Justin,

This is a remarkable piece of work. Congratulations on this impressive achievement. One of the best dissertations to ever come out of the QPM area (seriously– every time I had a question you ran a small simulation study and answered it). This is a very creative piece of work,

After you get a chance to read over my comments lets set up a zoom call asap so that you can get a cleaned up copy to your committee.

I couldn't be happier. You will be Dr Justin very soon!

Congrats again,

Niels

UNIVERSITY OF MINNESOTA

This is to certify that I have examined
this copy of a doctoral dissertation by

Justin D. Kracht

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Niels G. Waller

Name of Faculty Advisor

Signature of Faculty Advisor

Date

GRADUATE SCHOOL

Make Some Noise: Methods for Generating Data From Imperfect Factor Models

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Justin D. Kracht

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Niels G. Waller, Advisor

April 2022

Acknowledgements

I would like to acknowledge. . . Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas vel eros sed mauris porttitor semper nec a orci. Nullam vestibulum mi nec condimentum posuere. Pellentesque eget diam id sapien aliquet ullamcorper. Pellentesque blandit nec lectus ut mollis. Praesent in facilisis justo. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Sed eget congue leo, sed consequat libero. In rutrum malesuada nisi. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Morbi sollicitudin tortor ut sem facilisis mollis.

Dedication

This is for my mother who paved the way.

Abstract

Researchers conducting Monte Carlo simulation studies involving covariance structure models (e.g., the common factor model) have increasingly recognized the importance of incorporating error due to model misfit in simulated data. Incorporating this model error acknowledges that all models are literally false, and no covariance structure model will fit perfectly in the population. Several methods for generating data from error-perturbed models have been proposed, including the Tucker, Koopman, and Linn (TKL; 1969), Cudeck and Browne (CB; 1992), and Wu and Browne (WB; 2015) model-error methods. All of these methods require user-specified parameter values that determine the degree of model misfit to be introduced. In particular, the CB and WB methods each have a single parameter that is related to the desired Root Mean Square Error of Approximation (RMSEA) value for the simulated covariance matrix. In contrast, the TLK method includes two parameter values that are generally chosen to align with values used in previous simulation studies or by testing many combinations of parameter values until solutions have RMSEA values that are close to the desired value. However, although RMSEA has often been used to indicate the degree of misfit introduced by model-error methods, RMSEA alone does not provide a complete summary of model fit. To get a more complete summary of model fit, other types of fit indices like the Comparative Fit Index (CFI) should also be used. Unfortunately, the TKL, CB, and WB model-error methods do not provide a way to specify multiple target fit indices.

To address this issue, I proposed an optimization procedure that allows users to specify either a target RMSEA value, a target CFI value, or both simultaneously, and then attempts to find a combination of parameter values that produces a solution with fit indices close to the target values. To test the procedure, I conducted a simulation study using the

proposed multiple-target TKL method, the CB method, and the WB method to generate error-perturbed correlation matrices for models with varying numbers of factors, items per factor, salient factor loadings, factor correlations, and target model fit indices. The results of the simulation study showed that the multiple-target TKL method led to solutions with RMSEA and CFI values that indicated similar qualitative levels of model fit compared to the alternatives. Thus, the multiple-target TKL method should be a useful tool for researchers who wish to generate error-perturbed correlation matrices in Monte Carlo simulation studies. To facilitate its use, I wrote an R package (*noisemaker*) with implementations of the multiple-target TKL method, along with implementations of the CB and WB model-error methods.

This is not exactly clear for me.
How did the other misfit models
perform?

Table of Contents

List of Tables

You should check. The university may not allow colored text. I honestly do not know. In any case blue text is had to xerox or scan into BW

List of Figures

Chapter 1

Introduction

Cite Bollen 1989 instead. Cudeck's article is about R matrices.

Covariance structure models (also called structural equation models) are widely used in psychological research (Cudeck, 1989). These models allow a structured covariance matrix to be represented as a matrix-valued function of a vector of parameters such that $\boldsymbol{\Omega} = \boldsymbol{\Omega}(\boldsymbol{\gamma})$, where $\boldsymbol{\Omega}$ is a $p \times p$ covariance matrix and $\boldsymbol{\gamma}$ is a vector of free model parameters. Stated another way, covariance structure models attempt to represent the structural connections between a set of unobserved latent variables (factors) and a set of observed variables that are indicators of the latent variables. When assessing model fit or estimating the dispersion of the estimated structural parameters, $\hat{\gamma}$, researchers traditionally assume that the model holds perfectly in the population. It is assumed that there exists some vector $\boldsymbol{\gamma} = \boldsymbol{\gamma}_0$ such that the population covariance matrix $\boldsymbol{\Omega}$ can be perfectly reproduced (i.e., $\boldsymbol{\Omega} = \boldsymbol{\Omega}(\boldsymbol{\gamma}_0)$). If the model fits perfectly in the population, differences between the sample covariance matrix and the corresponding model-implied covariance matrix can only be due to sampling variability.

However, the assumption that the covariance structure is correctly specified in the population will nearly always be violated in practice (Browne et al., 2002; Cudeck & Henly, 1991; MacCallum et al., 2001; MacCallum & Tucker, 1991; Meehl & Waller, 2002; Tomarken & Waller, 2003). After all, "no mathematical model will fit real-world phenomena exactly" (MacCallum et al., 2001, p. 503) due, for example, to non-linearities in the relationships between factors and indicators, or to the effect of numerous minor factors of little theoretical interest (Cudeck & Browne, 1992). The idea that all models are imperfect representations of reality has been expressed many ways, perhaps most succinctly by Box's aphorism that "all models are wrong, but some are useful" (1987, p. 424). Further interesting discussions of the idea that models are only useful approximations of reality (and are often useful *because* they are approximations) can be found in both scientific and popular literature (Borges, 1998; Carroll, 1894; Eco, 1994; Gelman, 2008; Nester, 1996; Steele, 2008).

Recognizing that all models are literally false, and thus no covariance structure model will fit perfectly in the population, several authors have suggested that estimation error for these models can be attributed to two sources: error due to sampling variability and error due to model misfit (Browne & Cudeck, 1992; Cudeck & Henly, 1991; Tucker et al., 1969). This second type of error has been variously referred to as *model error* (MacCallum et al., 2001; Tucker et al., 1967, 1969), *error of approximation* (Cudeck & Henly, 1991), *specification error* (Satorra, 2015), or *adventitious error* (Wu & Browne, 2015a). Although some of these terms are related to specific views regarding the nature of model error (Tucker et al., 1969; c.f. Wu & Browne, 2015a), all of the terms refer to the discrepancy between the model-implied covariance matrix, $\mathbf{\Omega}$, and the error-perturbed population covariance matrix, $\mathbf{\Sigma} = \mathbf{\Omega} + \mathbf{E}$ (where $\mathbf{E}$ is a symmetric matrix representing the effects of model error). In this proposal, the term "model error" will generally be used to describe the discrepancy between ~~the~~ $\mathbf{\Sigma}$ and $\mathbf{\Omega}$.

Acknowledging model error is important because it can have significant implications for estimating covariance structure models. For instance, consider the traditional Chi-square test of exact model fit. Given a sample covariance matrix $\mathbf{S}$, the minimum objective function value for a hypothesized model $\hat{F} = F(\mathbf{S}, \mathbf{\Omega}(\hat{\gamma}))$ obtained by minimizing a discrepancy function $F(\mathbf{S}, \mathbf{\Omega}(\gamma))$ is assumed to follow a central Chi-square distribution when multiplied by $n = N - 1$, where N denotes the sample size (Olsson et al., 2004).[1] However, the test

[1]The maximum likelihood discrepancy function is commonly used, but other common discrepancy functions (e.g., generalized least squares, weighted least squares, asymptotically distribution free) will converge to the same minimum discrepancy values when the model is correctly specified and the observed variables

Is this at least 10 pt?

requires two stringent assumptions that are unlikely to both be satisfied in empirical settings: (a) that the observed variables are multivariate normal, and (b) that the model fits perfectly in the population (i.e., there is no model error; Browne, 1984). If (b) is not satisfied, then the test statistic $n\hat{F}$ will not follow a central Chi-square distribution and will lead to incorrect tests (Olsson et al., 2004). Moreover, Chi-square tests of exact fit are sensitive to sample size, with large sample sizes leading to almost certain rejection of the model, even with small amounts of misspecification (Bentler & Bonett, 1980; Tomarken & Waller, 2003; Yuan & Marshall, 2004). In any case, testing a model that is known not to be perfectly true against the null hypothesis of perfect fit would seem to have limited usefulness (Browne & Cudeck, 1992; Steiger, 2007).

Model error also has implications for covariance structure modeling beyond global tests of model fit. For instance, traditional methods of computing confidence intervals for model parameters assume that all error is sampling error (i.e., the model fits perfectly in the population). Thus, confidence intervals produced using these methods are overly-optimistic when model error is present (Wu & Browne, 2015a). Simulation studies have shown that the presence of model error can also impact parameter estimation for exploratory factor analysis (Briggs & MacCallum, 2003), dimensionality identification (Kracht & Waller, 2022), the behavior of confidence regions and fungible parameter estimates for structural equation models (Pek, 2012), and is important in other contexts as well (Beauducel & Hilger, 2016; de Winter & Dodou, 2016; Gnambs & Staufenbiel, 2016; Hsu et al., 2015; Trichtinger & Zhang, 2020).

These simulation studies represent a recent trend toward incorporating model error in Monte Carlo simulation studies involving covariance structure models. Including model error in these studies is important for at least two reasons. First, the addition of model error makes simulated data sets more representative of empirical data sets, which almost certainly do not have population covariance matrices that are perfectly fit by any simple covariance

are multivariate normal (Olsson et al., 2004).

structure model. Therefore, the inclusion of model error should lead to results that are more generalizable to empirical settings compared to Monte Carlo studies that do not include model error. Second, incorporating model error allows researchers to evaluate the robustness of methods when covariance structure models do not hold exactly.

Recognizing the importance of incorporating model error in Monte Carlo simulation studies, various authors have introduced methods for generating population covariance matrices with imperfect model fit. The three most popular of these methods were proposed by (a) Tucker, Koopman, and Linn (TKL; 1969), (b) Cudeck and Browne (CB; 1992), and (c) Wu and Browne (WB; 2015a).

1.1 Model-Error Methods

1.1.1 The Tucker, Koopman, and Linn Method

One of the first model-error methods was developed by Tucker et al. (1969) in the context of common factor analysis. The common factor analysis model in terms of p manifest variables and k common factors can be written as

That constraint does not produce unit diagonal Lambda must also be properly scaled —think of it this way. Regress y on x. Assume y has var of 2. Now regress y on z where z is x standardized. Predicted y does not have var 1.

$$\mathbf{\Omega} = \mathbf{\Lambda}\mathbf{\Phi}\mathbf{\Lambda}' + \mathbf{\Psi}, \tag{1.1}$$

where $\mathbf{\Omega}$ is the model-implied $p \times p$ covariance matrix, $\mathbf{\Lambda}$ is the $p \times k$ factor-pattern matrix, $\mathbf{\Phi}$ is the $k \times k$ common factor covariance matrix, and $\mathbf{\Psi}$ is the $p \times p$ diagonal matrix containing the uniqueness variances. Without loss of generality, we can define all common factors as having means of zero and standard deviations of one so that $\mathbf{\Omega}$ has a unit diagonal (i.e., is a correlation matrix). Tucker et al. (1969) proposed extending the common factor model to include many minor common factors of small effect in addition to the major common factors to represent "unsystematic or unknown aspects of the process that generates the data" (Cudeck & Browne, 1992, p. 358). The model they proposed can be written as

$$\mathbf{\Sigma} = \mathbf{\Lambda}\mathbf{\Phi}\mathbf{\Lambda}' + \mathbf{\Psi} + \mathbf{W}\mathbf{W}', \tag{1.2}$$

where $\mathbf{\Sigma}$ is the $p \times p$ error-perturbed population covariance matrix, $\mathbf{W}$ is the $p \times q$ matrix of minor common factor loadings, and all other terms are as previously defined. For convenience, we can define the observed variables to be in standard score form such that $\mathbf{\Sigma}$ has a unit diagonal. The combined influence of the q minor common factors represents the reliable common variance (and covariance) not accounted for by the major common factors and is considered to be due to model error.

Have you defined this?

The proportion of the uniqueness variance reapportioned to the minor common factors and how that variance is distributed to each minor common factor are determined by two user-specified parameters, $\nu_{\mathrm{e}} \in [0, 1]$ and $\epsilon \in [0, 1]$, respectively. To create the matrix of minor factor loadings, $\mathbf{W}$, a $p \times q$ provisional matrix, $\mathbf{W}^*$, is first created such that the ith column of $\mathbf{W}^*$ consists of p independent samples from $\mathcal{N}(0, (1-\epsilon)^{i-1})$, where $\mathcal{N}(0, (1-\epsilon)^{i-1})$ denotes a normal distribution with a mean of zero and a standard deviation of $(1-\epsilon)^{i-1}$. Because the standard deviation of the ith column of $\mathbf{W}^*$ is given by $(1-\epsilon)^{i-1}$, values of ϵ close to zero result in columns with relatively equal variance, corresponding to approximately equipotent minor factors. On the other hand, values of ϵ close to one result in error variance primarily being distributed to the first minor factor, with the remaining variance distributed to the other minor factors in a decreasing geometric sequence.

To ensure that the minor common factors account for the specified proportion of uniqueness variance (ν_{e}), $\mathbf{W}^*$ is then scaled to create $\mathbf{W}$. This scaling is done in several steps. First, a diagonal matrix $\mathbf{\Psi}^*_{p \times p}$ is created such that

You need to define i and its bounds

The normal is typically denoted by mean and Variance rather than sd

$$\mathbf{\Psi}^* = \mathbf{I}_p - \mathrm{dg}(\mathbf{\Lambda}\mathbf{\Phi}\mathbf{\Lambda}'), \tag{1.3}$$

where $\mathrm{dg}(\mathbf{\Lambda}\mathbf{\Phi}\mathbf{\Lambda}')$ is the diagonal matrix formed from the diagonal entries in $\mathbf{\Lambda}\mathbf{\Phi}\mathbf{\Lambda}'$ and $\mathbf{I}_p$ denotes a $p \times p$ identity matrix. Then the matrix $\mathbf{W}$ is formed using

$$\mathbf{W} = (\mathrm{dg}(\mathbf{W}^*\mathbf{W}^{*\prime})^{-1}\mathbf{\Psi}^*\nu_{\mathrm{e}})^{1/2}\mathbf{W}^*. \tag{1.4}$$

This process ensures that the q minor common factors account for the specified proportion of the variance not accounted for by the major common factors. The $\mathbf{W}$ matrix can then be used to create the diagonal matrix of unique variances, $\mathbf{\Psi} = \mathbf{I}_p - \mathrm{dg}(\mathbf{\Lambda\Phi\Lambda}' + \mathbf{WW}')$. The $\mathbf{\Lambda}$, $\mathbf{\Psi}$, and $\mathbf{W}$ matrices are then used to construct the population correlation matrix (with model error), $\mathbf{\Sigma}$, as shown in Equation (1.2). The TKL method is one of the most widely-used methods for generating Correlation? ~~covariance~~ matrices with imperfect model fit (Beauducel & Hilger, 2016; Chung & Cai, 2019; de Winter & Dodou, 2016; Gnambs & Staufenbiel, 2016; Kracht & Waller, 2022; Lorenzo-Seva & Ferrando, 2020a; Lorenzo-Seva & Ginkel, 2016; Myers et al., 2015).[2]

Although the original TKL method is still most commonly used in simulation studies, 24-1 at least two variants of the This ~~original TKL method~~ have since been developed. First, Hong (1999) introduced a variation of the TKL method that allowed for minor common factors that were correlated with each other and with the major common factors. In Hong's model, the population covariance matrix can be written as

$$\mathbf{\Sigma} = \mathbf{LBL}' + \mathbf{\Psi}, \tag{1.5}$$

where $\mathbf{L} = \begin{bmatrix} \mathbf{\Lambda} & \mathbf{W} \end{bmatrix}$ is the super matrix containing the major and minor factor loadings, and $\mathbf{\Psi}$ is the diagonal matrix of uniqueness variances, as defined in Equation (1.1). The matrix $\mathbf{B}$ is the correlation matrix for the major and minor factors such that

$$\mathbf{B} = \begin{bmatrix} \mathbf{\Phi} & \mathbf{\Upsilon} \\ \mathbf{\Upsilon}' & \mathbf{\Gamma} \end{bmatrix}, \tag{1.6}$$

[2] Tucker et al. (1969) was cited 163 times as of March 11, 2021, according to citation counts provided by Web of Science and Crossref.

where $\mathbf{\Phi}$ is the $k \times k$ major factor correlation matrix, $\mathbf{\Upsilon}$ is the $k \times q$ matrix of correlations between the major and minor factors, and $\mathbf{\Gamma}$ is the $q \times q$ minor factor correlation matrix.

Hong's (1999) article has been cited a number of times since its publication[3], but only one published study has applied Hong's method (Porritt, 2015). In Porritt's (2015) simulation study, all major and minor factor correlations were fixed at .3, following the example given by Hong (1999). However, neither Porritt (2015) nor Hong (1999) directly compared Hong's method with the original TKL method. Therefore, it remains unclear whether modeling major-minor factor correlations is important when simulating covariance matrices with model error.

A second variant of the TKL method was recently introduced by Trichtinger and Zhang (2020). Specifically, Trichtinger and Zhang's method adapted the TKL method to allow the simulation of covariance matrices with model error for multivariate time series data. Trichtinger and Zhang (2020) first introduced a novel test statistic appropriate for P-
Give ref for P technique fa
technique factor analysis. They then used their adapted TKL method to generate data with model error in a Monte Carlo simulation study to evaluate the empirical characteristics of their test statistic. Although their adapted TKL method has not yet been used in any other published simulation studies, it demonstrates that it is possible to extend the TKL method beyond the simple common factor model to other types of covariance structure models.

1.1.2 The Cudeck and Browne (CB; 1992) Method

An alternative to the TKL method for modeling imperfect model fit in simulation studies was described by Cudeck and Browne (1992). These authors agreed with Tucker et al. (1969) that no simple factor analysis model is likely to fit exactly in the population and that Monte Carlo simulation studies conducted to evaluate statistical methods should incorporate model error to test robustness against imperfect model fit. However, Cudeck and Browne wanted a model-error method that was more flexible than the TKL method and that allowed the

[3] Hong (1999) was cited twelve times as of March 11, 2021, according to Web of Science.

user to specify the desired discrepancy function value to control the amount of model error. Therefore, Cudeck and Browne proposed a new method of generating data from models with imperfect fit, building on the work of Tucker et al. (1969). They developed their new approach to satisfy three desiderata: First, that the approach is general to covariance structure models and not only factor analysis models; Second, that the approach allows the pre-specification of the amount of model error in terms of the maximum likelihood or least squares discrepancy function value, $F(\boldsymbol{\Sigma}, \boldsymbol{\Omega}(\boldsymbol{\gamma})) = \delta$; Third, that the minimizer of the discrepancy function is a specified vector of model parameters, $\boldsymbol{\gamma} = \boldsymbol{\gamma_0}$.

The Cudeck and Browne method (referred to as CB hereafter) works as follows. A $p \times p$ population covariance matrix is defined as the sum

$$\boldsymbol{\Sigma} = \boldsymbol{\Omega}(\boldsymbol{\gamma}) + \mathbf{E}, \tag{1.7}$$

where $\boldsymbol{\Omega}(\boldsymbol{\gamma})$ is a matrix-valued function of a vector of parameters, $\boldsymbol{\gamma}$, and $\mathbf{E}$ is a $p \times p$ symmetric matrix such that Equation (1.7) is positive definite. Moreover, let $\boldsymbol{\Sigma}_0 = \boldsymbol{\Omega}(\boldsymbol{\gamma}_0) + \mathbf{E}$ be the population covariance matrix for a particular vector of model parameters, $\boldsymbol{\gamma}_0$. The CB method works by finding an $\mathbf{E}$ matrix such that the discrepancy function $F(\boldsymbol{\Sigma}_0, \boldsymbol{\Omega}(\boldsymbol{\gamma}_0))$ is minimized at $\boldsymbol{\gamma}_0$ and the minimum is equal to a pre-specified value, δ. Cudeck and Browne (1992) considered discrepancy functions of the form

$$F(\boldsymbol{\Sigma}, \boldsymbol{\Omega}(\boldsymbol{\gamma})) = \frac{1}{2} \operatorname{tr}[\mathbf{Z}^{-1}(\boldsymbol{\Sigma} - \boldsymbol{\Omega}(\boldsymbol{\gamma}))^2], \tag{1.8}$$

where the fixed matrix $\mathbf{Z}$ does not depend on $\mathbf{E}$. Note that when $\mathbf{Z} = \mathbf{I}_p$, Equation (1.8) is the discrepancy function for ordinary least squares. The discrepancy function for ordinary-theory maximum likelihood can be written as

$$F_{\mathrm{ML}}(\boldsymbol{\Sigma}, \boldsymbol{\Omega}(\boldsymbol{\gamma})) = \ln|\boldsymbol{\Omega}(\boldsymbol{\gamma})| - \ln|\boldsymbol{\Sigma}| + \operatorname{tr}[\boldsymbol{\Sigma}\boldsymbol{\Omega}(\boldsymbol{\gamma})^{-1}] - p, \tag{1.9}$$

where $|\mathbf{\Sigma}|$ is the determinant of $\mathbf{\Sigma}$. Cudeck and Browne (1992) showed that the minimizer of Equation (1.8) is the same as the minimizer of Equation (1.9) when $\mathbf{Z} = \mathbf{\Omega}(\boldsymbol{\gamma}_{\text{ML}})$, where $\boldsymbol{\gamma}_{\text{ML}}$ is the minimizer of the maximum likelihood discrepancy function in Equation (1.9). Thus, Equation (1.8) is a general form of the discrepancy function that is equivalent to either the least squares discrepancy function or the maximum likelihood discrepancy function, depending on the value of $\mathbf{Z}$.

Recall that one objective of the CB method is to ensure that the discrepancy function $F(\mathbf{\Sigma}_0, \mathbf{\Omega}(\boldsymbol{\gamma}))$ is minimized when $\boldsymbol{\gamma} = \gamma_0$. To do this, it is necessary to find an $\mathbf{E}$ matrix such that the gradient $\partial F(\mathbf{\Sigma}_0, \mathbf{\Omega}(\boldsymbol{\gamma}))/\partial \boldsymbol{\gamma} = \mathbf{0}$. Cudeck and Browne (1992) showed the gradient can be written as

I am unclear where B came from

$$\frac{\partial F(\mathbf{\Sigma}_0, \mathbf{\Omega}(\boldsymbol{\gamma}))}{\partial \boldsymbol{\gamma}} = \mathbf{B}'\tilde{\mathbf{e}} \tag{1.10}$$

where $\mathbf{B}$ is a $p \times p$ symmetric matrix that depends on both $\mathbf{Z}$ and $\dot{\mathbf{\Sigma}}_i = [\partial \mathbf{\Sigma}(\gamma)/\partial \gamma_i]$, and $\tilde{\mathbf{e}} = \text{vecs}\,[\mathbf{\Sigma}_0 - \mathbf{\Omega}(\gamma)]$. The vecs operator is defined such that for a symmetric matrix, $\mathbf{A}$, $\text{vecs}\,\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{22} & a_{13} & \ldots & a_{pp} \end{bmatrix}'$. Put another way, the vecs $\mathbf{A}$ operator returns a vector of the stacked upper-diagonal elements of $\mathbf{A}$ (including the diagonal). Written in this form, the gradient can be set equal to the null vector, $\mathbf{B}'\tilde{\mathbf{e}} = \mathbf{0}|_{\gamma=\gamma_0}$ and solved for $\tilde{\mathbf{e}} = \text{vecs}\,\tilde{\mathbf{E}}$. To find a suitable $\tilde{\mathbf{e}}$, let $\mathbf{y}$ be a non-null $\frac{1}{2}(p^2+p)\times 1$ vector. Then the difference

Do you understand this step. I dont

$\tilde{\mathbf{e}} = \mathbf{y} - \mathbf{B}(\mathbf{B}'\mathbf{B})^{-1}\mathbf{B}'\mathbf{y}$ gives an $\tilde{\mathbf{e}}$ vector such that $\mathbf{B}'\tilde{\mathbf{e}} = \mathbf{0}|_{\gamma=\gamma_0}$.

After finding $\tilde{\mathbf{E}}$ such that $F(\mathbf{\Sigma}_0, \mathbf{\Omega}(\boldsymbol{\gamma}))$ is minimized at $\boldsymbol{\gamma} = \gamma_0$, the next step is to ensure that the minimum is equal to a specified value, δ. The population Root Mean Square Error of Approximation (RMSEA; Steiger, 1990) value is related to the objective function value by $\varepsilon = \sqrt{F(\mathbf{\Sigma}_0, \mathbf{\Omega}(\boldsymbol{\gamma}))/df}$, where ε and df denote the RMSEA and model degrees of freedom, respectively. Therefore, δ is generally selected to produced a desired RMSEA value such that $\delta = \varepsilon^2 df$. Cudeck and Browne defined the error matrix as $\mathbf{E} = \kappa\tilde{\mathbf{E}}$, where κ is a scaling term that is chosen so that the value of the objective function at its minimum is equal to δ. When

How large is too large

$\kappa = 0$, $\mathbf{\Sigma} = \mathbf{\Omega}$, whereas larger values of κ lead to larger discrepancy function values. Cudeck and Browne (1992) furthermore proved that $\boldsymbol{\gamma}_0$ is the global minimizer of $F(\mathbf{\Sigma}_0, \mathbf{\Omega}(\boldsymbol{\gamma}))$ as long as κ is not too large (pp. 360–361). Thus, their second desideratum (i.e., that $\boldsymbol{\gamma}_0$ be the minimizer of the objective function) and third desideratum (i.e., that the value of the discrepancy function at its minimum is δ) are both satisfied.

The CB method is appealing for use in simulation studies for a number of reasons. First, it allows the user to specify a desired RMSEA value. Second, unlike the TKL method, the CB method does not have any tuning parameters that need to be chosen (other than the target RMSEA). Third, the CB method is easily extendable to many types of covariance structure models. Likely because of these advantages, the CB method has been used in a relatively large number of Monte Carlo simulation studies (Lai, 2020a, 2018, 2020b, 2020c, 2019, 2020d; Lai & Zhang, 2017; Montoya & Edwards, 2020; Xia, 2021).

Although it is appealing for simulation work, the CB method carries with it an assumption about the nature of model error that might or might not be considered reasonable for empirical data sets. Recall that a requirement of the CB method is that the discrepancy function $F(\mathbf{\Sigma}_0, \mathbf{\Omega}(\boldsymbol{\gamma}_0))$ is minimized when $\boldsymbol{\gamma} = \boldsymbol{\gamma}_0$. When a model is mis-specified, there does not exist any $\boldsymbol{\gamma}_0$ such that $\mathbf{\Sigma}_0 = \mathbf{\Omega}(\boldsymbol{\gamma}_0)$. However, the maximum likelihood parameter estimate $\hat{\boldsymbol{\gamma}}_{\mathrm{ML}}$ is still consistent toward the minimizer of the maximum likelihood discrepancy function under mild regularity conditions (Shapiro, 1983, Theorem 5.4; 2007, sec. 5.3; Wu & Browne, 2015a). Moreover, if a sample covariance matrix $\mathbf{S}$ is an unbiased estimator of $\mathbf{\Sigma}_0$, then the expected value of $\mathbf{S}$ is $\mathbf{\Sigma}_0$. Taken together, these properties indicate that maximum likelihood parameter estimates converges to $\boldsymbol{\gamma}_0$ as $N \to \infty$ under the CB framework.

This result reflects the view that the "true" population parameter values are simply the parameter values obtained by fitting a model to $\mathbf{\Sigma}$ using a particular discrepancy function. In this view, the discrepancy between the $\mathbf{\Sigma}$ and the model-implied covariance matrix obtained from analyzing $\mathbf{\Sigma}$ (i.e., $\hat{\mathbf{\Omega}}$) is of primary interest, not the discrepancy between $\mathbf{\Sigma}$ and the implied covariance matrix for some ideal model ($\mathbf{\Omega}$). Because the CB method ensures that

You should spell out more clearly the definitions of \Sigma and \Sigma_0

We need to discuss this

Perhaps make a brief table with a list of key terms and parameters

γ_0 is a minimizer of the discrepancy function, $\hat{\mathbf{\Omega}} = \mathbf{\Omega}$. Thus, in this view it makes sense that the population parameters (γ_0) will be perfectly recovered as $N \to \infty$.

1.1.3 The Wu and Browne (WB; 2015) Method

A third model-error method was introduced by Wu and Browne (2015a) and is unique among the three approaches because it represents model error as a random effect rather than as a fixed quantity. Moreover, the TKL and CB model-error methods were developed for use in simulation studies. In contrast, the WB method was motivated by Wu and Browne's development of a method for estimating confidence intervals for model parameters, taking into account variability due to model error.

Before describing Wu and Browne's (2015a) estimation method, it will be useful to define two important terms. The term *model discrepancy*, as used by Wu and Browne (2015a), corresponds to what has been previously referred to as model error (i.e., the difference between the error-perturbed and model-implied population covariance matrices). The term *adventitious error* is used to describe the process underlying model discrepancy (Wu & Browne, 2015a). In Wu and Browne's (2015a) conceptualization, model discrepancy arises from differences between two populations: an operational population from which the observed sample is representative, and a theoretical general population. The theory (as represented by the covariance structure model) is hypothesized to hold exactly in the theoretical general population, but not in the operational population (Wu & Browne, 2015a, 2015b). The general population might also be referred to as the *ideal* population. This is not the terminology used by Wu and Browne (2015a), but more clearly reflects their view of the general population as being "…contained in the Platonic Aether" (attributed to Michael C. Edwards in Wu & Browne, 2015b, p. 620) as opposed to "a mundane collection of people that can be reached through more complicated designs" (Wu & Browne, 2015b, p. 621).

Wu and Browne (2015a) argued that the discrepancy between the operational and ideal population models is a source of variation that is not reflected in traditional methods for fitting covariance structures to sample covariance matrices. These approaches generally fit a covariance structure $\boldsymbol{\Omega}(\boldsymbol{\gamma})$ to a sample covariance matrix $\mathbf{S}$ by minimizing a discrepancy function $F(\mathbf{S}, \boldsymbol{\Omega}(\boldsymbol{\gamma}))$. For instance, a common choice of discrepancy function is the maximum likelihood discrepancy function,

$$F_{\mathrm{ML}}(\mathbf{S}, \boldsymbol{\Omega}(\boldsymbol{\gamma})) = \ln|\boldsymbol{\Omega}(\boldsymbol{\gamma})| - \ln|\mathbf{S}| + \mathrm{tr}[\mathbf{S}\boldsymbol{\Omega}(\boldsymbol{\gamma})^{-1}] - p. \tag{1.11}$$

Note that Equation (1.11) is equivalent to Equation (1.9) with $\mathbf{S}$ substituted for $\boldsymbol{\Sigma}$. The discrepancy function is called "maximum likelihood" because minimizing the function is equivalent to maximizing the likelihood function for the Wishart distribution, $\mathrm{W}_p(\boldsymbol{\Omega}(\boldsymbol{\gamma})/n, n)$. This Wishart distribution is the sampling distribution of $\mathbf{S}$ under the assumption that $\boldsymbol{\Sigma} = \boldsymbol{\Omega}(\boldsymbol{\gamma})$ for some $\boldsymbol{\gamma} = \boldsymbol{\gamma}_0$, and the assumption of normality (Wu & Browne, 2015a). The asymptotic distribution of the maximum likelihood parameter estimate, $\hat{\boldsymbol{\gamma}}_{\mathrm{ML}}$ can then be derived (e.g., Shapiro, 2007, Theorem 5.5, p. 249), and confidence intervals for parameters can be constructed (Jöreskog, 1969).

Unfortunately, using maximum likelihood estimation requires assumptions that are unlikely to hold in many applied settings. First, the maximum likelihood discrepancy function is derived using normal distribution theory and can be sensitive to violations of normality (Browne & Shapiro, 1988). However, maximum likelihood estimates can also be derived using a model that does not require any distributional assumptions (Howe, 1955; Mulaik, 2009, pp. 214–215) and normal theory methods have been shown to be robust under certain circumstances (Browne & Shapiro, 1988). A more troublesome issue highlighted by Wu and Browne (2015a) is that using the maximum likelihood discrepancy function defined in Equation (1.11) implicitly assumes that the theorized model holds perfectly in the population and that all differences between the sample and population covariance matrices are attributable to sampling error (Briggs & MacCallum, 2003; Wu & Browne, 2015a). Because adventitious error is not considered as a source of random variation, the variability estimates (and therefore confidence intervals) of parameter estimates and test statistics will be underestimated when using the traditional approach to model estimation (Wu & Browne, 2015a).

Unlike the traditional approach, Wu and Browne's (2015a) estimation method accounts for variability due to adventitious error by modeling adventitious error as a random effect with a distribution. The estimated dispersion parameter of this distribution can then be used as a measure of model misspecification. The statistical basis for their model is as follows. First, under the assumption of normality, the sample covariance matrix $\mathbf{S}$ has a Wishart distribution such that

$$(\mathbf{S}|\mathbf{\Sigma}) \sim \mathrm{W}_p(\mathbf{\Sigma}/n, n), \tag{1.12}$$

where n is the degrees of freedom. As opposed to the traditional method where $\mathbf{S}$ is assumed to be an unbiased estimator of the model-implied population covariance matrix $\mathbf{\Omega}(\boldsymbol{\gamma})$, here $\mathbf{S}$ is instead assumed to be an unbiased estimator of the error-perturbed population covariance matrix, $\mathbf{\Sigma}$. $\mathbf{\Sigma}$ is then assumed to follow an inverse-Wishart distribution such that

$$(\mathbf{\Sigma}|\mathbf{\Omega}, m) \sim \mathrm{W}_p^{-1}(m\mathbf{\Omega}, m), \tag{1.13}$$

where m is a continuous precision parameter such that $m > p-1$ (Wu & Browne, 2015a). Wu and Browne (Wu & Browne, 2015a) also introduced the inverse of this precision parameter, $v = 1/m \in [0, (p-1)^{-1})$, which gives the dispersion of the adventitious error and can also be interpreted as a measure of misspecification. In particular, Wu and Brown (2015a, p. 580) show that $v \approx \varepsilon^2$, where ε denotes the RMSEA. Because v has an upper bound of $1/(p-1)$, Wu and Browne suggested using $\sqrt{\tilde{v}} = (m - p + 1)^{-1/2}$ as the criterion of model admissibility. Specifically, they stated that $\sqrt{\tilde{v}} = 0.05$ is indicative of good model fit, values of $\sqrt{\tilde{v}}$ between 0.05 and 0.08 are indicative of acceptable model fit, and values above 0.08 are indicative of unacceptable model fit.

Wu and Browne's (2015a) model therefore has two parameters, $\boldsymbol{\gamma}$ and v, that require

This is a vector of parameters so
perhaps two sets of parameters?

estimation. They showed that these parameters can be estimated by maximizing the likelihood function corresponding to the marginal distribution of the sample covariance matrix, $(\mathbf{S}|\boldsymbol{\Omega}, m)$, with the population covariance matrix $\boldsymbol{\Sigma}$ integrated out. Wu and Browne's choice of a conjugate distribution for $\boldsymbol{\Sigma}$ means that the marginal distribution of the sample covariance matrix follows a Type II matrix-variate beta distribution (Gupta & Nagar, 2000, Chapter 5),

$$(\mathbf{S}|\boldsymbol{\Omega}, m) \sim \mathbf{B}_p^{\mathrm{II}}\left(\frac{n}{2}, \frac{m}{2}, \frac{m}{n}\boldsymbol{\Omega}\right), \tag{1.14}$$

from which the probability density function and log-likelihood functions can be obtained (see Wu & Browne, 2015a for additional details). The parameter estimate obtained by maximizing the beta marginal likelihood (or equivalently, minimizing the negative twice beta log-likelihood) is referred to as the maximum beta likelihood estimate (MBLE).

In addition to showing how to estimate $\boldsymbol{\gamma}$ and v, Wu and Browne (2015a) derived sampling distributions and confidence intervals for $\hat{\boldsymbol{\gamma}}$ and $\hat{v}$. An important advantage of Wu and Browne's method over more traditional estimation methods is that their confidence

Assuming that that form of model error is reasonable

intervals for $\hat{\boldsymbol{\gamma}}$ account for the amount of model error indicated by $\hat{v}$. They provided simulation evidence showing that confidence interval coverage rates for both estimated parameters were close to their nominal values, at least when n and m were relatively large and when $(\boldsymbol{\Sigma}|\boldsymbol{\Omega}, m) \sim \mathrm{W}_p^{-1}(m\boldsymbol{\Omega}, m)$. Moreover, they also showed that coverage rates for 95% confidence intervals estimated using their method were much closer to the nominal levels compared to confidence intervals estimated using the traditional approach.

Thus far, this section has focused on Wu and Browne's (2015a) method for estimating parameters and constructing confidence intervals for covariance structure models. However, as noted previously, Wu and Browne's model assumes that a covariance matrix with imperfect model fit is a random sample from an inverse-Wishart distribution, as shown in Equation (1.13). Given a model-implied population covariance matrix $\boldsymbol{\Sigma}$ and some chosen

Still not sure why they think
their model makes sense. Do they
give any hypothetical examples of
the types of subpopulations they
had in mind?   Can we anchor this
in psychology/sociology/etc rather
than in just math?

value of the precision parameter m, covariance matrices with imperfect fit can be easily sampled from an appropriate inverse-Wishart distribution using statistical software such as R (R Core Team, 2021), MATLAB (The Mathworks, 2019), or Julia (Bezanson et al., 2017). The degree of model error (in terms of RMSEA) can be controlled to some extent by choosing an appropriate value of m, which Wu and Browne showed has a relationship with RMSEA such that $\sqrt{1/m} = \sqrt{v} \approx \varepsilon$ (Wu & Browne, 2015a, Eq. 16, p. 580).

1.2 Population Model Fit Indices

This section is important but confusing. You really need a list of definitions for Simga Omega etc

In addition to being able to generate error-perturbed covariance matrices, it is important to be able to quantify the lack-of-fit between an error-perturbed population covariance matrix and a model-implied covariance matrix. Before describing the various model-fit indices that have been used to quantify model error, it will be helpful to first describe two different perspectives of model fit as it relates to model error. In the first, model error is quantified

Remind the reader what these symbols stand for.

as the lack-of-fit between $\mathbf{\Sigma}$ and $\mathbf{\Omega}$. In the second, model error is quantified as the lack-of-fit between $\mathbf{\Sigma}$ and $\hat{\mathbf{\Omega}}$, the implied covariance matrix from fitting a particular model to $\mathbf{\Sigma}$. These two perspectives are equivalent in the special case when $\hat{\mathbf{\Omega}} = \mathbf{\Omega}$, which occurs when the CB method is used (because $\boldsymbol{\gamma}_0$ is required to be a minimizer of the chosen discrepancy function). For the purposes of this dissertation, I will focus primarily on the first model fit perspective. Unless specified, all model fit indices reflect the first perspective, indicating the lack-of-fit between between $\mathbf{\Sigma}$ and $\mathbf{\Omega}$. Where I discuss a model fit index indicating the lack-of-fit between $\mathbf{\Sigma}$ and $\hat{\mathbf{\Omega}}$, a "$\hat{\mathbf{\Omega}}$" subscript is used to make it clear that model fit is being considered from the second perspective. I.e., I use "RMSEA" to indicate the root mean square error of approximation between $\mathbf{\Sigma}$ and $\mathbf{\Omega}$, and "RMSEA$_{\hat{\mathbf{\Omega}}}$" to indicate the root mean square error of approximation between $\mathbf{\Sigma}$ and $\hat{\mathbf{\Omega}}$.

As described in the previous section, the population RMSEA value is often used to describe the lack-of-fit between population covariance matrices due to model error. Other fit indices such as the Tucker-Lewis Index (TLI; Tucker & Lewis, 1973), the Comparative Fit Index (CFI; Bentler, 1990), and the Standardized Root Mean Square Residual (SRMR; Hu & Bentler, 1999) have also be used for this purpose. Although far from an exhaustive list, these fit indices are among the most popular in the psychometric literature and can be divided into two general categories (Hu & Bentler, 1999). First, RMSEA and SRMR are absolute fit indices that report model fit in terms of the difference between a particular covariance matrix and a model-implied covariance matrix, without using a reference or baseline model for comparison. In the population context, absolute fit indices answer the question, "How different is $\mathbf{\Sigma}$ to $\mathbf{\Omega}$?" On the other hand, the TLI and CFI are incremental fit indices, which reflect the improvement of fit of a particular model compared to a (restricted, nested) baseline model. The null or independence model is often used as the baseline model, answering the question, "How well is my model doing, compared with the worst model that there is?" (Miles & Shevlin, 2007, p. 870).

