and.real(eigens(j))<4.8) eigarray3(48,j)=eigarray3(48,j)+1
   if (real(eigens(j))>=4.8.and.real(eigens(j))<4.9) eigarray3(49,j)=eigarray3(49,j)+1
   if (real(eigens(j))>=4.9.and.real(eigens(j))<=5.0) eigarray3(50,j)=eigarray3(50,j)+1
      enddo
      vec(1)=A(2,1)
      vec(2)=A(3,1)
      vec(3)=A(4,1)
      vec(4)=A(o,1)
      vec(5)=A(o,2)
      vec(6)=A(o,3)
      do j=1,6
        xx=vec(j)
        if (xx>=0.00.and.xx<0.05) array3(1,j)=array3(1,j)+1
        if (xx>=0.05.and.xx<0.10) array3(2,j)=array3(2,j)+1
        if (xx>=0.10.and.xx<0.15) array3(3,j)=array3(3,j)+1
        if (xx>=0.15.and.xx<0.20) array3(4,j)=array3(4,j)+1
        if (xx>=0.20.and.xx<0.25) array3(5,j)=array3(5,j)+1
        if (xx>=0.25.and.xx<0.30) array3(6,j)=array3(6,j)+1
        if (xx>=0.30.and.xx<0.35) array3(7,j)=array3(7,j)+1
        if (xx>=0.35.and.xx<0.40) array3(8,j)=array3(8,j)+1
        if (xx>=0.40.and.xx<0.45) array3(9,j)=array3(9,j)+1
        if (xx>=0.45.and.xx<0.50) array3(10,j)=array3(10,j)+1
        if (xx>=0.50.and.xx<0.55) array3(11,j)=array3(11,j)+1
        if (xx>=0.55.and.xx<0.60) array3(12,j)=array3(12,j)+1
        if (xx>=0.60.and.xx<0.65) array3(13,j)=array3(13,j)+1
        if (xx>=0.65.and.xx<0.70) array3(14,j)=array3(14,j)+1
        if (xx>=0.70.and.xx<0.75) array3(15,j)=array3(15,j)+1
        if (xx>=0.75.and.xx<0.80) array3(16,j)=array3(16,j)+1
        if (xx>=0.80.and.xx<0.85) array3(17,j)=array3(17,j)+1
        if (xx>=0.85.and.xx<0.90) array3(18,j)=array3(18,j)+1
        if (xx>=0.90.and.xx<0.95) array3(19,j)=array3(19,j)+1
        if (xx>=0.95.and.xx<=1.00) array3(20,j)=array3(20,j)+1
      enddo
      do j=1,o ! write matrix to output to file.
        write (3,2) (r(j,k),k=1,o)
      enddo
      write (3,3)
    enddo outer
    call dsvrgn(6000,freq1,freq1) ! sort the freq1 array to get cumulative percentiles.
    do i=1,6000
      freq2(i,1)=freq1(i)
      freq2(i,2)=real(i)/6000
      write(11,5) (freq2(i,j),j=1,2)
    enddo
    do m=1,20 ! write tallies to output file for histograms.
      write(14,6) (array3(m,j),j=1,6)
    enddo
    do i=1,50
      write(18,8) (eigarray3(i,j),j=1,o)
    enddo
    close (3)
    close (8)
    close (11)
    close (14)
    close (18)
  endif
  deallocate (r,eigens,freq1,freq2,A,Q,aa,univec,y,z,B1,B2,tvec1,tvec2)
  deallocate (cs,B1b,B2b,d,w,u,zeros,yy,xcand,ycand,tmp,xtmp)
  deallocate (eigarray1,eigarray2,eigarray3)
enddo
```

```
end program randomr5
function dot(DX,DY,N)
  use msimsl
  implicit none
  real*8                 :: dot
  real*8, dimension(:)   :: DX,DY
  integer                :: N,INCX=1,INCY=1
  dot=DDOT(N,DX,INCX,DY,INCY)
end function dot
```

APPENDIX B

FORTRAN CODE FOR CONSTRUCTING LISREL SYNTAX

```
! The purpose of RandomR6_lisrel is to create LISREL input files for large batch runs.

program randomr6_lisrel
  implicit none

  integer               :: i,ios,j,k,mo,mx,nm,dim
  character(24)         :: mxfile
  character(45)         :: garbage
  character(49)         :: mod01,mod02,mod03,mod04,mod05,mod06,mod11,mod12,mod13,mod14
  character(49)         :: mod15,mod16,mod17,mod22,mod23,mod31,mod32,mod33,mod34,mod35
  character(49)         :: mod36,mod37,mod38,mod45,mod46,mod49,mod50,mod52,mod53
  character(51)         :: hdr01file,hdr02file,hdr03file,hdr04file,hdr05file,hdr06file
  character(52)         :: hdr11file,hdr12file,hdr13file,hdr14file,hdr15file,hdr16file
  character(52)         :: hdr17file,hdr22file,hdr23file,hdr31file,hdr32file,hdr33file
  character(52)         :: hdr34file,hdr35file,hdr36file,hdr37file,hdr38file,hdr45file
  character(52)         :: hdr46file,hdr49file,hdr50file,hdr52file,hdr53file
  character(100), allocatable, dimension(:) :: header
  real*8, allocatable, dimension(:,:)       :: matrix

  ! parameters to change each time:
  nm=5000 ! number of matrices to analyze
  dim=9 ! dimension of input matrices
  mxfile="c:\randomr\mxs_m2_05.out" ! file containing matrices
  ! LISREL header file labels (comment out undesired headers):
  hdr01file="c:\dissfiles\header1.tmp"  ! header, model 1, 0-factor model, Fig. 3.4-3.7
  hdr02file="c:\dissfiles\header2.tmp"  ! header, model 2, 1-factor model, Fig. 3.4-3.7
  hdr03file="c:\dissfiles\header3.tmp"  ! header, model 3, 2-factor model, Fig. 3.4-3.7
  hdr04file="c:\dissfiles\header4.tmp"  ! header, model 4, 3-factor model, Fig. 3.4-3.7
  hdr05file="c:\dissfiles\header5.tmp"  ! header, model 5, extra loading, Fig. 3.16
  hdr06file="c:\dissfiles\header6.tmp"  ! header, model 6, corr. factors, Fig. 3.16
  hdr11file="c:\dissfiles\header11.tmp" ! header, model 11, omit phi(1,2), Fig. 3.18
  hdr12file="c:\dissfiles\header12.tmp" ! header, model 12, omit phi(2,3), Fig. 3.18
  hdr13file="c:\dissfiles\header13.tmp" ! header, model 13, omit phi(3,4), Fig. 3.18
  hdr14file="c:\dissfiles\header14.tmp" ! header, model 14, no restrict., Fig. 3.26-3.27
  hdr15file="c:\dissfiles\header15.tmp" ! header, model 15, range restr., Fig. 3.26-3.27
  hdr16file="c:\dissfiles\header16.tmp" ! header, model 16, fixed loading, Fig. 3.13
  hdr17file="c:\dissfiles\header17.tmp" ! header, model 17, equal. constraint, Fig. 3.13
  hdr22file="c:\dissfiles\header22.tmp" ! header, model 22, equivalent model 1, Fig. 3.3
  hdr23file="c:\dissfiles\header23.tmp" ! header, model 23, equivalent model 2, Fig. 3.3
  hdr31file="c:\dissfiles\header31.tmp" ! header, model 31, no constraint, Fig. 3.11
  hdr32file="c:\dissfiles\header32.tmp" ! header, model 32, == constraint, Fig. 3.11
  hdr33file="c:\dissfiles\header33.tmp" ! header, model 33, equal loadings, Fig. 3.24
  hdr34file="c:\dissfiles\header34.tmp" ! header, model 34, no correlation, Fig. 3.24
  hdr35file="c:\dissfiles\header35.tmp" ! header, model 35, simplex model, Fig. 3.21
  hdr36file="c:\dissfiles\header36.tmp" ! header, model 36, factor model, Fig. 3.21
  hdr37file="c:\dissfiles\header37.tmp" ! header, model 37, CCA model, Fig. 4.15
  hdr38file="c:\dissfiles\header38.tmp" ! header, model 38, CDP model, Fig. 4.15
  hdr45file="c:\dissfiles\header45.tmp" ! header, model 45, Newsom H, Fig. 4.11
  hdr46file="c:\dissfiles\header46.tmp" ! header, model 46, Newsom 1, Fig. 4.11
```