Not all baseline models assume variable independence

RMSEA, CFI, TLI, and SRMR are generally defined in terms of the sample covariance matrix, but can be expressed in their population form (i.e., with reference to $\mathbf{\Sigma}$) as follows. Let F_h and F_b be the minimized discrepancy function values for the hypothesized and baseline models (respectively) at the population level. Furthermore, let df_h and df_b be the degrees of freedom for the hypothesized and baseline models. Then the population RMSEA value is given by

$$\text{RMSEA} = \varepsilon = \sqrt{\frac{F_h}{df_h}}, \tag{1.15}$$

the population CFI value is given by

$$\text{CFI} = 1 - \frac{F_h}{F_b}, \tag{1.16}$$

and the population TLI value is given by

$$\text{TLI} = 1 - \frac{F_h/df_h}{F_b/df_b}. \tag{1.17}$$

Unlike RMSEA, CFI, and TLI, the population SRMR does not depend on discrepancy function values and is given by

$$\text{SRMR} = \sqrt{\frac{1}{p(p+1)/2} \sum_{i \leq j} \left(\frac{\sigma_{ij} - \omega_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}} \right)^2}, \tag{1.18}$$

where p is the number of observed variables, and σ_{ij} and ω_{ij} are the i, jth elements of $\mathbf{\Sigma}$ and $\mathbf{\Omega}$, respectively (Maydeu-Olivares, 2017; Pavlov et al., 2021; Xia & Yang, 2019). Note that when $\mathbf{\Sigma}$ and $\mathbf{\Omega}$ are constrained to be correlation matrices with unit diagonals, the correlation root mean residual (CRMR; Kenneth A. Bollen, 1989; Ogasawara, 2001) is appropriate:

$$\text{CRMR} = \sqrt{\frac{1}{p(p-1)/2} \sum_{i<j} (\sigma_{ij} - \omega_{ij})^2}. \tag{1.19}$$

Although all of the fit indices discussed here have been used to quantify model error in simulation studies (e.g., Lai, 2020b, 2020c; Lai & Zhang, 2017; Xia et al., 2016), the population RMSEA is perhaps the most widely-used. Indeed, many simulation studies using the TKL, CB, or WB model-error methods have grouped simulated covariance matrices into model fit categories based on highly-cited "rule-of-thumb" RMSEA cutoff values. For instance, Lorenzo-Seva and Ferrando (2020a) used RMSEA values of 0.065 to represent models with fair fit, citing Browne and Cudeck (1992). Myers et al. (2015) used RMSEA values of 0.025, 0.065, and 0.090 to categorize models as having very good fit, fair fit, and poor fit (respectively), citing Browne and Cudeck (1992), Steiger (1989), and Jackson (2009). According to Lai and Green (2016, p. 220):

> The most widely used cutoffs for RMSEA yield the following interpretations: (a) Values less than .05 (Browne & Cudeck, 1992) or .06 (Hu & Bentler, 1999) suggest 'good' fit; (b) values between .05 and .10 suggest 'acceptable' fit (Browne

& Cudeck, 1992; MacCallum, Browne, and Sugawara, 1996); and (c) values larger than .10 suggest 'bad' fit (Browne & Cudeck, 1992).

However, rule-of-thumb cutoffs based on other common fit indices such as CFI can lead to conclusions about model fit that disagree with those indicated by RMSEA.

The problem of disagreement among fit indices was explored in detail by Lai and Green (2016), who derived necessary and sufficient conditions for disagreement between CFI and RMSEA (when used with common cutoff values). For instance, Lai and Green (2016) showed that "good" RMSEA values and "bad" CFI values (i.e., RMSEA ≤ 0.05 and CFI ≤ 0.90) will occur when $df_h \leq 400F_b \leq 10df_h$, where df_h denotes the model degrees of freedom and F_b is the discrepancy function value for the baseline null (independence) model. Lai and Green (2016) also showed that $F_b = -\ln|\mathbf{S}|$ for an observed correlation matrix and that $\ln|\mathbf{S}|$ will be large (and $-\ln|\mathbf{S}|$ small) when the elements of $\mathbf{S}$ are close to zero. Thus, models will have "good" RMSEA but "bad" CFI when the elements of $\mathbf{S}$ are small (but large enough that $F_b > df_h/400$) and when the model degrees of freedom value is large (but small enough that $df_h < 400F_b$; Lai and Green, 2016). Lai and Green (2016) concluded that disagreement between CFI and RMSEA does not reflect a failure on the part of either fit index. Rather, disagreement occurs because the two fit indices evaluate model fit from different perspectives and the cutoffs used to make qualitative judgments about model fit are arbitrary.

Unfortunately, disagreement between fit indices is a problem for researchers attempting to simulate covariance matrices with some particular qualitative level of model fit. Consider, for instance, a simulated covariance matrix with RMSEA $= 0.04$ and CFI $= 0.78$.[4] An RMSEA value of 0.04 indicates good model fit based on most rule-of-thumb cutoff values (Browne & Cudeck, 1992). However, CFI values less than 0.90 are seldom considered to indicate acceptable fit (Lai & Green, 2016). How should this covariance matrix (and others like it) be categorized in a simulation study? No straightforward answer is apparent.

[4]This is not just a hypothetical example. Simulated correlation matrices leading to similar RMSEA and CFI values were encountered in simulations conducted by Kracht and Waller (2020).

The essential problem, as stated by Kline (2011), is that "*there is no such thing as a magical, single-number summary that says everything worth knowing about model fit*" (p. 193). Therefore, a common recommendation is to report multiple fit indices to get a more complete picture of model fit (Kline, 2011; Lai & Green, 2016)

Despite the recommendation to report multiple fit indices, only a handful of simulation studies that incorporate error-perturbed covariance matrices report fit indices other than RMSEA (Kracht & Waller, 2022; Lai, 2020a, 2018, 2020b, 2019; Lai & Green, 2016; Lai & Zhang, 2017). The reliance on a single model fit index (usually RMSEA) when generating error-perturbed covariance matrices can be explained by the characteristics of the TKL, CB, and WB model-error methods.

In the case of the TKL method, little is known about which values of the TKL parameters (ϵ and ν_{e}) are reasonable for generating data that are representative of empirical data. Without clear guidance about which parameter values to use, researchers using the TKL method in simulation studies have often relied on RMSEA values to determine whether resulting correlation matrices were representative of empirical data sets. For instance, Briggs and MacCallum (2003) used the TKL method to generate data sets they categorized as having either good or moderate model fit. In their simulation study, error-perturbed covariance matrices were retained or rejected based on their RMSEA values. A matrix was retained if the RMSEA resulting from a maximum likelihood factor analysis fell within the range .030–.049 for the good fit condition and within the range .070–.089 for the moderate model fit condition.

Perhaps cite the new Hoang and Niels paper

Other researchers have also used RMSEA as a measure of model misfit when using the TKL method in Monte Carlo simulation studies (Gnambs & Staufenbiel, 2016; Kracht & Waller, 2022; Lorenzo-Seva & Ferrando, 2020a; Myers et al., 2015; Preacher et al., 2013; Preacher & MacCallum, 2002). One difficulty with this approach is that it can be challenging to choose values of ϵ and ν_{e} that lead to desired RMSEA values while varying factor model characteristics (e.g., number of factors, factor salience, factor correlations, etc.). Manually choosing parameters that routinely lead to desired RMSEA values and desired values of another fit statistic (such as CFI) is even more challenging. Thus, researchers who use the TKL method in simulation studies generally select the values of ϵ and ν_e based on only RMSEA.

Researchers who choose to instead generate error-perturbed covariance matrices using the CB method often use RMSEA as a measure of model misfit because the CB method allows them to specify a desired discrepancy function value (and therefore a desired RMSEA value). However, the CB method has no other user-specified parameters than can be changed to influence model fit indices that measure other aspects of model fit (such as CFI). Lai and Green (2016) used the CB method to generate covariance matrices with specified RMSEA and CFI values, but did so by updating the factor loadings and uniqueness variances in the population model while holding RMSEA constant. Although this approach is effective, it has the significant drawback of not allowing researchers to systematically vary model parameters across experimental conditions.

Similar to the CB method, the WB method allows the user to choose a desired RMSEA value but does not provide an easy way to specify a desired a CFI value (or any alternative fit index). The WB method is also less precise than the CB method in the sense that it only allows the user to specify a distribution for the model-perturbed covariance matrices and thus does not guarantee that the RMSEA value for a simulated covariance matrix will be very close to the target value.

Chapter 2

A Method for Generating Error-Perturbed Covariance Matrices with Specified Levels of Model Fit

Given that model fit cannot be fully described by a single fit index, it is desirable to have a model fit method that would allow for the simultaneous specification of multiple fit indices (e.g., RMSEA and CFI). To date, RMSEA has been used almost exclusively to quantify the amount of model error introduced by a model-error method, and is the only fit index that can be specified in advance when using the CB and WB methods (Briggs & MacCallum, 2003; Cudeck & Browne, 1992; Wu & Browne, 2015a).

In this section, I propose a procedure based on the TKL method that uses optimization to find parameter values that lead to error-perturbed covariance matrices with RMSEA and CFI values that are close to user-specified target values. RMSEA and CFI are used because they are among the most commonly-used model fit indices and reflect different aspects of model fit (absolute and incremental model fit; Kline, 2011). However, in theory the procedure could be extended to work with any two fit indices.[1] The proposed procedure also allows the user to impose constraints on the number of large minor factor loadings to ensure a clear delineation between major and minor factors in the TKL framework. Because

[1] In fact, the procedure could easily be extended to include three or more fit indices. However, the marginal benefit of including additional fit indices is unlikely to be large and could make optimization more difficult.

the optimization procedure allows for RMSEA and CFI targets and constraints on factor loadings, I will refer to it as the "multiple-target optimization method" or "the multiple-target method."

The proposed procedure uses the limited memory Broyden-Fletcher-Goldfarb-Shanno optimization algorithm with box constraints (L-BFGS-B; Byrd et al., 1995) to minimize the objective function

Why bold?

$$G(\nu_{\text{e}}, \epsilon) = b_1 \frac{[\varepsilon - \varepsilon_T]^2}{\varepsilon_T^2} + b_2 \frac{[\text{CFI} - \text{CFI}_T]^2}{(1 - \text{CFI}_T)^2} + \mathbf{1}_{\boldsymbol{W}} \lambda, \tag{2.1}$$

where $0 \leq \nu_{\text{e}} \leq 1$, $0 \leq \epsilon \leq 1$, b_1 and b_2 are user-specified weights constrained to sum to one. Setting $b_1 = b_2 = 0.5$ places equal weight on RMSEA and CFI, whereas unequal weights can be used to indicate a preference for one fit index over the other. If either weight is set to zero, the corresponding fit index has no effect on optimization. Thus, the optimization procedure seeks to find values of ν_{e} and ϵ such that the weighted sum of the mean squared error between the observed and target RMSEA and the mean squared error between the observed and target CFI is minimized.

The right-most term in Equation (2.1) consists of a user-defined penalty, λ, and an indicator function, $\mathbf{1}_{\boldsymbol{W}}$. The indicator function $\mathbf{1}_{\boldsymbol{W}}$ is equal to one whenever a user-specified number of (absolute) minor factor loadings are greater than some threshold value for any minor factor, and zero otherwise. The addition of this penalty term attempts to ensure a clear delineation between major and minor factors. For instance, a user could specify that no more than two minor factor loadings should be greater than 0.3 in absolute value. If the penalty term is set to some large value (e.g., $\lambda = 100$), TKL parameters that lead to models with too many large minor factor loadings will be heavily penalized.

Empirical testing of the multiple-target method showed that some combinations of target fit index values and constraints on the $\mathbf{W}$ matrix sometimes led to non-convergence when using the L-BFGS-B algorithm. In particular, non-convergence was observed when using relatively high target RMSEA values and relatively low target CFI values, and when there were only a small number of factors. Non-convergence can often be remedied by restarting optimization with different starting parameter values. If non-convergence still occurs after trying multiple different starting values, global optimization using a genetic algorithm (GA; Holland, 1975) can be used to minimize the objective function in Equation (2.1).

GAs take inspiration from biological evolutionary processes and natural selection to find candidate solutions that have a high level of fitness (i.e., that lead to high/low objective function values during maximization/minimization). Many variations of GAs have been proposed (Scrucca, 2013), but the procedure for most GAs can be generally described as follows. First, a random set of candidate solutions is generated and the fitness (i.e., objective function value) for each solution is evaluated. Next, a pair of "parent" solutions are selected (with replacement) from the set such that solutions with higher fitness have a higher selection probability. The parent solutions then produce two offspring with user-defined crossover and mutation probabilities. If crossover occurs, the children are formed as combinations of their parents using some crossover function. If crossover does not occur, the children are formed as exact copies of their parents. Similarly, if mutation occurs, the child solution is randomly perturbed. This process continues until there are as many children in the new generation as there were parents in the previous generation. Once the new generation is formed, selection occurs again and the process continues until a fixed number of generations have passed or until some stopping criterion is reached (Mitchell, 1996; Scrucca, 2013).

Although GAs work well for many problems where derivative-based methods have difficulty (e.g., when the objective function is not smooth or when there are local optima; Scrucca, 2013), a downside is that they can be relatively slow compared to derivative-based optimization methods when applied to more "well-behaved" optimization problems. Therefore, the proposed optimization procedure first attempts to use the L-BFGS-B method to find a solution (with multiple random starts if convergence does not occur). If convergence still does not occur after multiple random starts using L-BFGS-B, a GA is used to find a solution. The advantage of this approach is that the procedure quickly produces a solution when the user-specified values make the problem well-suited for derivative-based optimization. If the derivative-based optimization fails, the procedure can still find a solution using the GA, albeit somewhat more slowly.

It is important to note that the multiple-target method will not necessarily produce solutions with RMSEA and CFI values that are exactly equal to the target values. For some models, it can be difficult—potentially impossible—to obtain particular combinations of RMSEA and CFI values (e.g., when modeling correlation matrices with many items, a low ϵ_T value and a high CFI_T value). Moreover, the method is not guaranteed to find solutions that are global minima. Thus, users should check solutions to make sure that the model fit values are sufficiently close to the target values for their purposes. These issues will be investigated more thoroughly in the simulation study that is described in the next section.

This section as great. Overall, your writing is also great. Nice job

Chapter 3

Methods

The previous section only discussed TKL. I am not sure how you controlled CFI in CB and WB?

I conducted a simulation study to investigate and compare characteristics of the multiple-target TKL model-error method, the CB model-error method, and the WB model-error method. Moreover, the simulation study was also conducted to evaluate the effectiveness of the proposed multiple-target TKL method. Two questions were of particular interest. First, how did the model-error methods differ in terms of the characteristics of the error-perturbed covariance matrices they produced? Specifically, how did the model-error methods differ in terms of model fit statistics (e.g., RMSEA, SRMR, CFI, TLI) for the error-perturbed covariance matrices they generated? Second, how well did the model-error methods work in terms of producing error-perturbed covariance matrices with RMSEA (and CFI) values that were close to the target values? Answering these questions should be helpful to researchers who are planning Monte Carlo simulation studies involving common factor models and would like to understand how their choice of model-error method is likely to affect the characteristics of simulated, error-perturbed covariance matrices.

In the simulation study, I compared model-error methods by generating error-perturbed covariance matrices using a variety of population models and comparing the results based on several model fit indices. For simplicity, I used covariance matrices with unit diagonals (i.e., correlation matrices). Moreover, I focused on four model fit indices: the RMSEA, the CFI, the Tucker-Lewis Index (TLI; Tucker & Lewis, 1973), and the Correlation Mean Squared

Residual (CRMR; K. A. Bollen, 1989; Ogasawara, 2001). Although many other fit indices have been proposed (see Marsh et al., 2005), the selected fit indices are among the most commonly-used and include both measures of absolute fit (RMSEA, CRMR) and incremental fit (CFI, TLI; Kline, 2011).

Because the relationship between fit indices is affected by model characteristics (Lai & Green, 2016), I included a variety of distinct population models created by systematically varying: (a) the number of major factors (Factors $\in \{1, 3, 5, 10\}$), (b) the number of items per factor (Items/Factor $\in \{5, 15\}$), (c) the correlation between factors ($\phi \in \{0, .3, .6\}$), and (d) the strength of the Salient item factor loadings (Loadings $\in \{0.4, 0.6, 0.8\}$). Each item loaded on only a single factor and factor loadings were fixed at values representing weak, moderate, and strong factor loadings, respectively (Hair et al., 2018). Examples of factor loading matrices corresponding to the weak, moderate, and strong factor loading conditions are shown in Table 3.1. The factor loading and factor correlation values used in the simulation study were intended to represent a range of values representative of values observed in empirical research. For instance, in a confirmatory factor analysis of sub-tests from the Ball Aptitude Battery believed to measure aspects of intelligence, Neuman et al. (2000) reported estimated factor loadings between 0.26 and 0.95 and factor correlations between .18 and .73.

Forming a fully-crossed design from the levels of Factors, Items/Factor, and factor correlation (ϕ) would have resulted in 4 (Factors) $\times$ 2 (Items/Factor) $\times$ 3 (ϕ) $\times$ 3 (Loadings) = 72 unique conditions. However, the 12 conditions with one factor and factor correlations greater than zero were invalid because it was nonsensical to have correlated factors for one-factor models. For the 60 remaining conditions, I computed the model-implied population correlation matrix ($\mathbf{\Omega}$) corresponding to the population common factor model indicated by the condition. To generate the $\mathbf{\Omega}$ matrices, I used the `simFA()` function in the R *fungible* library (Version 1.0.8; Waller, 2021)[1]. The `simFA()` function computes population correlation

[1] Additionally, the following R (R Core Team, 2021) packages were used either in the simulation study or to create this manuscript: *bookdown* [Version 0.24; Xie (2016); Xie (2016)], *colorspace* [Version 2.0.3; Zeileis et al. (2020); Zeileis et al. (2009); Stauffer et al. (2009);], *crayon* [Version 1.5.0; Csárdi (2022)], *devtools* [Version

(a)

F1	F2	F3
0.4	0.0	0.0
0.4	0.0	0.0
0.4	0.0	0.0
0.4	0.0	0.0
0.4	0.0	0.0
0.0	**0.4**	0.0
0.0	**0.4**	0.0
0.0	**0.4**	0.0
0.0	**0.4**	0.0
0.0	**0.4**	0.0
0.0	0.0	**0.4**
0.0	0.0	**0.4**
0.0	0.0	**0.4**
0.0	0.0	**0.4**
0.0	0.0	**0.4**

(b)

F1	F2	F3
0.6	0.0	0.0
0.6	0.0	0.0
0.6	0.0	0.0
0.6	0.0	0.0
0.6	0.0	0.0
0.0	**0.6**	0.0
0.0	**0.6**	0.0
0.0	**0.6**	0.0
0.0	**0.6**	0.0
0.0	**0.6**	0.0
0.0	0.0	**0.6**
0.0	0.0	**0.6**
0.0	0.0	**0.6**
0.0	0.0	**0.6**
0.0	0.0	**0.6**

(c)

F1	F2	F3
0.8	0.0	0.0
0.8	0.0	0.0
0.8	0.0	0.0
0.8	0.0	0.0
0.8	0.0	0.0
0.0	**0.8**	0.0
0.0	**0.8**	0.0
0.0	**0.8**	0.0
0.0	**0.8**	0.0
0.0	**0.8**	0.0
0.0	0.0	**0.8**
0.0	0.0	**0.8**
0.0	0.0	**0.8**
0.0	0.0	**0.8**
0.0	0.0	**0.8**

Table 3.1

Matrices of factor loadings corresponding to the (a) Weak, (b) Moderate, and (c) Strong factor loading conditions.

You really didnt need simfa for this first step.

matrices for common factor models by taking the model parameters (e.g., factor loadings, number of items per factor, factor correlations) as arguments and then using the equation for the common factor model (i.e., Equation (1.1)) to produce the population correlation matrix corresponding to the specified model.

Having generated model-implied population correlation matrices without model error, the next step in the simulation procedure was to generate population correlation matrices with model error ($\mathbf{\Sigma}$) for each of the 60 population factor models using the multiple-target TKL, CB, and WB model-error methods. Each model-error method was repeated

2.4.3; Wickham, Hester, et al. (2021)], *dplyr* [Version 1.0.8; Wickham, François, et al. (2021)], *gghighlight* [Version 0.3.2; Yutani (2021)], *gopherdown* [Version 0.2.1; Zieffler & Ismay (2022)], *here* [Version 1.0.1; Müller (2020)], *kableExtra* [Version 1.3.4; Zhu (2020)], *knitr* [Version 1.37; Xie (2015)], *latex2exp* [Version 0.5.0; Meschiari (2021)], *MBESS* [Version 4.8.0; Kelley (2020)], *microbenchmark* [Version 1.4.9; Mersmann (2021)], *papaja* [Version 0.1.0.9997; Aust & Barth (2020)], *parallel* [Version 4.1.3; R Core Team (2021)], *patchwork* [Version 1.1.1; Pedersen (2020)], *pbmcapply* [Version 1.5.0; Kuang et al. (2019)], *purrr* [Version 0.3.4; Henry & Wickham (2020)], *purrrgress* [Version 0.0.1; Smith (2021)], *rmarkdown* [Version 2.13; Xie et al. (2018); Xie et al. (2020)], *scales* [Version 1.1.1; Wickham & Seidel (2020)], *stringr* [Version 1.4.0; Wickham (2019)], *thesisdown* [Version 0.2.0.9000; Ismay & Solomon (2022)], *tidyr* [Version 1.2.0; Wickham (2021)], *tidyverse* [Version 1.3.1; Wickham et al. (2019)], and *xfun* [Version 0.30; Xie (2022)].

with random starting conditions 500 times for each of three target RMSEA values ($\varepsilon_T \in \{0.025, 0.065, 0.090\}$) and each of the 60 population factor models. The target RMSEA values were chosen to represent models with very good, fair, and poor model fit, following the convention used by Myers et al. (2015) and MacCallum et al. (2001). To generate these Σ matrices, I wrote R code to implement each of the model error methods described in this section. Moreover, I created an R package (*noisemaker*) that provides a convenient, simple, and unified interface for generating correlation matrices with model error.[2] R code for all of the model-error method implementations discussed in this dissertation is provided in Appendix A.1.

Although the TKL method has so far been discussed as a single model-error method, several variations of the multiple-target TKL method were included in the simulation study. Specifically, I generated error-perturbed covariance matrices for each condition with the multiple-target method using (a) only target RMSEA values (denoted as $\text{TKL}_{\text{RMSEA}}$), (b) equally-weighted target RMSEA and CFI values ($\text{TKL}_{\text{RMSEA/CFI}}$), and (c) only target CFI values (TKL_{CFI}). I used target CFI values (denoted as CFI_{T}) corresponding to the same subjective levels of model fit as the previously-mentioned target RMSEA values (Hu & Bentler, 1999; Marcoulides & Yuan, 2017), forming conditions with Very Good ($\text{RMSEA}_{\text{T}} = 0.025$, $\text{CFI}_{\text{T}} = 0.99$), Fair ($\text{RMSEA}_{\text{T}} = 0.065$, $\text{CFI}_{\text{T}} = 0.95$), and Poor ($\text{RMSEA}_{\text{T}} = 0.090$, $\text{CFI}_{\text{T}} = 0.90$) model fit. Additionally, each of the multiple-target TKL methods included a penalty term, $\lambda = 1{,}000{,}000$, to heavily penalize solutions where any minor common factor had more than two loadings greater than 0.3 in absolute value. Throughout the rest of this dissertation I often refer to the $\text{TKL}_{\text{RMSEA}}$, TKL_{CFI}, and $\text{TKL}_{\text{RMSEA/CFI}}$ methods collectively as the "TKL-based" methods because they all utilize the original TKL method, albeit with ν_{e} and ϵ values selected via optimization.

For all of the TKL-based methods, the L-BFGS-B optimization procedure was restarted

[2]The TKL and CB implementations provided in the *noisemaker* package are based on implementations of those methods provided in the *fungible* (Waller, 2021) and *MBESS* (Kelley, 2020) packages, respectively.

with new starting values of ν_e if it failed to converge within 1,000 iterations. Starting values were randomly generated using $\nu_{e0} \sim \mathcal{U}(.2, .9)$ and $\epsilon_0 \sim \mathcal{U}(0, .8)$, where ν_{e0} and ϵ_0 denote the starting values of ν_e and ϵ and $\mathcal{U}(a, b)$ denotes a uniform distribution on the interval $[a, b]$. These distributions were chosen because initial testing indicated that the multiple-target TKL method was more likely to result in a converged solution if the range of the starting values were somewhat restricted.

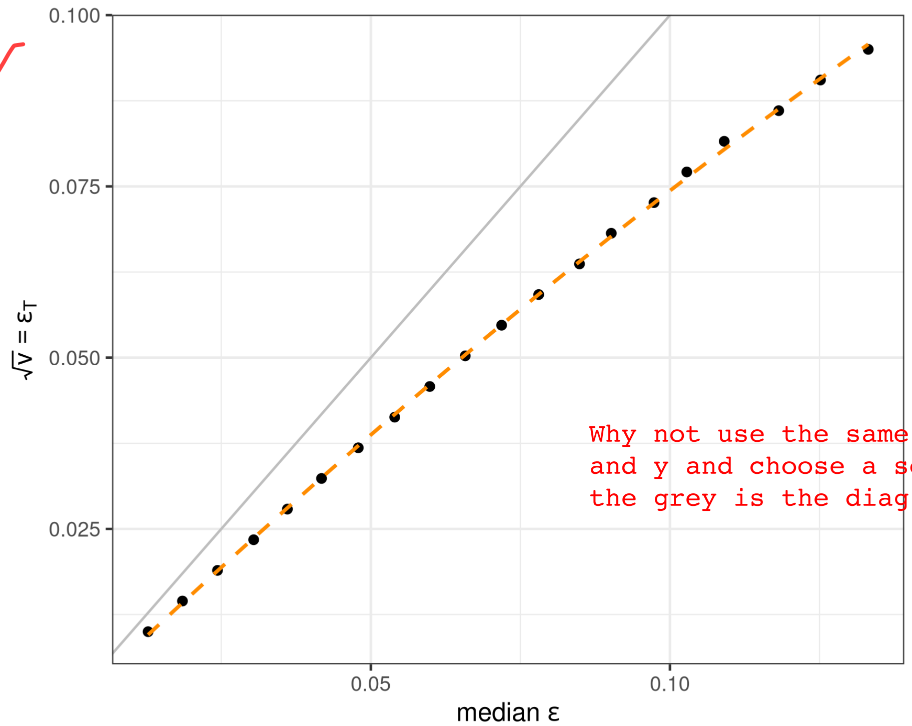
Refer to an earlier equation that defines this

In addition to randomly initializing the starting values of ν_e and ϵ, the values of the $\mathbf{W}^*$ matrix were also randomly initialized at each repetition. In the multiple-target TKL method, the $p \times q$ provisional matrix $\mathbf{W}^*$ was initialized such that each column consisted of p samples from a standard normal distribution. The $\mathbf{W}$ matrix was then obtained as follows. Let ϵ_j and ν_{ej} denote the proposed values of ϵ and ν_e at iteration j of the multiple-target TKL optimization procedure. At the jth iteration, the columns of the $\mathbf{W}^*$ were scaled using $\mathbf{W}^*_j = \mathbf{W}^*\mathbf{V}$, where $\mathbf{V}$ denotes a $q \times q$ diagonal matrix with diagonal elements $(1-\epsilon_j)^0, (1-\epsilon_j)^1, \ldots, (1-\epsilon_j)^{q-1}$. Then, $\mathbf{W}^*_j$ was scaled as described in Section 1.1.1 to form a $\mathbf{W}_j$ matrix to ensure that the proportion of variance accounted for by the q minor common factors was equal to ν_{ej}.

Repeating each of the TKL method variations 500 times with random starting values was important because the multiple-target TKL method is not guaranteed to find the global optimum for a particular problem. Moreover, the results of the multiple-target TKL method depend on the $\mathbf{W}^*$ matrix. Therefore, evaluating the multiple-target TKL method with a single set of starting values and a single $\mathbf{W}^*$ matrix might have led to results that were idiosyncratic and not representative of the method's general performance. Another reasonable question is why $\mathbf{W}^*$ was not held fixed over the 500 repetitions with random starting values to find the global optimum parameter values *for that particular* $\mathbf{W}^*$ *matrix*. However, the purpose of the multiple-target TKL method was to produce solutions with fit index values that were "close enough" to the target values to be reasonable over the entire space of $\mathbf{W}^*$ matrices, not to find the best solution for one particular $\mathbf{W}^*$ matrix.

As with the multiple-target TKL methods, the CB and WB methods were also repeated 500 times for each condition of the simulation design. For the CB method, each repetition produced slightly different results due to differences in the randomly-generated $\mathbf{y}$ vector (see Section 1.1.2). For the WB method, each repetition represented an independent sample from the inverse Wishart distribution associated with each condition. Therefore, a large number of samples were required to ensure that the simulation results for the CB and WB methods represented the full range of potential outcomes. Unlike the CB model-error method, the WB model-error method did not allow precise control of RMSEA values and produced RMSEA values that were only approximately equal to the target value. Simply using a target RMSEA value to solve for a value of v was unlikely to result in $\mathbf{\Sigma}$ matrices that had RMSEA values close to the target RMSEA value, unless the target RMSEA value was relatively small.

To resolve this issue, I developed a method to find a value of v such that the median RMSEA value from the resulting error-perturbed covariance matrices was close to the target RMSEA value. The method is as follows. First, initial values of v were obtained by squaring each element in a sequence of 20 equally-spaced RMSEA values ranging from 0.01 to 0.095. For each v value, 50 error-perturbed covariance matrices were sampled from the inverse Wishart distribution and the median RMSEA value was computed. Next, v was regressed on the linear and squared median RMSEA terms. The fitted model was then used to predict an appropriate v value such that the median RMSEA value for simulated, error-perturbed covariance matrices was close to the target value. An example is shown in Figure 3.1, which shows the $\sqrt{v} = \varepsilon_{\mathrm{T}}$ values plotted against the observed median ε values for 50 simulated Σ matrices corresponding to an orthogonal, five-factor model with salient factor loadings sampled uniformly from 0.3 and 0.7. The dashed orange curve indicates the $\sqrt{v}$ values related to each median ε value, as predicted by the fitted regression model.



Why not use the same limits for x and y and choose a square plot so the grey is the diag?

Figure 3.1. The relationship between $\sqrt{\tilde{v}}$ and ϵ for a 5-factor model. The solid gray line indicates where the target RMSEA and median RMSEA values (from 50 samples) would be equal. The dashed orange line indicates the predicted value of $\sqrt{v}$ that will produce a given RMSEA value.

Sorry, but this is not clear to me. You should flesh this out. What are the black dots?

In summary, the design of the simulation study was as follows. The crossed combinations of number of factors, number of items per factor, factor correlation, and factor loading configurations corresponded to 60 population major factor models. For each of those population models, I generated 500 error-perturbed correlation matrices using three variations of the multiple-target TKL method, the CB method, and the WB method at each of three target model fit conditions corresponding to Very Good, Fair, and Poor model fit. In total, this resulted in 900 unique conditions and a total of 450,000 simulated error-perturbed correlation matrices. All of the independent variables in the study (and the levels of those variables) are summarized in Table 3.2. R code for all aspects of the simulation study is provided in Appendix A.2.

Variable	Levels
Factors	1, 3, 5
Items/Factor	5, 15
Factor Correlation (ϕ)	0, .3, .6
Factor Loading	0.4 , 0.6, 0.8
Target Model Fit	Very Good ($\mathrm{RMSEA_T} = 0.025$, $\mathrm{CFI_T} = .99$), Fair ($\mathrm{RMSEA_T} = 0.065$, $\mathrm{CFI_T} = .95$), Poor ($\mathrm{RMSEA_T} = 0.090$, $\mathrm{CFI_T} = .90$)
Model-Error Method	$\mathrm{TKL_{RMSEA}}$, $\mathrm{TKL_{CFI}}$, $\mathrm{TKL_{RMSEA/CFI}}$, CB, WB

Note. $\mathrm{RMSEA_T}$ = Target RMSEA value; $\mathrm{CFI_T}$ = Target CFI value. TKL subscripts indicate which model fit indices were used as targets.

Table 3.2
Simulation Study Design Variables and Levels.

Chapter 4

Results

I strongly disagree with this claim since the models produce very different types of error structures. For instance the CB error seems very implausible to me. You do not need to agree with me, but you need to justify your claim that it makes no difference if the target rmsea is achieved

In the previous section, I described the simulation study I conducted to learn more about the behavior of different methods for generating error-perturbed population covariance (correlation) matrices. The simulation study included five model-error methods—the three multiple-target TKL variations, the CB method, and the WB method—and was designed to answer two primary questions.

First, I wanted to know how the model-error methods differed in terms of the CFI, TLI, and CRMR fit indices they led to when used with the same error-free models and target RMSEA values. If there were no meaningful differences among the methods, it would indicate that the choice of one particular model-error method over another (among the methods considered here) is not an important variable in the design of simulation studies. On the other hand, If the model-error methods led to systematically different values of the alternative fit indices when matched on RMSEA and all other characteristics, it would suggest that they are not exchangeable. In that case, it would be useful to know which model-error methods frequently produced solutions with multiple fit indices that indicated the target level of qualitative model fit.

A second purpose of the study was to evaluate the effectiveness of the multiple-target TKL method for generating correlation matrices with model error that had RMSEA and CFI values that were close to the specified target values. It was not expected that the multiple-target TKL method would be able to produce correlation matrices with RMSEA and CFI values that were very close to the target values for all of the major-factor population models because of the relationship between RMSEA, CFI, and population model characteristics (Lai & Green, 2016). Therefore, I used the absolute deviation between the observed and target RMSEA and CFI values to compare the results from the multiple-target TKL method to the results from the CB and WB methods used in Study 1.

The remainder of the section is structured as follows. First, I report how frequently simulated matrices had properties that might make them unsuitable for use in a simulation study. For instance, the CB method sometimes produced Σ matrices that were indefinite (i.e. having one or more negative eigenvalues). Additionally, the TKL-based model-error methods sometimes failed to converge using the L-BFGS-B method or led to solutions with

We should discuss this.

nominally minor factors that would more rightly be considered major factors due to the number of salient factor loadings. Second, I report the distributions of the five fit indices investigated in this study (RMSEA, CFI, TLI, and CRMR) for each of the five model-error methods, conditioned on the other variables in the study design. Third, I report the extent to which the RMSEA and CFI values for the Σ matrices produced by each of the model-error methods indicated similar levels of model fit. Fourth, I compare the RMSEA and CFI values corresponding to discrepancy between Σ and Ω to the RMSEA and CFI values corresponding to the discrepancy between Σ and $\hat{\Omega}$. Finally, I report results showing that the TKL-based optimization method was able to generate solutions with RMSEA and CFI values that were very close to the target values when the target value combinations were known to be possible.

Be kind, and remind (your readers what these mean)

4.1 Indefinite Matrices (CB)

One drawback of the CB model-error method is that the resulting correlation matrix with model error can be indefinite when the specified target RMSEA value is large (Cudeck &

Browne, 1992). These matrices are undesirable because correlation and covariance matrices are, by definition, at least positive semi-definite (i.e., having strictly non-negative eigenvalues). Of the 90,000 correlation matrices with model error that were generated using the CB method, 14,291 (16%) were indefinite. However, indefinite solutions were much more common for some conditions of the simulation design than others. Figure 4.1 shows the percent of indefinite CB solutions for each level of model fit, number of items per factor, number of factors, and factor loading strength. (Exact percentages are reported in Table 4.1).

Figure 4.1 shows at least three notable trends. First, the percent of indefinite solutions increased as model fit degraded. Second, the percent of indefinite solutions increased as the total number of items increased (i.e., as the number of factors and items per factor increased). Finally, the percent of indefinite solutions increased as factor loadings increased. In the best-case scenarios, conditions corresponding to models with 25 or fewer items led to indefinite solutions very infrequently (in less than 1% of cases). On the other hand, conditions with Interesting Poor model fit and 45 or more items led to indefinite correlation matrices in more than 90% of cases. These results show that the CB method would be an inefficient way to simulate positive semi-definite population correlation or covariance matrices with model error when input matrices are large and the target RMSEA value is relatively large.

A potential strategy for dealing with indefinite solutions when using the CB method would be to simply generate solutions using the CB method until a sufficient number of positive semi-definite solutions have been obtained. However, the amount of time taken by the CB method increases quickly as the number of items increase, making the oversampling strategy impractical for problems with many items. In fact, using the CB method to generate even a small number of solution matrices becomes impractical for large input matrices. This is shown in Figure 4.2, which plots the completion time for the CB method when applied to a one-factor model with salient loadings fixed at 0.6 and the number of items varying between 5 and 120. Using a computer with an Intel Core i5-4570 3.20GHz CPU and 16GB of RAM, the CB method took just over 30 seconds to complete (on average) for an input correlation matrix with 65 items. For an input matrix with 115 items, the CB method took approximately four and a half minutes to complete. Such long completion times are often impractical for large simulation studies, particularly if indefinite solutions are discarded. In fact, I had to skip using the CB method in simulation conditions with 10 factors and 15 items per factor (150 items) because those conditions would have taken an impractical amount of time to complete. Timing a single example, the CB method took 15 minutes and 48 seconds to complete with a 150-item input correlation matrix. At that rate, it would have taken almost 11 days to complete one (out of 36) conditions of the simulation design with 150 items.

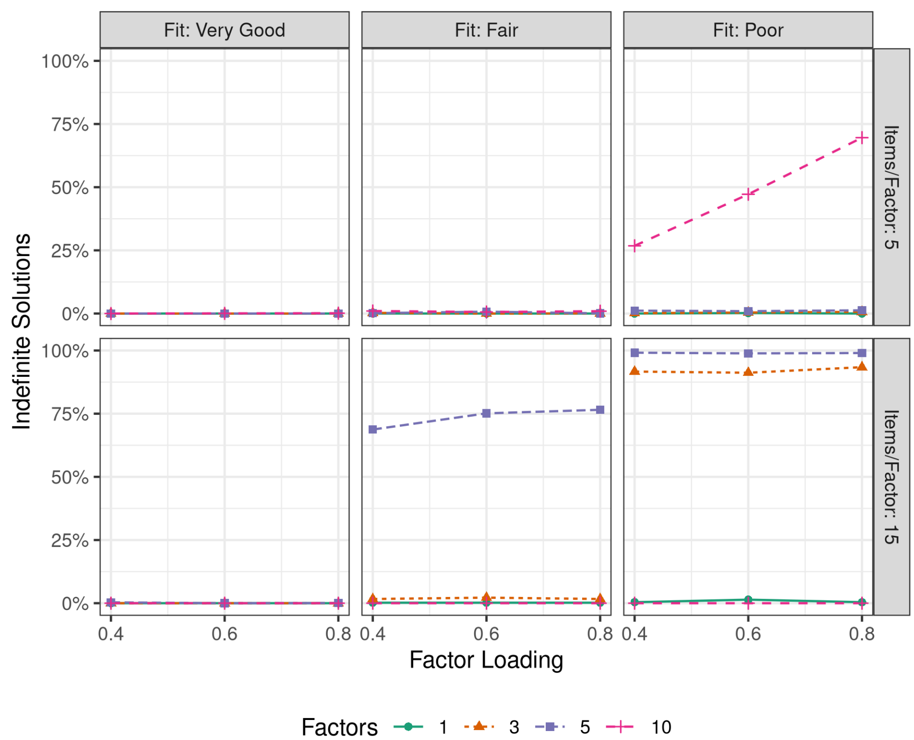


Figure 4.1. The percent of Cudeck-Browne (CB) method solutions that were indefinite, conditioned on number of factors, factor loading, number of items per factor, and model fit.

			Factors			
Items/Factor	Loading	Model Fit	1	3	5	10
5	0.4	Very Good	0.0	0.0	0.0	0.0
5	0.4	Fair	0.0	0.3	0.1	0.0
5	0.4	Poor	0.0	0.1	1.1	0.3
5	0.6	Very Good	0.0	0.0	0.0	0.0
5	0.6	Fair	0.0	0.0	0.7	0.0
5	0.6	Poor	0.2	0.4	0.9	0.5
5	0.8	Very Good	0.0	0.0	0.0	0.0
5	0.8	Fair	0.0	0.1	0.1	0.0
5	0.8	Poor	0.0	0.7	1.3	0.7
15	0.4	Very Good	0.0	0.0	0.3	—
15	0.4	Fair	0.2	1.7	68.7	—
15	0.4	Poor	0.4	91.7	99.1	—
15	0.6	Very Good	0.0	0.0	0.0	—
15	0.6	Fair	0.2	2.2	75.1	—
15	0.6	Poor	1.4	91.2	98.9	—
15	0.8	Very Good	0.0	0.0	0.1	—
15	0.8	Fair	0.2	1.7	76.5	—
15	0.8	Poor	0.4	93.4	99.0	—

Note. The Cudeck-Browne method was not used for conditions with 10 major factors and 15 items per factor because it was prohibitively slow for those conditions.

Table 4.1
The percent of Cudeck and Browne (CB) model-error method solutions that were indefinite.

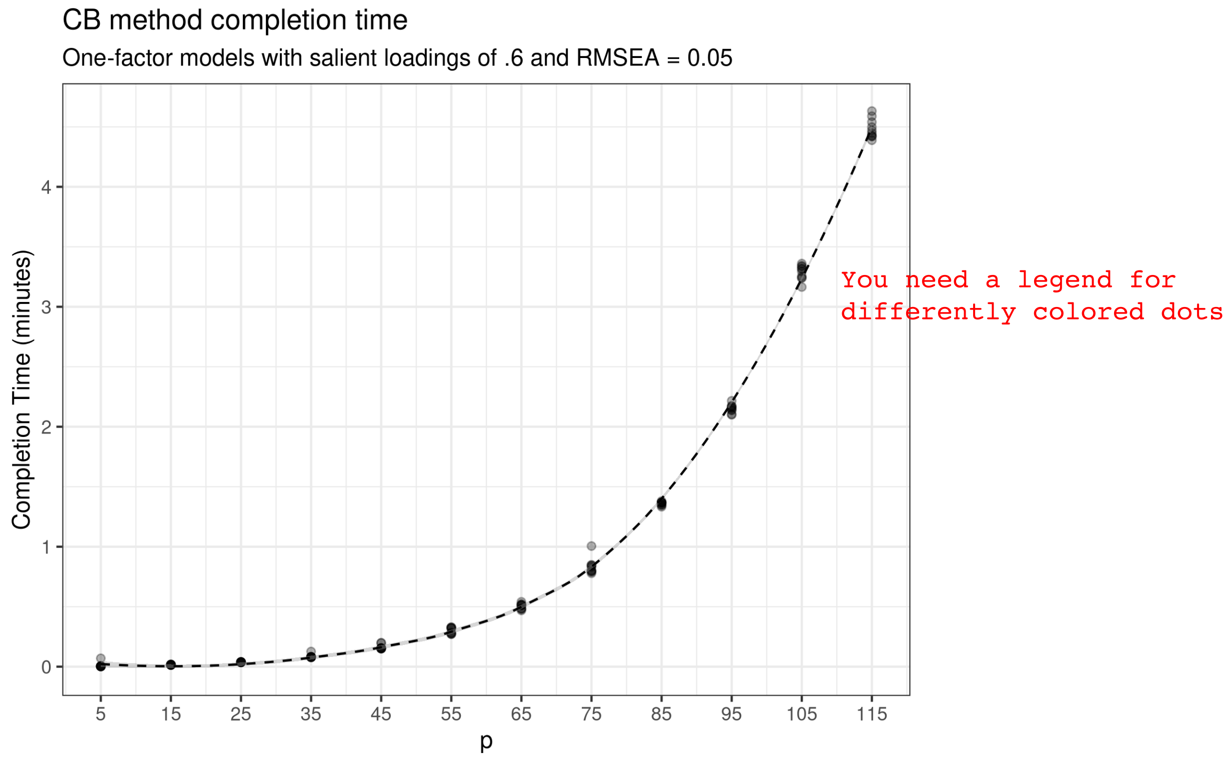


Figure 4.2. The amount of time (in minutes) taken to generate a single correlation matrix with model error using the CB method. Completion times were recorded 10 times for single-factor models with salient factor loadings fixed at 0.6 and number of items (p) varying between 5 and 115. The dashed black line is the LOESS regression line.

4.2 L-BFGS-B Non-convergence (Genetic Algorithm)

The default optimization method for the multiple-target TKL method was L-BFGS-B. In most cases, this method worked well and converged to a solution relatively quickly. However, there were a small number of cases where the L-BGFS-B method failed to converge. Specifically, the L-BFGS-B method failed to converge 14 times (< 1% of cases) and only failed to converge when the $\text{TKL}_{\text{RMSEA/CFI}}$ method was used and when model fit was Poor. Non-convergence was also somewhat more likely for conditions with few factors compared to conditions with many factors. This can be seen in Table 4.2, which shows the L-BFGS-B non-convergence rates by the number of factors, number of items per factor, and model fit

Dont you need two decimal places
if these are percentages. Why not
report counts or 100x%

		Model Fit		
Factors	Items/Factor	Very Good	Fair	Poor
1	5	0.0	0.0	0.1
1	15	0.0	0.0	0.4
3	5	0.0	0.0	0.1
3	15	0.0	0.0	0.0
5	5	0.0	0.0	0.0
5	15	0.0	0.0	0.0
10	5	0.0	0.0	0.0
10	15	0.0	0.0	0.0

Table 4.2
The percent of cases in each combination of conditions where the L-BFGS-B algorithm did not converge after 100 random starts and genetic optimization was used instead.

for the $\text{TKL}_{\text{RMSEA/CFI}}$ method.

Although non-convergence of the L-BFGS-B algorithm was rare, a natural question was whether the genetic algorithm that was used as a fallback option led to similar results compared to cases where the L-BFGS-B algorithm converged. This question was difficult to answer statistically because non-convergence occurred so infrequently. To get a general sense of whether results were similar for cases where the L-BFGS-B algorithm converged or did not converge, I plotted the CFI and RMSEA values for all converged and non-converged cases in conditions where the $\text{TKL}_{\text{RMSEA/CFI}}$ model-error method was used and model fit was Poor. Figure 4.3 shows that cases where the L-BFGS-B algorithm did not converge (shown in black) led to similar RMSEA and CFI values compared to cases where the algorithm converged (shown in gray). Thus, using a genetic algorithm seemed to provide reasonable results in the few cases where the L-BFGS-B algorithm failed to converge (albeit much more slowly than the L-BFGS-B algorithm).

How slowly. Did you tell us?

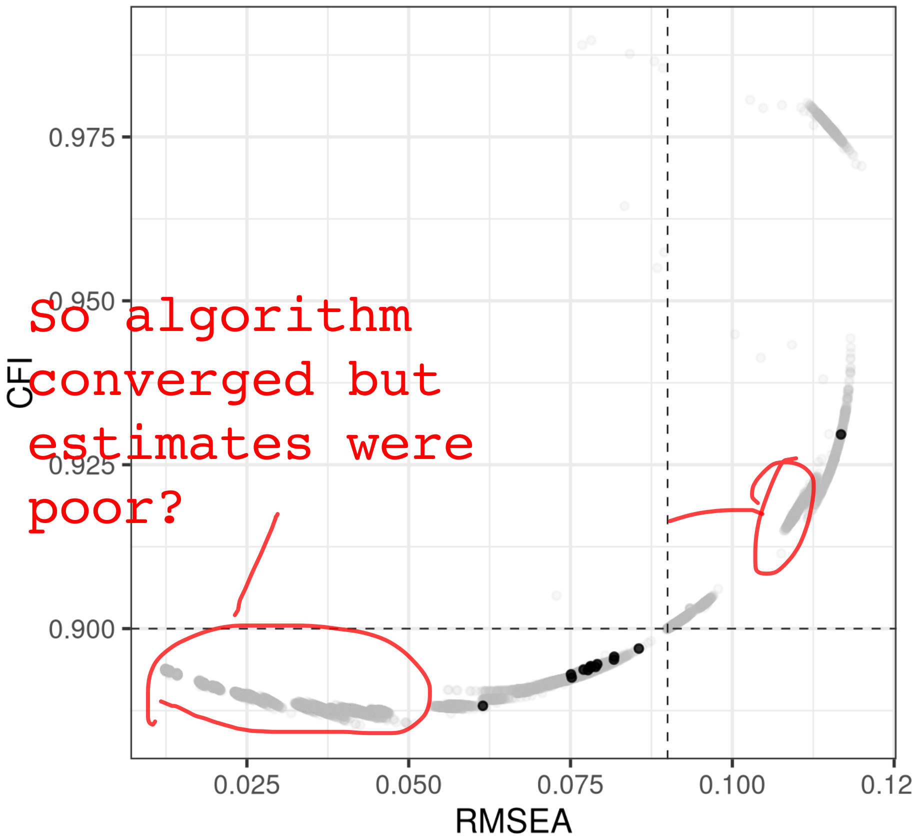


Figure 4.3. RMSEA and CFI values for cases where the model-error method was $\text{TKL}_{\text{RMSEA/CFI}}$ and model fit was Poor. Grey dots indicate cases where the L-BFGS-G algorithm converged; black dots indicate cases that did not converge and where a genetic algorithm was used instead. The dashed black lines indicate the target RMSEA and CFI values.

4.3 Major Minor Factors (W Matrix Constraint Violations)

Recall that the multiple-objective TKL method included an optional penalty that penalized cases that had strong minor factors. Specifically, the penalty was applied if any minor factor had two or more absolute factor loadings greater than or equal to a specified value. The purpose of the penalty was to avoid introducing minor factors that might be more accurately characterized as major factors. To determine whether the penalty was effective at helping avoid major minor factors, I checked each of the minor factor loading (**W**) matrices to determine whether any minor factor had two or more loadings greater than 0.3 in absolute value.

The percent of cases where the constraints on **W** were violated for each level of number of factors, number of items per factor, factor loading strength, and model fit are shown in Figure 4.4 and reported in Table 4.3. Only results for the $\text{TKL}_{\text{RMSEA}}$ method were included because the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} model-error methods seldom led to solutions that violated the constraints on **W**. In fact, only 24 out of 180,000 cases (<0.01%) had violated **W** constraints for the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} methods combined. Figure 4.4 shows that the **W** constraints were violated most often when model fit was Fair or Poor, factor loadings were relatively low, and there were many total items (i.e., many factors and items per factor). These variables were included in the figure because they were most important in terms of whether a solution was likely to violate the constraints on **W**, as indicated by the size of the logistic regression coefficients reported in Appendix B.1.1.

Have you mentioned how many minor factors (col order of W) were generated and whether you always generated the same number of minor factors

			Model Fit		
Factors	Items per Factor	Loading	Very Good	Fair	Poor
1	5	0.4	0.0	0.0	0.0
1	5	0.6	0.0	0.0	0.0
1	5	0.8	0.0	0.0	0.0
1	15	0.4	0.0	3.2	39.4
1	15	0.6	0.0	0.0	15.0
1	15	0.8	0.0	0.0	0.0
3	5	0.4	0.0	0.1	22.9
3	5	0.6	0.0	0.0	4.7
3	5	0.8	0.0	0.0	0.0
3	15	0.4	0.0	57.3	79.3
3	15	0.6	0.0	33.3	74.9
3	15	0.8	0.0	0.0	7.4
5	5	0.4	0.0	4.8	52.3
5	5	0.6	0.0	0.0	34.5
5	5	0.8	0.0	0.0	0.0
5	15	0.4	0.0	79.0	92.6
5	15	0.6	0.0	73.3	89.7
5	15	0.8	0.0	0.0	81.2
10	5	0.4	0.0	44.7	75.9
10	5	0.6	0.0	12.1	68.9
10	5	0.8	0.0	0.0	1.6
10	15	0.4	0.0	94.5	99.9
10	15	0.6	0.0	94.8	98.3
10	15	0.8	0.0	82.3	96.8

Table 4.3
The percent of cases that violated the minor common factor loading constraints when the TKL_{RMSEA} method was used.

This is very interesting and
suggests that in big models rmsea
is insensitive to misfit unless
you get the number of factors
question wrong.

Is this any different from the
above table or redundant

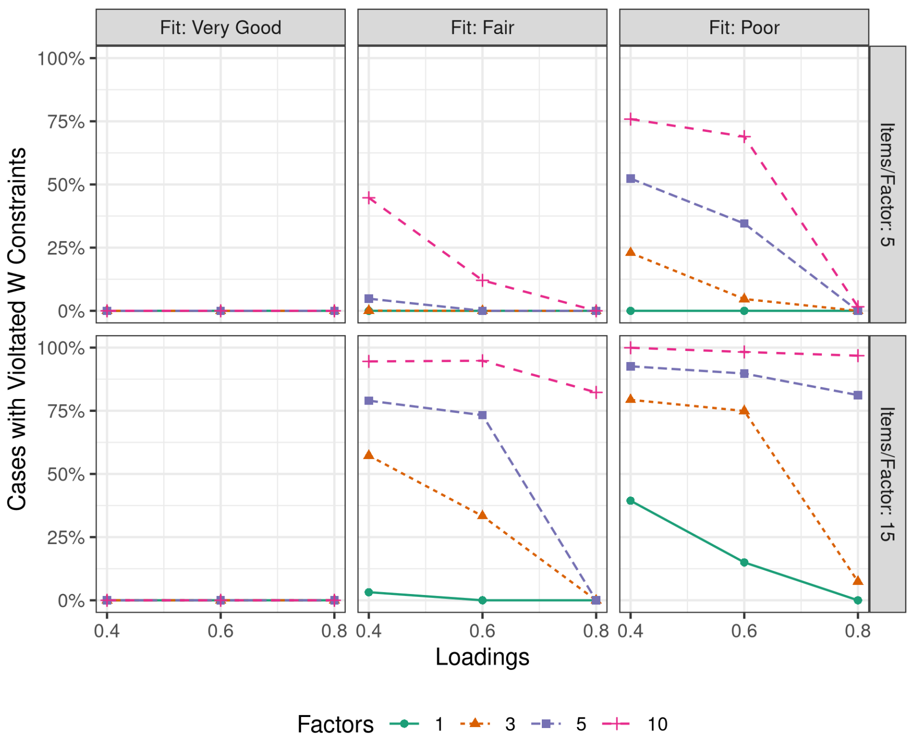


Figure 4.4. The percent of cases where the constraints on **W** were violated when the $\text{TKL}_{\text{RMSEA}}$ model-error method was used, conditioned on number of factors, number of items per factor, factor loading, and model fit.

To understand why certain conditions led to more **W** constraint violations than others, it is helpful to understand the relationship between those conditions and the TKL parameters (ϵ and ν_{e}). Figure 4.5 shows the ϵ and ν_{e} values for each of the TKL model-error methods, conditioned on factor loading strength, number of factors, and number of items per factor. To conserve space, only conditions with Poor model fit and 1, 5, or 10 major factors were included in the figure (a complete version of this figure is shown in Appendix B.2.3). Overall, the figure shows that there was a trade-off between ϵ and ν_{e} such that higher values of ν_{e} were related to lower values of ϵ (and *vice versa*). Moreover, Figure 4.5 shows that the distributions of ϵ and ν_{e} differed depending on which TKL variant was used. The differences between the error-method variants were largest when the there were many items (i.e., many items and many items per factor) and when factor loadings were relatively weak. Under those circumstances, the $\text{TKL}_{\text{RMSEA}}$ method led to higher values of ν_{e} than the $\text{TKL}_{\text{RMSEA/CFI}}$ or TKL_{CFI} methods. This effect suggested that higher values of ν_{e} were required to produce solutions with RMSEA values close to 0.09 when there were many items. On the other hand, the results for the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} methods indicated that lower ν_{e} values were required to obtain CFI values close to .90 when there were many items.

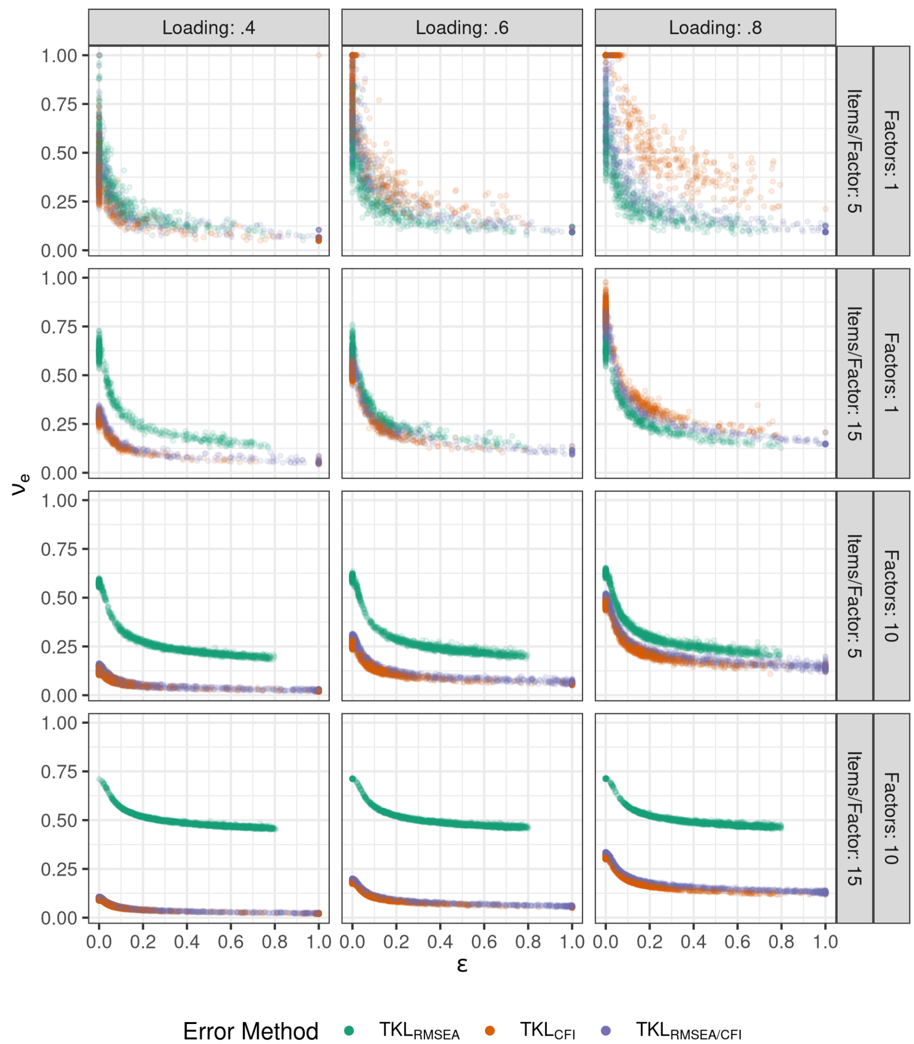


Figure 4.5. Values of the TKL parameters (ϵ and ν_{e}) by model-error method, number of factors, number of items per factor, and factor loading strength when model fit was Poor. Results for conditions with three or five major factors were omitted to conserve space. TKL = Tucker, Koopman, and Linn.

The apparent trade-off between ϵ and ν_{e} values made sense considering their roles in the TKL method. A high ν_{e} value indicated that much of the unique variance would be assigned to the minor common factors. If the value of ϵ was also high, it indicated that the first few minor factors would account for most of the minor factor variance. Therefore, the **W** constraints were more likely to be violated when both ϵ and ν_{e} were high, and less likely to be violated if either parameter was low. The link between ϵ and ν_{e} and constraint violations is shown in Figure 4.6, which shows the values of ϵ and ν_{e} produced by the three TKL variants for conditions with Poor model fit, 10 factors, 15 items per factor, and factor loadings of 0.8. Each point (corresponding to a single case) was colored according to whether or not the $\mathbf{W}$ constraints were violated. The figure shows that the $\mathbf{W}$ constraints were violated when the values of ϵ and ν_{e} were both higher than some threshold values, which only happened when the $\mathrm{TKL}_{\mathrm{RMSEA}}$ was used.

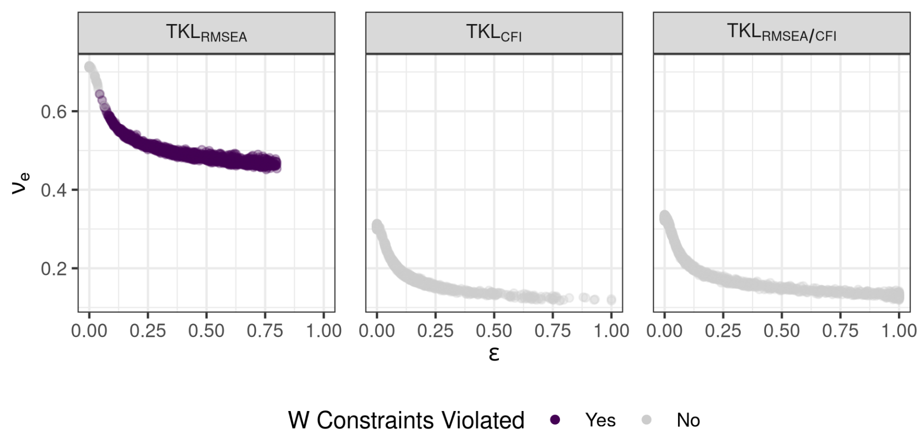


Figure 4.6. The distribution of ϵ and ν_{e} (and whether or not $\mathbf{W}$ constraints were violated) for conditions with Poor model fit, 10 factors, 15 items per factor, and factor loadings of 0.8. TKL = Tucker, Koopman, and Linn.

4.3.1 Choice of Penalty Value

The penalty value in Equation (2.1), λ, can be set at a higher or lower value to change the penalty that is incurred when a potential solution has a $\mathbf{W}$ matrix that violates the user-specified constraints. In the present study, the penalty was set to be $\lambda = 1{,}000{,}000$ to ensure that the penalty was sufficiently large. However, it is possible that smaller values of λ would have also been as effective or more effective at leading to solutions that satisfied the constraints on $\mathbf{W}$. To determine which values of λ worked best, I simulated 200 correlation matrices with model error using the $\text{TKL}_{\text{RMSEA}}$ method for each of 36 conditions in the main simulation study. These conditions were formed by crossing the number of major common factors (1, 3, 5, or 10), and nine values of λ ($\lambda \in [0, 0.1, 1, 10, 100, 1,000, 10,000, 100,000, 1,000,000]$). All factors were orthogonal, with factor loadings fixed at 0.4, 15 items per factor, and a target RMSEA value of 0.09. These sets of values were chosen because they often resulted in solutions with violated **W** constraints in the main simulation study.[1]

The percentages of solutions with violated **W** constraints for each combination of number of factors and λ are shown in Figure 4.7. The figure shows that for conditions with one, three, or five factors, the proportion of solutions with violations of the **W** constraints decreased as λ increased from zero to one and then leveled off for λ values greater than one. For conditions with ten factors, all solutions had violated **W** constraints, regardless of the λ value. Although only a subset of the conditions from the full study were included in this smaller simulation, the results suggested that using a smaller λ value is unlikely to have resulted in a substantial decrease in the number of solutions with violated **W** constraints. On the other hand, the results also suggested that λ values as small as one are likely just as effective as much larger values at preventing solutions with violated **W** constraints.

[1] Code for this simulation study is provided in Appendix A.3.

Thanks for looking into this

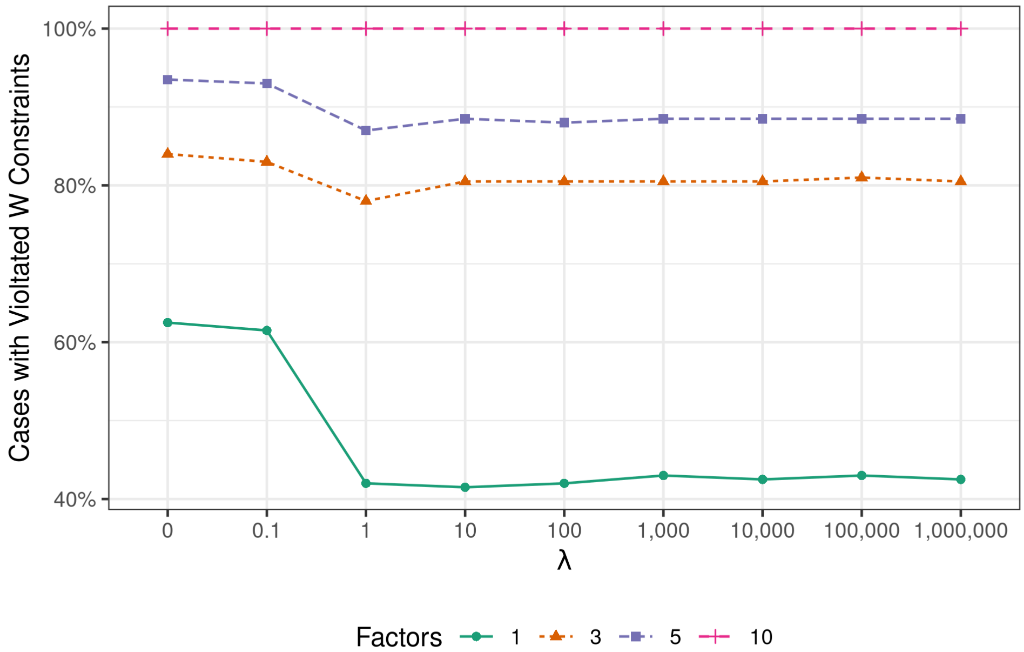


Figure 4.7. The percent of cases with violated **W** constraints conditioned on the constraint penalty value (λ) and the number of major common factors when the $\mathrm{TKL}_{\mathrm{RMSEA}}$ method was used. All conditions had 15 items per factor, major common factor loadings fixed at 0.4, and a target RMSEA value of 0.09. One hundred solution matrices were generated for each condition.

The results from both the main simulation study and the smaller study of λ values suggested that enforcing constraints on **W** using the penalty in Equation (2.1) was only somewhat effective when the $\mathrm{TKL}_{\mathrm{RMSEA}}$ method was used. Regardless of the λ value used, the $\mathrm{TKL}_{\mathrm{RMSEA}}$ method often led to solutions with violated **W** constraints in conditions with Poor model fit, weak factor loadings, and many items or factors. In contrast, the $\mathrm{TKL}_{\mathrm{CFI}}$ and $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$ methods rarely produced solutions that violated the constraints on **W**. However, it was unclear whether the the penalty worked better for the $\mathrm{TKL}_{\mathrm{CFI}}$ and $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$ methods for some reason, or whether the inclusion of a target CFI value simply made it less likely for the **W** constraints to be violated regardless of whether the penalty was included.

To test whether the inclusion of the target CFI value was sufficient to avoid violated $\mathbf{W}$ constraints when no penalty was applied, I conducted another small-scale simulation study. Specifically, I generated 200 Σ matrices using a population correlation matrix with ten (orthogonal) major common factors, 15 items per factor, and salient factor loadings fixed at 0.4. Each of the Σ matrices were generated using the $\text{TKL}_{\text{RMSEA/CFI}}$ method with target RMSEA and CFI values of 0.09 and 0.90, respectively, and $\lambda = 0$ so that no penalty was applied. I chose this condition because nearly 100% of the solutions generated by the $\text{TKL}_{\text{RMSEA}}$ for this condition in the main simulation study had violated $\mathbf{W}$ constraints, whereas the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods did produce any solutions with violated $\mathbf{W}$. Thus, generating Σ matrices for this condition using the $\text{TKL}_{\text{RMSEA/CFI}}$ with $\lambda = 0$ was a reasonable test of whether using a penalty to enforce the $\mathbf{W}$ constraints was necessary when the $\text{TKL}_{\text{RMSEA/CFI}}$ was used.[2] Somewhat surprisingly, I found that none of the 200 solutions that were generated violated the $\mathbf{W}$ constraints when λ was set to zero. This result suggested that the addition of a target CFI value in and of itself was useful for producing solutions without unacceptably strong minor factors.

To better understand why the $\text{TKL}_{\text{RMSEA}}$ method so often led to solutions that violated the constraints on $\mathbf{W}$ compared to the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods, I evaluated the objective function over a grid of $62,500$ ϵ and ν_{e} values for a single pair of Ω and $\mathbf{W}^*$ matrices and each of the three TKL model-error methods. The Ω population correlation matrix without model error was again set to correspond to a model with 10 orthogonal major factors and 15 items per factor, with salient factor loadings fixed at 0.4. The penalty value was fixed at 1,000 to penalize solutions with minor factors that had more than two factor loadings greater or equal to 0.3 in absolute value. The objective function surfaces for each of the three TKL model-error methods are shown in Figure 4.8. To make it easier to compare the objective function surfaces, I made separate plots for cases where the $\mathbf{W}$ constraints were and were not violated. Moreover, I scaled the objective function values (G_{obj}) in each

[2] Code for this simulation study is provided in Appendix A.4.

panel of Figure 4.8 to fall between zero and one so that the surfaces could be easily compared. The figure showed clear differences between the objective function surfaces for the $\text{TKL}_{\text{RMSEA}}$ method compared to the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods. In particular, the figure showed that the values of F_{obj} decreased as ϵ and ν_{e} increased for the $\text{TKL}_{\text{RMSEA}}$ method. In contrast, the values of F_{obj} increased as ϵ and ν_{e} increased for the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods. This is goofy

The results shown in Figure 4.8 helped explain why the $\text{TKL}_{\text{RMSEA}}$ method often led to solutions with violated **W** constraints even when such solutions were heavily penalized. The figure shows that for ϵ and ν_{e} needed to be relatively large to produce solutions with RMSEA values that were close to the target RMSEA value of 0.09. In fact, the lowest F_{obj} values were produced when ϵ and ν_{e} were as large as possible without violating the **W** constraints. The objective function value increased sharply after ϵ and ν_{e} became large enough to violate the **W** constraints, but then decreased again as ϵ and ν_{e} increased. Therefore, the optimization procedure was likely to move toward larger values of ϵ and ν_{e} corresponding to local minima, regardless of whether the constraints on **W** were violated.

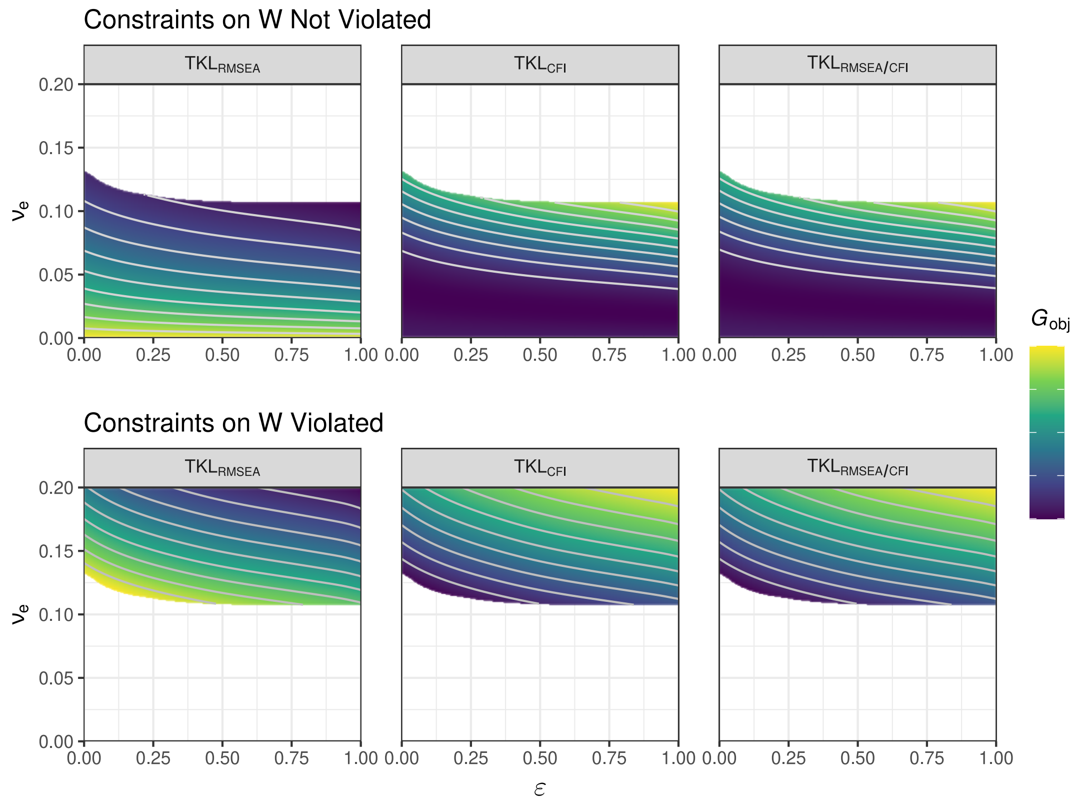


Figure 4.8. Heatmaps of the scaled objective function surface for combinations of ϵ and ν_{e} values and each of the Tucker, Koopman, and Linn (TKL; 1969) model-error methods. Objective function values (G_{obj}) were computed using a $\mathbf{\Omega}$ matrix with ten orthogonal major factors, 15 items per factor, and salient factor loadings fixed at 0.4. The penalty value, λ, was fixed at 1,000 to penalize solutions with minor factors that had more than two factor loadings greater or equal to 0.3 in absolute value. To aid visualization, the the objective function values for combinations of ϵ and ν_{e} where the constraints on $\mathbf{W}$ were violated were scaled and plotted separately from combinations that did not lead to constraint violations.

In contrast to the $\mathrm{TKL}_{\mathrm{RMSEA}}$ method, the $\mathrm{TKL}_{\mathrm{CFI}}$ and $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$ methods rarely produced solutions that violated the constraints on $\mathbf{W}$. This result can be at least partially explained by comparing the objective function surfaces shown in Figure 4.8. Unlike the $\mathrm{TKL}_{\mathrm{RMSEA}}$ method, the objective function surfaces for the $\mathrm{TKL}_{\mathrm{CFI}}$ and $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$ methods were lowest when both ϵ and ν_{e} were small. The optimization procedure was therefore unlikely to move toward values of ϵ and ν_{e} that produced solutions with violated $\mathbf{W}$

constraints. Moreover, even if the optimization procedure was initialized at large values of of ϵ and ν_e corresponding to violated $\mathbf{W}$ constraints, the gradient of the objective-function surface would have directed the optimization procedure back toward smaller ϵ and ν_e values that did not violate the $\mathbf{W}$ constraints. Therefore, adding a target CFI value in addition to (or instead of) a target RMSEA value ended up being far more effective than using an explicit penalty in Equation (2.1) for ensuring a meaningful distinction between major and minor common factors.

This is a Very interesting result

4.4 Distributions of Fit Statistics

One of the primary questions the simulation study was intended to answer was whether the five model-error methods produced solutions with different fit index values when used with the same error-free models and target RMSEA and CFI values. In this section, I report the distributions of the RMSEA, CFI, TLI and CRMR model-fit indices for solutions produced by each of the five model-error methods ($\text{TKL}_{\text{RMSEA}}$, TKL_{CFI}, $\text{TKL}_{\text{RMSEA/CFI}}$, CB, and WB). All of the fit indices reported in this section reflect the discrepancy between Σ and Ω. Plots for fit indices reflecting the discrepancy between Σ and $\hat{\Omega}$ similar to those presented in this section can be found in Appendix B.2.8.

4.4.1 RMSEA

Of the fit indices investigated in this study, the RMSEA value has been most often used as a measure of model fit when generating covariance or correlation matrices with model error (Briggs & MacCallum, 2003; Kracht & Waller, 2022; Lorenzo-Seva & Ferrando, 2020a; MacCallum et al., 2001; Myers et al., 2015). Figure 4.9 shows box-plots summarizing the distributions of RMSEA values for each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. These variables were the most important in terms of the absolute difference between the target and observed RMSEA values, as evidenced by the effect sizes reported in the ANOVA summary table in Appendix B.1.2. Notice in Figure 4.9 that the $\text{TKL}_{\text{RMSEA}}$ and CB methods almost always produced solutions with RMSEA values that were very close to the target RMSEA values. This result makes sense because both methods use optimization to produce solutions with RMSEA values close to a specified target. However, a somewhat unexpected result was that the CB method occasionally produced solutions with RMSEA values that were much higher or lower than the target value, particularly in conditions with ten major factors.

After the $\text{TKL}_{\text{RMSEA}}$ and CB methods, Figure 4.9 shows that the WB method was the next best model-error method in terms of producing solutions with RMSEA values close to the target values. In fact, the WB method produced solutions with median RMSEA values that were as close to the target values as those from $\text{TKL}_{\text{RMSEA}}$ and CB solutions. However, the WB method also led to more variable RMSEA values, particularly in conditions with Poor model fit and many factors.

The two methods that performed worst in terms of producing solutions with RMSEA values close to the targets were the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} methods. Figure 4.9 shows that these methods often led to RMSEA values that were lower than the target values, except when there were relatively few factors and strong factor loadings. The largest differences between the observed and target RMSEA values for the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} methods occurred in conditions with Poor model fit and weak factor loadings. In those conditions, both methods led to RMSEA values that were considerably lower than the target values.

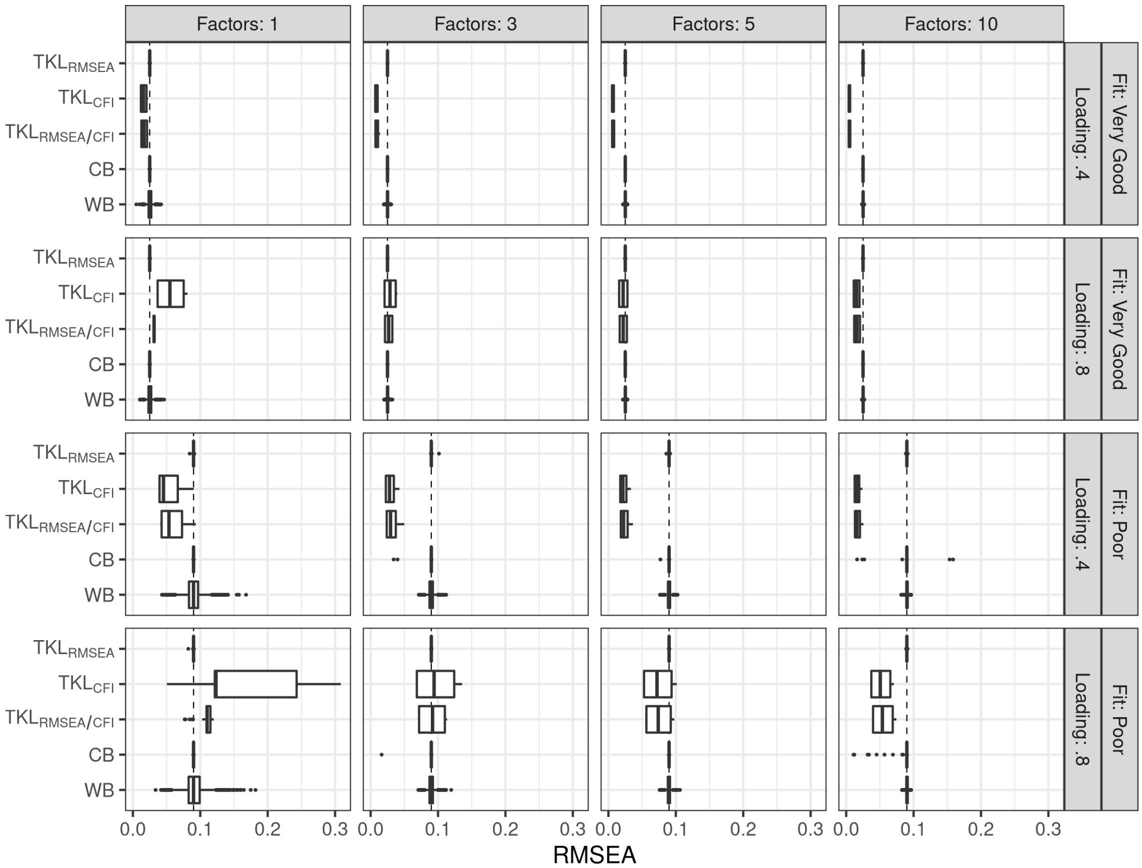


Figure 4.9. Distributions of the RMSEA values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. The dashed lines indicate the target RMSEA value for each condition. Note that some levels of model fit and factor loading strength were omitted to conserve space. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

4.4.2 CFI

The distributions of CFI values for the solutions produced using the five model-error methods are shown in Figure 4.10, conditioned on number of factors, model fit, and factor loading strength. These were the most important variables in terms of the absolute difference between the target and observed CFI values, as indicated by the effect sizes shown in the ANOVA summary table in Appendix B.1.3. As with Figure 4.9, the middle levels of model fit and factor loading strength were omitted to converse space. Figure 4.10 shows that the results for

CFI were nearly opposite to the results for RMSEA. Specifically, whereas the $\text{TKL}_{\text{RMSEA}}$, CB, and WB methods produced solutions with RMSEA values much closer to the target values than those produced by the $\text{TKL}_{\text{RMSEA/CFI}}$ or TKL_{CFI} methods, the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} produced solutions with CFI values that were closer to the target CFI values compared to the other model-error methods in most conditions. The differences between the methods that optimized for CFI ($\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI}) and the other model-error methods were largest for conditions with Poor model fit, low factor loadings, and many factors (see the third row of Figure 4.10). On the other hand, Figure 4.10 shows that all of the model-error methods produced similar CFI values (that were close to the target CFI value) for conditions with Very Good model fit and strong factor loadings.

It is worth highlighting the result that the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} produced solutions with very similar CFI values in most conditions. These two methods also produced solutions with very similar RMSEA values in many conditions, as shown in Figure 4.9. Indeed, the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} methods led to much more similar results in terms of both RMSEA and CFI than the $\text{TKL}_{\text{RMSEA/CFI}}$ and $\text{TKL}_{\text{RMSEA}}$ methods. Put another way, optimizing for CFI alone generally led to similar results compared to optimizing for both CFI and RMSEA, whereas optimizing for RMSEA alone generally led to solutions with much different RMSEA and CFI values compared to optimizing for both CFI and RMSEA. This suggests that CFI was more sensitive to small changes in parameter values than RMSEA, a hypothesis that is explored in greater depth later on in this section. Nice!

The distributions of CFI values shown in Figure 4.10 also emphasize the importance of reporting multiple fit indices when simulating correlation or covariance matrices with model error when compared with the distributions of RMSEA values in Figure 4.9. For instance, the $\text{TKL}_{\text{RMSEA}}$ and CB methods produced solutions with RMSEA values close to the target RMSEA value of .09 in conditions with Poor model fit and weak factor loadings. However, Figure 4.10 shows that those two model-error methods led to solutions with unacceptably low CFI values in the same conditions.