```
hdr49file="c:\dissfiles\header49.tmp" ! header, model 49, Newsom 4, Fig. 4.11
hdr50file="c:\dissfiles\header50.tmp" ! header, model 50, Beier, Fig. 4.2
hdr52file="c:\dissfiles\header52.tmp" ! header, model 52, Coffman, Fig. 3.30, 4.4
hdr53file="c:\dissfiles\header53.tmp" ! header, model 53, contrived 5-MV, Fig. 3.30
! LISREL script filenames (comment out undesired headers):
mod01="c:\msdev\projects\randomr6_lisrel\debug\mod01.pr2"
mod02="c:\msdev\projects\randomr6_lisrel\debug\mod02.pr2"
mod03="c:\msdev\projects\randomr6_lisrel\debug\mod03.pr2"
mod04="c:\msdev\projects\randomr6_lisrel\debug\mod04.pr2"
mod05="c:\msdev\projects\randomr6_lisrel\debug\mod05.pr2"
mod06="c:\msdev\projects\randomr6_lisrel\debug\mod06.pr2"
mod11="c:\msdev\projects\randomr6_lisrel\debug\mod11.pr2"
mod12="c:\msdev\projects\randomr6_lisrel\debug\mod12.pr2"
mod13="c:\msdev\projects\randomr6_lisrel\debug\mod13.pr2"
mod14="c:\msdev\projects\randomr6_lisrel\debug\mod14.pr2"
mod15="c:\msdev\projects\randomr6_lisrel\debug\mod15.pr2"
mod16="c:\msdev\projects\randomr6_lisrel\debug\mod16.pr2"
mod17="c:\msdev\projects\randomr6_lisrel\debug\mod17.pr2"
mod22="c:\msdev\projects\randomr6_lisrel\debug\mod22.pr2"
mod23="c:\msdev\projects\randomr6_lisrel\debug\mod23.pr2"
mod31="c:\msdev\projects\randomr6_lisrel\debug\mod31.pr2"
mod32="c:\msdev\projects\randomr6_lisrel\debug\mod32.pr2"
mod33="c:\msdev\projects\randomr6_lisrel\debug\mod33.pr2"
mod34="c:\msdev\projects\randomr6_lisrel\debug\mod34.pr2"
mod35="c:\msdev\projects\randomr6_lisrel\debug\mod35.pr2"
mod36="c:\msdev\projects\randomr6_lisrel\debug\mod36.pr2"
mod37="c:\msdev\projects\randomr6_lisrel\debug\mod37.pr2"
mod38="c:\msdev\projects\randomr6_lisrel\debug\mod38.pr2"
mod45="c:\msdev\projects\randomr6_lisrel\debug\mod45.pr2"
mod46="c:\msdev\projects\randomr6_lisrel\debug\mod46.pr2"
mod49="c:\msdev\projects\randomr6_lisrel\debug\mod49.pr2"
mod50="c:\msdev\projects\randomr6_lisrel\debug\mod51.pr2"
mod52="c:\msdev\projects\randomr6_lisrel\debug\mod52.pr2"
mod53="c:\msdev\projects\randomr6_lisrel\debug\mod53.pr2"
allocate (header(100),matrix(dim,dim))
open(unit=7,file=mxfile,position="rewind",iostat=ios)

model_loop: do mo=1,4 ! change for each desired set of models.
  ! Read in program template for LISREL input and open LISREL input file.
  select case (mo)
    case(1)
      open(unit=36,file=hdr01file,position="rewind",iostat=ios)
      j=0
      m01loop: do i=1,100
        read(36,10) header(i)
        10 format (A100)
        j=j+1
        if (header(i)(1:2)=="OU") exit m01loop
      enddo m01loop
    case(2)
      open(unit=37,file=hdr02file,position="rewind",iostat=ios)
      j=0
      m02loop: do i=1,100
        read(37,10) header(i)
        j=j+1
        if (header(i)(1:2)=="OU") exit m02loop
      enddo m02loop
    case(3)
      open(unit=38,file=hdr03file,position="rewind",iostat=ios)
      j=0
      m03loop: do i=1,100
        read(38,10) header(i)
        j=j+1
        if (header(i)(1:2)=="OU") exit m03loop
      enddo m03loop
    case(4)
      open(unit=39,file=hdr04file,position="rewind",iostat=ios)
```

```
  j=0
  m04loop: do i=1,100
    read(39,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m04loop
  enddo m04loop
case(5)
  open(unit=52,file=hdr05file,position="rewind",iostat=ios)
  j=0
  m05loop: do i=1,100
    read(52,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m05loop
  enddo m05loop
case(6)
  open(unit=53,file=hdr06file,position="rewind",iostat=ios)
  j=0
  m06loop: do i=1,100
    read(53,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m06loop
  enddo m06loop
case(11)
  open(unit=56,file=hdr11file,position="rewind",iostat=ios)
  j=0
  m11loop: do i=1,100
    read(56,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m11loop
  enddo m11loop
case(12)
  open(unit=57,file=hdr12file,position="rewind",iostat=ios)
  j=0
  m12loop: do i=1,100
    read(57,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m12loop
  enddo m12loop
case(13)
  open(unit=58,file=hdr13file,position="rewind",iostat=ios)
  j=0
  m13loop: do i=1,100
    read(58,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m13loop
  enddo m13loop
case(14)
  open(unit=1,file=hdr14file,position="rewind",iostat=ios)
  j=0
  m14loop: do i=1,100
    read(1,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m14loop
  enddo m14loop
case(15)
  open(unit=2,file=hdr15file,position="rewind",iostat=ios)
  j=0
  m15loop: do i=1,100
    read(2,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m15loop
  enddo m15loop
case(16)
  open(unit=3,file=hdr16file,position="rewind",iostat=ios)
  j=0
  m16loop: do i=1,100
    read(3,10) header(i)
```