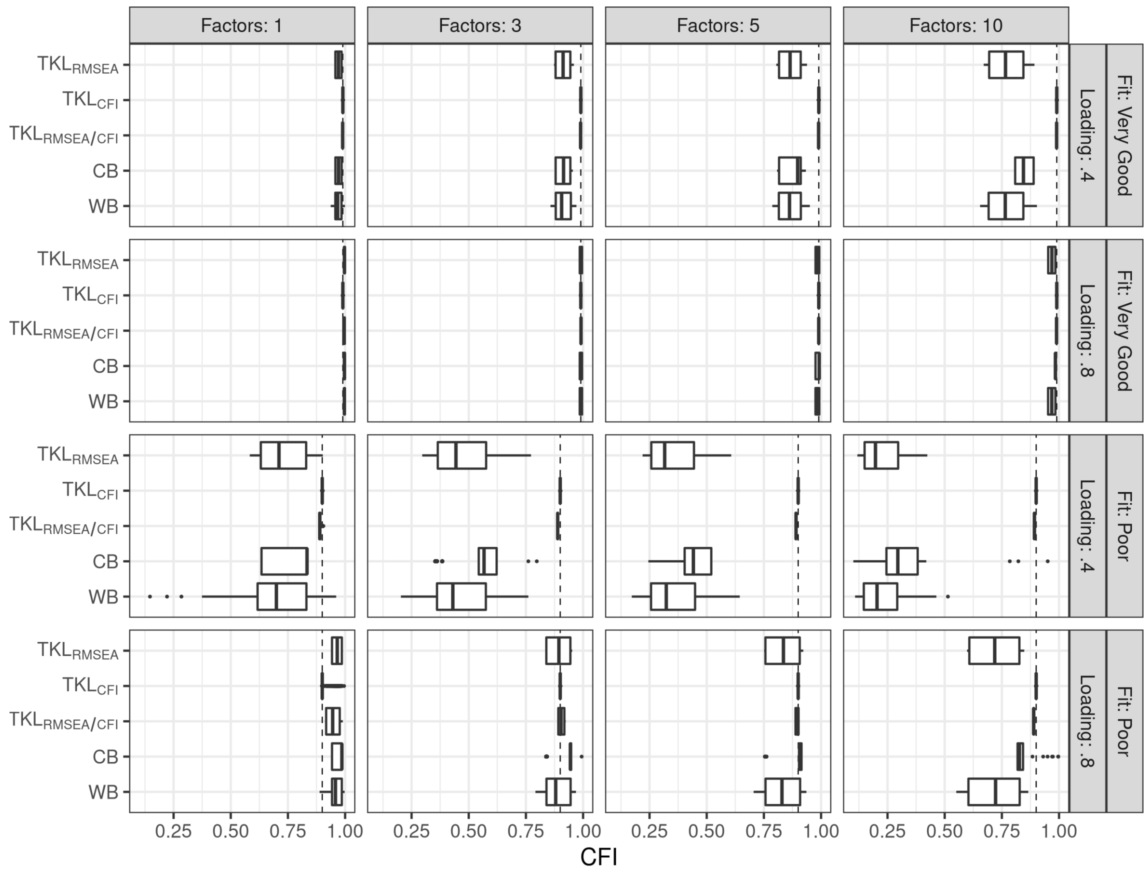


Figure 4.10. Distributions of the CFI values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. The dashed lines indicate the target CFI value for each condition. Note that some levels of model fit and factor loading strength were omitted to conserve space. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

4.4.3 TLI

The distributions of TLI values for the solutions produced using the five model-error methods are shown in Figure 4.11, conditioned on number of factors, model fit, and factor loading strength. Overall, the distributions of TLI values were quite similar to the distributions of CFI values shown in Figure 4.10. In particular, the $\mathrm{TKL}_{\mathrm{CFI}}$ and $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$ methods tended to produce solutions with higher TLI values than the other model-error methods, except for conditions with a single factor, Poor model fit, and strong factor loadings. Similar to CFI, the differences in TLI values resulting from the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods compared to the other model-error methods were most pronounced for conditions with many factors, weak factor loadings, and Poor model fit.

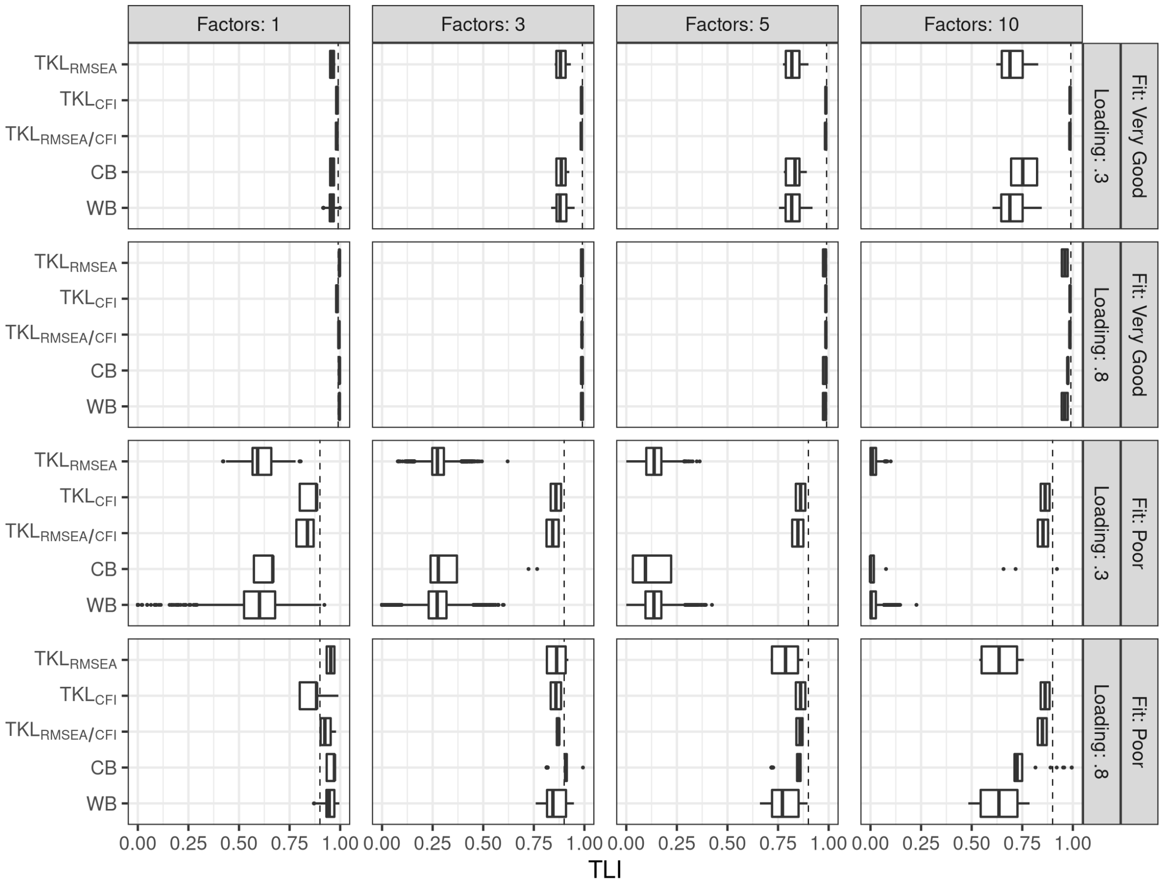


Figure 4.11. Distributions of the TLI values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. The dashed lines indicate the threshold values of TLI that correspond to the targeted levels of model fit, according to Hu and Bentler (1999). Note that some levels of model fit and factor loading strength were omitted to conserve space. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

4.4.4 CRMR

The distributions of the CRMR model fit index by level of model fit, factor loading, number of major common factors, and model-error method are represented as boxplots in Figure 4.12.

The figure shows that in conditions with Very Good model fit, all of the model-error methods led to CRMR values that were generally at or below 0.05, the rule-of-thumb CRMR threshold for good model fit given by Hu and Bentler (1999). In general, all of the model fit methods led to similar CRMR distributions when model fit was Very Good. In those conditions, the largest differences in CRMR values between model-error methods occurred when there were many factors and low factor loadings, where the $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$ and $\mathrm{TKL}_{\mathrm{CFI}}$ model-error methods led to lower CRMR values compared to the other methods.

Larger differences between the model-error methods emerged when model fit was Poor. In those conditions, only the $\mathrm{TKL}_{\mathrm{RMSEA}}$ and WB methods consistently led to CRMR values that were near or above the threshold CRMR value for acceptable model fit given by Hu and Bentler (1999). In fact, the $\mathrm{TKL}_{\mathrm{CFI}}$ and $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$ methods often led to CRMR values that were less than the Hu and Bentler (1999) threshold for good model fit and were frequently the methods the led to the lowest CRMR values. The only conditions where the $\mathrm{TKL}_{\mathrm{CFI}}$ and $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$ methods led to higher median CRMR values were conditions with one-factor models, salient factor loadings of 0.8, and Poor model fit.

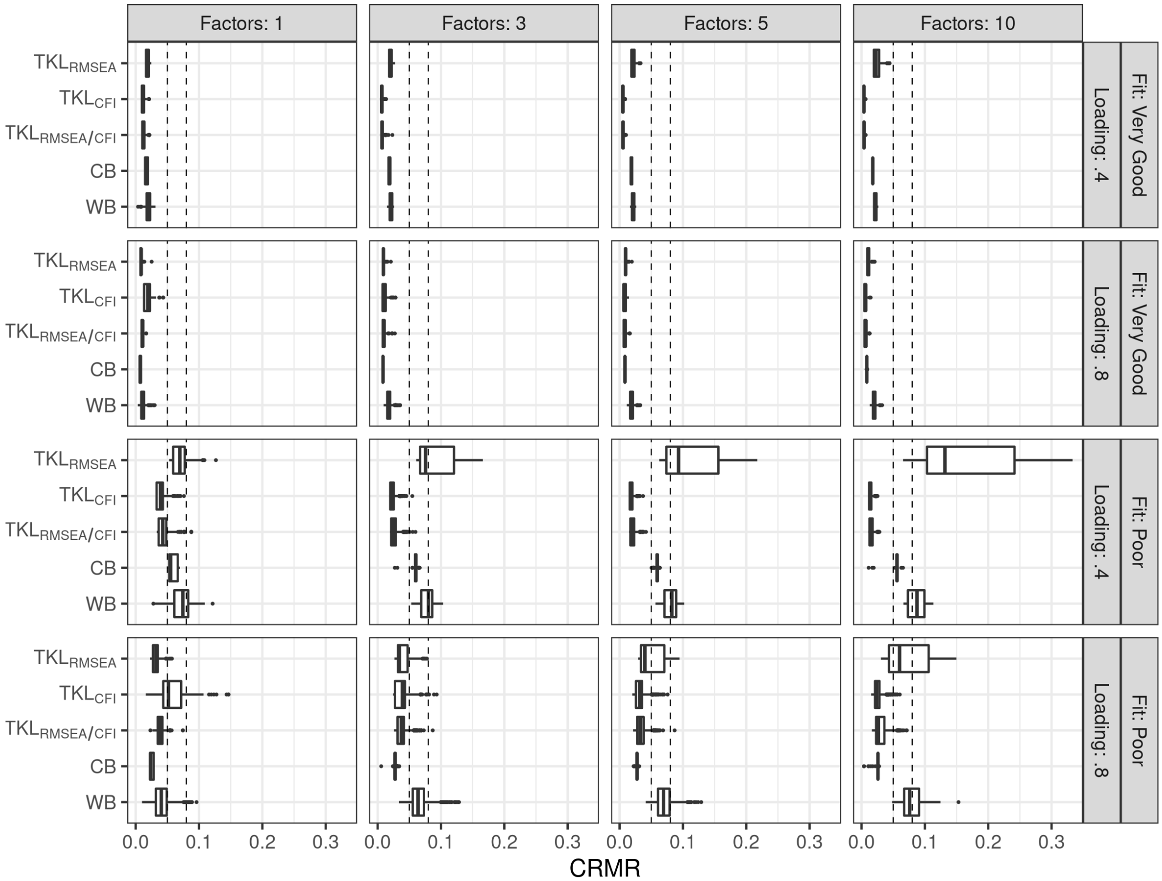


Figure 4.12. Distributions of the CRMR values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. Recommendations for threshold values of SRMR/CRMR are not as fine-grained as for some other fit indices, but threshold values of 0.05 and 0.08 were proposed by Hu and Bentler (1999) as more and less conservative upper-bounds for acceptable SRMR/CRMR values. The dashed lines indicate these two threshold values. Note that some levels of model fit and factor loading strength were omitted to conserve space. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

4.5 Fit Index Agreement

In the previous section, I reported results in terms of the RMSEA, CFI, TLI, and CRMR values individually. However, it is often recommended to evaluate model fit using more than one fit index (Hu & Bentler, 1999; Raykov & Marcoulides, 2012). Therefore, I was interested in the extent to which the model-error methods included in this simulation study led to solutions with fit indices indicating similar levels of model fit. In this section, I report the results in terms of fit index agreement. Specifically, I address two main questions. First, how well did model-error methods do at producing solutions with RMSEA and CFI values that were close to the target values corresponding to a particular level of model fit? Second, to what extent did the model-error methods produce solutions with fit index values that corresponded to the same qualitative interpretation of model fit? For both questions, I focused on the agreement between RMSEA and CFI, both for the sake of simplicity and because RMSEA and CFI were the fit indices that I used as target values for the model-error methods.

To evaluate how well each of the model error-methods did at producing solutions that had both RMSEA and CFI values that were close to the target values, I used the metric

$$D = |\mathrm{RMSEA}_{\mathrm{obs}} - \mathrm{RMSEA}_{\mathrm{T}}| + |\mathrm{CFI}_{\mathrm{obs}} - \mathrm{CFI}_{\mathrm{T}}|, \tag{4.1}$$

where $\mathrm{RMSEA}_{\mathrm{obs}}$ and $\mathrm{CFI}_{\mathrm{obs}}$ denote the observed RMSEA and CFI values for each solution and $\mathrm{RMSEA}_{\mathrm{T}}$ and $\mathrm{CFI}_{\mathrm{T}}$ denote the target RMSEA and CFI values. A small D value indicated that a solution was good in the sense that the observed RMSEA and CFI values were close to the target RMSEA and CFI values. On the other hand, a large D value indicated a poor solution with either one or both of the observed RMSEA and CFI values far from the corresponding target values.

Figure 4.13 shows box-plots of the D values for each of the model-error methods conditioned on the number of major common factors, model fit, and factor loading strength.[3] These variables had the largest effect on D values, as indicated by the effect sizes reported in the ANOVA summary table in Appendix B.1.4. The figure shows that all of the model-error method led to good results (i.e., small D values) in conditions with Very Good model

[3]Some conditions were omitted from this figure to conserve space. A full version is provided in Appendix B.2.9.

fit and strong factor loadings. Similarly, all of the model-error methods led to reasonably good results in conditions with one or three major factors, Poor model fit, and strong factor loadings. In the remaining conditions where the model-error methods did not all lead to small D values, the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods led to smaller D values than the other model-error methods. In particular, the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods led to much smaller D values than the alternative model-error methods in conditions with many major common factors and weak factor loadings, especially when model fit was Poor. The CB method typically led to the next-smallest D values in these conditions, followed by the $\text{TKL}_{\text{RMSEA}}$ and WB methods.

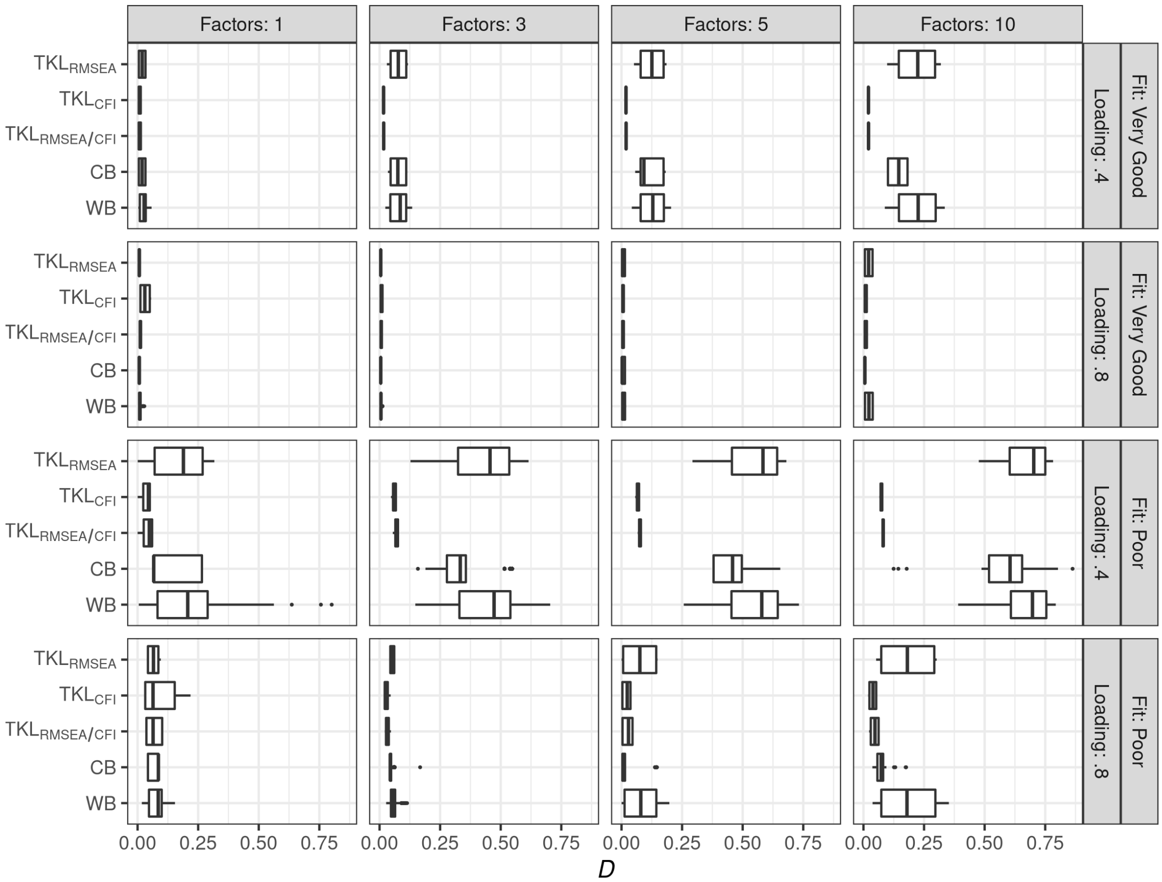


Figure 4.13. The sum of the absolute differences between the observed and target RMSEA and CFI values (D), conditioned on number of factors, model fit, and factor loading strength. Note that some levels of model fit and factor loading strength were omitted to conserve space. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

In addition to determining how far the observed RMSEA and CFI values differed from the target RMSEA and CFI values (in terms of D), I obtained a second perspective on model fit agreement by determining how often fit indices led to the same qualitative assessment of model fit using rule-of-thumb threshold values of RMSEA and CFI such as those reported by Hu and Bentler (1999). To calculate the rates of qualitative model fit agreement, I first assigned each simulated correlation matrix into one of three qualitative model fit categories based on its observed RMSEA and CFI values. For RMSEA, a simulated correlation matrix was considered to have good model fit if the observed RMSEA value was less than or equal to 0.05. If the RMSEA was greater than 0.05 but less than .10, it was considered to have acceptable model fit. RMSEA values greater than 0.10 were considered to represent unacceptable model fit. Similarly, observed CFI values greater than .95 were considered to represent good model fit, CFI values between .95 and .90 were considered to represent acceptable model fit, and CFI values below .90 were considered to represent unacceptable model fit.

The percent of simulated Σ matrices that led to fit indices indicating qualitative agreement on model fit are shown in Figure 4.14, conditioned on number of factors, factor loading strength, model fit, and model-error method. These variables had the largest effects on qualitative model fit index agreement rates, as indicated by the logistic regression coefficient estimates reported in Appendix B.1.5. Figure 4.14 shows that no single model-error method always led to the highest rates of qualitative fit agreement. When averaged over all conditions, the TKL_{CFI} model-error method led to the highest rate of qualitative fit index agreement (66.1%), followed by the CB (41.7%), $\text{TKL}_{\text{RMSEA/CFI}}$ (38.4%), $\text{TKL}_{\text{RMSEA}}$ (30.3%), and WB (29.4%) methods.

Breaking these results out further by level of target model fit, the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods almost always led to qualitative model fit agreement rates of 100% in conditions with Very Good model fit. The other model-error methods also led to qualitative model fit agreement rates of nearly 100% in conditions with Very Good model fit and strong factor loadings (i.e., Loading = 0.8), but led to much lower rates of agreement in conditions with Very Good model fit and weaker factor loadings. Results for conditions with Fair or Poor model fit were less straightforward. When target model fit was Fair, the TKL_{CFI} method often led to the highest rates of qualitative fit agreement of the model-error methods, particularly when factor loadings were weak and when there were many major common factors. When target model fit was Poor, all of the model-error methods had low qualitative fit agreement rates except in conditions with strong factor loadings.

Can you not print the numbers not
included in the design for factors
(e.g. 7). This is a minor issue
unless you think this should be
included in the publication

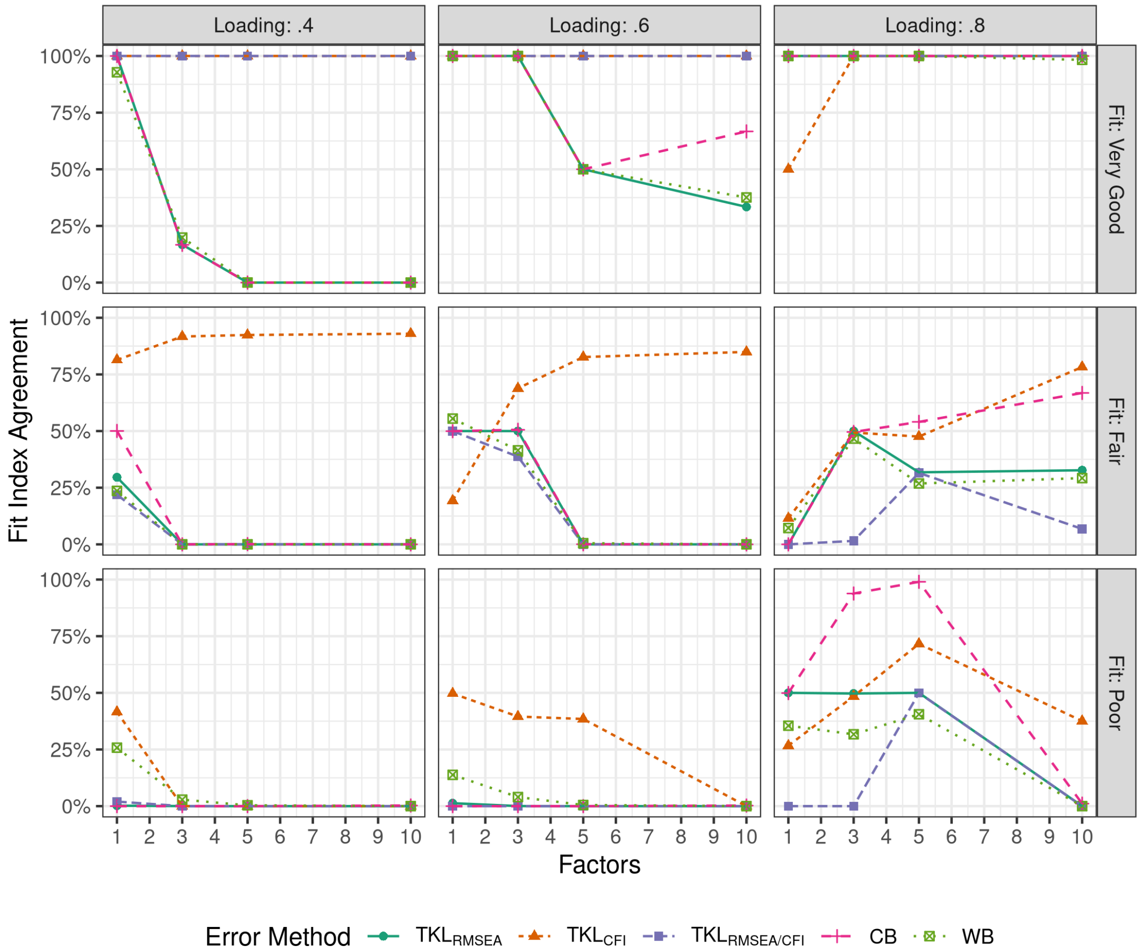


Figure 4.14. The percent of cases where the observed RMSEA and CFI values led to the same qualitative evaluation of model fit based on the threshold values suggested by Hu and Bentler (1999). TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

Although the TKL$_{\text{CFI}}$ and TKL$_{\text{RMSEA/CFI}}$ methods led to somewhat different results in terms of qualitative model fit agreement, it was both interesting and unexpected that these methods led to similar results in terms of D and observed RMSEA and CFI values, whereas the TKL$_{\text{RMSEA}}$ method often led to results that were more similar to those of the CB and WB methods. The similarity between the TKL$_{\text{CFI}}$ and TKL$_{\text{RMSEA/CFI}}$ results suggested that the CFI had more influence in Equation (2.1) than RMSEA, even when the

Is this redundant with above figure?

			Qualitative Fit Agreement (%)				
Factors	Loading	Model Fit	$\text{TKL}_{\text{RMSEA}}$	TKL_{CFI}	$\text{TKL}_{\text{RMSEA/CFI}}$	CB	WB
1	0.4	Very Good	100.0	100.0	100.0	100.0	92.8
1	0.4	Fair	29.6	81.4	21.9	50.1	23.6
1	0.4	Poor	0.2	41.6	2.0	0.0	25.8
1	0.6	Very Good	100.0	100.0	100.0	100.0	100.0
1	0.6	Fair	50.0	19.3	50.0	49.9	55.5
1	0.6	Poor	1.3	49.7	0.0	0.0	13.8
1	0.8	Very Good	100.0	50.0	100.0	100.0	100.0
1	0.8	Fair	0.0	11.5	0.0	0.0	7.2
1	0.8	Poor	50.0	26.7	0.0	49.9	35.5
3	0.4	Very Good	16.7	100.0	100.0	16.7	19.9
3	0.4	Fair	0.0	91.7	0.0	0.0	0.0
3	0.4	Poor	0.0	0.0	0.0	0.0	2.9
3	0.6	Very Good	100.0	100.0	100.0	100.0	100.0
3	0.6	Fair	50.0	68.9	38.7	50.6	41.5
3	0.6	Poor	0.0	39.5	0.0	0.0	4.0
3	0.8	Very Good	100.0	100.0	100.0	100.0	100.0
3	0.8	Fair	50.0	49.3	1.5	49.6	46.7
3	0.8	Poor	49.7	48.5	0.0	93.8	31.7
5	0.4	Very Good	0.0	100.0	100.0	0.0	0.0
5	0.4	Fair	0.0	92.4	0.0	0.1	0.0
5	0.4	Poor	0.0	0.0	0.0	0.0	0.4
5	0.6	Very Good	50.0	100.0	100.0	50.0	50.0
5	0.6	Fair	0.0	82.7	0.0	0.1	0.5
5	0.6	Poor	0.0	38.5	0.0	0.0	0.5
5	0.8	Very Good	100.0	100.0	100.0	100.0	100.0
5	0.8	Fair	31.8	47.6	31.5	54.1	26.9
5	0.8	Poor	50.0	71.6	49.9	99.0	40.5
10	0.4	Very Good	0.0	100.0	100.0	0.0	0.0
10	0.4	Fair	0.0	93.0	0.0	0.0	0.0
10	0.4	Poor	0.0	0.0	0.0	0.3	0.0
10	0.6	Very Good	33.4	100.0	100.0	66.7	37.6
10	0.6	Fair	0.0	84.9	0.0	0.0	0.0
10	0.6	Poor	0.0	0.0	0.0	0.3	0.0
10	0.8	Very Good	100.0	100.0	100.0	100.0	98.3
10	0.8	Fair	32.7	78.3	6.8	66.8	29.2
10	0.8	Poor	0.0	37.5	0.0	1.1	0.0

Note. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

Table 4.4
The percent of cases where the observed RMSEA and CFI values led to the same qualitative evaluation of model fit based on the threshold values suggested by Hu and Bentler (1999).

fit indices were equally weighted. Evidence for this hypothesis is provided in Figure 4.15, which shows observed RMSEA and CFI values for solutions from the $\text{TKL}_{\text{RMSEA}}$, TKL_{CFI}, and $\text{TKL}_{\text{RMSEA/CFI}}$ methods, conditioned on number of factors and model fit. The figure shows that solutions generated using the $\text{TKL}_{\text{RMSEA}}$ method had little variability in terms of RMSEA values. On the other hand, the CFI values for those solutions often had much more variability, particularly in conditions with weak factor loadings. When the TKL_{CFI} method was used, solutions had RMSEA and CFI values that were generally constrained to a small range within each condition. These results indicated that the range of possible CFI values was much larger for solutions with a fixed RMSEA value than the range of RMSEA values for solutions with a fixed CFI value in many conditions. Therefore, the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ might have led to such similar results because including a CFI target constrained the range of RMSEA values. Put another way, incorporating a target RMSEA value in addition to a target CFI value might have had little effect compared to using only a target CFI value because making small changes to CFI often led to large changes in RMSEA.

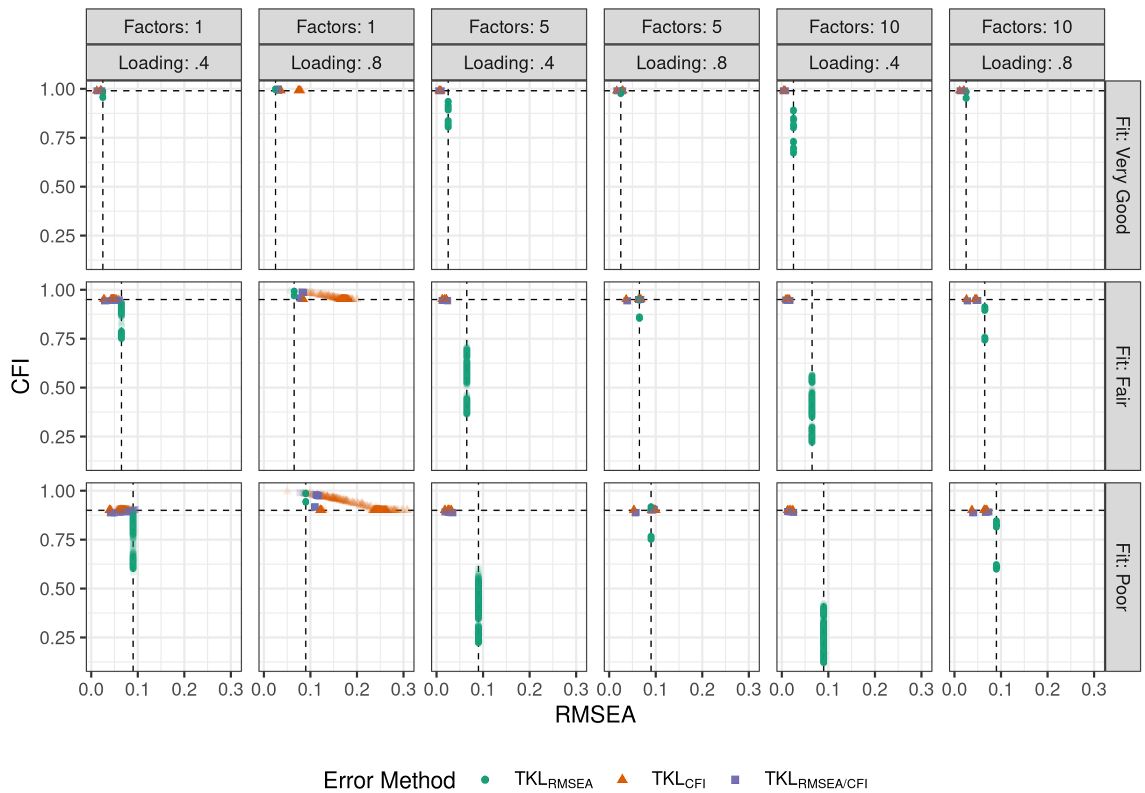


Figure 4.15. RMSEA and CFI values for the TKL-based model-error methods, conditioned on number of factors and model fit. The dashed vertical and horizontal lines indicate the target RMSEA and CFI values, respectively. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

To better understand the conditional distribution of CFI values for solutions with fixed RMSEA values, I used the $\text{TKL}_{\text{RMSEA}}$ method to generate solutions with a range of target RMSEA values for a subset of conditions from the main simulation study. Specifically, I generated 100 solutions for each of 16 target RMSEA values equally-spaced between 0.025 and 0.100 for each condition of the main simulation design with uncorrelated major common factors. (Conditions with correlated major common factors were omitted to make the number of conditions manageable.) The results can be seen in Figure 4.16, which shows the CFI and RMSEA values for each condition. The solid black lines in the figure indicate where RMSEA is equal to $1-\text{CFI}$ and can be used to easily determine which fit index changed most quickly as a function of the other fit index. RMSEA and $1 - \text{CFI}$ are comparable in this context because they indicate roughly the same level of qualitative model fit over the range of target RMSEA values. For example, both RMSEA $= 0.05$ and CFI $= .95$ have been used as threshold values for good model fit (Hu & Bentler, 1999). Figure 4.16 shows that CFI values decreased faster than RMSEA values increased for most of the included conditions, with the exception of conditions with relatively few major factors and strong factor loadings. Moreover, the figure shows that there was little variability in the conditional CFI values for a particular RMSEA value, except in conditions with few items and weak factor loadings.

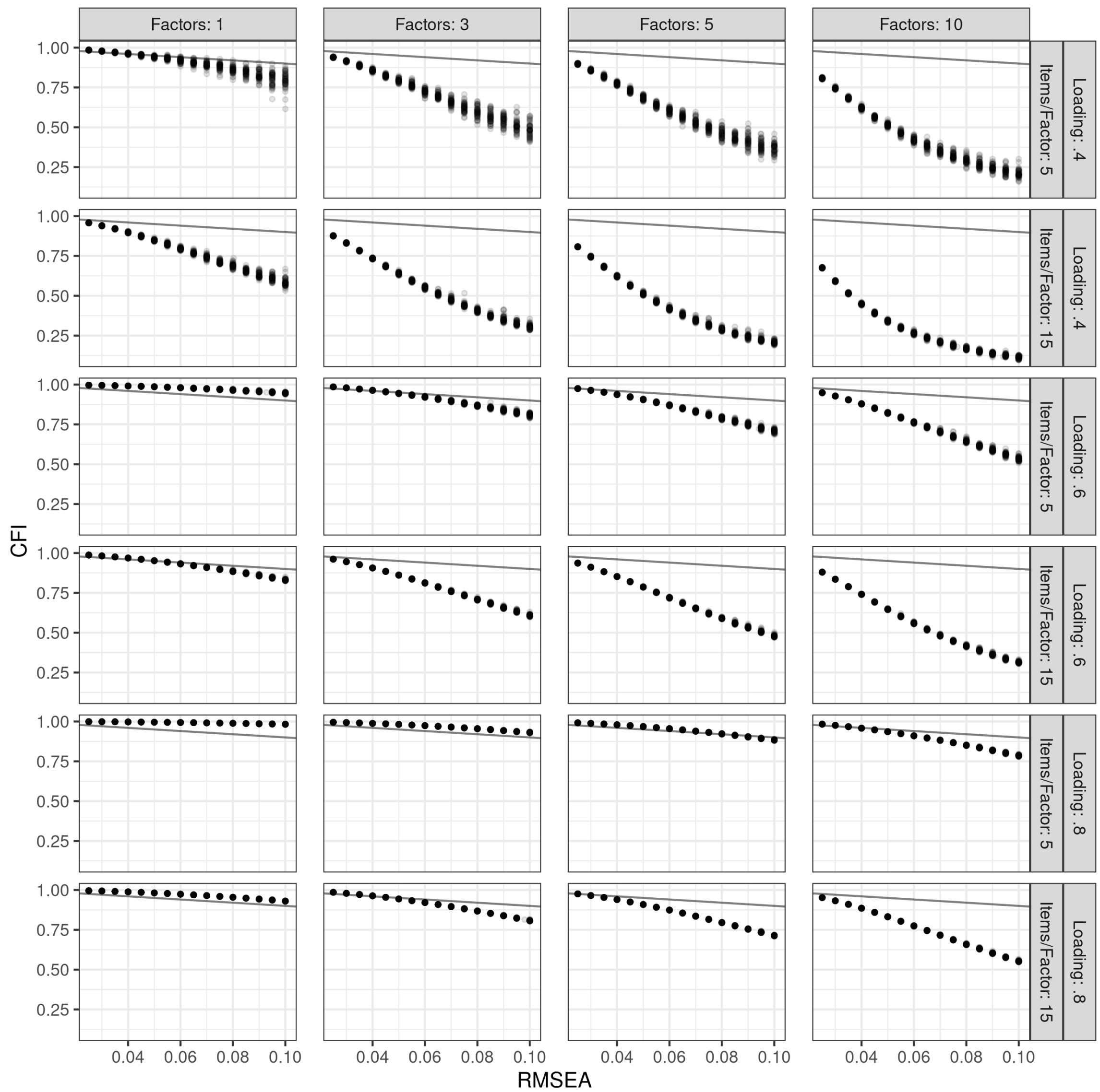


Figure 4.16. Observed CFI and RMSEA values for solutions generated using the $\text{TKL}_{\text{RMSEA}}$ method. For each combination of number of factors, number of items per factor, and factor loading strength, 100 solutions were generated for each of 16 target RMSEA values equally-spaced between 0.025 and 0.100. The solid black line indicates where RMSEA and $1 - \text{CFI}$ were equal.

In addition to looking at the conditional distribution of CFI values at fixed RMSEA values, I also looked at the conditional distribution of RMSEA values at fixed CFI values. Similar to the previous procedure for RMSEA, I used the TKL_{CFI} method to generate 100 solutions with 19 target CFI values equally-spaced between .90 and .99 for each condition of the main simulation design with uncorrelated major common factors. The results are shown in Figure 4.17. As in the previous figure, the points in Figure 4.17 were often below the solid black line indicating (roughly) equivalent qualitative model fit. Points falling below the line denoted cases with RMSEA values that indicated better model fit than their corresponding CFI values. The figure also shows that CFI changed more quickly as a function of RMSEA than RMSEA changed as a function of CFI. Finally, the figure shows that there was little variability in the conditional RMSEA values for a particular CFI value in all of the conditions except conditions with one factor and five items per factor.

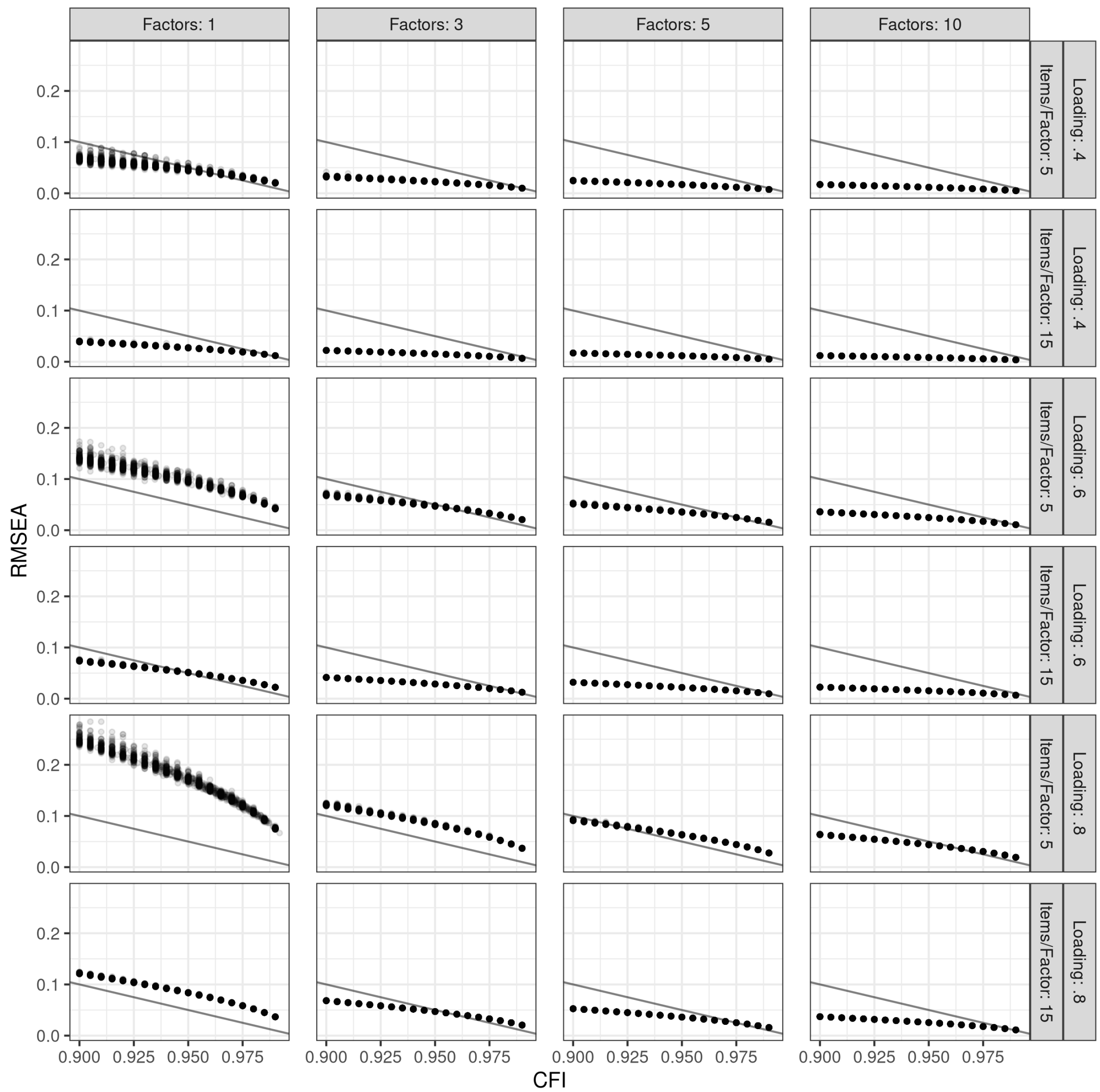


Figure 4.17. Observed RMSEA and CFI values for solutions generated using the TKL_{CFI} method. For each combination of number of factors, number of items per factor, and factor loading strength, 100 solutions were generated for each of 19 target CFI values equally-spaced between 0.90 and 0.99. The solid black line indicates where RMSEA and $1 - \text{CFI}$ were equal.