```
    j=j+1
    if (header(i)(1:2)=="OU") exit m16loop
  enddo m16loop
case(17)
  open(unit=4,file=hdr17file,position="rewind",iostat=ios)
  j=0
  m17loop: do i=1,100
    read(4,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m17loop
  enddo m17loop
case(22)
  open(unit=20,file=hdr22file,position="rewind",iostat=ios)
  j=0
  m22loop: do i=1,100
    read(20,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m22loop
  enddo m22loop
case(23)
  open(unit=21,file=hdr23file,position="rewind",iostat=ios)
  j=0
  m23loop: do i=1,100
    read(21,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m23loop
  enddo m23loop
case(31)
  open(unit=44,file=hdr31file,position="rewind",iostat=ios)
  j=0
  m31loop: do i=1,100
    read(44,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m31loop
  enddo m31loop
case(32)
  open(unit=45,file=hdr32file,position="rewind",iostat=ios)
  j=0
  m32loop: do i=1,100
    read(45,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m32loop
  enddo m32loop
case(33)
  open(unit=48,file=hdr33file,position="rewind",iostat=ios)
  j=0
  m33loop: do i=1,100
    read(48,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m33loop
  enddo m33loop
case(34)
  open(unit=49,file=hdr34file,position="rewind",iostat=ios)
  j=0
  m34loop: do i=1,100
    read(49,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m34loop
  enddo m34loop
case(35)
  open(unit=62,file=hdr35file,position="rewind",iostat=ios)
  j=0
  m35loop: do i=1,100
    read(62,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m35loop
  enddo m35loop
```

```
case(36)
  open(unit=63,file=hdr36file,position="rewind",iostat=ios)
  j=0
  m36loop: do i=1,100
    read(63,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m36loop
  enddo m36loop
case(37)
  open(unit=66,file=hdr37file,position="rewind",iostat=ios)
  j=0
  m37loop: do i=1,100
    read(66,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m37loop
  enddo m37loop
case(38)
  open(unit=67,file=hdr38file,position="rewind",iostat=ios)
  j=0
  m38loop: do i=1,100
    read(67,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m38loop
  enddo m38loop
case(45)
  open(unit=82,file=hdr45file,position="rewind",iostat=ios)
  j=0
  m45loop: do i=1,100
    read(82,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m45loop
  enddo m45loop
case(46)
  open(unit=83,file=hdr46file,position="rewind",iostat=ios)
  j=0
  m46loop: do i=1,100
    read(83,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m46loop
  enddo m46loop
case(49)
  open(unit=86,file=hdr49file,position="rewind",iostat=ios)
  j=0
  m49loop: do i=1,100
    read(86,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m49loop
  enddo m49loop
case(50)
  open(unit=92,file=hdr50file,position="rewind",iostat=ios)
  j=0
  m50loop: do i=1,100
    read(92,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m50loop
  enddo m50loop
case(52)
  open(unit=96,file=hdr52file,position="rewind",iostat=ios)
  j=0
  m52loop: do i=1,100
    read(96,10) header(i)
    j=j+1
    if (header(i)(1:2)=="OU") exit m52loop
  enddo m52loop
case(53)
  open(unit=98,file=hdr53file,position="rewind",iostat=ios)
  j=0
```

```
    m53loop: do i=1,100
      read(98,10) header(i)
      j=j+1
      if (header(i)(1:2)=="OU") exit m53loop
    enddo m53loop
end select

! Open output files (LISREL scripts)
if (mo==1) then
  open(unit=40,file=mod01,position="append",iostat=ios)
  rewind (40)
elseif (mo==2) then
  open(unit=41,file=mod02,position="append",iostat=ios)
  rewind (41)
elseif (mo==3) then
  open(unit=42,file=mod03,position="append",iostat=ios)
  rewind (42)
elseif (mo==4) then
  open(unit=43,file=mod04,position="append",iostat=ios)
  rewind (43)
elseif (mo==5) then
  open(unit=54,file=mod05,position="append",iostat=ios)
  rewind (54)
elseif (mo==6) then
  open(unit=55,file=mod06,position="append",iostat=ios)
  rewind (55)
elseif (mo==11) then
  open(unit=59,file=mod11,position="append",iostat=ios)
  rewind (59)
elseif (mo==12) then
  open(unit=60,file=mod12,position="append",iostat=ios)
  rewind (60)
elseif (mo==13) then
  open(unit=61,file=mod13,position="append",iostat=ios)
  rewind (61)
elseif (mo==14) then
  open(unit=14,file=mod14,position="append",iostat=ios)
  rewind (14)
elseif (mo==15) then
  open(unit=15,file=mod15,position="append",iostat=ios)
  rewind (15)
elseif (mo==16) then
  open(unit=8,file=mod16,position="append",iostat=ios)
  rewind (8)
elseif (mo==17) then
  open(unit=9,file=mod17,position="append",iostat=ios)
  rewind (9)
elseif (mo==22) then
  open(unit=22,file=mod22,position="append",iostat=ios)
  rewind (22)
elseif (mo==23) then
  open(unit=23,file=mod23,position="append",iostat=ios)
  rewind (23)
elseif (mo==31) then
  open(unit=46,file=mod31,position="append",iostat=ios)
  rewind (46)
elseif (mo==32) then
  open(unit=47,file=mod32,position="append",iostat=ios)
  rewind (47)
elseif (mo==33) then
  open(unit=50,file=mod33,position="append",iostat=ios)
  rewind (50)
elseif (mo==34) then
  open(unit=51,file=mod34,position="append",iostat=ios)
  rewind (51)
elseif (mo==35) then
  open(unit=64,file=mod35,position="append",iostat=ios)
```

```
  rewind (64)
elseif (mo==36) then
  open(unit=65,file=mod36,position="append",iostat=ios)
  rewind (65)
elseif (mo==37) then
  open(unit=68,file=mod37,position="append",iostat=ios)
  rewind (68)
elseif (mo==38) then
  open(unit=69,file=mod38,position="append",iostat=ios)
  rewind (69)
elseif (mo==45) then
  open(unit=87,file=mod45,position="append",iostat=ios)
  rewind (87)
elseif (mo==46) then
  open(unit=88,file=mod46,position="append",iostat=ios)
  rewind (88)
elseif (mo==49) then
  open(unit=91,file=mod49,position="append",iostat=ios)
  rewind (91)
elseif (mo==50) then
  open(unit=93,file=mod50,position="append",iostat=ios)
  rewind (93)
elseif (mo==52) then
  open(unit=97,file=mod52,position="append",iostat=ios)
  rewind (97)
elseif (mo==53) then
  open(unit=99,file=mod53,position="append",iostat=ios)
  rewind (99)
endif

matrix_loop: do mx=1,nm

  ! Read next dim x dim lower triangular matrix for LISREL syntax file.
  do i=1,dim
    read (7,20) (matrix(i,k),k=1,i)
    20 format (20F10.6)
  enddo
  read (7,10) garbage