Taken together, the results in Figure 4.15, Figure 4.16, and Figure 4.17 can help explain why the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ model-error methods often led to similar results. Because CFI values decreased more quickly than RMSEA values increased in most conditions,

The writing in this section is
very good.

changes to parameter values that led to an RMSEA value that was closer to the target RMSEA value but moved the CFI value further from the target CFI value were more costly than changes that prioritized the target CFI value. Moreover, Figure 4.16 and Figure 4.17 also provided indications of which conditions were most likely to lead to conflicting qualitative interpretations of RMSEA and CFI. Points far from the line indicated RMSEA and CFI pairs that led to conflicting qualitative interpretations of model fit because RMSEA and $1 - \text{CFI}$ indicated approximately the same qualitative model fit over the range of target RMSEA values. Both figures show that conditions with weak factor loadings, many factors and items, and high RMSEA values (or low CFI values) were most likely to result in conflicting fit index interpretations.

The frequent disagreement between RMSEA and CFI can be also be understood by considering how these fit indices differ in how they describe model fit. Recall from Section 1.2 that in the population context, absolute fit indices (e.g., RMSEA) indicate how different the Σ and Ω matrices are from one another. On the other hand, relative model fit indices (e.g., CFI) indicate how much better the population model fits Σ compared to the independence model. Generating a Σ matrix corresponding to a particular RMSEA value is therefore a relatively straightforward matter of perturbing Ω until the discrepancy between Σ and Ω results in the target RMSEA value. On the other hand, generating a Σ matrix corresponding to a particular CFI value is more complex because CFI takes into account both the fit of the hypothesized model indicated by Ω and the fit of the baseline (independence) model. Thus, perturbing the elements of Ω to create Σ negatively affects the fit of the baseline model in addition to the fit of the hypothesized model because the independence model implies that Σ should be an identity matrix. However, unless the hypothesized model leads to a much worse fit (i.e., a much smaller minimized discrepancy function value) than the baseline model, the resulting CFI value will be relatively small.

This is demonstrated in Figure 4.18, which shows the minimized discrepancy function values for the hypothesized and baseline models as the number of major factors increased

Not clear which discrepancy
function you mean here. Smaller
MLE F means better fit

from one to ten, along with the corresponding CFI and RMSEA values. The hypothesized models (i.e., the population models without model error) were all orthogonal models with salient major factor loadings of 0.4, 15 items per factor, and between 1 and 10 common factors. The Σ matrices were generated using the $\text{TKL}_{\text{RMSEA}}$ method with a target RMSEA value of 0.09. To interpret Panel A of the figure, recall that $\text{CFI} = 1 - F_t/F_b$ as defined in Equation (1.16), where F_t/F_b is the ratio of the minimized discrepancy function values for the hypothesized and baseline models. Panel A of Figure 4.18 shows that both F_t and F_b increased as the number of factors increased, holding everything else constant. However, F_t grew fast enough that F_t/F_b decreased as the number of factors increased, resulting in lower CFI values as RMSEA remained fixed (as shown in Panels B and C). To keep CFI constant as the number of factors increased, F_t would have needed to increase at the same rate as F_b. For instance, Panel A includes a line indicating the values of F_t required to produce a CFI value of 0.90 for each number of factors. Although the figure only shows results for a specific set of conditions, it demonstrates the tension between RMSEA and CFI that occurs as the order of Ω increases. Specifically, it shows that fixing RMSEA at a value indicating Fair or Poor model fit when there are many items or factors requires an F_m value that is too large (relative to F_b) to produce an acceptable CFI value.

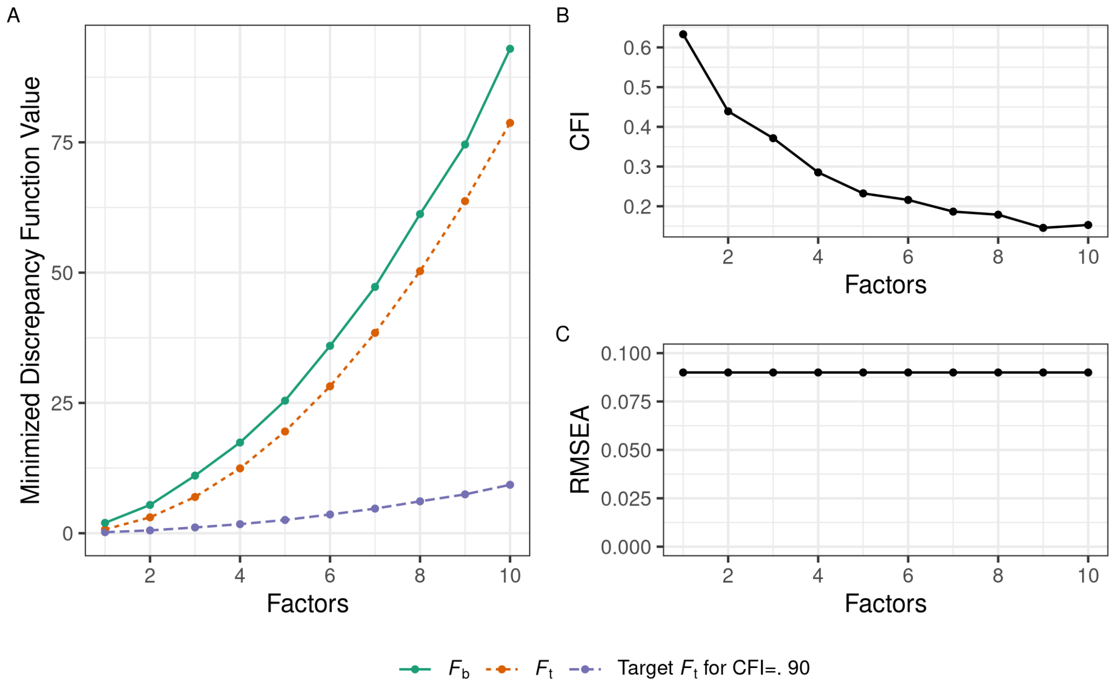


Figure 4.18. Panel A: Minimized discrepancy function values, CFI, and RMSEA values for Σ matrices generated from orthogonal models with salient major factor loadings of 0.4, 15 items per factor, and between 1 and 10 factors. The Σ matrices were generated using the $\text{TKL}_{\text{RMSEA}}$ model-error method with a target RMSEA value of 0.09. The line in the left-most panel labeled "Target for CFI = .90" indicates the value of F_t that would be needed to obtain a CFI value of .90, given the value of F_b. Panels B and C: Observed CFI and RMSEA values for each simulated Σ matrix in Panel A.

The trade-off between CFI and RMSEA can also be seen by examining the differences between the $\mathbf{\Sigma}$ matrices produced by each of the model-error methods for a condition that often led to conflicting RMSEA and CFI values. Figure 4.19 shows heat-maps of the $\mathbf{\Sigma}$ matrices for each of the five model-error methods. The Σ matrices corresponded to the condition with ten orthogonal major common factors, five items per factor, weak factor loadings of 0.3, and Poor model fit.[4] The figure shows clear differences between the model-

[4]Heat-maps of both the Σ and $\hat{\Omega}$ matrices for the five model-error methods and conditions with ten orthogonal factors, weak factor loadings, Poor model fit, and five or ten items per factor are shown in Appendix B.2.12.

error methods that incorporated a target CFI value (TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$) and the remaining model-error methods that only incorporated a target RMSEA value. The Σ matrices produced by the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods included relatively small amounts of model error (in terms of the differences between the elements of Σ and Ω), as can be seen by looking at their heat-map representations in Figure 4.19. The block-diagonal structure of Ω was well-preserved in both of the Σ matrices produced by the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} model-error methods, with no large off-block-diagonal correlations. On the other hand, the Σ matrices produced by the other model-error methods had many off-block-diagonal correlations that were relatively large, many as large (or larger) than the block-diagonal correlations. The figure thus reaffirms the previously reported result that the model-error methods that included a target CFI value led to solutions that perturbed Ω less than solutions from model-error methods that did not include a target CFI value.

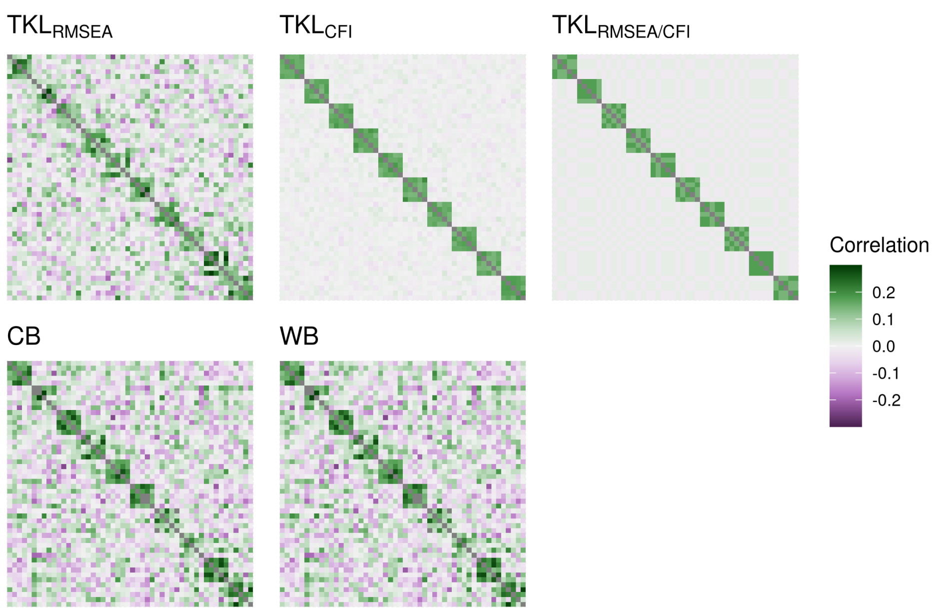


Figure 4.19. Correlation matrices with model error ($\mathbf{\Sigma}$) for each model-error method from the condition with ten orthogonal factors, five items per factor, weak factor loadings of 0.3, and Poor model fit. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

4.6 Fit Indices Indicating Lack-of-Fit Between Σ and $\hat{\Omega}$

Although the lack-of-fit between Σ and Ω was of primary interest in this dissertation, the lack-of-fit between Σ and $\hat{\Omega}$ might also be of interest to some researchers. Recall that $\hat{\Omega}$ denotes the implied population correlation matrix obtained by analyzing Σ using the major-factor model. Thus, the fit index values based on Σ and $\hat{\Omega}$ represent the "best-case scenario" fit index values a researcher could expect to obtain because the lack-of-fit is due only to model approximation error without any sampling error.

Figures reporting the distributions of RMSEA, CFI, TLI, and CRMR values representing the lack-of-fit between Σ and $\hat{\Omega}$ (analogous to Figure 4.9, Figure 4.10, Figure 4.11, and Figure 4.12) are provided in Appendix B.2.8. In addition to the distributions of the $\hat{\Omega}$ fit indices, I was also interested in how they related to the corresponding fit indices indicating lack-of-fit between Σ and Ω. In particular, I was interested in determining how $\text{RMSEA}_{\hat{\Omega}}$ was related to RMSEA and how $\text{CFI}_{\hat{\Omega}}$ was related to CFI.

4.6.1 Differences between RMSEA and $\text{RMSEA}_{\hat{\Omega}}$

To better understand the relationship between the observed $\text{RMSEA}_{\hat{\Omega}}$ and RMSEA values from the simulation study, Figure 4.20 shows box-plots of the differences between $\text{RMSEA}_{\hat{\Omega}}$ and RMSEA, denoted as

$$\text{RMSEA}_{\Delta} = \text{RMSEA} - \text{RMSEA}_{\hat{\Omega}}, \tag{4.2}$$

conditioned on each of the model-error methods, factor loading strength, and level of model fit. These variables had the largest effect on RMSEA_{Δ}, as indicated by the effect sizes reported in the ANOVA summary table in Appendix B.1.6. Figure 4.20 shows that the $\text{RMSEA}_{\hat{\Omega}}$ values were almost always lower than the RMSEA values corresponding to the same Σ matrix, resulting in positive values of RMSEA_{Δ}. The vast majority of these differences were quite small, with a median difference of only .003. However, Figure 4.20 shows that RMSEA_Δ was affected by the level of model fit such that RMSEA_Δ was largest when model fit was Poor and smallest when model fit was Very Good. In the most extreme instance, a Σ matrix with Poor model fit had an $\text{RMSEA}_{\hat{\Omega}}$ value of .006 and an RMSEA value of .308, indicating very different qualitative interpretations of model fit. Thus, although the differences between RMSEA and $\text{RMSEA}_{\hat{\Omega}}$ were often negligible, in some cases they were large enough to indicate completely different qualitative interpretations of model fit for RMSEA and $\text{RMSEA}_{\hat{\Omega}}$.

Haven't you switched ordering of model fit. Earlier it was good models on top. Be consistent throughout document.

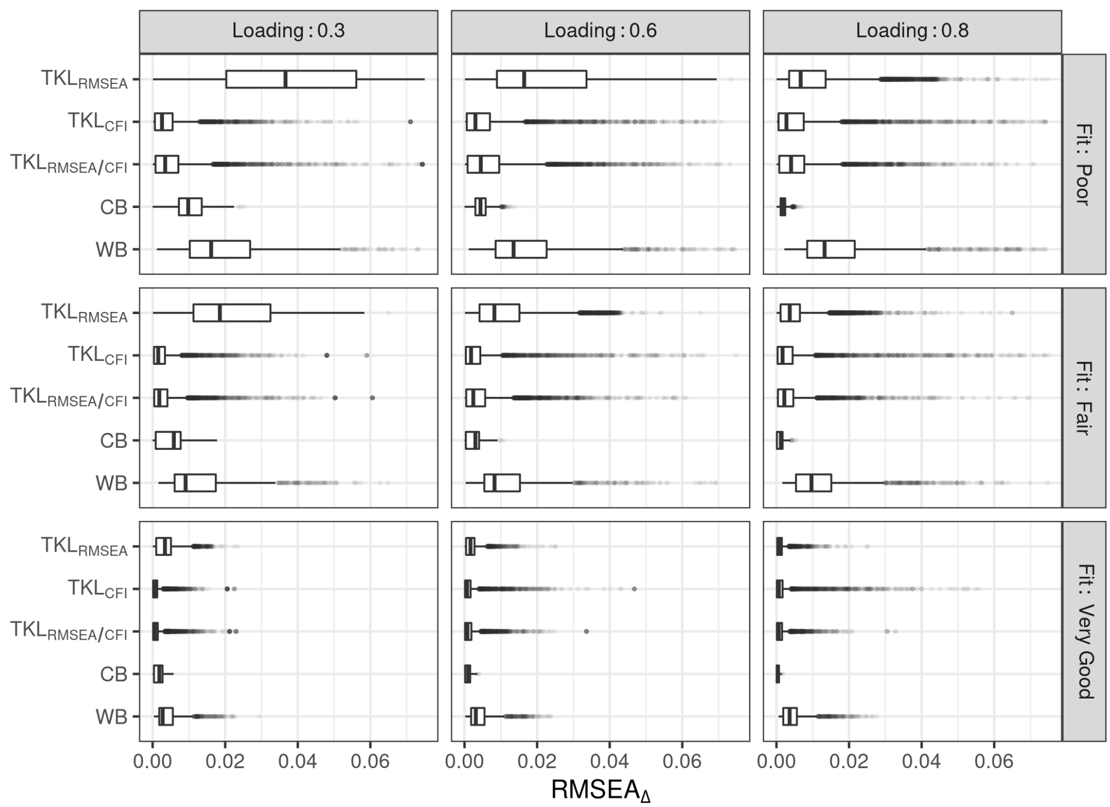


Figure 4.20. Box-plots of the differences between RMSEA and $\text{RMSEA}_{\hat{\Omega}}$ (denoted as RMSEA_Δ) conditioned on model-error method, levels of model fit, and factor loading. Note that some outliers were omitted from the plot to aid visualization. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

Figure 4.20 also shows that the relationship between RMSEA and $\text{RMSEA}_{\hat{\Omega}}$ differed depending on the model-error method that was used. For instance, the CB method consistently produced Σ matrices with small RMSEA$_\Delta$ values, whereas the other model-error methods often produced solutions with large RMSEA$_\Delta$ values. In theory, the CB method should have led to solutions with equal RMSEA and RMSEA$_{\hat{\Omega}}$ values because $\hat{\Omega} = \Omega$ when the CB method converges to a valid solution (see Section 1.2). However, in the present simulation study the CB method produced RMSEA and RMSEA$_{\hat{\Omega}}$ values that were often very close but not exactly equal (corresponding to small but non-zero RMSEA$_\Delta$ values). The RMSEA$_\Delta$ values for the CB method were largest when model fit was Poor and smallest when model fit was Very Good, as can be seen by moving from top to bottom of Figure 4.20. Although these RMSEA$_\Delta$ values were often non-zero, they were generally small enough to be of little concern to most researchers. It is likely that these discrepancies were due to the CB method converging to local minima, which Cudeck and Browne (1992) acknowledge is possible when the target RMSEA value is large.

In addition to being affected by choice of model-error method, RMSEA$_\Delta$ was affected by the strength of the major common factor loadings. Moving across Figure 4.20 from left to right, the differences between RMSEA and RMSEA$_{\hat{\Omega}}$ values tended to decrease as factor loadings increased from 0.3 to 0.8. The figure also shows that there was an interaction between model-error method and factor loading such that RMSEA$_\Delta$ decreased the most for the TKL$_{\text{RMSEA}}$ and CB methods as factor loadings increased, compared to the alternative model-error methods.

The number of major common factors and the number of items per factor also had effects on RMSEA$_\Delta$. These effects are shown in Figure 4.21, which contains box-plots of RMSEA$_\Delta$ conditioned on number of factors and number of items per factor. The figure shows that the TKL-based methods were most affected by the number of factors and the number of items per factor, but that the direction of the effect depended on the particular method. For instance, the TKL$_{\text{RMSEA}}$ method led to solutions with median RMSEA$_\Delta$ values that tended to increase along with the number of major factors when there were 15 items per factor. For conditions with only five items per factor, the median RMSEA$_\Delta$ first decreased as the number of factors increased from one to three, and then increased slightly as the number of factors increased further. In contrast to $\text{TKL}_{\text{RMSEA}}$ method, the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} methods led to median RMSEA_{Δ} values that decreased as the number of factors increased for conditions with five items per factor. For conditions with 15 items per factor, the median RMSEA_{Δ} remained stable as the number of factors increased. For the CB and WB model-error methods, the number of factors had little effect on median RMSEA_{Δ}. Finally, all of the model-error methods produced solutions with smaller median RMSEA_{Δ} values as the number of items per factor increased, with the exception of the $\text{TKL}_{\text{RMSEA}}$ method for conditions with five or ten factors.

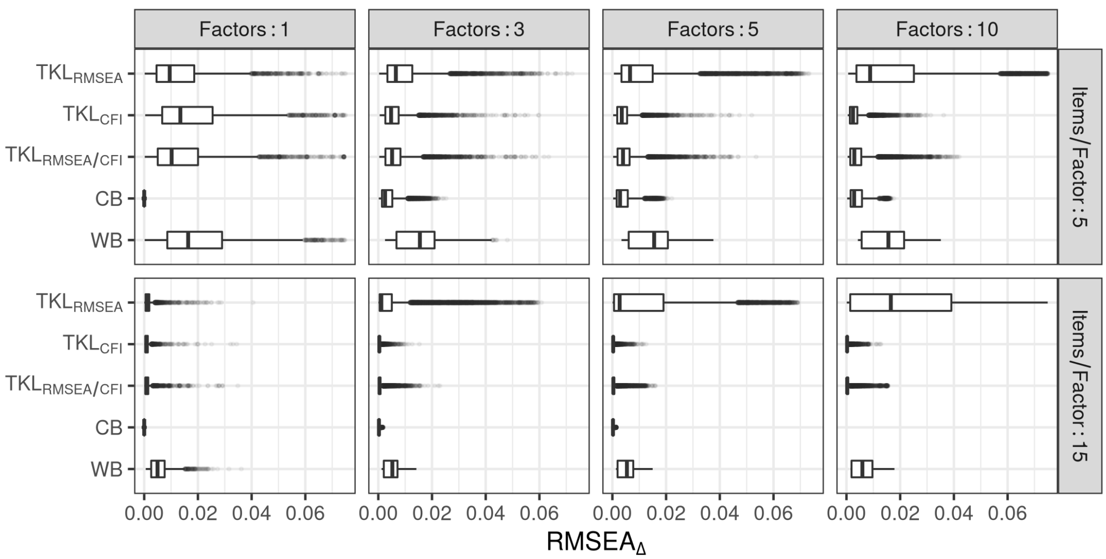


Figure 4.21. Box-plots of the differences between RMSEA and $\text{RMSEA}_{\hat{\Omega}}$ (denoted as RMSEA_{Δ}) conditioned on model-error method, number of factors, and number of items per factor. Note that some outliers were omitted from the plot to aid visualization. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

I would add Omega subscript to distinguish rmsea_\Sigma

4.6.2 Differences between CFI and $\text{CFI}_{\hat{\Omega}}$

As with RMSEA and $\text{RMSEA}_{\hat{\Omega}}$, the difference between CFI and $\text{CFI}_{\hat{\Omega}}$ was denoted as

$$\text{CFI}_\Delta = \text{CFI} - \text{CFI}_{\hat{\Omega}}. \tag{4.3}$$

Figure 4.22 contains box-plots of CFI_Δ conditioned on model-error method, level of model fit, and factor loading strength. Similar to the previous results for RMSEA_Δ, the figure shows that CFI_Δ values were largest when model fit was Poor and smallest when model fit was Very Good. Moreover, the figure also shows differences among the model-error methods indices. In general, the $\text{TKL}_{\text{RMSEA}}$ and WB methods led to the most extreme CFI_Δ values[5], whereas the $\text{TKL}_{\text{RMSEA/CFI}}$, TKL_{CFI}, and CB methods often led to less extreme values, particularly in conditions with weak factor loadings or Poor model fit. In fact, the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} model-error methods sometimes led to CFI_Δ values that were less extreme than those from the CB method. As discussed in the previous section, this was somewhat surprising because the CB method was expected to produce solutions such that $\Omega = \hat{\Omega}$ and $\text{CFI}_\Delta = 0$ and suggests that the CB method often converged to local minima in conditions with weak factor loadings and Poor or Fair model fit.

[5] For instance, the most extreme value, $\text{CFI}_\Delta = .818$, was obtained using the $\text{TKL}_{\text{RMSEA}}$ model-error method and a model with ten factors (correlated .3 with one another), 15 items per factor, weak factor loadings, and Poor model fit.

I am surprised that is
possible

Switch order for fit to make all figs consistent

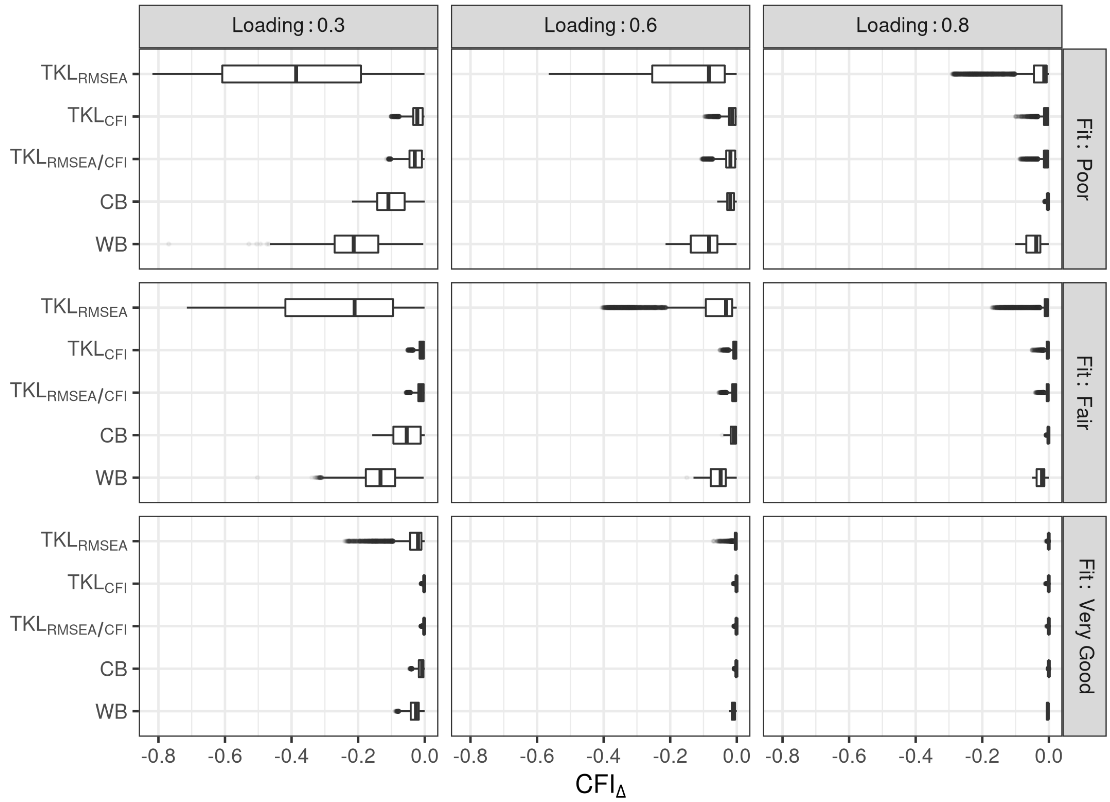


Figure 4.22. Box-plots of the differences between CFI and CFI$_{\hat{\Omega}}$ (denoted as CFI$_\Delta$) conditioned on model-error method, levels of model fit, and factor loading. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

In addition to the effects of factor loading and model fit, CFI$_\Delta$ was also affected by the number of factors and items per factor. To help understand these effects, Figure 4.23 shows box-plots of CFI$_\Delta$ conditioned on model-error method, number of factors, and number of items per factor. The figure shows that CFI$_\Delta$ tended to increase as the number of factors increased for the TKL$_{\text{RMSEA}}$ and WB methods, but that the number of factors had only a small effect (or no effect at all) on CFI$_\Delta$ for the other model-error methods. Considering the effect of number of items per factor, Figure 4.23 shows that the number of items per factor had almost no effect on the median CFI$_\Delta$ values for each of the model-error methods except for the TKL$_{\text{RMSEA}}$ method in conditions with ten factors. However, there was more variation in CFI_Δ values for the $\text{TKL}_{\text{RMSEA}}$ method in conditions with 15 items per factor compared to conditions with five items per factor. In contrast, the CB and WB methods led to less variable CFI_Δ values as the number of items per factor increased from five to 15.

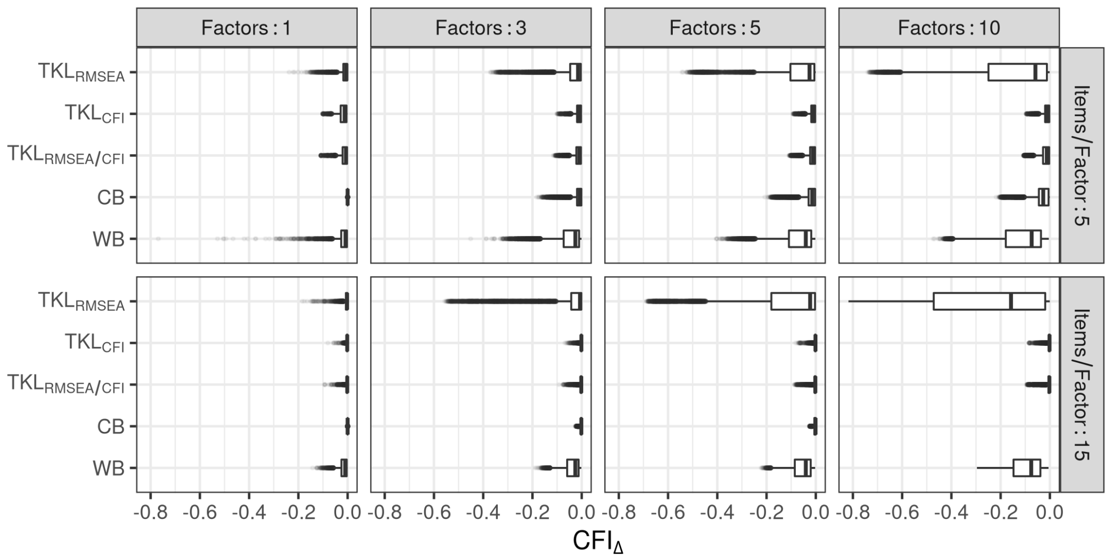


Figure 4.23. Box-plots of the differences between CFI and $\text{CFI}_{\hat{\Omega}}$ (denoted as CFI_Δ) conditioned on model-error method, number of factors, and number of items per factor. Note that some outliers were omitted from the plot to aid visualization. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

Overall, the results of the present simulation study suggested that any of the model-error methods evaluated in this simulation study could provide a useful starting point for researchers who would like to generate Σ matrices with particular $\text{RMSEA}_{\hat{\Omega}}$ or $\text{CFI}_{\hat{\Omega}}$ values. However, all of the model-error methods (including the CB methods) sometimes produced solutions with large RMSEA_Δ or CFI_Δ values, particularly in conditions with Poor model fit and weak factor loadings. Thus, researchers interested in generating solutions with particular $\text{RMSEA}_{\hat{\Omega}}$ or $\text{CFI}_{\hat{\Omega}}$ values should check the fit indices of each solution and reject solutions with fit indices outside of the desired range. Because the fit indices based on $\hat{\Omega}$ almost always indicated better fit than the indices based on Ω, it might also be useful to specify target RMSEA values that are slightly higher (or CFI values that are slightly lower) than the desired $\text{RMSEA}_{\hat{\Omega}}$ or $\text{CFI}_{\hat{\Omega}}$ values.

4.7 Model Fit Index Recovery for the TKL-Based Methods

The TKL-based model-error methods were designed to find a Σ matrix that had either an RMSEA value or a CFI value (or both) that were close to a specified value. When only one target model-fit index was used (i.e., $\text{TKL}_{\text{RMSEA}}$ or TKL_{CFI}), the observed RMSEA and CFI values were very close to the target values. However, when both RMSEA and CFI fit index targets were used simultaneously with the $\text{TKL}_{\text{RMSEA/CFI}}$ method, many solutions failed to have RMSEA and CFI values that were both very close to the target values. This could indicate that the optimization procedure was not working well and often failed to find an optimal solution. However, an alternative explanation is that some combinations of RMSEA and CFI might not have been possible for certain conditions (e.g., with major factor loadings fixed at a particular value, or with a certain number of major common factors).

Good

To determine whether the $\text{TKL}_{\text{RMSEA/CFI}}$ was able to find near-optimal solutions, I first needed to find combinations of RMSEA and CFI values that were known to be possible. If the $\text{TKL}_{\text{RMSEA/CFI}}$ method was able to produce solutions with RMSEA and CFI values that were close to these target RMSEA and CFI values, it would suggest that the $\text{TKL}_{\text{RMSEA/CFI}}$ was working well. More importantly, if the $\text{TKL}_{\text{RMSEA/CFI}}$ method was not able to produce solutions with fit indices close to these known-to-be-possible combinations of RMSEA and CFI, it would suggest that researchers cannot rely upon the $\text{TKL}_{\text{RMSEA/CFI}}$ model-error method to produce optimal or near-optimal solutions.

How does that follow. You just need to search for a global from multiple starts

To find RMSEA and CFI value combinations that were known to be possible, I used the standard TKL method implemented in the `simFA()` function to generate a correlation matrix with model error for every condition in the simulation design. Next, I computed the RMSEA and CFI values for each simulated correlation matrix. I then used those values as target RMSEA and CFI values for the $\text{TKL}_{\text{RMSEA/CFI}}$ method and generated 50 correlation matrices with model error for each condition .[6]

The results from this small simulation study are reported in Figure 4.24, which shows the known-to-be-possible target values of RMSEA and CFI (indicated by solid black lines) and the observed RMSEA and CFI values from the $\text{TKL}_{\text{RMSEA/CFI}}$ method for each condition. The figure shows that the observed RMSEA and CFI values were nearly identical to the target values for most conditions. The conditions where the observed fit indices had the most variability were conditions with one or three factors, five items, and weak factor loadings. However, even in those conditions many solutions had CFI and RMSEA values that were nearly identical to the target values. These results indicate that using the $\text{TKL}_{\text{RMSEA/CFI}}$ method repeatedly with different initial values of $\mathbf{W}$, ϵ, and ν_{e} and then selecting the solution with RMSEA and CFI values closest to the target values seems to be an effective approach for obtaining optimal (or nearly-optimal) solutions.

[6] Code for this simulation study is provided in Appendix A.5.

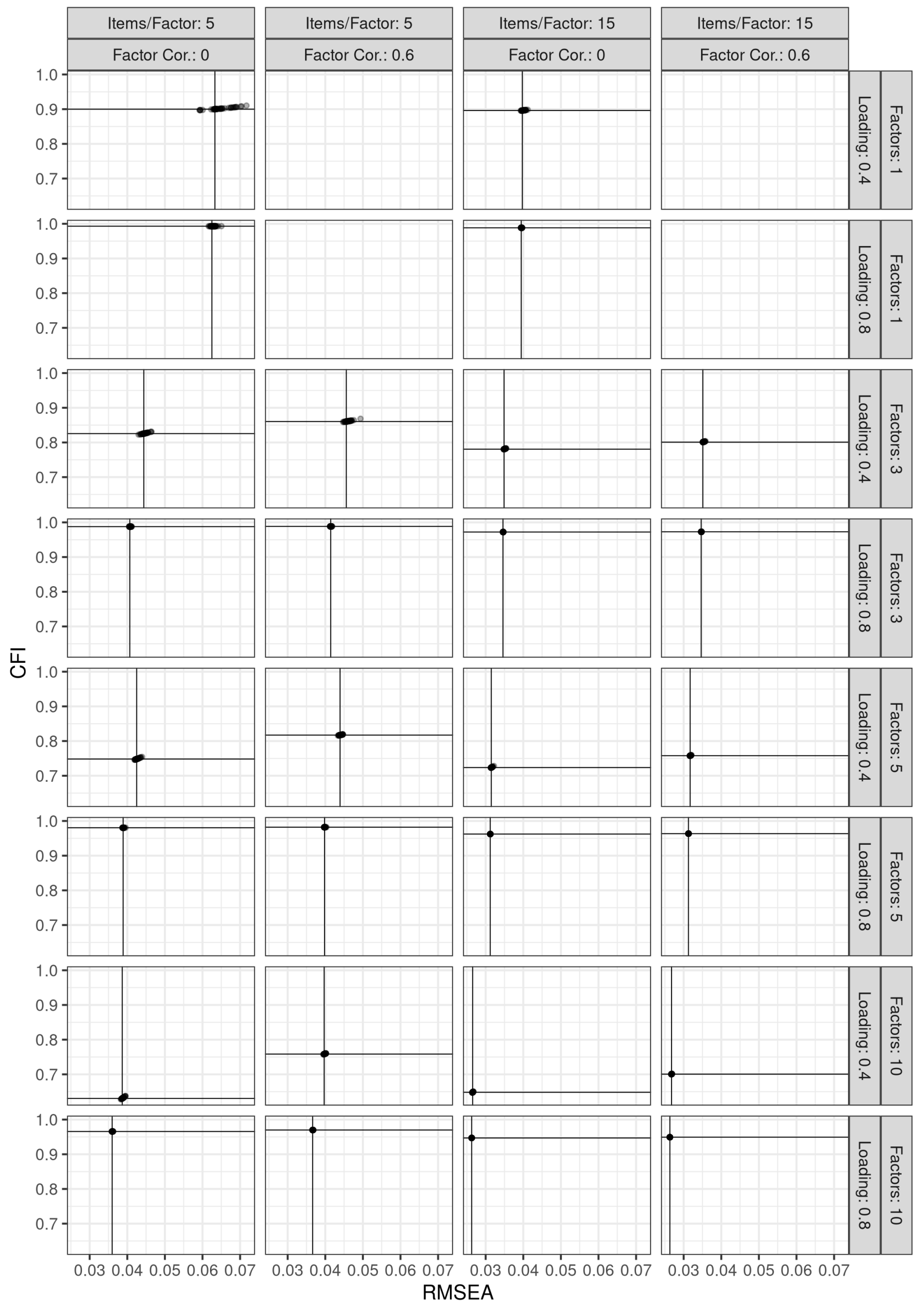


Figure 4.24. Observed RMSEA and CFI values for the $\text{TKL}_{\text{RMSEA/CFI}}$ model-error method and the corresponding known-to-be-possible target RMSEA and CFI values (indicated by the black lines), conditioned on number of factors, number of items per factor, factor loading strength, and factor correlation. Note that some levels were omitted to conserve space.

Are they reported in a supplement. Did they show anything interesting?

Chapter 5

Discussion

In this dissertation, I conducted a large-scale simulation study to compare several methods for simulating population correlation matrices with model error. The model-error methods I compared were the Cudeck and Browne (CB; 1992) method, the Wu and Browne (WB; 2015a) method, and three variations of the Tucker, Koopman, and Linn (TKL; 1969) model-error method using a novel optimization procedure to automatically select values of the TKL method parameters, ϵ and ν_{e}. The addition of the optimization procedure allowed the TKL method to be used with specified target values of RMSEA (the $\mathrm{TKL}_{\mathrm{RMSEA}}$ variation), CFI (the $\mathrm{TKL}_{\mathrm{CFI}}$ variation), or both fit indices simultaneously (the $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$ variation). Moreover, the optimization procedure also allowed users to impose constraints on the loadings of the minor common factors introduced by the TKL method to ensure that there was a clear delineation between major and minor common factors. To facilitate the use of all of the model-error methods discussed in this dissertation, I also developed an R package (*noisemaker*) that serves as an easy-to-use, unified interface for simulating correlation matrices with model error.

Through this simulation study, I hoped to answer two primary questions about the model-error methods I investigated. First, I wanted to know whether the five model-error methods included in the study (the $\mathrm{TKL}_{\mathrm{RMSEA}}$, $\mathrm{TKL}_{\mathrm{CFI}}$, $\mathrm{TKL}_{\mathrm{RMSEA/CFI}}$, CB, and WB methods) led to different values of the CFI, TLI, and CRMR fit indices when used with the same

No! They are only fungible
w.r.t. fit indices not the type
of error structure or
implications for parameter
estimates.

error-free population correlation matrices and target RMSEA values. If all of the model-error methods led to the same (or similar) fit index values, it would have suggested that the choice of which model-error method to use is not very important when conducting simulation studies involving covariance structure models. The second question I wanted to answer was related to the efficacy of the modified TKL method with the proposed optimization procedure (referred to as the multiple-target TKL method). That is, I was interested in determining how well the multiple-target TKL method was able to generate correlation matrices with model error that had RMSEA and CFI values that were close to the specified values. Note that in the following discussion of the simulation results I focus on the RMSEA and CFI fit indices for two reasons. First, RMSEA and CFI target values were used in the simulation study and are often used as indications of model fit when generating population correlation matrices with model error (Cudeck & Browne, 1992; Kracht & Waller, 2022; Trichtinger & Zhang, 2020; Tucker et al., 1969). Second, the CRMR and TLI indices led to results that were similar to the results for RMSEA and CFI, respectively.

Concerning the first primary research question, the results indicated that there were important differences between the five model-error methods in terms of the observed model fit indices they led to. Although all of the model-error methods led to similar RMSEA, CFI, TLI, and CRMR values in conditions with few major factors, strong factor loadings, and good model fit, they led to much more disparate results in other conditions. In particular, the results of the simulation study indicated that there were important differences between model-error methods that incorporated only a target RMSEA value (i.e., the $\text{TKL}_{\text{RMSEA}}$, CB, and WB methods) and methods that incorporated a target CFI value (the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods).

When evaluated on RMSEA, the model-error methods that only incorporated target RMSEA values generally produced solutions with observed RMSEA values very close to the target values. In particular, the $\text{TKL}_{\text{RMSEA}}$ and CB methods led to RMSEA values that almost always extremely close to the target values, whereas there was slightly more variability in the RMSEA values from the WB method. On the other hand, the model-error methods that incorporated CFI target values often led to RMSEA values that were lower than the target values, particularly in conditions with many major common factors and weak factor loadings. The result that the $\text{TKL}_{\text{RMSEA}}$, CB, and WB model-error methods led to solutions with RMSEA values that were close to the target values is perhaps unsurprising, given that this was what all three methods were designed to do. However, it confirms that the CB method worked as expected and provides evidence that the modifications of the TKL and WB methods to incorporate target RMSEA values were successful. On the other hand, the $\text{TKL}_{\text{RMSEA}}$ and $\text{TKL}_{\text{RMSEA/CFI}}$ methods both led to solutions with RMSEA values that were further from the target RMSEA values in most conditions, particularly when model fit was Poor or when there were many major common factors.

When evaluated on CFI, the two model-error methods that incorporated target CFI values (TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$) generally produced observed CFI values that were closer to the target values than the other model error methods. Although all of the model-error methods led to observed CFI values that were close the target values in conditions with strong factor loadings and Very Good model fit, the $\text{TKL}_{\text{RMSEA}}$, CB, and WB methods often led to unacceptably low CFI values in other conditions. Interestingly, the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods often led to quite similar results, despite the $\text{TKL}_{\text{RMSEA/CFI}}$ method incorporating a target RMSEA value. This seemed to be because CFI was more sensitive to changes in parameter values compared to RMSEA, particularly in conditions with many factors, many items per factor, and low factor loadings. A small change in parameter values that produced an RMSEA value slightly closer to the target value often resulted in a large change in CFI away from the target value. Correspondingly, small changes in parameter values that produced a CFI value closer to the target value generally had only a small effect on the RMSEA value. As a result, the $\text{TKL}_{\text{RMSEA/CFI}}$ method tended to "prioritize" target CFI values, despite both RMSEA and CFI being weighted equally in the objective function.