  ! Write header and matrix to LISREL syntax file.
  if (mo==1) then
    do i=1,4
      write (40,30) header(i)
      30 format (1X,A100)
    enddo
    do i=1,dim
      write (40,20) (matrix(i,k),k=1,i)
    enddo
    do i=5,j
      write (40,30) header(i)
    enddo
  elseif (mo==2) then
    do i=1,4
      write (41,30) header(i)
    enddo
    do i=1,dim
      write (41,20) (matrix(i,k),k=1,i)
    enddo
    do i=5,j
      write (41,30) header(i)
    enddo
  elseif (mo==3) then
    do i=1,4
      write (42,30) header(i)
    enddo
    do i=1,dim
      write (42,20) (matrix(i,k),k=1,i)
```

```
  enddo
  do i=5,j
    write (42,30) header(i)
  enddo
elseif (mo==4) then
  do i=1,4
    write (43,30) header(i)
  enddo
  do i=1,dim
    write (43,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (43,30) header(i)
  enddo
elseif (mo==5) then
  do i=1,4
    write (54,30) header(i)
  enddo
  do i=1,dim
    write (54,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (54,30) header(i)
  enddo
elseif (mo==6) then
  do i=1,4
    write (55,30) header(i)
  enddo
  do i=1,dim
    write (55,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (55,30) header(i)
  enddo
elseif (mo==11) then
  do i=1,4
    write (59,30) header(i)
  enddo
  do i=1,dim
    write (59,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (59,30) header(i)
  enddo
elseif (mo==12) then
  do i=1,4
    write (60,30) header(i)
  enddo
  do i=1,dim
    write (60,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (60,30) header(i)
  enddo
elseif (mo==13) then
  do i=1,4
    write (61,30) header(i)
  enddo
  do i=1,dim
    write (61,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (61,30) header(i)
  enddo
elseif (mo==14) then
  do i=1,4
    write (14,30) header(i)
```

```
  enddo
  do i=1,dim
    write (14,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (14,30) header(i)
  enddo
elseif (mo==15) then
  do i=1,4
    write (15,30) header(i)
  enddo
  do i=1,dim
    write (15,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (15,30) header(i)
  enddo
elseif (mo==16) then
  do i=1,4
    write (8,30) header(i)
  enddo
  do i=1,dim
    write (8,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (8,30) header(i)
  enddo
elseif (mo==17) then
  do i=1,4
    write (9,30) header(i)
  enddo
  do i=1,dim
    write (9,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (9,30) header(i)
  enddo
elseif (mo==22) then
  do i=1,4
    write (22,30) header(i)
  enddo
  do i=1,dim
    write (22,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (22,30) header(i)
  enddo
elseif (mo==23) then
  do i=1,4
    write (23,30) header(i)
  enddo
  do i=1,dim
    write (23,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (23,30) header(i)
  enddo
elseif (mo==31) then
  do i=1,4
    write (46,30) header(i)
  enddo
  do i=1,dim
    write (46,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (46,30) header(i)
  enddo
```

```
elseif (mo==32) then
  do i=1,4
    write (47,30) header(i)
  enddo
  do i=1,dim
    write (47,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (47,30) header(i)
  enddo
elseif (mo==33) then
  do i=1,4
    write (50,30) header(i)
  enddo
  do i=1,dim
    write (50,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (50,30) header(i)
  enddo
elseif (mo==34) then
  do i=1,4
    write (51,30) header(i)
  enddo
  do i=1,dim
    write (51,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (51,30) header(i)
  enddo
elseif (mo==35) then
  do i=1,4
    write (64,30) header(i)
  enddo
  do i=1,dim
    write (64,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (64,30) header(i)
  enddo
elseif (mo==36) then
  do i=1,4
    write (65,30) header(i)
  enddo
  do i=1,dim
    write (65,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (65,30) header(i)
  enddo
elseif (mo==37) then
  do i=1,4
    write (68,30) header(i)
  enddo
  do i=1,dim
    write (68,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (68,30) header(i)
  enddo
elseif (mo==38) then
  do i=1,4
    write (69,30) header(i)
  enddo
  do i=1,dim
    write (69,20) (matrix(i,k),k=1,i)
  enddo
```

```
  do i=5,j
    write (69,30) header(i)
  enddo
elseif (mo==45) then
  do i=1,4
    write (87,30) header(i)
  enddo
  do i=1,dim
    write (87,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (87,30) header(i)
  enddo
elseif (mo==46) then
  do i=1,4
    write (88,30) header(i)
  enddo
  do i=1,dim
    write (88,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (88,30) header(i)
  enddo
elseif (mo==49) then
  do i=1,4
    write (91,30) header(i)
  enddo
  do i=1,dim
    write (91,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (91,30) header(i)
  enddo
elseif (mo==50) then
  do i=1,4
    write (93,30) header(i)
  enddo
  do i=1,dim
    write (93,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (93,30) header(i)
  enddo
elseif (mo==52) then
  do i=1,4
    write (97,30) header(i)
  enddo
  do i=1,dim
    write (97,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (97,30) header(i)
  enddo
elseif (mo==53) then
  do i=1,4
    write (99,30) header(i)
  enddo
  do i=1,dim
    write (99,20) (matrix(i,k),k=1,i)
  enddo
  do i=5,j
    write (99,30) header(i)
  enddo
endif

write(6,50) 'Just did model ',mo,' matrix ',mx
50 format (1X,A15,I2,A8,I4)
```

```
    enddo matrix_loop
    rewind 7 ! Rewind matrix file for next method.

  enddo model_loop
  deallocate (header,matrix) ! Deallocate everything.

end program randomr6_lisrel
```

APPENDIX C

LISREL SYNTAX FILES

These LISREL syntax files were written for use with LISREL 8.53. The Fortran program in Appendix B inserts randomly generated data after the fourth line of code in each LISREL syntax file, then stacks 5000 copies of the syntax (one for each random correlation matrix) into a single large syntax file. This large syntax file was then executed from the DOS command line using the command:

```
lisrel85 mod#.pr2 mod#.out -nwin,
```

where "#" is replaced by the model's number.

```
MODEL1 - 0 FACTORS
DA NI=12 NO=1000 MA=KM
KM SY
(12F10.6)
MO NY=12 NE=12 LY=DI,FI PS=DI,FR TE=ZE
LE
F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12
VA 1 LY 1 1 LY 2 2 LY 3 3 LY 4 4 LY 5 5 LY 6 6 LY 7 7 LY 8 8 LY 9 9
VA 1 LY 10 10 LY 11 11 LY 12 12
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL2 - 1 FACTOR
DA NI=12 NO=1000 MA=KM
KM SY
(12F10.6)
MO NY=12 NE=1 LY=FU,FR PS=ST TE=DI,FR
LE
F1
```

```
FI PS 1 1
VA 1 PS 1 1
FR LY 1 1
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL3 - 2 FACTORS
DA NI=12 NO=1000 MA=KM
KM SY
(12F10.6)
MO NY=12 NE=2 LY=FU,FR PS=SY,FI TE=DI,FR
LE
F1 F2
FI LY 1 2 LY 2 1
VA 0 LY 1 2 LY 2 1
VA 1 PS 1 1 PS 2 2
FR PS 2 1 LY 1 1 LY 2 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL4 - 3 FACTORS
DA NI=12 NO=1000 MA=KM
KM SY
(12F10.6)
MO NY=12 NE=3 LY=FU,FR PS=SY,FR TE=DI,FR
LE
F1 F2 F3
FI LY 1 2 LY 1 3 LY 2 3 LY 2 1 LY 3 1 LY 3 2 PS 1 1 PS 2 2 PS 3 3
VA 0 LY 1 2 LY 1 3 LY 2 3 LY 2 1 LY 3 1 LY 3 2
VA 1 PS 1 1 PS 2 2 PS 3 3
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL5 - EXTRA LOADING
DA NI=12 NO=1000 MA=KM
KM SY
(12F10.6)
MO NY=12 NE=2 LY=FU,FI PS=DI,FR TE=DI,FR
LE
F1 F2
FR LY 2 1 LY 3 1 LY 4 1 LY 5 1 LY 6 1
FR LY 8 2 LY 9 2 LY 10 2 LY 11 2 LY 12 2 LY 1 2
VA 1 LY 1 1 LY 7 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL6 – CORRELATED FACTORS
DA NI=12 NO=1000 MA=KM
KM SY
(12F10.6)
MO NY=12 NE=2 LY=FU,FI PS=SY,FR TE=DI,FR
LE
F1 F2
FR LY 2 1 LY 3 1 LY 4 1 LY 5 1 LY 6 1
FR LY 8 2 LY 9 2 LY 10 2 LY 11 2 LY 12 2
VA 1 LY 1 1 LY 7 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL11 - NO PHI12
DA NI=12 NO=1000 MA=KM
KM SY
(12F10.6)
MO NX=12 NK=4 LX=FU,FI PH=SY,FR TD=DI,FR
LE
F1 F2 F3 F4
FR LX 1 1 LX 2 1 LX 3 1 LX 4 1 LX 5 2 LX 6 2 LX 7 2 LX 8 2
FR LX 9 3 LX 10 3 LX 11 4 LX 12 4
FI PH 1 1 PH 2 2 PH 3 3 PH 4 4 PH 2 1
VA 1 PH 1 1 PH 2 2 PH 3 3 PH 4 4
VA 0 PH 2 1
OU ME=UL AD=OFF ND=4 IT=1000 XM
```

```
MODEL12 - NO PHI23
DA NI=12 NO=1000 MA=KM
KM SY
(12F10.6)
MO NX=12 NK=4 LX=FU,FI PH=SY,FR TD=DI,FR
LE
F1 F2 F3 F4
FR LX 1 1 LX 2 1 LX 3 1 LX 4 1 LX 5 2 LX 6 2 LX 7 2 LX 8 2
FR LX 9 3 LX 10 3 LX 11 4 LX 12 4
FI PH 1 1 PH 2 2 PH 3 3 PH 4 4 PH 3 2
VA 1 PH 1 1 PH 2 2 PH 3 3 PH 4 4
VA 0 PH 3 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL13 - NO PHI34
DA NI=12 NO=1000 MA=KM
KM SY
(12F10.6)
MO NX=12 NK=4 LX=FU,FI PH=SY,FR TD=DI,FR
LE
F1 F2 F3 F4
FR LX 1 1 LX 2 1 LX 3 1 LX 4 1 LX 5 2 LX 6 2 LX 7 2 LX 8 2
FR LX 9 3 LX 10 3 LX 11 4 LX 12 4
FI PH 1 1 PH 2 2 PH 3 3 PH 4 4 PH 4 3
VA 1 PH 1 1 PH 2 2 PH 3 3 PH 4 4
VA 0 PH 4 3
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL14 - CORRELATED FACTORS
DA NI=6 NO=1000 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=2 LY=FI PS=SY,FR TE=DI,FR
LE
F1 F2
FI PS 1 1 PS 2 2
VA 1 PS 1 1 PS 2 2
FR LY 1 1 LY 2 1 LY 3 1
FR LY 4 2 LY 5 2 LY 6 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL15 - CORRELATED FACTORS WITH RANGE RESTRICTION
DA NI=6 NO=1000 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=2 LY=FI PS=SY,FR TE=DI,FR
LE
F1 F2
FI PS 1 1 PS 2 2
VA 1 PS 1 1 PS 2 2
FR LY 1 1 LY 2 1 LY 3 1
FR LY 4 2 LY 5 2 LY 6 2
IR TE 1 1 <.1
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL16 - FIXED LY 1 1
DA NI=6 NO=1000 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=2 LY=FI PS=SY,FR TE=DI,FR
LE
F1 F2
FI PS 1 1 PS 2 2
VA 1 PS 1 1 PS 2 2
FR LY 2 1 LY 3 1
FR LY 4 2 LY 5 2 LY 6 2
```

```
OU ME=UL ND=4 IT=1000 XM

MODEL17 - EQUAL LY 1 1 and LY 4 2
DA NI=6 NO=1000 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=2 LY=FI PS=SY,FR TE=DI,FR
LE
F1 F2
FI PS 1 1 PS 2 2
VA 1 PS 1 1 PS 2 2
FR LY 1 1 LY 2 1 LY 3 1
FR LY 4 2 LY 5 2 LY 6 2
EQ LY 1 1 LY 4 2
OU ME=UL ND=4 IT=1000 XM

MODEL22 - F1<--F2
DA NI=6 NO=1000 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=2 LY=FI BE=FU,FI PS=DI,FR TE=DI,FR
LE
F1 F2
FR LY 2 1 LY 3 1 LY 5 2 LY 6 2 BE 1 2
VA 1 LY 1 1 LY 4 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL23 - F1-->F2
DA NI=6 NO=1000 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=2 LY=FI BE=FU,FI PS=DI,FR TE=DI,FR
LE
F1 F2
FR LY 2 1 LY 3 1 LY 5 2 LY 6 2 BE 2 1
VA 1 LY 1 1 LY 4 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL31 - NO CONSTRAINT
DA NI=9 NO=1000 MA=KM
KM SY
(9F10.6)
MO NX=3 NY=6 NK=1 NE=2 LX=FU,FI LY=FU,FI GA=FU,FI BE=FU,FI PH=DI,FR PS=DI,FR TD=DI,FR
TE=DI,FR
LE
KSI1 ETA1 ETA2
FR LX 2 1 LX 3 1 LY 2 1 LY 3 1 LY 5 2 LY 6 2 BE 2 1 GA 1 1
VA 1 LX 1 1 LY 1 1 LY 4 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL32 - == CONSTRAINT
DA NI=9 NO=1000 MA=KM
KM SY
(9F10.6)
MO NX=3 NY=6 NK=1 NE=2 LX=FU,FI LY=FU,FI GA=FU,FI BE=FU,FI PH=DI,FR PS=DI,FR TD=DI,FR
TE=DI,FR
LE
KSI1 ETA1 ETA2
FR LX 2 1 LX 3 1 LY 2 1 LY 3 1 LY 5 2 LY 6 2 BE 2 1 GA 1 1
VA 1 LX 1 1 LY 1 1 LY 4 2
EQ TD 1 1 TD 2 2 TD 3 3
EQ TE 1 1 TE 2 2 TE 3 3
EQ TE 4 4 TE 5 5 TE 6 6
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL33 - EQUAL LOADINGS
DA NI=6 NO=1000 MA=KM
```