We should discuss this. Aren't you assuming that CFI is an equally valid measure of model fit whatever the heck validity means here

The result that the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods led to RMSEA values that were often far from the target suggested a problem with using only target RMSEA values to generate correlation matrices with model error. Namely, the problem was that solutions with RMSEA values that were close to target RMSEA values often had CFI values that indicated a worse qualitative level of model fit. For instance, in conditions with five factors, weak factor loadings, and Very Good (target) model fit, the $\text{TKL}_{\text{RMSEA}}$ method produced solutions with RMSEA values that were very close to the target RMSEA value of 0.025. However, none of corresponding CFI values for those solutions reached the target value of .99 (considered to represent Very Good model fit), and many CFI values were below .90 (a liberal threshold for acceptable model fit). These results indicate that using RMSEA alone to adjudicate model fit makes it not only possible, but *likely* that simulated population

ImportNt result. Publishable in Top shelf journals

correlation matrices would be included in conditions with nominally excellent model fit as indicated by RMSEA values, but with unacceptably poor model fit as indicated by CFI values. These results agreed with results reported by Kracht and Waller (2022), who used the TKL method with manually-selected parameter values to produce matrices with RMSEA values in a particular range. They found that although they were able to select parameter values that led to solutions with RMSEA values in the desired ranges, the CFI values for those solutions were often below the standard cutoff values. Furthermore, they reported that CFI values were lowest in conditions with many items, low factor loadings, and poor model fit.

The fact that all of the model error method often produced solutions with RMSEA and CFI values indicating different levels of qualitative model fit presents a problem for researchers who would like to generate population correlation matrices with fit indices indicating a particular degree of model fit. Choosing to use the $\text{TKL}_{\text{RMSEA}}$, CB, or WB methods would likely lead to correlation matrices with the desired RMSEA values for most conditions, but unacceptably low CFI values. On the other hand, choosing to use the TKL_{CFI} or $\text{TKL}_{\text{RMSEA/CFI}}$ would lead to almost the complete opposite problem; solutions would be likely to have observed CFI values near the target values, but would also be likely to have smaller-than-desired RMSEA values.

To determine which of the model-error methods led to the highest rates of fit index agreement, I evaluated fit index agreement in two ways. First, I evaluated each model error method in terms of the sum of the absolute differences between the observed and target RMSEA and CFI values, defined as D in Equation (4.1). When evaluated on this criterion, the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods led to much better results than the other investigated model-error methods, having the lowest median D values over all conditions. Moreover, the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods often led to much lower D values than the alternatives (particularly in conditions with many factors, weak factor loadings, and Poor model fit) and rarely led to higher D values. The second way I evaluated fit-index agreement was by using rule-of-thumb RMSEA and CFI threshold values to categorize correlation matrices as having good, acceptable, or unacceptable model fit and then determining how often RMSEA and CFI values led to the same level of qualitative model fit for each model-error method. The results of the simulation study indicated that the TKL_{CFI} method was the most likely to produce solutions with qualitative model fit agreement, followed by the CB, $\text{TKL}_{\text{RMSEA/CFI}}$, $\text{TKL}_{\text{RMSEA}}$, and WB methods. These results suggest that the TKL_{CFI} model is the best choice (of the model-error methods considered here) for researchers who would like to generate correlation matrices with model error and who would like to ensure that the matrices they generate have RMSEA and CFI values that indicate the same level of qualitative model fit. However, it is also important to note that in many conditions (e.g., in conditions with many factors, Weak factor loadings, and Poor model fit), all of the model-error methods had qualitative fit agreement rates close to zero.

The result that the TKL_{CFI} method led to both the smallest average D values and the highest rate of qualitative fit agreement (using RMSEA and CFI) was quite surprising. Before conducting the simulation study, I predicted that the $\text{TKL}_{\text{RMSEA/CFI}}$ method would lead to both the smallest D values and the highest rates of qualitative fit agreement because it incorporated both target RMSEA and CFI values. However, the simulation study results indicated that CFI values tended to change more quickly as a function of RMSEA values than *vice versa*, particularly for conditions with many factors, many items per factor, and strong factor loadings. Put another way, changing the TKL parameter values to produce a solution with an RMSEA value incrementally closer to the target value often resulted in a large change to the CFI value. This provided a plausible explanation for why the $\text{TKL}_{\text{RMSEA/CFI}}$ and TKL_{CFI} methods often led to similar results. Namely, because CFI values were more sensitive to changes to the TKL parameters than RMSEA, the $\text{TKL}_{\text{RMSEA/CFI}}$ solutions were primarily influenced by the CFI targets (and thus often produced results similar to the TKL_{CFI} method) despite the CFI and RMSEA targets being weighted equally in the objective function. Very interesting

Based on the results of the simulation study, I recommend that researchers who want to generate population correlation matrices with a particular level of model fit (as indicated by RMSEA and CFI values) should use the TKL_{CFI} model-error method. I also recommend the $\text{TKL}_{\text{RMSEA/CFI}}$ method as an acceptable alternative. This method produced lower rates of qualitative fit agreement compared to the TKL_{CFI} method, but very similar results in terms of D. Although the TKL_{CFI} method led to the smallest D values for the particular combinations of RMSEA and CFI target values included in this study, it is possible that the $\text{TKL}_{\text{RMSEA/CFI}}$ method might lead to smaller D values when different combinations of target RMSEA and CFI values are used. I recommend that researchers experiment with both options before committing to use either in a particular study.

Although the CB method led to the second-highest rate of qualitative model fit agreement, it also often led to substantially higher D values compared to the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ model-error methods. Moreover, it had several drawbacks that keep me from recommending it for use in simulation studies. First, the CB method was prohibitively slow whenever Ω was large (i.e., whenever there were many factors or items per factor). Using the CB method with 150-variable conditions was so time-consuming that I had to

Remove parentheses. It is important to acknowledge that these results are conditioned on accepting these fit indices– from the zillion that are out there– as definitive of model fit.

drop them from my simulation design. The fact that the CB method is time consuming when Ω is large was all the more problematic because the CB method often produced indefinite Σ matrices in those conditions. As noted in Section 4.1, indefinite Σ matrices are unacceptable candidates for population correlation matrices with model error because all correlation and covariance matrices are at least positive semi-definite by definition (Kracht & Waller, 2022; Lorenzo-Seva & Ferrando, 2020b; Wothke, 1993). Researchers hoping to obtain positive semi-definite Σ matrices using the CB method with large Ω matrices could simply generate a large number of solutions, rejecting indefinite Σ matrices. However, given the completion time of the CB method and the high rates of indefinite solutions reported for many conditions of the simulation study, this is unlikely to be a feasible approach. Finally, it is unclear whether all of the desiderata of the CB method are, in fact, desirable. Specifically, it is not self-evident that the vector of population parameters should be perfectly recovered when the model is applied to Σ using maximum likelihood (as it is in the CB method). Certainly, this constraint is not enforced by any of the alternative model-error methods. Even if a researcher finds the constraint reasonable and chooses to use the CB method as a result, the simulation results showed that the CB method often failed to find a solution such that $\hat{\Omega} = \Omega$ in some conditions.

Most of the issues associated with the CB method did not affect the $\text{TKL}_{\text{RMSEA}}$ or WB model-error methods. For instance, neither method produced indefinite Σ matrices and both methods had much shorter completion times compared with the CB method. However, I do not recommend either model error method for general use in simulation studies. Both the $\text{TKL}_{\text{RMSEA}}$ and WB methods led to relatively high D values and relatively low rates of qualitative model fit agreement compared to the alternative model-error methods. Additionally, the $\text{TKL}_{\text{RMSEA}}$ method often led to solutions with strong minor factors that would more appropriately be considered major factors. Even the inclusion of a large penalty term (λ) did not prevent the $\text{TKL}_{\text{RMSEA}}$ method from producing solutions that violated the constraint that no minor factor should have more than two absolute loadings greater than 0.3. On the other hand, the TKL_{CFI} or $\text{TKL}_{\text{RMSEA/CFI}}$ methods almost never led to solutions that violated the minor factor constraints. In fact, the TKL_{CFI} and $\text{TKL}_{\text{RMSEA/CFI}}$ methods did not produce solutions that violated the minor factor constraints even when $\lambda = 0$ (i.e., when no penalty was applied). Thus, the study results indicated that the inclusion of a reasonable target CFI value was almost always sufficient to avoid generating solutions with strong minor factors. Researchers who want to use the $\text{TKL}_{\text{RMSEA}}$ method could discard solutions that violated the minor factor constraints and continue generating new solutions until a sufficient number of acceptable solutions were found. However, the study results indicated the $\text{TKL}_{\text{RMSEA}}$ method almost always led to solutions that violated the minor factor constraints in conditions with many factors, many items per factor, and Poor model fit. Therefore, the strategy of discarding solutions that violated the minor factor constraints is likely to be highly inefficient (if not completely impractical) for those conditions.

5.1 Limitations and Future Work

As with any study, I acknowledge that the simulation study reported in this dissertation was subject to certain limitations. For instance, I designed the study to include a wide range of models that might plausibly be encountered in psychological research. However, there were still many types of models that were not included in the study design. For instance, the study design only included models with equal numbers of salient items per factor, all factor correlations fixed at the same value, and all non-zero factor loadings fixed at the same value. Although these models were artificially simple, they were chosen because they made it easier to isolate the effects of each of the independent variables (e.g., number of factors, number of items per factor, factor loading, etc.) and because similar models have been used in previous Monte Carlo simulation studies (Auerswald & Moshagen, 2019; Debelak & Tran, 2013, 2016; Kracht & Waller, 2022). Nevertheless, future research should investigate how the results from the CB, WB, and TKL-based model-error methods differ when used with more complex models. The implementations of these model error methods in the *noisemaker* R package, along with the simulation code in Appendix A.2 should facilitate this future work.

A second limitation of the present study was that it only investigated model error only in the context of factor analysis models and not covariance structure models more generally. The choice to focus on factor analysis models was largely motivated by the fact that the TKL method is specific to the factor analysis model. Unlike the CB and WB model-error methods, which can be used with any covariance structure model, the TKL method requires modification to be applied to covariance structure models other than the common factor model (e.g., Trichtinger & Zhang, 2020). Where possible, it would be useful to extend the TKL method (and the multiple-objective TKL method) to additional types of covariance structure models. It might also be useful to investigate whether automated procedures could be developed to incorporate target CFI values into the CB or WB methods. For instance, the target RMSEA value used in the CB and WB methods could be treated as a tuning parameter and optimized to find the value that leads to solutions with CFI values that are close to a user-specified target value. As with the multiple-objective TKL method, users could also specify how much weight to give each fit index. Although evaluating the CB method many times in an optimization loop is likely to be prohibitively time-consuming with large models, the procedure could work well for the WB method (or for the CB method when models are small). Future work should investigate the effectiveness of optimizing CB and WB target RMSEA values to incorporate CFI targets.

In addition to extending the simulation design and investigating ways to incorporate CFI targets in the CB and WB methods, future work should be done to build and improve flexible, robust, and easy-to-use implementations of model-error methods for use in Monte Carlo simulation studies. The *noisemaker* package (and the function of the same name) that was developed as a part of this dissertation provides a simple, unified interface for generating correlation matrices with model error using the new multiple-objective TKL method, the CB method, and WB method. Moreover, the *noisemaker* handles specification of the population model (without model error) using the `simFA()` function from the *fungible* package. Together, the *noisemaker* and *fungible* packages provide a powerful collection of functions for researchers who wish to simulate factor analysis models and data sets.

Future work should continue to improve the *noisemaker* package by adding features and improving ease-of-use. For example, an update to the `noisemaker()` function has been planned that will let users ~~to~~ specify allowable ranges of ϵ and ν_{e} parameters when using the multiple-objective TKL method. Although it is difficult to know which precise values of ϵ and ν_{e} are reasonable for a particular model, it is sometimes possible to specify plausible ranges of these parameters. For instance, consider a population model with one major common factor and ten items, each with a factor loading of 0.8. The major common factor therefore accounts for 64% of the variance in each item. Setting $\nu_{\mathrm{e}} = .75$ would then mean that 91% of the item variance would be accounted for by the (reliable) major and minor common factors. In some contexts, this might be reasonable; in many psychological contexts, it would be considered unreasonable. Similarly, a researcher might consider it unlikely that all of the minor factors should be equally strong. In that case, values of ϵ very close to zero would be considered unrealistic. To allow users to specify allowable ranges of ϵ and ν_{e} for the multiple-objective TKL method, those ranges could be used as the parameter boundary constraints in the L-BFGS-B algorithm instead of $\epsilon, \nu_{\mathrm{e}} \in [0, 1]$. This feature has already been implemented in a development version of *noisemaker*[1] that will be submitted to CRAN once testing has been completed.

In conclusion, the work in this dissertation should provide a valuable resource for researchers who would like to incorporate model error into Monte Carlo simulation studies of factor analysis models. I have reported an overview of existing model-error methods and have proposed an extension of the TKL model-error method that allows researchers to specify target RMSEA values, target CFI values, or both simultaneous. By conducting an extensive

[1] The most recent development version of the *noisemaker* package is available at github.com/JustinKracht/noisemaker, along with a vignette demonstrating how it can be used.

simulation study, I showed that using the proposed multiple-target TKL method with target RMSEA and CFI values (or with only a target CFI value) often led to solutions with better quantitative and qualitative fit index agreement compared to the alternative model-error methods. Finally, I developed the R *noisemaker* package to make it easy for researchers to use any of the model-error methods investigated in this simulation study.

Fantastic!!! Extreme impressive.
The writing was very clear and
your results are important. This
is highly publishable.

References

Auerswald, M., & Moshagen, M. (2019). How to determine the number of factors to retain in exploratory factor analysis: A comparison of extraction methods under realistic conditions. *Psychological Methods*, *24*(4), 468–491. https://doi.org/gf5spw

Aust, F., & Barth, M. (2020). *papaja: Create APA manuscripts with R Markdown* [Manual].

Beauducel, A., & Hilger, N. (2016). On the correlation of common factors with variance not accounted for by the factor model. *Communications in Statistics - Simulation and Computation*, *45*(6), 2145–2157. https://doi.org/gh64s7

Bentler, P. M. (1990). Comparative fit indexes in structural models. *Psychological Bulletin*, *107*(2), 238–246. https://doi.org/dbj

Bentler, P. M., & Bonett, D. G. (1980). Significance tests and goodness of fit in the analysis of covariance structures. *Psychological Bulletin*, *88*(3), 588–606. https://doi.org/dbm

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/f9wkpj

Bollen, K. A. (1989). A new incremental fit index for general structural equation models. *Sociological Methods & Research*, *17*(3), 303–316. https://doi.org/cfgdt5

Bollen, Kenneth A. (1989). *Structural Equations with Latent Variables*. John Wiley & Sons, Inc. https://doi.org/10.1002/9781118619179

Borges, J. L. (1998). On exactitude in science. In A. Hurley (Ed.), *Collected Fictions*. New York:Viking.

Box, G. E. P., & Draper, N. R. (1987). *Empirical model-building and response surfaces.* (pp. xiv, 669). John Wiley & Sons.

Briggs, N. E., & MacCallum, R. C. (2003). Recovery of weak common factors by maximum likelihood and ordinary least squares estimation. *Multivariate Behavioral Research*, *38*(1), 25–56. https://doi.org/bgnz9w

Browne, M. W. (1984). Asymptotically distribution-free methods for the analysis of covariance structures. *British Journal of Mathematical and Statistical Psychology*, *37*(1), 62–83. https://doi.org/cd4vn7

Browne, M. W., & Cudeck, R. (1992). Alternative ways of assessing model fit. *Sociological Methods & Research*, *21*(2), 230–258. https://doi.org/dbn

Browne, M. W., MacCallum, R. C., Kim, C. T., Andersen, B. L., & Glaser, R. (2002). When fit indices and residuals are incompatible. *Psychological Methods*, *7*(4), 403–421. https://doi.org/fhq5t3

Browne, M. W., & Shapiro, A. (1988). Robustness of normal theory methods in the analysis of linear latent variate models. *British Journal of Mathematical and Statistical Psychology*, *41*(2), 193–208. https://doi.org/bdk8xk

Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, *16*(5), 1190–1208. https://doi.org/bpjm24

Carroll, L. (1894). *Sylvie and Bruno, Volume II, by Lewis Carroll, illustrated by Harry Furniss*. https://www.gutenberg.org/files/48795/48795-h/48795-h.htm.

Chung, S., & Cai, L. (2019). Alternative multiple imputation inference for categorical structural equation modeling. *Multivariate Behavioral Research*, *54*(3), 323–337. https://doi.org/gh64rk

Csárdi, G. (2022). *Crayon: Colored terminal output* [Manual].

Cudeck, R. (1989). Analysis of correlation matrices using covariance structure models. *Psychological Bulletin*, *105*(2), 317–327. https://doi.org/fqks6n

Cudeck, R., & Browne, M. W. (1992). Constructing a covariance matrix that yields a specified minimizer and a specified minimum discrepancy function value. *Psychometrika*, *57*(3), 357–369. https://doi.org/cq6ckd

Cudeck, R., & Henly, S. J. (1991). Model selection in covariance structures analysis and the "problem" of sample size: A clarification. *Psychological Bulletin*, *109*(3), 512. https://doi.org/bpqvdv

de Winter, J. C. F., & Dodou, D. (2016). Common factor analysis versus principal component analysis: A comparison of loadings by means of simulations. *Communications in Statistics - Simulation and Computation*, *45*(1), 299–321. https://doi.org/gh64cn

Debelak, R., & Tran, U. S. (2013). Principal component analysis of smoothed tetrachoric correlation matrices as a measure of dimensionality. *Educational and Psychological Measurement*, *73*(1), 63–77. https://doi.org/ghx8xx

Debelak, R., & Tran, U. S. (2016). Comparing the effects of different smoothing algorithms on the assessment of dimensionality of ordered categorical items with parallel analysis. *PLOS ONE*, *11*(2), 1–18. https://doi.org/f8q4j2

Eco, U. (1994). On the impossibility of drawing a map of the empire on a scale of 1 to 1. In W. Weaver & U. Eco (Eds.), *How to Travel With a Salmon & Other Essays* (1st ed..). New York:Harcourt, Brace.

Gelman, A. (2008). Some thoughts on the saying, "All models are wrong, but some are useful." In *Statistical Modeling, Causal Inference, and Social Science*. https://bit.ly/3P5lXvg.

Gnambs, T., & Staufenbiel, T. (2016). Parameter accuracy in meta-analyses of factor structures. *Research Synthesis Methods*, *7*(2), 168–186. https://doi.org/gcz6x6

Gupta, A. K., & Nagar, D. K. (2000). *Matrix Variate Distributions*. Chapman & Hall.

Hair, J. F., Black, W. C., Babin, B. J., & Anderson, R. E. (2018). *Multivariate Data Analysis*. Cengage.

Henry, L., & Wickham, H. (2020). *Purrr: Functional programming tools* [Manual].

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press.

Hong, S. (1999). Generating correlation matrices with model error for simulation studies in factor analysis: A combination of the Tucker-Koopman-Linn model and Wijsman's algorithm. *Behavior Research Methods, Instruments, & Computers*, *31*(4), 727–730. https://doi.org/dsjcnm

Howe, W. G. (1955). *Some contributions to factor analysis* (Technical {{Report}} ONRL-1919).

Hsu, H.-Y., Kwok, O., Lin, J. H., & Acosta, S. (2015). Detecting misspecified multilevel structural equation models with common fit indices: A Monte Carlo study. *Multivariate Behavioral Research*, *50*(2), 197–215. https://doi.org/gg5fk7

Hu, L., & Bentler, P. M. (1999). Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural Equation Modeling: A Multidisciplinary Journal*, *6*(1), 1–55. https://doi.org/dbt

Ismay, C., & Solomon, N. (2022). *Thesisdown: An updated R Markdown thesis template using the bookdown package* [Manual].

Jackson, D. L. (2009). Reporting practices in confirmatory factor analysis: An overview and some recommendations. *Psychological Methods*, *14*(1), 6. https://doi.org/cbd6zd

Jöreskog, K. G. (1969). A general approach to confirmatory maximum likelihood factor analysis. *Psychometrika*, *34*(2, Pt.1), 183–202. https://doi.org/cs6df7

Kelley, K. (2020). *MBESS: The MBESS R package* [Manual].

Kline, R. B. (2011). *Principles and Practice of Structural Equation Modeling* (3rd ed). Guilford Press.

Kracht, J. D., & Waller, N. G. (2022). Assessing Dimensionality in Non-Positive Definite Tetrachoric Correlation Matrices: Does Matrix Smoothing Help? *Multivariate Behavioral Research*, *57*(2-3), 385–407. https://doi.org/ghq7p7

Kuang, K., Kong, Q., & Napolitano, F. (2019). *Pbmcapply: Tracking the progress of Mc*pply with progress bar* [Manual].

Lai, K. (2020a). Better confidence intervals for RMSEA in growth models given nonnormal data. *Structural Equation Modeling: A Multidisciplinary Journal*, *27*(2), 255–274. https://doi.org/gh7r8k

Lai, K. (2018). Estimating standardized SEM parameters given nonnormal data and incorrect model: Methods and comparison. *Structural Equation Modeling: A Multi-*

disciplinary Journal, *25*(4), 600–620. https://doi.org/gc4pjh

Lai, K. (2020b). Correct estimation methods for RMSEA under missing data. *Structural Equation Modeling: A Multidisciplinary Journal*, *0*(0), 1–12. https://doi.org/gh7r9d

Lai, K. (2020c). Fit difference between nonnested models given categorical data: Measures and estimation. *Structural Equation Modeling: A Multidisciplinary Journal*, *0*(0), 1–22. https://doi.org/gh7r9f

Lai, K. (2019). A simple analytic confidence interval for CFI given nonnormal data. *Structural Equation Modeling: A Multidisciplinary Journal*, *26*(5), 757–777. https://doi.org/gh7r8d

Lai, K. (2020d). Correct point estimator and confidence interval for RMSEA given categorical data. *Structural Equation Modeling: A Multidisciplinary Journal*, *27*(5), 678–695. https://doi.org/gh7r82

Lai, K., & Green, S. B. (2016). The problem with having two watches: Assessment of fit when RMSEA and CFI disagree. *Multivariate Behavioral Research*, *51*(2-3), 220–239. https://doi.org/gfpnvb

Lai, K., & Zhang, X. (2017). Standardized parameters in misspecified structural equation models: Empirical performance in point estimates, standard errors, and confidence intervals. *Structural Equation Modeling: A Multidisciplinary Journal*, *24*(4), 571–584. https://doi.org/gcph63

Lorenzo-Seva, U., & Ferrando, P. J. (2020a). Unrestricted factor analysis of multidimensional test items based on an objectively refined target matrix. *Behavior Research Methods*, *52*(1), 116–130. https://doi.org/gh64j4

Lorenzo-Seva, U., & Ferrando, P. J. (2020b). Not positive definite correlation matrices in exploratory item factor analysis: Causes, consequences and a proposed solution. *Structural Equation Modeling: A Multidisciplinary Journal*, 1–10. https://doi.org/gg9czw

Lorenzo-Seva, U., & Ginkel, J. R. V. (2016). Multiple imputation of missing values in exploratory factor analysis of multidimensional scales: Estimating latent trait scores. *Anales de Psicología / Annals of Psychology*, *32*(2), 596–608. https://doi.org/dvrt

MacCallum, R. C., & Tucker, L. R. (1991). Representing sources of error in the common-factor model: Implications for theory and practice. *Psychological Bulletin*, *109*(3), 502–511. https://doi.org/cgsqhv

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. https://doi.org/b3v4p6

Marcoulides, K. M., & Yuan, K.-H. (2017). New Ways to Evaluate Goodness of Fit: A Note on Using Equivalence Testing to Assess Structural Equation Models. *Structural Equation Modeling: A Multidisciplinary Journal*, *24*(1), 148–153. https://doi.org/ggf5hd

Marsh, H. W., Hau, K.-T., & Grayson, D. (2005). Goodness of Fit in Structural Equation Models. In *Contemporary Psychometrics: A Festschrift for Roderick P. Mcdonald.* (pp. 275–340). Lawrence Erlbaum Associates Publishers.

Maydeu-Olivares, A. (2017). Assessing the size of model misfit in structural equation models. *Psychometrika*, *82*(3), 533–558. https://doi.org/gbthc8

Meehl, P. E., & Waller, N. G. (2002). The path analysis controversy: A new statistical approach to strong appraisal of verisimilitude. *Psychological Methods*, *7*(3), 283–300.

https://doi.org/dbzsmd

Mersmann, O. (2021). *Microbenchmark: Accurate timing functions* [Manual].

Meschiari, S. (2021). *Latex2exp: Use LaTeX expressions in plots* [Manual].

Miles, J., & Shevlin, M. (2007). A time and a place for incremental fit indices. *Personality and Individual Differences*, *42*(5), 869–874. https://doi.org/fhdt3m

Mitchell, M. (1996). *An Introduction to Genetic Algorithms* (I. NetLibrary, Ed.). MIT Press.

Montoya, A. K., & Edwards, M. C. (2020). The poor fit of model fit for selecting number of factors in exploratory factor analysis for scale evaluation. *Educational and Psychological Measurement*, 413–440. https://doi.org/gh7r86

Mulaik, S. A. (2009). *Foundations of factor analysis*. Chapman and Hall/CRC. https://doi.org/10.1201/b15851

Müller, K. (2020). *Here: A simpler way to find your files* [Manual].

Myers, N. D., Jin, Y., Ahn, S., Celimli, S., & Zopluoglu, C. (2015). Rotation to a partially specified target matrix in exploratory factor analysis in practice. *Behavior Research Methods*, *47*(2), 494–505. https://doi.org/f7jmdz

Nester, M. R. (1996). An applied statistician's creed. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *45*(4), 401–410. https://doi.org/bk59hb

Neuman, G. A., Bolin, A. U., & Briggs, T. E. (2000). Identifying General Factors of Intelligence: A Confirmatory Factor Analysis of the Ball Aptitude Battery. *Educational and Psychological Measurement*, *60*(5), 697–712. https://doi.org/dxx8x4

Ogasawara, H. (2001). Approximations to the distributions of fit indexes for misspecified structural equation models. *Structural Equation Modeling: A Multidisciplinary*

Journal, *8*(4), 556–574. https://doi.org/bv4mwk

Olsson, U. H., Foss, T., & Breivik, E. (2004). Two equivalent discrepancy functions for maximum likelihood estimation: Do their test statistics follow a non-central chi-square distribution under model misspecification? *Sociological Methods & Research*, *32*(4), 453–500. https://doi.org/ft43gk

Pavlov, G., Maydeu-Olivares, A., & Shi, D. (2021). Using the Standardized Root Mean Squared Residual (SRMR) to Assess Exact Fit in Structural Equation Models. *Educational and Psychological Measurement*, *81*(1), 110–130. https://doi.org/gmchg6

Pedersen, T. L. (2020). *Patchwork: The composer of plots* [Manual].

Pek, J. (2012). *Fungible parameter contours and confidence regions in structural equation models* [PhD thesis, University of North Carolina]. https://doi.org/10.17615/S9V9-5M31

Porritt, M. (2015). *Performance of number of factors procedures in small sample sizes* [PhD thesis]. Loma Linda University.

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. https://doi.org/cghpnv

Preacher, K. J., Zhang, G., Kim, C., & Mels, G. (2013). Choosing the optimal number of factors in exploratory factor analysis: A model selection perspective. *Multivariate Behavioral Research*, *48*(1), 28–56. https://doi.org/gckf6t

R Core Team. (2021). *R: A language and environment for statistical computing* [Manual]. R Foundation for Statistical Computing.

Raykov, T., & Marcoulides, G. A. (2012). *A First Course in Structural Equation Modeling* (Second Edition). Routledge. https://doi.org/10.4324/9780203930687

Satorra, A. (2015). A Comment on a paper by H. Wu and M. W. Browne (2014). *Psychometrika*, *80*(3), 613–618. https://doi.org/gjrkc3

Scrucca, L. (2013). GA: A package for genetic algorithms in R. *Journal of Statistical Software*, *53*(1), 1–37. https://doi.org/gft29t

Shapiro, A. (1983). Asymptotic distribution theory in the analysis of covariance structures. *South African Statistical Journal*, *17*(1), 33–81. https://doi.org/10.10520/AJA0038271X_800

Shapiro, A. (2007). Statistical inference of moment structures. In *Handbook of Latent Variable and Related Models* (1st ed.., pp. 229–260). Amsterdam.

Smith, T. G. (2021). *Purrrgress: Progress bars for purrr* [Manual].

Stauffer, R., Mayr, G. J., Dabernig, M., & Zeileis, A. (2009). Somewhere over the rainbow: How to make effective use of colors in meteorological visualizations. *Bulletin of the American Meteorological Society*, *96*(2), 203–216. https://doi.org/f3sttk

Steele, J. M. (2008). *Models: Masterpieces and lame excuses.* https://whr.tn/3L1ULKE.

Steiger, J. H. (1990). Structural model evaluation and modification: An interval estimation approach. *Multivariate Behavioral Research*, *25*(2), 173–180. https://doi.org/db5

Steiger, J. H. (2007). Understanding the limitations of global fit assessment in structural equation modeling. *Personality and Individual Differences*, *42*(5), 893–898. https://doi.org/dsxstk

Steiger, J. H. (1989). *Ezpath: Causal Modeling: A Supplementary Module for SYSTAT and SYGRAPH: PC-MS-DOS, Version 1.0*. Systat.

The Mathworks, Inc. (2019). *Matlab and Statistics Toolbox R2019a*.

Tomarken, A. J., & Waller, N. G. (2003). Potential problems with "well fitting" models. *Journal of Abnormal Psychology*, *112*(4), 578. https://doi.org/fpmxjq

Trichtinger, L. A., & Zhang, G. (2020). Quantifying model error in P-technique factor analysis. *Multivariate Behavioral Research*, *0*(0), 1–16. https://doi.org/gh64s8

Tucker, L. R., Koopman, R. F., & Linn, R. L. (1967). *Evaluation of factor analytic research procedures by means of simulated correlation matrices*. Department of Psychology, University of Illinois, Urbana.

Tucker, L. R., Koopman, R. F., & Linn, R. L. (1969). Evaluation of factor analytic research procedures by means of simulated correlation matrices. *Psychometrika*, *34*(4), 421–459. https://doi.org/chcxvf

Tucker, L. R., & Lewis, C. (1973). A reliability coefficient for maximum likelihood factor analysis. *Psychometrika*, *38*(1), 1–10. https://doi.org/bcz7k9

Waller, N. G. (2021). *Fungible: Psychometric functions from the Waller lab*. [Manual].

Wickham, H. (2019). *Stringr: Simple, consistent wrappers for common string operations* [Manual].

Wickham, H. (2021). *Tidyr: Tidy messy data* [Manual].

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., ... Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43),

1686. https://doi.org/ggddkj

Wickham, H., François, R., Henry, L., & Müller, K. (2021). *Dplyr: A grammar of data manipulation* [Manual].

Wickham, H., Hester, J., Chang, W., & Bryan, J. (2021). *Devtools: Tools to make developing R packages easier* [Manual].

Wickham, H., & Seidel, D. (2020). *Scales: Scale functions for visualization* [Manual].

Wothke, W. (1993). Testing structural equation models. In K. A. Bollen & J. S. Long (Eds.), *Testing structural equation models* (Vol. 154, pp. 256–293). A Sage Focus Edition.

Wu, H., & Browne, M. W. (2015a). Quantifying adventitious error in a covariance structure as a random effect. *Psychometrika*, *80*(3), 571–600. https://doi.org/gjrkc4

Wu, H., & Browne, M. W. (2015b). Random model discrepancy: Interpretations and technicalities (a rejoinder). *Psychometrika*, *80*(3), 619–624. https://doi.org/ggfngm

Xia, Y. (2021). Determining the number of factors when population models can be closely approximated by parsimonious models. *Educational and Psychological Measurement*, 0013164421992836. https://doi.org/gh68s2

Xia, Y., & Yang, Y. (2019). RMSEA, CFI, and TLI in structural equation modeling with ordered categorical data: The story they tell depends on the estimation methods. *Behavior Research Methods*, *51*(1), 409–428. https://doi.org/ggssdx

Xia, Y., Yung, Y.-F., & Zhang, W. (2016). Evaluating the selection of normal-theory weight matrices in the Satorra–Bentler correction of chi-square and standard errors. *Structural Equation Modeling: A Multidisciplinary Journal*, *23*(4), 585–594. https://doi.org/gcz6zq

Xie, Y. (2015). *Dynamic documents with R and knitr* (Second). Chapman and Hall/CRC. https://doi.org/10.1201/b15166

Xie, Y. (2016). *Bookdown: Authoring books and technical documents with R markdown*. Chapman and Hall/CRC.

Xie, Y. (2022). *Xfun: Supporting functions for packages maintained by Yihui Xie* [Manual].

Xie, Y., Allaire, J. J., & Grolemund, G. (2018). *R markdown: The definitive guide*. Chapman and Hall/CRC.

Xie, Y., Dervieux, C., & Riederer, E. (2020). *R markdown cookbook*. Chapman and Hall/CRC.

Yuan, K.-H., & Marshall, L. L. (2004). A new measure of misfit for covariance structure models. *Behaviormetrika*, *31*(1), 67–90. https://doi.org/cr2mb8

Yutani, H. (2021). *Gghighlight: Highlight Lines and Points in 'Ggplot2'* [Manual].

Zeileis, A., Fisher, J. C., Hornik, K., Ihaka, R., McWhite, C. D., Murrell, P., Stauffer, R., & Wilke, C. O. (2020). colorspace: A toolbox for manipulating and assessing colors and palettes. *Journal of Statistical Software*, *96*(1), 1–49. https://doi.org/ghnnvz

Zeileis, A., Hornik, K., & Murrell, P. (2009). Escaping RGBland: Selecting colors for statistical graphics. *Computational Statistics & Data Analysis*, *53*(9), 3259–3270. https://doi.org/ds8dbd

Zhu, H. (2020). *kableExtra: Construct complex table with 'kable' and pipe syntax* [Manual].

Zieffler, A., & Ismay, C. (2022). *Gopherdown: An updated R Markdown thesis template for the University of Minneosta using the bookdown package* [Manual].