```
KM SY
(6F10.6)
MO NY=6 NE=2 LY=FU,FI BE=FU,FI PS=DI,FR TE=DI,FR
LE
F1 F2
FR LY 2 1 LY 3 1 LY 5 2 LY 6 2 BE 2 1
EQ TE 1 1 TE 2 2 TE 3 3
EQ TE 4 4 TE 5 5 TE 6 6
VA 1 LY 1 1 LY 4 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL34 - NO CORRELATION
DA NI=6 NO=1000 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=2 LY=FU,FI BE=FU,FI PS=DI,FR TE=DI,FR
LE
F1 F2
FR LY 2 1 LY 3 1 LY 5 2 LY 6 2
VA 1 LY 1 1 LY 4 2
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL35 - SIMPLEX
DA NI=6 NO=1000 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=6 LY=ID BE=FU,FI PS=DI,FR TE=DI,FI
LE
F1 F2 F3 F4 F5 F6
FR BE 2 1 BE 3 2 BE 4 3 BE 5 4 BE 6 5
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL36 - FACTOR MODEL
DA NI=6 NO=1000 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=1 LY=FU,FR PS=SY,FR TE=DI,FR
LE
F1
FI LY 1 1
VA 1 LY 1 1
EQ LY 2 1 LY 3 1
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL 37: SCALE-FREE BLOCK-DIAGONAL CCA
DA NI=9 NO=1000
KM SY
(9F10.6)
MO NY=9 NE=9 NK=5 PS=ZE LY=DI,FR GA=FU,FI PH=SY,FI
MA GA
 0.333333      0.471405             0      0.471405             0
 0.333333     -0.235702      0.408248      0.471405             0
 0.333333     -0.235702     -0.408248      0.471405             0
 0.333333      0.471405             0     -0.235702      0.408248
 0.333333     -0.235702      0.408248     -0.235702      0.408248
 0.333333     -0.235702     -0.408248     -0.235702      0.408248
 0.333333      0.471405             0     -0.235702     -0.408248
 0.333333     -0.235702      0.408248     -0.235702     -0.408248
 0.333333     -0.235702     -0.408248     -0.235702     -0.408248
FI PH 1 1
FR PH 2 2 PH 3 3 PH 4 4 PH 5 5 PH 3 2 PH 5 4
VA 1 LY 1 1 LY 2 2 LY 3 3 LY 4 4 LY 5 5 LY 6 6 LY 7 7 LY 8 8 LY 9 9 PH 1 1
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL 38 - CDPM MODEL
DA NI=9 NO=1000
KM SY
```

```
(9F10.6)
MO NY=9 NE=9 NK=18 LY=DI,FR GA=FU,FR PH=SY,FR PS=ZE BE=ZE TE=ZE
ST .5 LY 1 - LY 9
PA GA
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0
MA GA
1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0
1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0
0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0
0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1
FI LY 1 1 LY 4 4 LY 7 7
VA 1 LY 1 1 LY 4 4 LY 7 7
EQ GA 1 1 GA 2 2 GA 3 3
EQ GA 4 1 GA 5 2 GA 6 3
EQ GA 4 4 GA 5 5 GA 6 6
EQ GA 7 1 GA 8 2 GA 9 3
EQ GA 7 4 GA 8 5 GA 9 6
EQ GA 7 7 GA 8 8 GA 9 9
PA PH
0
1 0
1 1 0
0 0 0 0
0 0 0 1 0
0 0 0 1 1 0
0 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 0 0 1 1 0
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
MA PH
1
.5 1
.5 .5 1
0 0 0 1
0 0 0 .5 1
0 0 0 .5 .5 1
0 0 0 0 0 0 1
0 0 0 0 0 0 .5 1
0 0 0 0 0 0 .5 .5 1
0 0 0 0 0 0 0 0 0 .2
0 0 0 0 0 0 0 0 0 0 .2
0 0 0 0 0 0 0 0 0 0 0 .2
0 0 0 0 0 0 0 0 0 0 0 0 .2
0 0 0 0 0 0 0 0 0 0 0 0 0 .2
0 0 0 0 0 0 0 0 0 0 0 0 0 0 .2
```

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 .2
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 .2
EQ PH 2 1 PH 5 4 PH 8 7
EQ PH 3 1 PH 6 4 PH 9 7
EQ PH 3 2 PH 6 5 PH 9 8
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL45 - NEWSOM H
DA NI=6 NO=4734 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=6 LY=DI,FI PS=SY,FI TE=ZE BE=FU,FI
LE
PI BS AS TS LS DE
VA 1 LY 1 1 LY 2 2 LY 3 3 LY 4 4 LY 5 5 LY 6 6
FR BE 2 1 BE 3 1 BE 4 1
FR BE 5 2 BE 5 3 BE 5 4 BE 6 2 BE 6 3 BE 6 4
FR PS 1 1 PS 2 2 PS 3 3 PS 4 4 PS 5 5 PS 6 6
FR PS 3 2 PS 4 3 PS 4 2 PS 6 5
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL46 - NEWSOM 1
DA NI=6 NO=4734 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=6 LY=DI,FI PS=SY,FI TE=ZE BE=FU,FI
LE
PI BS AS TS LS DE
VA 1 LY 1 1 LY 2 2 LY 3 3 LY 4 4 LY 5 5 LY 6 6
FR BE 1 2 BE 1 3 BE 1 4 BE 2 4 BE 6 4
FR PS 1 1 PS 2 2 PS 3 3 PS 4 4 PS 5 5 PS 6 6
FR PS 3 2 PS 4 2 PS 4 3 PS 6 5
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL49 - NEWSOM 4
DA NI=6 NO=4734 MA=KM
KM SY
(6F10.6)
MO NY=6 NE=6 LY=DI,FI PS=SY,FI TE=ZE BE=FU,FI
LE
PI BS AS TS LS DE
VA 1 LY 1 1 LY 2 2 LY 3 3 LY 4 4 LY 5 5 LY 6 6
FR BE 5 1 BE 6 1 BE 2 5 BE 2 6 BE 3 5 BE 3 6 BE 4 5 BE 4 6
FR PS 1 1 PS 2 2 PS 3 3 PS 4 4 PS 5 5 PS 6 6
FR PS 6 5 PS 3 2 PS 4 2 PS 4 3
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL50 - BEIER AND ACKERMAN
DA NI=11 NO=153 MA=KM
KM SY
(11F10.6)
MO NY=11 NE=11 LY=DI,FI PS=SY,FI TE=ZE BE=FU,FI
LE
AGE GF GC MA TR OE TIE BF SK VSC CEK
VA 1 LY 1 1 LY 2 2 LY 3 3 LY 4 4 LY 5 5 LY 6 6
VA 1 LY 7 7 LY 8 8 LY 9 9 LY 10 10 LY 11 11 PS 1 1
FR BE 2 1 BE 3 1 BE 3 2 BE 3 4 BE 3 5 BE 3 6 BE 3 7 BE 3 8
FR BE 4 2 BE 5 2 BE 6 2 BE 7 2 BE 8 2 BE 9 3 BE 10 3
FR BE 11 9 BE 11 10 BE 11 3
FR PS 2 2 PS 3 3 PS 4 4 PS 5 5 PS 6 6 PS 7 7 PS 8 8
FR PS 9 9 PS 10 10 PS 11 11 PS 6 5 PS 7 6 PS 8 7 PS 8 6
FR PS 8 5 PS 10 9
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL52 - COFFMAN ET AL.
DA NI=5 NO=83 MA=KM
KM SY
```

```
(5F10.6)
MO NY=5 NE=5 LY=DI,FI PS=SY,FI TE=DI,FI BE=FU,FI
LE
CS EC RS AF AT
VA 1 LY 1 1 LY 2 2 LY 3 3 LY 4 4 LY 5 5
FR PS 1 1 PS 2 2 PS 3 3 PS 4 4 PS 5 5 PS 2 1
FR BE 3 1 BE 3 2 BE 4 1 BE 4 2 BE 4 3 BE 5 4
OU ME=UL AD=OFF ND=4 IT=1000 XM

MODEL53 - CONTRIVED 5-MV 2-FACTOR MODEL
DA NI=5 NO=1000 MA=KM
KM SY
(5F10.6)
MO NY=5 NE=2 LY=FU,FI PS=SY,FR TE=DI,FR
LE
F1 F2
FR LY 2 1 LY 3 1 LY 5 2
VA 1 LY 1 1 LY 4 2
EQ TE 1 1 TE 2 2 TE 3 3
OU ME=UL AD=OFF ND=4 IT=1000 XM
```