Appendix A

R Code

A.1 Implementations of Model Error Methods

The implementations of the model error methods used are also available bundled as an R package, *noisemaker*, which can be downloaded from https://www.github.com/JustinKracht/noisemaker.

```
#' Cudeck & Browne (1992) model error method
#'
#' Generate a population correlation matrix using the model described in Cudeck
#' and Browne (1992).
#'
#' @param mod A `fungible::simFA()` model object.
#' @param target_rmsea (scalar) Target RMSEA value.
#' @export
#' @references Cudeck, R., & Browne, M. W. (1992). Constructing a covariance
#'   matrix that yields a specified minimizer and a specified minimum
#'   discrepancy function value. *Psychometrika*, *57*(3), 357-369.
#'   <https://doi.org/10/cq6ckd>

cb <- function(mod,
               target_rmsea) {

  if (target_rmsea < 0 | target_rmsea > 1) {
    stop("The target RMSEA value must be a number between 0 and 1.\n",
         crayon::cyan("\u2139"), " You've specified a target RMSEA value of ",
         target_rmsea, ".", call. = F)
  }

  if (!(is.list(mod)) |
      is.null(mod$loadings) |
      is.null(mod$Phi) |
```

```
      is.null(mod$Rpop)) {
    stop("`mod` must be a valid `simFA()` model object.", call. = F)
  }

  p <- nrow(mod$loadings)
  k <- ncol(mod$loadings)
  df <- (p * (p - 1) / 2) - (p * k) + (k * (k - 1) / 2)
  discrep <- target_rmsea^2 * df
  sem_mod <- semify(mod)

  Sigma <- MBESS::Sigma.2.SigmaStar(
    model = sem_mod$model,
    model.par = sem_mod$theta,
    latent.var = sem_mod$latent_var,
    discrep = discrep
  )$Sigma.star

  # Check positive definiteness
  lambda_min <- min(eigen(Sigma, symmetric = TRUE, only.values = TRUE)$values)
  if (lambda_min < 0) {
    stop("Sigma is indefinite.\n",
         crayon::cyan("\u2139"), " The minimum eigenvalue is ",
         round(lambda_min, 2), call. = FALSE)
  }

  Sigma
}
#' Calculate CFI for two correlation matrices
#'
#' Given two correlation matrices of the same dimension, calculate the CFI value
#' value using the independence model as the null model.
#'
#' @param Sigma (matrix) Population correlation or covariance matrix (with model
#'   error).
#' @param Omega (matrix) Model-implied population correlation or covariance
#'   matrix.
#'
#' @export
#'
#' @examples
#' library(fungible)
#' library(noisemaker)
#'
#' mod <- fungible::simFA(Model = list(NFac = 3),
#'                        Seed = 42)
#' set.seed(42)
#' Omega <- mod$Rpop
#' Sigma <- noisemaker(
```

```
#'   mod = mod,
#'   method = "CB",
#'   target_rmsea = 0.05
#' )$Sigma
#' cfi(Sigma, Omega)
cfi <- function(Sigma, Omega) {
  if (!is.matrix(Sigma) | !is.matrix(Omega)) {
    stop("Sigma and Omega must be matrices.", call. = F)
  } else if (all.equal(Sigma, t(Sigma)) != TRUE) {
    stop("Sigma must be a symmetric matrix.", call. = F)
  } else if (all.equal(Omega, t(Omega)) != TRUE) {
    stop("Omega must be a symmetric matrix.", call. = F)
  } else if (all.equal(dim(Omega), dim(Sigma)) != TRUE) {
    stop("Sigma and Omega must have the same dimensions.", call. = F)
  }

  p <- nrow(Sigma)
  Ft <- log(det(Omega)) - log(det(Sigma)) +
    sum(diag(Sigma %*% solve(Omega))) - p
  cfi <- 1 - (Ft / -log(det(Sigma)))
  cfi
}
#' Find an `lm` model to use with the Wu & Browne (2015) model error method
#'
#' The Wu & Browne (2015) model error method takes advantage of the relationship
#' between v and RMSEA:
#'
#' \deqn{v = RMSEA^2 + o(RMSEA^2).}
#'
#' As RMSEA increases, the approximation \eqn{v ~= RMSEA^2} becomes worse. This
#' function generates population correlation matrices with model error for
#' multiple target RMSEA values and then regresses the target RMSEA values on
#' the median observed RMSEA values for each target. The fitted model can then
#' be used to predict a `target_rmsea` value that will give solutions with RMSEA
#' values that are close to the desired value.
#'
#' @param mod A `fungible::simFA()` model object.
#' @param n The number of times to evaluate `wb()` at each point.
#' @param values The number of target RMSEA values to evaluate between 0.02 and
#'   0.1.
#' @param lower (scalar) The smallest target RMSEA value to use.
#' @param upper (scalar) The largest target RMSEA value to use.
#'
#' @return (`lm` object) An `lm` object to use with the \code{\link{wb}}
#'   function to obtain population correlation matrices with model error that
#'   have RMSEA values closer to the target RMSEA values. The `lm` object will
#'   predict a `target_rmsea` value that will give solutions with (median) RMSEA
#'   values close to the desired RMSEA value.
```

```r
#' @export
#'
#' @examples
#' mod <- fungible::simFA(Seed = 42)
#' set.seed(42)
#' wb_mod <- get_wb_mod(mod)
#' noisemaker(mod, method = "WB", target_rmsea = 0.05, wb_mod = wb_mod)

get_wb_mod <- function(mod, n = 50, values = 10, lower = .01, upper = .095) {
  # Check arguments
  if (!(is.list(mod))) {
    stop("`mod` must be a valid `simFA()` model object.", call. = F)
  }
  if (is.null(mod$loadings) |
      is.null(mod$Phi) |
      is.null(mod$Rpop)) {
    stop("`mod` must be a valid `simFA()` model object.", call. = F)
  }
  if (length(n) != 1L | !is.numeric(n) | n <= 0) {
    stop("`n` must be a number greater than zero.\n",
         crayon::cyan("\u2139"), " You've specified an `n` value of ",
         n, ".", call. = F)
  }
  if (length(values) != 1L | !is.numeric(values) | values < 2) {
    stop("`values` must be a number greater than two.\n",
         crayon::cyan("\u2139"), " You've specified a `values` value of ",
         values, ".", call. = F)
  }
  if (length(lower) != 1L | !is.numeric(lower) | lower <= 0) {
    stop("`lower` must be a number greater than zero.\n",
         crayon::cyan("\u2139"), " You've specified a `lower` value of ",
         lower, ".", call. = F)
  }
  if (length(upper) != 1L | !is.numeric(upper)) {
    stop("`upper` must be a number.\n",
         crayon::cyan("\u2139"), " You've specified an `upper` value of ",
         upper, ".", call. = F)
  }

  k <- ncol(mod$loadings)
  Omega <- mod$Rpop
  p <- nrow(Omega)

  # WB requires m < p; calculate upper bound
  # (1 / target_rmsea^2) > p means that target_rmsea < 1 / sqrt(p)
  max_target_rmsea <- 1 / sqrt(p)
  if (upper >= max_target_rmsea) {
    warning("Specified upper bound was too large and was reduced.")
```

```
    # Set new upper bound to the maximum target RMSEA, minus 5% to avoid hitting
    # a boundary
    upper <- max_target_rmsea - 0.05 * max_target_rmsea
  }

  rmsea_values <- seq(lower, upper, length.out = values)

  rmsea_medians <- sapply(
    X = rmsea_values,
    FUN = function(target_rmsea,
                   mod,
                   Omega,
                   k) {
      obs_rmsea <- replicate(n = n, expr = {
        rmsea(wb(mod, target_rmsea, adjust_target = FALSE)$Sigma, Omega, k)
      })
      stats::median(obs_rmsea)
    },
    mod = mod,
    Omega = Omega,
    k = k
  )

  m1 <- stats::lm(rmsea_values ~ poly(rmsea_medians, 2))
  m1
}
#' Simulate a population correlation matrix with model error
#'
#' This tool lets the user generate a population correlation matrix with model
#' error using one of three methods: (1) the Tucker, Koopman, and Linn (TKL;
#' 1969) method, (2) the Cudeck and Browne (CB; 1992) method, or (3) the Wu and
#' Browne (WB; 2015) method. If the CB or WB methods are used, the user can
#' specify the desired RMSEA value. If the TKL method is used, an optimization
#' procedure finds a solution that produces RMSEA and/or CFI values that are
#' close to the user-specified values.
#'
#' @param mod A `fungible::simFA()` model object.
#' @param method (character) Model error method to use ("TKL", "CB", or "WB").
#' @param target_rmsea (scalar) Target RMSEA value.
#' @param target_cfi (scalar) Target CFI value.
#' @param tkl_ctrl (list) A control list containing the following TKL-specific
#'   arguments. See the `tkl()` help file for more details.
#' @param wb_mod (`lm` object) An optional `lm` object used to find a target
#'   RMSEA value that results in solutions with RMSEA values close to the
#'   desired value. Note that if no `wb_mod` is provided, a model will be
#'   estimated at run time. If many population correlation matrices are going to
#'   be simulated using the same model, it will be considerably faster to
#'   estimate `wb_mod` ahead of time. See also `get_wb_mod()`.
```

```r
#'
#' @return A list containing \eqn{\Sigma}, RMSEA and CFI values, and the TKL
#'   parameters (if applicable).
#' @export
#'
#' @examples
#' mod <- fungible::simFA(Seed = 42)
#'
#' set.seed(42)
#' # Simulate a population correlation matrix using the TKL method with target
#' # RMSEA and CFI values specified.
#' noisemaker(mod, method = "TKL",
#'            target_rmsea = 0.05,
#'            target_cfi = 0.95,
#'            tkl_ctrl = list(optim_type = "optim"))
#'
#' # Simulate a population correlation matrix using the CB method with target
#' # RMSEA value specified.
#' noisemaker(mod, method = "CB",
#'            target_rmsea = 0.05)
#'
#' # Simulation a population correlation matrix using the WB method with target
#' # RMSEA value specified.
#' noisemaker(mod,
#'            method = "WB",
#'            target_rmsea = 0.05)
noisemaker <- function(mod,
                       method = c("TKL", "CB", "WB"),
                       target_rmsea = 0.05,
                       target_cfi = NULL,
                       tkl_ctrl = list(),
                       wb_mod = NULL) {

  if (is.null(target_rmsea) & is.null(target_cfi)) {
    stop("Either target RMSEA or target CFI must be specified.",
         call. = F)
  }
  if (!is.numeric(target_rmsea) & !is.null(target_rmsea)) {
    stop("Target RMSEA value must be a number or NULL.\n",
         crayon::cyan("\u2139"), " You've specified a target RMSEA value of ",
         target_rmsea, ".", call. = F)
  }
  if (!is.numeric(target_cfi) & !is.null(target_cfi)) {
    stop("Target CFI value must be either a number or NULL.\n",
         crayon::cyan("\u2139"), " You've specified a target CFI value of ",
         target_cfi, ".", call. = F)
  }
  if (!is.null(target_rmsea)) {
```

```
    if (target_rmsea < 0 | target_rmsea > 1) {
      stop("The target RMSEA value must be a number between 0 and 1.\n",
           crayon::cyan("\u2139"), " You've specified a target RMSEA value of ",
           target_rmsea, ".", call. = F)
    }
  }
  if (!is.null(target_cfi) & (method != "TKL")) {
    stop(
      "The TKL method must be used when a CFI value is specified.\n",
      crayon::cyan("\u2139")," You've selected the ", method," method.\n",
      crayon::cyan("\u2139")," You've specified a target CFI value of ",
      target_cfi, "."
    )
  }
  if (!(method %in% c("TKL", "WB", "CB"))) {
    stop("`method` must be `TKL`, `CB`, or `WB`.\n",
         crayon::cyan("\u2139"), " You've specified ",
         method, " as `method`.", call. = F)
  }
  if (!(is.list(mod)) |
      is.null(mod$loadings) |
      is.null(mod$Phi) |
      is.null(mod$Rpop)) {
    stop("`mod` must be a valid `simFA()` model object.", call. = F)
  }
  if (mod$cn$ModelError$ModelError == TRUE) {
    warning(paste0("The `simFA()` object you provided includes model error",
                   " parameters that will be ignored by this function."))
  }

  out_list <- list(Sigma = NA,
                   rmsea = NA,
                   cfi = NA,
                   fn_value = NA,
                   m = NA,
                   v = NA,
                   eps = NA,
                   W = NA)

  k <- ncol(mod$loadings) # number of major factors

  if (method == "WB") {
    wb_out <- wb(mod = mod,
              target_rmsea = target_rmsea,
              wb_mod = wb_mod)
    out_list$Sigma <- wb_out$Sigma
    out_list$rmsea <- rmsea(out_list$Sigma, mod$Rpop, k)
    out_list$cfi <- cfi(out_list$Sigma, mod$Rpop)
```

```
    out_list$m <- wb_out$m
  } else if (method == "CB") {
    out_list$Sigma <- cb(mod = mod,
                         target_rmsea = target_rmsea)
    out_list$rmsea <- rmsea(out_list$Sigma, mod$Rpop, k)
    out_list$cfi <- cfi(out_list$Sigma, mod$Rpop)
  } else if (method == "TKL") {
    tkl_out <- tkl(mod = mod,
                   target_rmsea = target_rmsea,
                   target_cfi = target_cfi,
                   tkl_ctrl = tkl_ctrl)

    out_list$Sigma <- tkl_out$RpopME
    out_list$rmsea <- tkl_out$rmsea
    out_list$cfi <- tkl_out$cfi
    out_list$v <- tkl_out$v
    out_list$eps <- tkl_out$eps
    out_list$W <- tkl_out$W
    out_list$fn_value <- tkl_out$fn_value
  }

  out_list
}
#' Objective function for optimizing RMSEA and CFI
#'
#' This is the objective function that is minimized by the `tkl()` function.
#'
#' @param par (vector) Values of model error variance (\eqn{\upsilon}) and
#'   epsilon (\eqn{\epsilon}).
#' @param Rpop (matrix) The model-implied correlation matrix.
#' @param W (matrix) Matrix of provisional minor common factor loadings with
#'   unit column variances.
#' @param p (scalar) Number of variables.
#' @param u (vector) Major common factor variances.
#' @param df (scalar) Model degrees of freedom.
#' @param target_rmsea (scalar) Target RMSEA value.
#' @param target_cfi (scalar) Target CFI value.
#' @param weights (vector) Vector of length two indicating how much weight to
#'   give RMSEA and CFI, e.g., `c(1,1)` (default) gives equal weight to both
#'   indices; `c(1,0)` ignores the CFI value.
#' @param WmaxLoading (scalar) Threshold value for `NWmaxLoading`.
#' @param NWmaxLoading (scalar) Maximum number of absolute loadings \eqn{\ge}
#'   `WmaxLoading` in any column of `W`.
#' @param penalty (scalar) Large (positive) penalty value to apply if the
#'   NWmaxLoading condition is violated.
#' @param return_values (boolean) If `TRUE`, return the objective function value
#'   along with `Rpop`, `RpopME`, `W`, `RMSEA`, `CFI`, `v`, and `eps` values. If
#'   `FALSE`, return only the objective function value.
```

```
#'
#' @export

obj_func <- function(par = c(v, eps),
                     Rpop, W, p, u, df,
                     target_rmsea, target_cfi,
                     weights = c(1, 1),
                     WmaxLoading = NULL,
                     NWmaxLoading = 2,
                     penalty = 0,
                     return_values = FALSE) {
  v <- par[1] # error variance
  eps <- par[2] # epsTKL

  # Rescale W using eps
  scaling_matrix <- diag((1 - eps)^(0:(ncol(W) - 1)))
  W <- W %*% scaling_matrix

  # Create W matrix such that the proportion of unique variance accounted for by
  # the minor common factors is v.
  # Adapted from simFA() (lines 691--698)
  wsq <- diag(tcrossprod(W))
  ModelErrorVar <- v * u
  W <- diag(sqrt(ModelErrorVar / wsq)) %*% W
  RpopME <- Rpop + tcrossprod(W)
  diag(RpopME) <- 1

  # ML objective function value for the full model
  # Adapted from simFA() (lines 651--660)
  Ft <- log(det(Rpop)) - log(det(RpopME)) +
    sum(diag(RpopME %*% solve(Rpop))) - p

  # ML objective function value for the baseline (independence) model
  Fb <- -log(det(RpopME))

  # Compute RMSEA and CFI values
  # Adapted from simFA() (lines 651--660)
  rmsea <- sqrt(Ft / df)
  cfi <- 1 - (Ft / -log(det(RpopME)))

  # Define penalty if WmaxLoading and NWmaxLoading are defined
  if (!is.null(WmaxLoading)) {
    # Takes the value 1 if any column of W has more than NWmaxLoading
    # abs(loadings) >= WmaxLoading
    max_loading_indicator <- any(
      max(apply(abs(W) >= WmaxLoading, 2, sum)) > NWmaxLoading
    )
  } else {
```

```
    max_loading_indicator <- 0
  }

  weights <- weights / sum(weights) # scale weights to sum to one

  # Compute objective function value
  # fn_value <- weights[1] * (rmsea - target_rmsea)^2 +
  #   weights[2] * (cfi - target_cfi)^2 +
  #   penalty * max_loading_indicator
  fn_value <- weights[1] * ((rmsea - target_rmsea)^2 / target_rmsea^2) +
    weights[2] * (((1 - cfi) - (1 - target_cfi))^2 / (1 - target_cfi)^2) +
    penalty * max_loading_indicator

  # Objective function value weights RMSEA and CFI differences equally; could be
  # changed, if necessary
  if (return_values == FALSE) {
    fn_value
  } else {
    names(fn_value) <- names(v) <- names(eps) <- NULL
    list(
      fn_value = fn_value,
      Rpop = Rpop,
      RpopME = RpopME,
      W = W,
      rmsea = rmsea,
      cfi = cfi,
      v = v,
      eps = eps
    )
  }
}
#' Calculate RMSEA between two correlation matrices
#'
#' Given two correlation matrices of the same dimension, calculate the RMSEA
#' value using the degrees of freedom for the exploratory factor analysis model
#' (see details).
#'
#' @param Sigma (matrix) Population correlation or covariance matrix (with model
#'   error).
#' @param Omega (matrix) Model-implied population correlation or covariance
#'   matrix.
#' @param k (scalar) Number of major common factors.
#'
#' @details Note that this function uses the degrees of freedom for an
#'   exploratory factor analysis model: \deqn{df = p(p-1)/2-(pk)+k(k-1)/2,}
#'   where \eqn{p} is the number of items and \eqn{k} is the number of major
#'   factors.
#'
```

```
#' @md
#' @export
#'
#' @examples
#' mod <- fungible::simFA(Model = list(NFac = 3),
#'                        Seed = 42)
#' set.seed(42)
#' Omega <- mod$Rpop
#' Sigma <- noisemaker(
#'   mod = mod,
#'   method = "CB",
#'   target_rmsea = 0.05
#' )$Sigma
#' rmsea(Sigma, Omega, k = 3)

rmsea <- function(Sigma, Omega, k) {
  if (!is.matrix(Sigma) | !is.matrix(Omega)) {
    stop("Sigma and Omega must be matrices.", call. = F)
  } else if (all.equal(Sigma, t(Sigma)) != TRUE) {
    stop("Sigma must be a symmetric matrix.", call. = F)
  } else if (all.equal(Omega, t(Omega)) != TRUE) {
    stop("Omega must be a symmetric matrix.", call. = F)
  } else if (all.equal(dim(Omega), dim(Sigma)) != TRUE) {
    stop("Sigma and Omega must have the same dimensions.", call. = F)
  } else if (!is.numeric(k) | ((k %% 1) != 0) | k < 0) {
    stop("`k` must be a non-negative integer.\n",
         crayon::cyan("\u2139"), " You've specified ", k, " as `k`.", call. = F)
  }

  p <- nrow(Sigma)
  df <- (p * (p - 1) / 2) - (p * k) + (k * (k - 1) / 2)
  Fm <- log(det(Omega)) - log(det(Sigma)) + sum(diag(Sigma %*% solve(Omega))) - p
  sqrt(Fm / df)
}
#' Generate an sem model from a simFA model object
#'
#' @param mod A `fungible::simFA()` model object.
#'
#' @export
#'
#' @examples
#' ex_mod <- fungible::simFA(Seed = 42)
#' semify(mod = ex_mod)
semify <- function(mod) {
  L <- mod$loadings
  Phi <- mod$Phi
  u <- 1 - mod$h2
```

```
# Specify factor loadings
loading_spec <- ""
loadings <- numeric(length = sum(L != 0))
loading_names <- character(length = sum(L != 0))
i <- 1
for (item in seq_len(nrow(L))) {
  for (factor in seq_len(ncol(L))) {
    if (L[item, factor] == 0) next
    loading_spec <- paste0(
      loading_spec, "F", factor, " -> ", "V", item,
      ", lambda", i, ", ", L[item, factor], "\n"
    )
    loadings[i] <- L[item, factor]
    loading_names[i] <- paste0("lambda", i)
    i <- i + 1
  }
}

# Specify latent variable variances (fixed at 1)
latent_var_spec <- ""
latent_var_names <- colnames(L)
for (factor in seq_len(ncol(L))) {
  latent_var_spec <- paste0(
    latent_var_spec, "F", factor,
    " <-> ", "F", factor, ", NA, 1\n"
  )
}

# Specify latent variable correlations
latent_cor_spec <- ""
latent_cor <- NULL
latent_cor_names <- NULL
if (ncol(L) > 1) {
  var_pairs <- utils::combn(nrow(Phi), 2)
  latent_cor <- numeric(length = ncol(var_pairs))
  latent_cor_names <- character(length = ncol(var_pairs))
  for (pair in seq_len(ncol(var_pairs))) {
    fi <- var_pairs[1, pair]
    fj <- var_pairs[2, pair]
    latent_cor_spec <- paste0(
      latent_cor_spec, "F", fi, " <-> ", "F", fj,
      ", phi", fi, fj, ", ", Phi[fi, fj], "\n"
    )
    latent_cor[pair] <- Phi[fi, fj]
    latent_cor_names[pair] <- paste0("phi", fi, fj)
  }
}
```

```r
  # Specify observed variable variances
  obs_var_spec <- ""
  obs_var <- numeric(length = nrow(L))
  obs_var_names <- paste0("psi", seq_len(nrow(L)))
  for (item in seq_len(nrow(L))) {
    obs_var_spec <- paste0(
      obs_var_spec, "V", item, " <-> ", "V", item, ", psi", item, ", ",
      u[item], "\n"
    )
    obs_var[item] <- u[item]
  }

  # Combine the loading, factor variance, and factor correlation specifications
  # to form the complete model specification
  model <- paste(
    loading_spec,
    latent_var_spec,
    latent_cor_spec,
    obs_var_spec, "\n"
  )

  # Vector of model parameters
  theta <- c(loadings, latent_cor, obs_var)
  names(theta) <- c(loading_names, latent_cor_names, obs_var_names)

  # Return sem model, vector of (named) model parameters, and factor names
  list(
    model = sem::specifyModel(text = model, quiet = TRUE),
    theta = theta,
    latent_var = latent_var_names
  )
}
#' Optimize TKL parameters to find a solution with target RMSEA and CFI values
#'
#' Find the optimal W matrix such that the RMSEA and CFI values are as close as
#' possible to the user-specified target values.
#'
#' @param mod A `fungible::simFA()` model object.
#' @param target_rmsea (scalar) Target RMSEA value.
#' @param target_cfi (scalar) Target CFI value.
#' @param tkl_ctrl (list) A control list containing the following TKL-specific
#'   arguments:
#'   * weights (vector) Vector of length two indicating how much weight to give
#'   RMSEA and CFI, e.g., `c(1,1)` (default) gives equal weight
#'   to both indices; `c(1,0)` ignores the CFI value.
#'   * v_start (scalar) Starting value to use for \eqn{\upsilon}, the proportion
#'   of uniqueness variance reallocated to the minor common factors. Note that
#'   only `v` as a proportion of the unique (not total) variance is supported
```

```
#'   in this function.
#'   * eps_start (scalar) Starting value to use for \eqn{\epsilon}, which
#'   controls how common variance is distributed among the minor common factors.
#'   * NminorFac (scalar) Number of minor common factors.
#'   * WmaxLoading (scalar) Threshold value for `NWmaxLoading`.
#'   * NWmaxLoading (scalar) Maximum number of absolute loadings \eqn{\ge}
#'   `WmaxLoading` in any column of \eqn{W}.
#'   * penalty (scalar) Penalty applied to objective function if the
#'   `NmaxLoading` condition isn't satisfied.
#'   * optim_type (character)  Which optimization function to use, `optim` or
#'   `ga`? `optim()` is faster, but might not converge in some cases. If `optim`
#'    doesn't converge, `ga` will be used as a fallback option.
#'   * max_tries (numeric) How many times to restart optimization with new start
#'   parameter values if optimization doesn't converge?
#'   * factr (numeric) controls the convergence of the "L-BFGS-B" method.
#'   Convergence occurs when the reduction in the objective is within this
#'   factor of the machine tolerance. Default is 1e7, that is a tolerance of
#'   about 1e-8. (when using `optim`).
#'   * maxit (number) Maximum number of iterations to use (when using `optim`).
#'   * ncores (boolean/scalar) Controls whether `ga()` optimization is done in
#'   parallel. If `TRUE`, uses the maximum available number of processor cores.
#'   If `FALSE`, does not use parallel processing. If an integer is provided,
#'   that's how many processor cores will be used (if available).
#' @md
#'
#' @details This function attempts to find optimal values of the TKL parameters
#'   \eqn{\upsilon} and \eqn{\epsilon} such that the resulting correlation
#'   matrix with model error (\eqn{\Sigma}) has population RMSEA and/or CFI
#'   values that are close to the user-specified values. It is important to note
#'   that solutions are not guaranteed to produce RMSEA and CFI values that are
#'   reasonably close to the target values; in fact, some combinations of RMSEA
#'   and CFI will be difficult or impossible to obtain for certain models (see
#'   Lai & Green, 2016). It can be particularly difficult to find good solutions
#'   when additional restrictions are placed on the minor factor loadings (i.e.,
#'   using the `WmaxLoading` and `NWmaxLoading` arguments).
#'
#'   Optimization is fastest when the `optim_type = optim` optimization method
#'   is chosen. This indicates that optimization should be done using the
#'   `L-BFGS-B` algorithm implemented in the `optim()` function. However, this
#'   method can sometimes fail to find a solution. In that case, I recommend
#'   setting `optim_type = ga`, which indicates that a genetic algorithm
#'   (implemented in `GA::ga()`) will be used. This method takes longer than
#'   `optim()` but is more likely to find a solution.
#'
#' @export
#' @references Tucker, L. R., Koopman, R. F., & Linn, R. L. (1969). Evaluation
#'   of factor analytic research procedures by means of simulated correlation
#'   matrices. *Psychometrika*, *34*(4), 421-459.
```

```r
#'   <https://doi.org/10/chcxvf>

tkl <- function(mod,
                target_rmsea = NULL,
                target_cfi = NULL,
                tkl_ctrl = list()) {

  # Create default tkl_ctrl list; modify elements if changed by the user
  tkl_ctrl_default <- list(weights = c(rmsea = 1, cfi = 1),
                           v_start = stats::runif(1, 0.02, 0.9),
                           eps_start = stats::runif(1, 0, 0.8),
                           NMinorFac = 50,
                           WmaxLoading = NULL,
                           NWmaxLoading = 2,
                           debug = FALSE,
                           penalty = 1e6,
                           optim_type = "optim",
                           max_tries = 100,
                           factr = 1e6,
                           maxit = 5000,
                           ncores = FALSE)

  # Update the elements of the default tkl_ctrl list that have been changed by
  # the user
  tkl_ctrl_default <- tkl_ctrl_default[sort(names(tkl_ctrl_default))]
  tkl_ctrl_default[names(tkl_ctrl)] <- tkl_ctrl

  # Create objects for each of the elements in tkl_ctrl
  weights <- tkl_ctrl_default$weights
  v_start <- tkl_ctrl_default$v_start
  eps_start <- tkl_ctrl_default$eps_start
  NMinorFac <- tkl_ctrl_default$NMinorFac
  WmaxLoading <- tkl_ctrl_default$WmaxLoading
  NWmaxLoading <- tkl_ctrl_default$NWmaxLoading
  debug <- tkl_ctrl_default$debug
  penalty <- tkl_ctrl_default$penalty
  optim_type <- tkl_ctrl_default$optim_type
  ncores <- tkl_ctrl_default$ncores
  max_tries <- tkl_ctrl_default$max_tries
  factr <- tkl_ctrl_default$factr

  # Check arguments
  if (!is.null(target_rmsea)) {
    if (target_rmsea < 0 | target_rmsea > 1) {
      stop("The target RMSEA value must be a number between 0 and 1.\n",
           crayon::cyan("\u2139"), " You've specified a target RMSEA value of ",
           target_rmsea, ".", call. = F)
    }
```

```
}
if (!is.null(target_cfi)) {
  if (target_cfi > 1 | target_cfi < 0) {
    stop("Target CFI value must be between 0 and 1\n",
         crayon::cyan("\u2139"), " You've specified a target CFI value of ",
         target_cfi, ".", call. = F)
  }
}
if (is.null(target_cfi) & is.null(target_rmsea)) {
  stop("Either target RMSEA or target CFI (or both) must be specified.")
}
if (eps_start < 0 | eps_start > 1) {
  stop("The value of eps_start must be between 0 and 1.", call. = F)
}
if (v_start < 0 | v_start > 1) {
  stop("The value of v_start must be between 0 and 1.", call. = F)
}
if (!(is.list(mod)) |
    is.null(mod$loadings) |
    is.null(mod$Phi) |
    is.null(mod$Rpop)) {
  stop("`mod` must be a valid `simFA()` model object.", call. = F)
}
if (!is.numeric(weights) | length(weights) != 2) {
  stop("`weights` must be a numeric vector of length two.", call. = F)
}
if (NMinorFac < 0) {
  stop("The number of minor factors must be non-negative.\n",
       crayon::cyan("\u2139"), " You've asked for ", NMinorFac,
       " minor factors.", call. = F)
}
if (!(optim_type %in% c("optim", "ga"))) {
  stop("`optim_type` must be either `optim` or `ga`.\n",
       crayon::cyan("\u2139"), " You've supplied ", optim_type,
       " as `optim_type`.", call. = F)
}
if (!is.numeric(penalty) | penalty < 0) {
  stop("`penalty` must be a postive number.\n",
       crayon::cyan("\u2139"), " You've supplied ", penalty,
       " as `penalty`.", call. = F)
}
if (!is.null(WmaxLoading)) {
  if (!is.numeric(WmaxLoading) | WmaxLoading <= 0) {
    stop("`WmaxLoading` must be a positive number.\n",
         crayon::cyan("\u2139"), " You've supplied ", WmaxLoading,
         " as `WmaxLoading`.", call. = F)
  }
}
```

```
if (((NWmaxLoading %% 1) != 0) | NWmaxLoading < 0) {
  stop("`NWmaxLoading` must be a non-negative integer.\n",
       crayon::cyan("\u2139"), " You've supplied ", NWmaxLoading,
       " as `NWmaxLoading`.", call. = F)
}

# If no CFI value is given, set the weight to zero and set target_cfi to a
# no-null value (it will be ignored in the optimization)
if (is.null(target_cfi)) {
  weights[2] <- 0
  target_cfi <- 999
}

# Same for RMSEA
if (is.null(target_rmsea)) {
  weights[1] <- 0
  target_rmsea <- 999
}

L <- mod$loadings
Phi <- mod$Phi

# Create W with eps = 0
W <- MASS::mvrnorm(
  n = nrow(L),
  mu = rep(0, NMinorFac),
  Sigma = diag(NMinorFac)
)

p <- nrow(L) # number of items
k <- ncol(L) # number of major factors

CovMajor <- L %*% Phi %*% t(L)
u <- 1 - diag(CovMajor)
Rpop <- CovMajor
diag(Rpop) <- 1 # ensure unit diagonal

df <- (p * (p - 1) / 2) - (p * k) + (k * (k - 1) / 2) # model df

start_vals <- c(v_start, eps_start)

if (optim_type == "optim") {
  ctrl <- list(factr = factr)
  if (debug == TRUE) {
    ctrl$trace <- 5
    ctrl$REPORT <- 1
  }
  # Try optim(); if it fails, then use GA instead
```

```
  opt <- NULL
  tries <- 0
  converged <- FALSE
  while (converged == FALSE & (tries <= max_tries)) {
    if (tries > 1) start_vals <- c(v_start = stats::runif(1, 0.02, 0.9),
                                   eps_start = stats::runif(1, 0, 0.8))
    tryCatch(
      {
        opt <- stats::optim(
          par = start_vals,
          fn = obj_func,
          method = "L-BFGS-B",
          lower = c(0.001, 0), # can't go lower than zero
          upper = c(1, 1), # can't go higher than one
          Rpop = Rpop,
          W = W,
          p = p,
          u = u,
          df = df,
          target_rmsea = target_rmsea,
          target_cfi = target_cfi,
          weights = weights,
          WmaxLoading = WmaxLoading,
          NWmaxLoading = NWmaxLoading,
          control = ctrl,
          penalty = penalty
        )
        par <- opt$par
      },
      error = function(e) NULL
    )

    tries <- tries + 1
    if (is.null(opt)) {
      converged <- FALSE
      next
    }

    converged <- opt$convergence == 0
  }

  # If the algorithm fails to converge or produces NULL output, try GA instead
  if (is.null(opt)) {
    opt <- list(convergence = FALSE)
  }

  if (opt$convergence != 0) {
    optim_type <- "ga"
```

```
    warning("`optim()` failed to converge, using `ga()` instead.",
            call. = FALSE)
  }
}

if (optim_type == "ga") {
  opt <- GA::ga(
    type = "real-valued",
    fitness = function(x) {
      -obj_func(x,
        Rpop = Rpop,
        W = W,
        p = p,
        u = u,
        df = df,
        target_rmsea = target_rmsea,
        target_cfi = target_cfi,
        weights = weights,
        WmaxLoading = WmaxLoading,
        NWmaxLoading = NWmaxLoading,
        penalty = penalty
      )
    },
    lower = c(0, 0),
    upper = c(1, 1),
    popSize = 50,
    maxiter = 1000,
    run = 100,
    parallel = ncores,
    monitor = FALSE
  )
  par <- opt@solution[1, ]
}

obj_func(
  par = par,
  Rpop = Rpop,
  W = W,
  p = p,
  u = u,
  df = df,
  target_rmsea = target_rmsea,
  target_cfi = target_cfi,
  weights = weights,
  WmaxLoading = WmaxLoading,
  NWmaxLoading = NWmaxLoading,
  return_values = TRUE,
  penalty = penalty
```

```
  )
}
#' Wu & Browne model error method
#'
#' Generate a population correlation matrix using the model described in Wu and
#' Browne (2015).
#'
#' @param mod A `fungible::simFA()` model object.
#' @param target_rmsea (scalar) Target RMSEA value.
#' @param wb_mod (`lm` object) An optional `lm` object used to find a target
#'   RMSEA value that results in solutions with RMSEA values close to the
#'   desired value. Note that if no `wb_mod` is provided, a model will be
#'   estimated at run time. If many population correlation matrices are going to
#'   be simulated using the same model, it will be considerably faster to
#'   estimate `wb_mod` ahead of time. See also `get_wb_mod()`.
#' @param adjust_target (TRUE; logical) Should the target_rmsea value be
#'   adjusted to ensure that solutions have RMSEA values that are close to the
#'   provided target RMSEA value? Defaults to TRUE and should stay there unless
#'   you have a compelling reason to change it.
#'
#' @author Justin Kracht <krach018@umn.edu>
#' @references Wu, H., & Browne, M. W. (2015). Quantifying adventitious error in
#'   a covariance structure as a random effect. *Psychometrika*, *80*(3),
#'   571-600. <https://doi.org/10/gjrkc4>
#'
#' @export
#' @details The Wu and Browne method generates a correlation matrix with model
#'   error (\eqn{\Sigma}) using
#'
#'   \deqn{(\Sigma | \Omega) ~ IW(m, m \Omega),}
#'
#'   where \eqn{m ~= 1/\epsilon^2} is a precision parameter related to RMSEA
#'   (\eqn{\epsilon}) and \eqn{IW(m, m \Omega)} denotes an inverse Wishart
#'   distribution. Note that *there is no guarantee that the RMSEA will be very
#'   close to the target RMSEA*, particularly when the target RMSEA value is
#'   large. Based on experience, the method tends to give solutions with RMSEA
#'   values that are larger than the target RMSEA values. Therefore, it might be
#'   worth using a target RMSEA value that is somewhat lower than what is
#'   actually needed. Alternatively, the \code{\link{get_wb_mod}} function can
#'   be used to estimate a coefficient to shrink the target RMSEA value by an
#'   appropriate amount so that the solution RMSEA values are close to the
#'   (nominal) target values.
#'
#' @examples
#' # Specify a default model using simFA()
#' mod <- fungible::simFA(Seed = 42)
#'
#' set.seed(42)
```

```
#' wb(mod, target_rmsea = 0.05)

wb <- function(mod,
               target_rmsea,
               wb_mod = NULL,
               adjust_target = TRUE) {

  if (!(is.list(mod)) |
      is.null(mod$loadings) |
      is.null(mod$Phi) |
      is.null(mod$Rpop)) {
    stop("`mod` must be a valid `simFA()` model object.", call. = F)
  }
  if (target_rmsea < 0 | target_rmsea > 1) {
    stop("The target RMSEA value must be a number between 0 and 1.\n",
         crayon::cyan("\u2139"), " You've specified a target RMSEA value of ",
         target_rmsea, ".", call. = F)
  }
  if (!is.null(wb_mod)) {
    if (class(wb_mod) != "lm") {
      stop("`wb_mod` must be an object of class `lm`.", call. = F)
    }
  }

  if (is.null(wb_mod) & (adjust_target == TRUE)) {
    if (target_rmsea >= 0.095) {
      upper <- target_rmsea + 0.01
    } else {
      upper <- 0.095
    }
    wb_mod <- get_wb_mod(mod, upper = upper)
  }

  # Use wb_mod to find the correct target_rmsea value to use
  if (!is.null(wb_mod) & (adjust_target == TRUE)) {
    target_rmsea <- stats::predict(
      wb_mod,
      newdata = data.frame(rmsea_medians = target_rmsea)
    )
  }

  v <- target_rmsea^2
  m <- v^-1 # m is the precision parameter, Wu and Browne (2015), p. 576

  Omega <- mod$Rpop
  p <- nrow(Omega)
  if (m < p) {
    stop("Target RMSEA value is too large, try a smaller value.", call. = FALSE)
```

```
  }

  Sigma <- MCMCpack::riwish(m, m * Omega)
  list(Sigma = stats::cov2cor(Sigma),
       m = m)
}
```

A.2 Main Simulation Code

```
if (!require(noisemaker)) {
  devtools::install_github("JustinKracht/noisemaker",
                           dependencies = TRUE)
}

library(noisemaker)
library(here)
library(fungible)
library(tidyverse)
library(pbmcapply)

# Set variable values ---------------------------------------------------

DATA_DIR <- here("data")

# Number of simulation reps
reps <- 500

# Create a matrix of all of the simulation conditions
condition_matrix <- tidyr::expand_grid(
  factors = c(1, 3, 5, 10),
  items_per_factor = c(5, 15),
  factor_cor = c(0, .3, .6),
  loading = c("weak", "moderate", "strong"),
  target_rmsea = c(0.025, 0.065, 0.090)
) %>% filter(
  !(factors == 1 & factor_cor != 0)
) %>% mutate(
  condition_num = 1:n()
) %>% dplyr::relocate(condition_num)

# Add target CFI values
condition_matrix <- condition_matrix %>% mutate(
  target_cfi = case_when(target_rmsea == 0.025 ~ 0.99,
                         target_rmsea == 0.065 ~ 0.95,
                         target_rmsea == 0.090 ~ 0.90)
```

```
)

saveRDS(condition_matrix, here("data", "condition_matrix.RDS"))

# Make matrix of optimization method choices for TKL
method_matrix <- tidyr::expand_grid(
  rmsea_weight = c(0,1),
  cfi_weight = c(0,1)
) %>% filter(rmsea_weight != 0 | cfi_weight != 0)

# tryCatch function to wrap noisemaker so that warnings and errors are captured
myTryCatch <- function(expr) {
  warn <- err <- NULL
  value <- withCallingHandlers(
    tryCatch(expr, error=function(e) {
      err <<- e
      NULL
    }), warning=function(w) {
      warn <<- w
      invokeRestart("muffleWarning")
    })
  list(value=value, warning=warn, error=err)
}

# Simulation loop ---------------------------------------------------------

RNGkind("L'Ecuyer-CMRG")
set.seed(666)
seed_list <- sample(1e7, size = nrow(condition_matrix), replace = FALSE)

results_list <- pbmclapply(
  X = seq_along(condition_matrix$condition_num),
  # X = 154:nrow(condition_matrix),
  FUN = function(condition) {

    set.seed(seed_list[condition])

    factors <- condition_matrix$factors[condition]
    items_per_factor <- condition_matrix$items_per_factor[condition]
    factor_cor <- condition_matrix$factor_cor[condition]
    loading <- condition_matrix$loading[condition]
    target_rmsea <- condition_matrix$target_rmsea[condition]
    target_cfi <- condition_matrix$target_cfi[condition]

    FacLoadRange <- switch(loading,
                           "weak" = 0.4,
                           "moderate" = 0.6,
                           "strong" = 0.8)
```

```
# Generate factor model
mod <- simFA(Model = list(NFac = factors,
                          NItemPerFac = items_per_factor,
                          Model = "oblique"),
             Loadings = list(FacLoadDist = "fixed",
                             FacLoadRange = FacLoadRange),
             Phi = list(PhiType = "fixed",
                        MaxAbsPhi = factor_cor))

wb_mod <- get_wb_mod(mod, n = 100, values = 15)

sigma_list <- purrr::map(
  .x = seq_len(reps),
  .f = function(i, mod, target_rmsea) {

    # TKL method variations
    sigma_tkl_rmsea <- myTryCatch(
      noisemaker(
        mod,
        method = "TKL",
        target_rmsea = target_rmsea,
        target_cfi = NULL,
        tkl_ctrl = list(WmaxLoading = .3,
                        NWmaxLoading = 2,
                        max_tries = 100,
                        maxit = 1000)
      )
    )

    sigma_tkl_cfi <- myTryCatch(
      noisemaker(
        mod,
        method = "TKL",
        target_rmsea = NULL,
        target_cfi = target_cfi,
        tkl_ctrl = list(WmaxLoading = .3,
                        NWmaxLoading = 2,
                        max_tries = 100,
                        maxit = 1000)
      )
    )

    sigma_tkl_rmsea_cfi <- myTryCatch(
      noisemaker(
        mod,
        method = "TKL",
        target_rmsea = target_rmsea,
        target_cfi = target_cfi,
```

```
            tkl_ctrl = list(WmaxLoading = .3,
                            NWmaxLoading = 2,
                            max_tries = 100,
                            maxit = 1000)
          )
        )

        # Other methods
        # Skip CB for the largest conditions; takes way too long.
        if (condition %in% 154:nrow(condition_matrix)) {
          sigma_cb <- list("value" = NA,
                           "warning" = NULL,
                           "error" = NULL)
        } else {
          sigma_cb  <- myTryCatch(noisemaker(mod, method = "CB",
                                             target_rmsea = target_rmsea))
        }
        sigma_wb  <- myTryCatch(noisemaker(mod, method = "WB",
                                           target_rmsea = target_rmsea,
                                           wb_mod = wb_mod))

        list(sigma_tkl_rmsea = sigma_tkl_rmsea,
             sigma_tkl_cfi = sigma_tkl_cfi,
             sigma_tkl_rmsea_cfi = sigma_tkl_rmsea_cfi,
             sigma_cb  = sigma_cb,
             sigma_wb  = sigma_wb)
      },
      mod = mod,
      target_rmsea = target_rmsea
    )

    # Save condition results
    saveRDS(
      sigma_list,
      file = here(
        "data",
        paste0("results_",
               formatC(condition, width = 3, flag = 0),
               ".RDS"
        )
      )
    )

  },
  mc.preschedule = FALSE,
  mc.cores = 18
)
# Extract simulation data from lists and calculate alternative fit statistics
```

```
library(dplyr)
library(tidyr)
library(purrr)
library(parallel)
library(stringr)
library(fungible)
library(here)
library(purrrgress)

condition_matrix <- readRDS(here("data", "condition_matrix.RDS"))

# Create results matrix -----------------------------------------------

reps <- 500

# Create a matrix to hold results
results <- condition_matrix[rep(1:nrow(condition_matrix), each = reps),]

# For each rep in each condition, record the rep number
results$rep_num <- rep(1:reps, times = nrow(condition_matrix))

# Create model error method variable
results <- expand_grid(
  results,
  error_method = c("TKL_rmsea", "TKL_cfi", "TKL_rmsea_cfi", "CB", "WB")
)

# Function definitions ------------------------------------------------

# Compute the fit statistics that were not computed in the main simulation
# loop

compute_fit_stats <- function(error_method_results, Rpop, k) {

  # Check if value is NULL; if so, return empty list
  if (is.null(error_method_results$value)) {
    out = data.frame(RMSEA_thetahat = NA,
                     CFI_thetahat = NA,
                     CRMR_theta = NA,
                     CRMR_thetahat = NA,
                     TLI_theta = NA,
                     TLI_thetahat = NA)
    return(out)
  }

  RpopME <- pluck(error_method_results, "value", "Sigma")

  p <- ncol(RpopME)
```

```
tp <- p * (p + 1)/2

fout <- factanal(covmat = RpopME, factors = k,
                 rotation = "none", n.obs = 1000,
                 control = list(nstart = 100))

Fhat <- fout$loadings
Sigma_k <- Fhat %*% t(Fhat)
diag(Sigma_k) <- 1

Tr <- function(X) sum(diag(X))

num_thetahat <- log(det(Sigma_k)) - log(det(RpopME)) +
  Tr(RpopME %*% solve(Sigma_k)) - p

k <- ncol(Fhat)
DF <- (p * (p - 1)/2) - (p * k) + (k * (k - 1)/2)
DF_B <- (p * (p - 1)/2) - (p * p) + (p * (p - 1)/2)

F_T_thetahat <- log(det(Sigma_k)) - log(det(RpopME)) +
  Tr(RpopME %*% solve(Sigma_k)) - p
F_B_thetahat <- -log(det(RpopME))

F_T_theta <- log(det(Rpop)) - log(det(RpopME)) +
  Tr(RpopME %*% solve(Rpop)) - p
F_B_theta <- -log(det(RpopME))

# Calculate RMSEA thetahat
RMSEA_thetahat <- sqrt(num_thetahat/DF)

# Calculate CFI thetahat
CFI_thetahat <- 1 - F_T_thetahat/F_B_thetahat

# Calculate CRMR values
CRMR_theta <- sqrt(
  sum(((RpopME - Rpop)[upper.tri(Rpop, diag = FALSE)]^2)) / (tp - p)
)

CRMR_thetahat <- sqrt(
  sum(((RpopME - Sigma_k)[upper.tri(Rpop, diag = FALSE)]^2)) / (tp - p)
)

# Calculate TLI values (from Xia & Yang, 2019, p. 412)
TLI_theta <- 1 - ( (F_T_theta / DF) / (F_B_theta / DF_B) )
TLI_thetahat <- 1 - ( (F_T_thetahat / DF) / (F_B_thetahat / DF_B) )

data.frame(RMSEA_thetahat = RMSEA_thetahat,
           CFI_thetahat = CFI_thetahat,
```

```
              CRMR_theta = CRMR_theta,
              CRMR_thetahat = CRMR_thetahat,
              TLI_theta = TLI_theta,
              TLI_thetahat = TLI_thetahat)
}

# Compute d values
compute_d_values <- function(error_method_data, target_rmsea, target_cfi) {

  if (is.null(error_method_data$value)) {
    out = data.frame(d1 = NA,
                     d2 = NA,
                     d3 = NA)
    return(out)
  }

  rmsea <- pluck(error_method_data, "value", "rmsea")
  cfi <- pluck(error_method_data, "value", "cfi")

  d1 <- abs(rmsea - target_rmsea)
  d2 <- abs(cfi - target_cfi)
  d3 <- d1 + d2

  data.frame(
    d1 = d1,
    d2 = d2,
    d3 = d3
  )
}

# Check if there are any major factors in W
check_w_major_factors <- function(W) {
  if (!is.matrix(W)) {
    NA
  } else {
    sum(W[,1] >= .3) > 2
  }
}

# Create a vector of result file paths ------------------------------------

results_files <- list.files(
  path = "data",
  pattern = ".*[0-9]+\\.RDS",
  full.names = TRUE
)

# Read data from each condition and calculate statistics ------------------
```

```
# Calculate alternative fit statistics

pbmcapply::pbmclapply(
  X = seq_along(results_files),
  FUN = function(i, results_files, condition_matrix) {

    cat("/nWorking on condition:", i)

    # Read in loading matrix
    condition_results <- readRDS(results_files[i])

    # Extract the condition number from the results file name
    condition_num <- as.numeric(
      str_extract(results_files[i], pattern = "[0-9]+")
    )

    # Extract condition information (loading strength, num. factors, num. items)
    j <- which(condition_matrix$condition_num == condition_num)
    loading_condition <- condition_matrix$loading[j]
    k <- condition_matrix$factors[j]
    p <- condition_matrix$items_per_factor[j] * k

    # Extract target RMSEA and CFI values
    target_rmsea <- condition_matrix$target_rmsea[j]
    target_cfi <- condition_matrix$target_cfi[j]

    # Create factor loading matrix
    loadings <- switch(loading_condition,
                       "weak" = .4,
                       "moderate" = .6,
                       "strong" = .8)

    # Extract Rpop
    mod <- fungible::simFA(
      Model = list(NFac = condition_matrix$factors[j],
                   NItemPerFac = condition_matrix$items_per_factor[j],
                   Model = "oblique"),
      Loadings = list(FacLoadDist = "fixed",
                      FacLoadRange = loadings),
      Phi = list(PhiType = "fixed",
                 MaxAbsPhi = condition_matrix$factor_cor[j])
    )

    Rpop <- mod$Rpop

    condition_results_matrix <- purrr::map_dfr(
      .x = seq_along(condition_results),
      .f = function(z, Rpop, p, k, target_rmsea, target_cfi,
```

```
                condition_num) {

        rep_results <- condition_results[[z]]
        other_fit_stats <- map_dfr(.x = rep_results,
                                   ~ compute_fit_stats(., Rpop = Rpop, k = k))
        d_values <- map_dfr(.x = rep_results,
                            ~ compute_d_values(.,
                                               target_rmsea = target_rmsea,
                                               target_cfi = target_cfi))

        out <- data.frame(
          condition_num = rep(condition_num, 5),
          rep_num = rep(z, 5),
          cfi = map_dbl(rep_results, ~ pluck(., "value", "cfi", .default = NA)),
          cfi_thetahat = other_fit_stats$CFI_thetahat,
          rmsea = map_dbl(rep_results, ~ pluck(., "value", "rmsea",
                                               .default = NA)),
          rmsea_thetahat = other_fit_stats$RMSEA_thetahat,
          crmr = other_fit_stats$CRMR_theta,
          crmr_thetahat = other_fit_stats$CRMR_thetahat,
          tli = other_fit_stats$TLI_theta,
          tli_thetahat = other_fit_stats$TLI_thetahat,
          m = map_dbl(rep_results, ~ pluck(., "value", "m", .default = NA)),
          v = map_dbl(rep_results, ~ pluck(., "value", "v", .default = NA)),
          eps = map_dbl(rep_results, ~ pluck(., "value", "eps", .default = NA)),
          fn_value = map_dbl(rep_results, ~ pluck(., "value", "fn_value",
                                                  .default = NA)),
          error_method = c("tkl_rmsea", "tkl_cfi", "tkl_rmsea_cfi", "cb", "wb"),
          w_has_major_factors = map_lgl(rep_results,
                                        ~ check_w_major_factors(
                                          pluck(., "value", "W", .default = NA)
                                        )),
          d1 = d_values$d1,
          d2 = d_values$d2,
          d3 = d_values$d3,
          warning = map_chr(rep_results, .f = function(x) {
            warning <- pluck(x, "warning")
            if (is.null(warning)) {
              warning <- NA
            } else {
              warning <- as.character(warning)
            }
            warning
          }),
          error = map_chr(rep_results, .f = function(x) {
            error <- pluck(x, "error")
            if (is.null(error)) {
              error <- NA
```

```
          } else {
            error <- as.character(error)
          }
          error
        })
      )

      rownames(out) <- NULL
      out
    },
    Rpop = Rpop, p = p, k = k, target_rmsea = target_rmsea,
    target_cfi = target_cfi, condition_num
  )

  saveRDS(condition_results_matrix,
          file = paste0("data/results_matrix_",
                        formatC(i, width = 3, flag = 0),
                        ".RDS"))

}, results_files = results_files, condition_matrix = condition_matrix,
mc.cores = 8
)
```