APPENDIX D

FORTRAN CODE FOR SUMMARIZING FIT INFORMATION

```
! The purpose of RandomR7_lisrel is to read output files from LISREL,
! summarizing fit statistics and information about nonconvergence.

program randomr7_lisrel
  implicit none

  integer               :: flag,i,ios,j,k,mo,nm
  character(51)         :: garbage
  character(57)         :: mod14,mod15,mod16,mod17,mod22,mod23,mod31,mod32,mod33
  character(57)         :: mod34,mod35,mod36,mod37,mod38,mod45,mod46,mod52,mod53
  character(59)         :: mod01,mod02,mod03,mod04,mod05,mod06,mod11,mod12,mod13,mod50
  character(60)         :: mod14out,mod15out,mod16out,mod17out,mod22out,mod23out
  character(60)         :: mod31out,mod32out,mod33out,mod34out,mod35out,mod36out
  character(60)         :: mod37out,mod38out,mod45out,mod46out,mod52out,mod53out
  character(62)         :: mod01out,mod02out,mod03out,mod04out,mod05out,mod06out
  character(62)         :: mod11out,mod12out,mod13out
  character(62)         :: mod50out
  real*8, allocatable, dimension(:,:,:)     :: matrix

  nm=5000 ! number of matrices to analyze
  ! LISREL output files (comment out undesired files):
  mod01="c:\dissfiles\mcmc12x12\mod01.out"       ! LISREL output for model 01
  mod02="c:\dissfiles\mcmc12x12\mod02.out"       ! LISREL output for model 02
  mod03="c:\dissfiles\mcmc12x12\mod03.out"       ! LISREL output for model 03
  mod04="c:\dissfiles\mcmc12x12\mod04.out"       ! LISREL output for model 04
  mod05="c:\dissfiles\mcmc12x12\mod05.out"       ! LISREL output for model 05
  mod06="c:\dissfiles\mcmc12x12\mod06.out"       ! LISREL output for model 06
  mod11="c:\dissfiles\mcmc12x12\mod11.out"       ! LISREL output for model 11
  mod12="c:\dissfiles\mcmc12x12\mod12.out"       ! LISREL output for model 12
  mod13="c:\dissfiles\mcmc12x12\mod13.out"       ! LISREL output for model 13
  mod14="c:\dissfiles\mcmc6x6\mod14.out"         ! LISREL output for model 14
  mod15="c:\dissfiles\mcmc6x6\mod15.out"         ! LISREL output for model 15
  mod16="c:\dissfiles\mcmc6x6\mod16.out"         ! LISREL output for model 16
  mod17="c:\dissfiles\mcmc6x6\mod17.out"         ! LISREL output for model 17
  mod22="c:\dissfiles\mcmc6x6\mod22.out"         ! LISREL output for model 22
  mod23="c:\dissfiles\mcmc6x6\mod23.out"         ! LISREL output for model 23
  mod31="c:\dissfiles\mcmc9x9\mod31.out"         ! LISREL output for model 31
  mod32="c:\dissfiles\mcmc9x9\mod32.out"         ! LISREL output for model 32
  mod33="c:\dissfiles\mcmc6x6\mod33.out"         ! LISREL output for model 33
  mod34="c:\dissfiles\mcmc6x6\mod34.out"         ! LISREL output for model 34
  mod35="c:\dissfiles\mcmc6x6\mod35.out"         ! LISREL output for model 35
  mod36="c:\dissfiles\mcmc6x6\mod36.out"         ! LISREL output for model 36
  mod37="c:\dissfiles\mcmc9x9\mod37.out"         ! LISREL output for model 37
  mod38="c:\dissfiles\mcmc9x9\mod38.out"         ! LISREL output for model 38
  mod45="c:\dissfiles\mcmc6x6\mod45.out"         ! LISREL output for model 45
  mod46="c:\dissfiles\mcmc6x6\mod46.out"         ! LISREL output for model 46
  mod50="c:\dissfiles\mcmc11x11\mod50.out"       ! LISREL output for model 50
  mod52="c:\dissfiles\mcmc5x5\mod52.out"         ! LISREL output for model 52
  mod53="c:\dissfiles\mcmc5x5\mod53.out"         ! LISREL output for model 53
```

```
! Summary files (comment out undesired files):
mod01out="c:\dissfiles\out\mcmc12x12\mod01out.out" ! Summarized output for model 01
mod02out="c:\dissfiles\out\mcmc12x12\mod02out.out" ! Summarized output for model 02
mod03out="c:\dissfiles\out\mcmc12x12\mod03out.out" ! Summarized output for model 03
mod04out="c:\dissfiles\out\mcmc12x12\mod04out.out" ! Summarized output for model 04
mod05out="c:\dissfiles\out\mcmc12x12\mod05out.out" ! Summarized output for model 05
mod06out="c:\dissfiles\out\mcmc12x12\mod06out.out" ! Summarized output for model 06
mod11out="c:\dissfiles\out\mcmc12x12\mod11out.out" ! Summarized output for model 11
mod12out="c:\dissfiles\out\mcmc12x12\mod12out.out" ! Summarized output for model 12
mod13out="c:\dissfiles\out\mcmc12x12\mod13out.out" ! Summarized output for model 13
mod14out="c:\dissfiles\out\mcmc6x6\mod14out.out"   ! Summarized output for model 14
mod15out="c:\dissfiles\out\mcmc6x6\mod15out.out"   ! Summarized output for model 15
mod16out="c:\dissfiles\out\mcmc6x6\mod16out.out"   ! Summarized output for model 16
mod17out="c:\dissfiles\out\mcmc6x6\mod17out.out"   ! Summarized output for model 17
mod22out="c:\dissfiles\out\mcmc6x6\mod22out.out"   ! Summarized output for model 22
mod23out="c:\dissfiles\out\mcmc6x6\mod23out.out"   ! Summarized output for model 23
mod31out="c:\dissfiles\out\mcmc9x9\mod31out.out"   ! Summarized output for model 31
mod32out="c:\dissfiles\out\mcmc9x9\mod32out.out"   ! Summarized output for model 32
mod33out="c:\dissfiles\out\mcmc6x6\mod33out.out"   ! Summarized output for model 33
mod34out="c:\dissfiles\out\mcmc6x6\mod34out.out"   ! Summarized output for model 34
mod35out="c:\dissfiles\out\mcmc6x6\mod35out.out"   ! Summarized output for model 35
mod36out="c:\dissfiles\out\mcmc6x6\mod36out.out"   ! Summarized output for model 36
mod37out="c:\dissfiles\out\mcmc9x9\mod37out.out"   ! Summarized output for model 37
mod38out="c:\dissfiles\out\mcmc9x9\mod38out.out"   ! Summarized output for model 38
mod45out="c:\dissfiles\out\mcmc6x6\mod45out.out"   ! Summarized output for model 45
mod46out="c:\dissfiles\out\mcmc6x6\mod46out.out"   ! Summarized output for model 46
mod50out="c:\dissfiles\out\mcmc11x11\mod50out.out" ! Summarized output for model 50
mod52out="c:\dissfiles\out\mcmc5x5\mod52out.out"   ! Summarized output for model 52
mod53out="c:\dissfiles\out\mcmc5x5\mod53out.out"   ! Summarized output for model 53

allocate (matrix(50,nm,2))
do i=1,50
  do j=1,nm
    do k=1,2
      matrix(i,j,k)=0
    enddo
  enddo
enddo
model_loop: do mo=1,4 ! change for each desired set of models.
  select case(mo)
    case(1)
      open(unit=3,file=mod01,position="rewind",iostat=ios)
      open(unit=4,file=mod01out,position="rewind",iostat=ios)
    case(2)
      open(unit=3,file=mod02,position="rewind",iostat=ios)
      open(unit=4,file=mod02out,position="rewind",iostat=ios)
    case(3)
      open(unit=3,file=mod03,position="rewind",iostat=ios)
      open(unit=4,file=mod03out,position="rewind",iostat=ios)
    case(4)
      open(unit=3,file=mod04,position="rewind",iostat=ios)
      open(unit=4,file=mod04out,position="rewind",iostat=ios)
    case(5)
      open(unit=3,file=mod05,position="rewind",iostat=ios)
      open(unit=4,file=mod05out,position="rewind",iostat=ios)
    case(6)
      open(unit=3,file=mod06,position="rewind",iostat=ios)
      open(unit=4,file=mod06out,position="rewind",iostat=ios)
    case(11)
      open(unit=3,file=mod11,position="rewind",iostat=ios)
      open(unit=4,file=mod11out,position="rewind",iostat=ios)
    case(12)
      open(unit=3,file=mod12,position="rewind",iostat=ios)
      open(unit=4,file=mod12out,position="rewind",iostat=ios)
    case(13)
      open(unit=3,file=mod13,position="rewind",iostat=ios)
      open(unit=4,file=mod13out,position="rewind",iostat=ios)
```

```
  case(14)
    open(unit=3,file=mod14,position="rewind",iostat=ios)
    open(unit=4,file=mod14out,position="rewind",iostat=ios)
  case(15)
    open(unit=3,file=mod15,position="rewind",iostat=ios)
    open(unit=4,file=mod15out,position="rewind",iostat=ios)
  case(16)
    open(unit=3,file=mod16,position="rewind",iostat=ios)
    open(unit=4,file=mod16out,position="rewind",iostat=ios)
  case(17)
    open(unit=3,file=mod17,position="rewind",iostat=ios)
    open(unit=4,file=mod17out,position="rewind",iostat=ios)
  case(22)
    open(unit=3,file=mod22,position="rewind",iostat=ios)
    open(unit=4,file=mod22out,position="rewind",iostat=ios)
  case(23)
    open(unit=3,file=mod23,position="rewind",iostat=ios)
    open(unit=4,file=mod23out,position="rewind",iostat=ios)
  case(31)
    open(unit=3,file=mod31,position="rewind",iostat=ios)
    open(unit=4,file=mod31out,position="rewind",iostat=ios)
  case(32)
    open(unit=3,file=mod32,position="rewind",iostat=ios)
    open(unit=4,file=mod32out,position="rewind",iostat=ios)
  case(33)
    open(unit=3,file=mod33,position="rewind",iostat=ios)
    open(unit=4,file=mod33out,position="rewind",iostat=ios)
  case(34)
    open(unit=3,file=mod34,position="rewind",iostat=ios)
    open(unit=4,file=mod34out,position="rewind",iostat=ios)
  case(35)
    open(unit=3,file=mod35,position="rewind",iostat=ios)
    open(unit=4,file=mod35out,position="rewind",iostat=ios)
  case(36)
    open(unit=3,file=mod36,position="rewind",iostat=ios)
    open(unit=4,file=mod36out,position="rewind",iostat=ios)
  case(37)
    open(unit=3,file=mod37,position="rewind",iostat=ios)
    open(unit=4,file=mod37out,position="rewind",iostat=ios)
  case(38)
    open(unit=3,file=mod38,position="rewind",iostat=ios)
    open(unit=4,file=mod38out,position="rewind",iostat=ios)
  case(45)
    open(unit=3,file=mod45,position="rewind",iostat=ios)
    open(unit=4,file=mod45out,position="rewind",iostat=ios)
  case(46)
    open(unit=3,file=mod46,position="rewind",iostat=ios)
    open(unit=4,file=mod46out,position="rewind",iostat=ios)
  case(50)
    open(unit=3,file=mod50,position="rewind",iostat=ios)
    open(unit=4,file=mod50out,position="rewind",iostat=ios)
  case(52)
    open(unit=3,file=mod52,position="rewind",iostat=ios)
    open(unit=4,file=mod52out,position="rewind",iostat=ios)
  case(53)
    open(unit=3,file=mod53,position="rewind",iostat=ios)
    open(unit=4,file=mod53out,position="rewind",iostat=ios)
end select
i=0
matrix_loop: do
  read (unit=3,fmt=40,iostat=ios) garbage
  40 format (A51)
  if (ios/=0) exit
  if (garbage(33:43)=="L I S R E L") then
    i=i+1
    flag=0
    write (6,34) ' Working on ',i
```

```
        34 format (A12,I6)
      endif
      if (garbage(2:28)=="W_A_R_N_I_N_G: The solution") then
        matrix(mo,i,2)=1
        read (3,40) garbage
      endif
      if (garbage(21:45)=="Root Mean Square Residual") then
        backspace(3)
        read(3,37) matrix(mo,i,1)
        37 format(54X,F7.5)
      endif
      if (garbage(22:46)=="Root Mean Square Residual") then
        backspace(3)
        read(3,36) matrix(mo,i,1)
        36 format(55X,F6.4)
      endif
    enddo matrix_loop
    close(3)
    do i=1,nm
      write (4,35) (matrix(mo,i,j),j=1,2)
      35 format (2F12.5)
    enddo
    close(4)
  enddo model_loop
  deallocate (matrix)
end program randomr7_lisrel
```