A.3 Secondary Simulation: Check Effect of λ Values

```
# Set the number of reps
reps <- 200

# Create a matrix of fully-crossed conditions
factors <- unique(results_matrix$factors)
items_per_factor <- 15
loading <- c(.4)
target_rmsea <- c(0.090)
penalty <- c(0, .1, 1, 1e1, 1e2, 1e3, 1e4, 1e5, 1e6)

conditions_matrix <- expand.grid(
  factors = factors,
  items_per_factor = items_per_factor,
  loading = loading,
  target_rmsea = target_rmsea
)


# Function to check whether a W matrix has more than two factor loadings greater
# than 0.3 for any minor factor.
```

```r
check_constraints <- function(W) {
  any(max(apply(abs(W) >= .3, 2, sum)) > 2)
}

# For each condition, generate a population correlation matrix without model
# error and then generate 200 population correlations with model error using
# the TKL (RMSEA) method
set.seed(666)
constraint_violations <- map_dfr(
  .x = 1:nrow(conditions_matrix),
  .f = function(condition) {

    cat("\nWorking on condition", condition, "of", nrow(conditions_matrix))

    # Generate population correlation matrix without model error
    mod <- simFA(
      Model = list(NFac = conditions_matrix$factors[condition],
                   NItemPerFac = conditions_matrix$items_per_factor[condition],
                   Model = "orthogonal"),
      Loadings = list(FacLoadDist = "fixed",
                      FacLoadRange = conditions_matrix$loading[condition])
    )

    # Generate 200 population correlation matrices with model error
    pro_map_dfr(.x = 1:reps,
                .f = function(x, target_rmsea) {
                  map_dfr(.x = penalty,
                          .f = function(penalty, mod, target_rmsea, seed) {
                            set.seed(seed)
                            sol <- noisemaker(mod = mod,
                                              method = "TKL",
                                              target_rmsea = target_rmsea,
                                              target_cfi = NULL,
                                              tkl_ctrl = list(penalty = penalty,
                                                              NWmaxLoading = 2,
                                                              WmaxLoading = .3))
                            w_constraints_violated <- check_constraints(sol$W)

                            c(penalty = penalty,
                              constraints_violated = w_constraints_violated)
                          },
                          mod = mod,
                          target_rmsea = target_rmsea,
                          seed = sample(1e6, 1))
                }, target_rmsea = conditions_matrix$target_rmsea[condition])
  }
)
```

```
# Bind the conditions matrix to the results to indicate which result belongs to
# which condition
constraint_violations <- bind_cols(
  conditions_matrix[rep(1:nrow(conditions_matrix),
                        each = length(penalty) * reps),],
  constraint_violations
)

# Plot the results
constraint_violations %>%
  mutate(factors = as.factor(factors),
         penalty = factor(penalty,
                          labels = c("0", "0.1", "1", "10",
                                     "100", "1,000", "10,000",
                                     "100,000", "1,000,000"))) %>%
  group_by(factors, penalty) %>%
  summarise(percent = mean(constraints_violated, na.rm = TRUE)) %>%
  ggplot(aes(x = penalty, y = percent, color = factors, shape = factors,
             linetype = factors, group = factors)) +
  geom_point() +
  geom_line() +
  scale_y_continuous(labels = scales::percent) +
  scale_color_brewer(palette = "Dark2", type = "qual") +
  labs(y = "Cases with Violtated W Constraints",
       x = latex2exp::TeX("$\\lambda$"),
       color = "Factors", shape = "Factors", linetype = "Factors") +
  theme_bw() +
  theme(legend.position = "bottom")

# Save the plot
ggsave(filename = here("img/penalty_values.png"),
       dpi = 320,
       height = 4,
       width = 6)
```

A.4 Secondary Simulation: Check Whether TKL_{CFI} Requires A Penalty

```
# Check whether the TKL (CFI) method leads to solutions that violate the W
# constraints when no penalty is applied
set.seed(42)

# Create a population correlation matrix without model error (Omega)
mod <- simFA(Model = list(NFac = 10,
                          NItemPerFac = 15,
```

```
                           Model = "orthogonal"),
              Loadings = list(FacLoadDist = "fixed",
                              FacLoadRange = .4))

# Function to check whether a W matrix has more than two factor loadings greater
# than 0.3 for any minor factor.
check_constraints <- function(W) {
  any(max(apply(abs(W) >= .3, 2, sum)) > 2)
}

# Generate 200 population correlation matrices with model error (Sigma) using
# the TKL (CFI) method without a penalty, then check how often the constraints
# on W were violated
constraints_violated_vec <- pbmclapply(
  X = 1:200,
  FUN = function(x) {
    sol <- noisemaker(mod = mod,
                      method = "TKL",
                      target_rmsea = 0.09,
                      target_cfi = 0.9,
                      tkl_ctrl = list(penalty = 0))
    check_constraints(sol$W)
  },
  mc.cores = 4
)

# What percent of cases had violated W constraints?
mean(constraints_violated_vec, na.rm = TRUE)
```

A.5 Secondary Simulation: Check Recovery of Model Fit Indices with Known-Possible Values

```
# Check to see if RMSEA/CFI combinations can be recovered by noisemaker()

library(fungible)
library(noisemaker)
library(tidyverse)

# Define conditions ----

factors <- c(1, 3, 5, 10)
items_per_factor <- c(5, 15)
factor_corr <- c(0, .3, .6)
factor_loading <- c(.4, .6, .8)
```

```
conditions_matrix <- expand.grid(
  factors = factors,
  items_per_factor = items_per_factor,
  factor_corr = factor_corr,
  factor_loading = factor_loading
)

conditions_matrix <- filter(conditions_matrix,
                            !(factors == 1 & factor_corr != 0))

# Generate a simFA model for each condition -------------------------------

simFA_mod_list <- purrrgress::pro_pmap(
  .l = conditions_matrix,
  ~ simFA(Model = list(NFac = ..1,
                       NItemPerFac = ..2,
                       Model = "oblique"),
          Loadings = list(FacLoadRange = ..4),
          Phi = list(MaxAbsPhi = ..3,
                     PhiType = "fixed"),
          ModelError = list(ModelError = TRUE),
          Seed = 123)
)

# Extract RMSEA and CFI fit statistics ----

fit_stats <- purrr::map(
  .x = simFA_mod_list,
  ~ list(RMSEA = pluck(.x, "ModelErrorFitStats", "RMSEA_theta"),
         CFI = pluck(.x, "ModelErrorFitStats", "CFI_theta"))
)

# Use noisemaker to generate models with error ----

set.seed(123)

noisy_mod_list <- purrrgress::pro_pmap(
  .l = list(mod = simFA_mod_list,
            target_rmsea = purrr::map(fit_stats, ~ pluck(., "RMSEA")),
            target_cfi = purrr::map(fit_stats, ~ pluck(., "CFI"))),
  function(mod, target_rmsea, target_cfi) {
    suppressWarnings(
      replicate(n = 50,
                noisemaker(
                  mod = mod, method = "TKL", target_rmsea = target_rmsea,
                  target_cfi = target_cfi,
                  tkl_ctrl = list(factr = 1e7,
                                  max_tries = 1000,
```

```
                                  WmaxLoading = .3,
                                  NWmaxLoading = 2,
                                  factr = 1e5))
      )
    )
  }
)

# Unpack list data into a data frame for plotting ----

noisy_data <- map_dfr(noisy_mod_list,
                      function(x) {as.data.frame(t(x)[,-c(1,5,8)])})
noisy_data <- noisy_data %>%
  mutate(across(rmsea:eps, ~unlist(.)))

noisy_data$condition <- rep(1:nrow(conditions_matrix), each = 50)
noisy_data$target_rmsea <- rep(purrr::map_dbl(fit_stats, ~ pluck(., "RMSEA")),
                               each = 50)
noisy_data$target_cfi <- rep(purrr::map_dbl(fit_stats, ~ pluck(., "CFI")),
                             each = 50)
noisy_data$condition <- as.factor(noisy_data$condition)
noisy_data <- as_tibble(cbind(noisy_data,
                              conditions_matrix[noisy_data$condition,]))

# Create variable labels
noisy_data <- noisy_data %>%
  mutate(factors = factor(factors,
                          levels = c(1, 3, 5, 10),
                          labels = paste("Factors:", c(1, 3, 5, 10))),
         items_per_factor = factor(items_per_factor,
                                   levels = c(5, 15),
                                   labels = paste("Items/Factor:", c(5, 15))),
         factor_corr = factor(factor_corr,
                              levels = c(0, .3, .6),
                              labels = paste("Factor Cor.:", c(0.0, 0.3, 0.6))),
         factor_loading = factor(factor_loading,
                                 levels = c(0.4, 0.6, 0.8),
                                 labels = paste("Loading:", c(0.4, 0.6, 0.8))))

saveRDS(noisy_data, file = "data/noisy_data.RDS")
# Set the number of reps
reps <- 200

# Create a matrix of fully-crossed conditions
factors <- unique(results_matrix$factors)
items_per_factor <- 15
loading <- c(.4)
target_rmsea <- c(0.090)
```

```
penalty <- c(0, .1, 1, 1e1, 1e2, 1e3, 1e4, 1e5, 1e6)

conditions_matrix <- expand.grid(
  factors = factors,
  items_per_factor = items_per_factor,
  loading = loading,
  target_rmsea = target_rmsea
)


# Function to check whether a W matrix has more than two factor loadings greater
# than 0.3 for any minor factor.
check_constraints <- function(W) {
  any(max(apply(abs(W) >= .3, 2, sum)) > 2)
}

# For each condition, generate a population correlation matrix without model
# error and then generate 200 population correlations with model error using
# the TKL (RMSEA) method
set.seed(666)
constraint_violations <- map_dfr(
  .x = 1:nrow(conditions_matrix),
  .f = function(condition) {

    cat("\nWorking on condition", condition, "of", nrow(conditions_matrix))

    # Generate population correlation matrix without model error
    mod <- simFA(
      Model = list(NFac = conditions_matrix$factors[condition],
                   NItemPerFac = conditions_matrix$items_per_factor[condition],
                   Model = "orthogonal"),
      Loadings = list(FacLoadDist = "fixed",
                      FacLoadRange = conditions_matrix$loading[condition])
    )

    # Generate 200 population correlation matrices with model error
    pro_map_dfr(.x = 1:reps,
                .f = function(x, target_rmsea) {
                  map_dfr(.x = penalty,
                          .f = function(penalty, mod, target_rmsea, seed) {
                            set.seed(seed)
                            sol <- noisemaker(mod = mod,
                                              method = "TKL",
                                              target_rmsea = target_rmsea,
                                              target_cfi = NULL,
                                              tkl_ctrl = list(penalty = penalty,
                                                              NWmaxLoading = 2,
                                                              WmaxLoading = .3))
```

```
                             w_constraints_violated <- check_constraints(sol$W)

                             c(penalty = penalty,
                               constraints_violated = w_constraints_violated)
                           },
                           mod = mod,
                           target_rmsea = target_rmsea,
                           seed = sample(1e6, 1))
                 }, target_rmsea = conditions_matrix$target_rmsea[condition])
  }
)

# Bind the conditions matrix to the results to indicate which result belongs to
# which condition
constraint_violations <- bind_cols(
  conditions_matrix[rep(1:nrow(conditions_matrix),
                        each = length(penalty) * reps),],
  constraint_violations
)

# Plot the results
constraint_violations %>%
  mutate(factors = as.factor(factors),
         penalty = factor(penalty,
                          labels = c("0", "0.1", "1", "10",
                                     "100", "1,000", "10,000",
                                     "100,000", "1,000,000"))) %>%
  group_by(factors, penalty) %>%
  summarise(percent = mean(constraints_violated, na.rm = TRUE)) %>%
  ggplot(aes(x = penalty, y = percent, color = factors, shape = factors,
             linetype = factors, group = factors)) +
  geom_point() +
  geom_line() +
  scale_y_continuous(labels = scales::percent) +
  scale_color_brewer(palette = "Dark2", type = "qual") +
  labs(y = "Cases with Violtated W Constraints",
       x = latex2exp::TeX("$\\lambda$"),
       color = "Factors", shape = "Factors", linetype = "Factors") +
  theme_bw() +
  theme(legend.position = "bottom")

# Save the plot
ggsave(filename = here("img/penalty_values.png"),
       dpi = 320,
       height = 4,
       width = 6)
```

Appendix B

Supplemental Tables and Figures

B.1 Supplemental Tables

B.1.1 Logistic Regression on W Constraint Violations

Table B.1 gives a summary of the logistic regression of each of the independent simulation study variables and their interactions on a variable indicating whether a solution violated the constraints imposed on W (i.e., whether the solution had major minor factors). Note that some standard errors were quite large because the data were linearly separable due to there being some combinations of conditions that did not produce solutions with violated $\mathbf{W}$ constraints. All numeric predictors were scaled to have a mean of zero and a standard deviation of one to make the coefficient estimates more easily comparable. The "Model Fit (Very Good)" and "Error Method ($\text{TKL}_{\text{RMSEA}}$)" terms were used as baseline levels and were subsumed within the "Constant" term.

Constant	−26.88 (353.54)
Factors	3.87 (273.30)
Items/Factor	4.06 (273.65)
Factor Correlation	0.56 (110.83)
Loading	−5.22 (451.83)
Model Fit (Fair)	23.97 (353.54)
Model Fit (Poor)	26.34 (353.54)
Error Method (TKL_{CFI})	2.15 (579.47)
Error Method ($\text{TKL}_{\text{RMSEA/CFI}}$)	5.28 (425.86)
Factors × Items/Factor	1.08 (0.08)***
Factors × Factor Correlation	−0.28 (0.04)***
Factors × Loading	−0.23 (0.12)
Factors × Model Fit (Fair)	−1.08 (273.30)
Factors × Model Fit (Poor)	−0.99 (273.30)
Factors × Error Method (TKL_{CFI})	−8.08 (1.77)***

Factors × Error Method ($TKL_{RMSEA/CFI}$)	-4.59 (0.87)***
Items/Factor × Factor Correlation	0.01 (0.05)
Items/Factor × Loading	1.80 (0.18)***
Items/Factor × Model Fit (Fair)	1.02 (273.65)
Items/Factor × Model Fit (Poor)	-0.07 (273.65)
Items/Factor × Error Method (TKL_{CFI})	-5.71 (0.79)***
Items/Factor × Error Method ($TKL_{RMSEA/CFI}$)	-5.09 (0.82)***
Factor Correlation × Loading	-0.05 (0.07)
Factor Correlation × Model Fit (Fair)	-0.47 (110.83)
Factor Correlation × Model Fit (Poor)	-0.44 (110.83)
Factor Correlation × Error Method (TKL_{CFI})	-0.61 (0.75)
Factor Correlation × Error Method ($TKL_{RMSEA/CFI}$)	-0.15 (0.41)
Loading × Model Fit (Fair)	-1.00 (451.83)
Loading × Model Fit (Poor)	0.65 (451.83)
Loading × Error Method (TKL_{CFI})	13.33 (2.94)***
Loading × Error Method ($TKL_{RMSEA/CFI}$)	5.02 (1.19)***
Model Fit (Fair) × Error Method (TKL_{CFI})	-13.51 (579.47)
Model Fit (Poor) × Error Method (TKL_{CFI})	-14.03 (579.47)
Model Fit (Fair) × Error Method ($TKL_{RMSEA/CFI}$)	-24.05 (451.18)
Model Fit (Poor) × Error Method ($TKL_{RMSEA/CFI}$)	-13.22 (425.86)
AIC	35836.61
BIC	36204.33
Log Likelihood	-17883.30
Deviance	35766.61
Num. obs.	270000

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table B.1

Coefficient estimates and standard errors for the logistic regression model predicting the probability of producing a solution with violated W constraints (i.e., at least one minor factor with more than two absolute factor loadings greater than 0.3).

B.1.2 $|\mathbf{RMSEA_{obs}} - \mathbf{RMSEA_T}|$ Analysis of Variance

Table B.2 gives a summary of the analysis of variance for the absolute difference between the observed and target RMSEA values ($|\text{RMSEA}_{\text{obs}} - \text{RMSEA}_{\text{T}}|$). Small values of $|\text{RMSEA}_{\text{obs}} - \text{RMSEA}_{\text{T}}|$ indicated that a model-error method recovered the target RMSEA value well, whereas large values indicated that a model-error method recovered the target RMSEA value poorly. All numeric predictors were scaled to have a mean of zero and a standard deviation of one to make the coefficient estimates more easily comparable.

Effect	F	df_1	df_2	MSE	p	$\hat{\eta}^2_G$
Factors	18,898.11	1	422,156	0.00	$<$ .001	.043
Items/Factor	21,774.22	1	422,156	0.00	$<$ .001	.049
Factor Correlation	913.28	1	422,156	0.00	$<$ .001	.002
Loading	52,417.83	1	422,156	0.00	$<$ .001	.110
Model Fit	72,266.96	2	422,156	0.00	$<$ .001	.255
Error Method	176,258.22	4	422,156	0.00	$<$ .001	.625
Factors × Items/Factor	5,282.93	1	422,156	0.00	$<$ .001	.012
Factors × Factor Correlation	78.69	1	422,156	0.00	$<$ .001	.000
Factors × Loading	3,443.56	1	422,156	0.00	$<$ .001	.008
Factors × Model Fit	2,105.57	2	422,156	0.00	$<$ .001	.010
Factors × Error Method	6,383.80	4	422,156	0.00	$<$ .001	.057
Items/Factor × Factor Correlation	1,173.91	1	422,156	0.00	$<$ .001	.003
Items/Factor × Loading	451.76	1	422,156	0.00	$<$ .001	.001
Items/Factor × Model Fit	2,741.32	2	422,156	0.00	$<$ .001	.013
Items/Factor × Error Method	5,063.19	4	422,156	0.00	$<$ .001	.046
Factor Correlation × Loading	632.53	1	422,156	0.00	$<$ .001	.001
Factor Correlation × Model Fit	59.53	2	422,156	0.00	$<$ .001	.000
Factor Correlation × Error Method	348.18	4	422,156	0.00	$<$ .001	.003
Loading × Model Fit	6,569.05	2	422,156	0.00	$<$ .001	.030
Loading × Error Method	18,023.17	4	422,156	0.00	$<$ .001	.146
Model Fit × Error Method	17,916.27	8	422,156	0.00	$<$ .001	.253

Note. $\hat{\eta}^2_G$ denotes the estimate of the generalized eta-squared effect size.

Table B.2
Analysis of Variance for the absolute difference between the observed and target RMSEA values ($|RMSEA_{obs} - RMSEA_T|$).

B.1.3 $|\mathbf{CFI_{obs}} - \mathbf{CFI_T}|$ Analysis of Variance

Table B.3 gives a summary of the analysis of variance for the absolute difference between the observed and target CFI values ($|\mathrm{CFI_{obs}} - \mathrm{CFI_T}|$). Small values of $|\mathrm{CFI_{obs}} - \mathrm{CFI_T}|$ indicated that a model-error method recovered the target CFI value well, whereas large values indicated that a model-error method recovered the target CFI value poorly. All numeric predictors were scaled to have a mean of zero and a standard deviation of one to make the coefficient estimates more easily comparable.

Effect	F	df_1	df_2	MSE	p	$\hat{\eta}^2_G$
Factors	199,859.17	1	422,156	0.00	$< .001$	.321
Items/Factor	92,175.19	1	422,156	0.00	$< .001$	.179
Factor Correlation	2,944.60	1	422,156	0.00	$< .001$	.007
Loading	637,465.49	1	422,156	0.00	$< .001$	.602
Model Fit	192,400.20	2	422,156	0.00	$< .001$	.477
Error Method	285,742.77	4	422,156	0.00	$< .001$	.730
Factors × Items/Factor	7,994.17	1	422,156	0.00	$< .001$	.019
Factors × Factor Correlation	5,100.42	1	422,156	0.00	$< .001$	.012
Factors × Loading	44,849.10	1	422,156	0.00	$< .001$	.096
Factors × Model Fit	16,448.91	2	422,156	0.00	$< .001$	.072
Factors × Error Method	48,082.33	4	422,156	0.00	$< .001$	.313
Items/Factor × Factor Correlation	2,293.57	1	422,156	0.00	$< .001$	.005
Items/Factor × Loading	6,760.03	1	422,156	0.00	$< .001$	.016
Items/Factor × Model Fit	8,114.61	2	422,156	0.00	$< .001$	.037
Items/Factor × Error Method	28,681.51	4	422,156	0.00	$< .001$	.214
Factor Correlation × Loading	474.87	1	422,156	0.00	$< .001$	.001
Factor Correlation × Model Fit	57.72	2	422,156	0.00	$< .001$	.000
Factor Correlation × Error Method	508.94	4	422,156	0.00	$< .001$	.005
Loading × Model Fit	60,935.61	2	422,156	0.00	$< .001$	.224
Loading × Error Method	127,372.38	4	422,156	0.00	$< .001$	.547
Model Fit × Error Method	37,155.12	8	422,156	0.00	$< .001$	.413

Note. $\hat{\eta}^2_G$ denotes the estimate of the generalized eta-squared effect size.

Table B.3
Analysis of Variance for the absolute difference between the observed and target CFI values ($|CFI_{obs} - CFI_T|$).

B.1.4 D Analysis of Variance

Table B.4 gives a summary of the analysis of variance for the sum of the absolute differences between the observed and target CFI values and the observed and target RMSEA values, $D = |\text{RMSEA}_{\text{obs}} - \text{RMSEA}_{\text{T}}| + |\text{CFI}_{\text{obs}} - \text{CFI}_{\text{T}}|$. Small D values indicated that a model-error method recovered both the target CFI and target RMSEA values well, whereas large D values indicated that a model-error method recovered one (or both) of the target RMSEA or CFI values poorly. All numeric predictors were scaled to have a mean of zero and a standard deviation of one to make the coefficient estimates more easily comparable.

Effect	F	df_1	df_2	MSE	p	$\hat{\eta}^2_G$
Factors	232,107.54	1	422,156	0.00	$< .001$	.355
Items/Factor	114,684.42	1	422,156	0.00	$< .001$	.214
Factor Correlation	3,762.85	1	422,156	0.00	$< .001$	.009
Loading	734,305.08	1	422,156	0.00	$< .001$	.635
Model Fit	250,482.08	2	422,156	0.00	$< .001$	.543
Error Method	205,516.31	4	422,156	0.00	$< .001$	.661
Factors × Items/Factor	11,218.14	1	422,156	0.00	$< .001$	.026
Factors × Factor Correlation	4,961.79	1	422,156	0.00	$< .001$	.012
Factors × Loading	51,465.67	1	422,156	0.00	$< .001$	.109
Factors × Model Fit	19,457.68	2	422,156	0.00	$< .001$	.084
Factors × Error Method	42,811.29	4	422,156	0.00	$< .001$	.289
Items/Factor × Factor Correlation	3,107.40	1	422,156	0.00	$< .001$	.007
Items/Factor × Loading	7,700.49	1	422,156	0.00	$< .001$	.018
Items/Factor × Model Fit	10,305.73	2	422,156	0.00	$< .001$	.047
Items/Factor × Error Method	24,818.06	4	422,156	0.00	$< .001$	.190
Factor Correlation × Loading	280.38	1	422,156	0.00	$< .001$	.001
Factor Correlation × Model Fit	84.90	2	422,156	0.00	$< .001$	.000
Factor Correlation × Error Method	408.73	4	422,156	0.00	$< .001$	.004
Loading × Model Fit	71,005.69	2	422,156	0.00	$< .001$	.252
Loading × Error Method	111,162.03	4	422,156	0.00	$< .001$	.513
Model Fit × Error Method	27,940.24	8	422,156	0.00	$< .001$	.346

Note. $\hat{\eta}^2_G$ denotes the estimate of the generalized eta-squared effect size.

Table B.4
Analysis of Variance for D.

B.1.5 Logistic Regression on Qualitative Model Fit Index Agreement

Table B.1 gives a summary of the logistic regression of each of the independent simulation study variables and their interactions on a variable indicating whether a solution produced RMSEA and CFI values that indicated the same qualitative degree of model fit. Note that some standard errors were quite large because the data were linearly separable due to there being some combinations of conditions with either 100% or 0% rates of qualitative fit agreement. All numeric predictors were scaled to have a mean of zero and a standard deviation of one to make the coefficient estimates more easily comparable. The "Model Fit (Very Good)" and "Error Method (TKL_{RMSEA})" terms were used as baseline levels and were subsumed within the "Constant" term.

Constant	1.30 (0.03)***
Factors	−3.76 (0.04)***
Items/Factor	−3.02 (0.04)***
Factor Correlation	0.59 (0.02)***
Loading	16.25 (0.11)***
Model Fit (Fair)	−4.45 (0.04)***
Model Fit (Poor)	−9.82 (0.06)***
Error Method (TKL_{CFI})	5.14 (0.06)***
Error Method ($TKL_{RMSEA/CFI}$)	29.97 (36.77)
Error Method (CB)	−0.08 (0.04)*
Error Method (WB)	−0.13 (0.03)***
Factors × Items/Factor	−3.24 (0.03)***
Factors × Factor Correlation	−0.04 (0.01)**
Factors × Loading	7.37 (0.06)***
Factors × Model Fit (Fair)	0.09 (0.03)**
Factors × Model Fit (Poor)	−2.86 (0.04)***
Factors × Error Method (TKL_{CFI})	4.52 (0.04)***
Factors × Error Method ($TKL_{RMSEA/CFI}$)	0.33 (0.05)***
Factors × Error Method (CB)	0.19 (0.04)***
Factors × Error Method (WB)	0.21 (0.04)***
Items/Factor × Factor Correlation	−0.56 (0.02)***
Items/Factor × Loading	4.85 (0.07)***
Items/Factor × Model Fit (Fair)	0.40 (0.04)***
Items/Factor × Model Fit (Poor)	−3.57 (0.05)***
Items/Factor × Error Method (TKL_{CFI})	4.25 (0.05)***
Items/Factor × Error Method ($TKL_{RMSEA/CFI}$)	−1.09 (0.06)***
Items/Factor × Error Method (CB)	0.29 (0.04)***
Items/Factor × Error Method (WB)	0.11 (0.04)**
Factor Correlation × Loading	0.97 (0.04)***
Factor Correlation × Model Fit (Fair)	−0.58 (0.02)***
Factor Correlation × Model Fit (Poor)	−0.65 (0.03)***
Factor Correlation × Error Method (TKL_{CFI})	0.05 (0.02)*
Factor Correlation × Error Method ($TKL_{RMSEA/CFI}$)	−0.15 (0.03)***
Factor Correlation × Error Method (CB)	0.02 (0.02)
Factor Correlation × Error Method (WB)	−0.05 (0.02)*

Loading × Model Fit (Fair)	-8.37 (0.09)***
Loading × Model Fit (Poor)	4.22 (0.11)***
Loading × Error Method (TKL_{CFI})	-14.16 (0.11)***
Loading × Error Method ($\text{TKL}_{\text{RMSEA/CFI}}$)	-2.52 (0.12)***
Loading × Error Method (CB)	-0.51 (0.10)***
Loading × Error Method (WB)	-1.51 (0.10)***
Model Fit (Fair) × Error Method (TKL_{CFI})	-0.59 (0.07)***
Model Fit (Poor) × Error Method (TKL_{CFI})	1.42 (0.08)***
Model Fit (Fair) × Error Method ($\text{TKL}_{\text{RMSEA/CFI}}$)	-30.79 (36.77)
Model Fit (Poor) × Error Method ($\text{TKL}_{\text{RMSEA/CFI}}$)	-31.44 (36.77)
Model Fit (Fair) × Error Method (CB)	0.44 (0.05)***
Model Fit (Poor) × Error Method (CB)	0.72 (0.07)***
Model Fit (Fair) × Error Method (WB)	0.23 (0.05)***
Model Fit (Poor) × Error Method (WB)	0.32 (0.07)***
AIC	181690.26
BIC	182226.97
Log Likelihood	-90796.13
Deviance	181592.26
Num. obs.	422205

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table B.5

Coefficient estimates and standard errors for the logistic regression model predicting the probability of producing a solution with qualitative fit index agreement.

B.1.6 RMSEA$_\Delta$ Analysis of Variance

Table B.6 gives a summary of the analysis of variance for the difference between RMSEA and RMSEA$_{\hat{\Omega}}$, RMSEA$_\Delta$ = RMSEA − RMSEA$_{\hat{\Omega}}$. Small values of RMSEA$_\Delta$ indicated that a model-error method led to similar values of RMSEA and RMSEA$_{\hat{\Omega}}$ whereas large values indicated that the two fit indices were quite different. All numeric predictors were scaled to have a mean of zero and a standard deviation of one to make the coefficient estimates more easily comparable.

Effect	F	df_1	df_2	MSE	p	$\hat{\eta}^2_G$
Factors	6,642.41	1	422,101	0.00	< .001	.015
Items/Factor	57,703.85	1	422,101	0.00	< .001	.120
Factor Correlation	2,898.51	1	422,101	0.00	< .001	.007
Loading	23,880.14	1	422,101	0.00	< .001	.054
Model Fit	87,799.46	2	422,101	0.00	< .001	.294
Error Method	46,404.47	4	422,101	0.00	< .001	.305
Factors × Items/Factor	10,863.30	1	422,101	0.00	< .001	.025
Factors × Factor Correlation	6,560.58	1	422,101	0.00	< .001	.015
Factors × Loading	6,870.58	1	422,101	0.00	< .001	.016
Factors × Model Fit	4,216.85	2	422,101	0.00	< .001	.020
Factors × Error Method	15,926.54	4	422,101	0.00	< .001	.131
Items/Factor × Factor Correlation	224.02	1	422,101	0.00	< .001	.001
Items/Factor × Loading	72.93	1	422,101	0.00	< .001	.000
Items/Factor × Model Fit	4,554.76	2	422,101	0.00	< .001	.021
Items/Factor × Error Method	7,779.53	4	422,101	0.00	< .001	.069
Factor Correlation × Loading	1,599.28	1	422,101	0.00	< .001	.004
Factor Correlation × Model Fit	1,063.70	2	422,101	0.00	< .001	.005
Factor Correlation × Error Method	1,608.15	4	422,101	0.00	< .001	.015
Loading × Model Fit	5,773.44	2	422,101	0.00	< .001	.027
Loading × Error Method	17,813.67	4	422,101	0.00	< .001	.144
Model Fit × Error Method	10,384.75	8	422,101	0.00	< .001	.164

Note. $\hat{\eta}^2_G$ denotes the estimate of the generalized eta-squared effect size.

Table B.6
Analysis of Variance for RMSEA$_\Delta$.

B.1.7 CFI_Δ Analysis of Variance

Table B.7 gives a summary of the analysis of variance for the difference between CFI and $\text{CFI}_{\hat{\Omega}}$, $\text{CFI}_\Delta = \text{CFI} - \text{CFI}_{\hat{\Omega}}$. Small values of CFI_Δ indicated that a model-error method led to similar values of CFI and $\text{CFI}_{\hat{\Omega}}$ whereas large values indicated that the two fit indices were quite different. All numeric predictors were scaled to have a mean of zero and a standard deviation of one to make the coefficient estimates more easily comparable.

Effect	F	df_1	df_2	MSE	p	$\hat{\eta}^2_G$
Factors	93,288.82	1	422,154	0.00	$< .001$	.181
Items/Factor	188.76	1	422,154	0.00	$< .001$	.000
Factor Correlation	1,217.80	1	422,154	0.00	$< .001$	.003
Loading	164,264.30	1	422,154	0.00	$< .001$	.280
Model Fit	79,055.01	2	422,154	0.00	$< .001$	.272
Error Method	63,019.04	4	422,154	0.00	$< .001$	.374
Factors × Items/Factor	3,419.08	1	422,154	0.00	$< .001$	.008
Factors × Factor Correlation	89.44	1	422,154	0.00	$< .001$	.000
Factors × Loading	32,507.98	1	422,154	0.00	$< .001$	.071
Factors × Model Fit	18,211.40	2	422,154	0.00	$< .001$	.079
Factors × Error Method	26,666.12	4	422,154	0.00	$< .001$	.202
Items/Factor × Factor Correlation	516.03	1	422,154	0.00	$< .001$	.001
Items/Factor × Loading	221.66	1	422,154	0.00	$< .001$	.001
Items/Factor × Model Fit	212.94	2	422,154	0.00	$< .001$	.001
Items/Factor × Error Method	7,280.09	4	422,154	0.00	$< .001$	.065
Factor Correlation × Loading	429.67	1	422,154	0.00	$< .001$	.001
Factor Correlation × Model Fit	514.03	2	422,154	0.00	$< .001$	.002
Factor Correlation × Error Method	442.50	4	422,154	0.00	$< .001$	.004
Loading × Model Fit	32,638.93	2	422,154	0.00	$< .001$	.134
Loading × Error Method	40,788.22	4	422,154	0.00	$< .001$	.279
Model Fit × Error Method	14,407.88	8	422,154	0.00	$< .001$	.214

Note. $\hat{\eta}^2_G$ denotes the estimate of the generalized eta-squared effect size.

Table B.7
Analysis of Variance for CFI_Δ.

B.2 Supplemental Figures

B.2.1 Percent of CB Solutions that were Indefinite

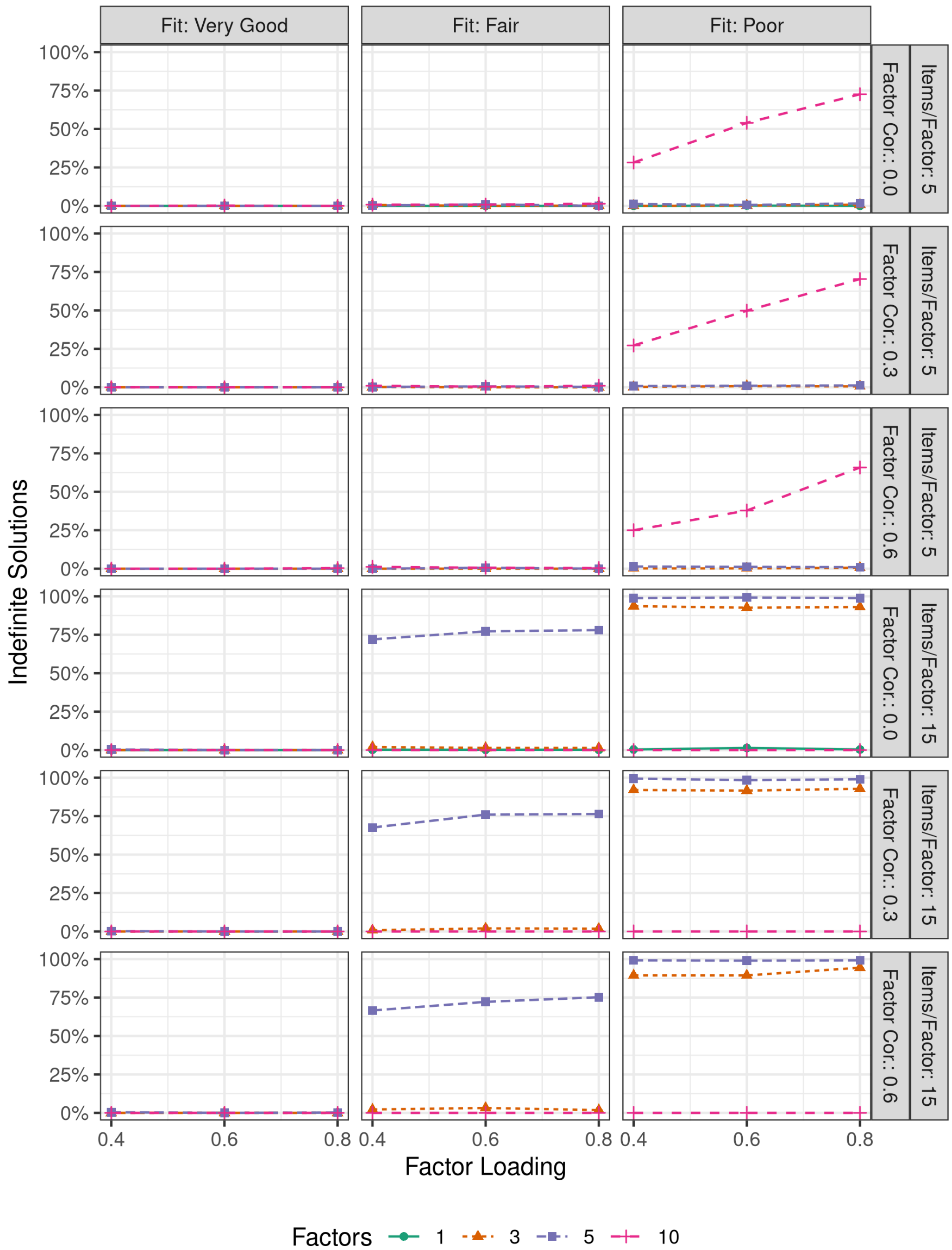


Figure B.1. The percent of Cudeck-Browne (CB) method solutions that were indefinite, conditioned on number of factors, factor loading, number of items per factor, factor correlation, and model fit.

B.2.2 Percent of TKL Solutions Where the Constraints on W were Violated

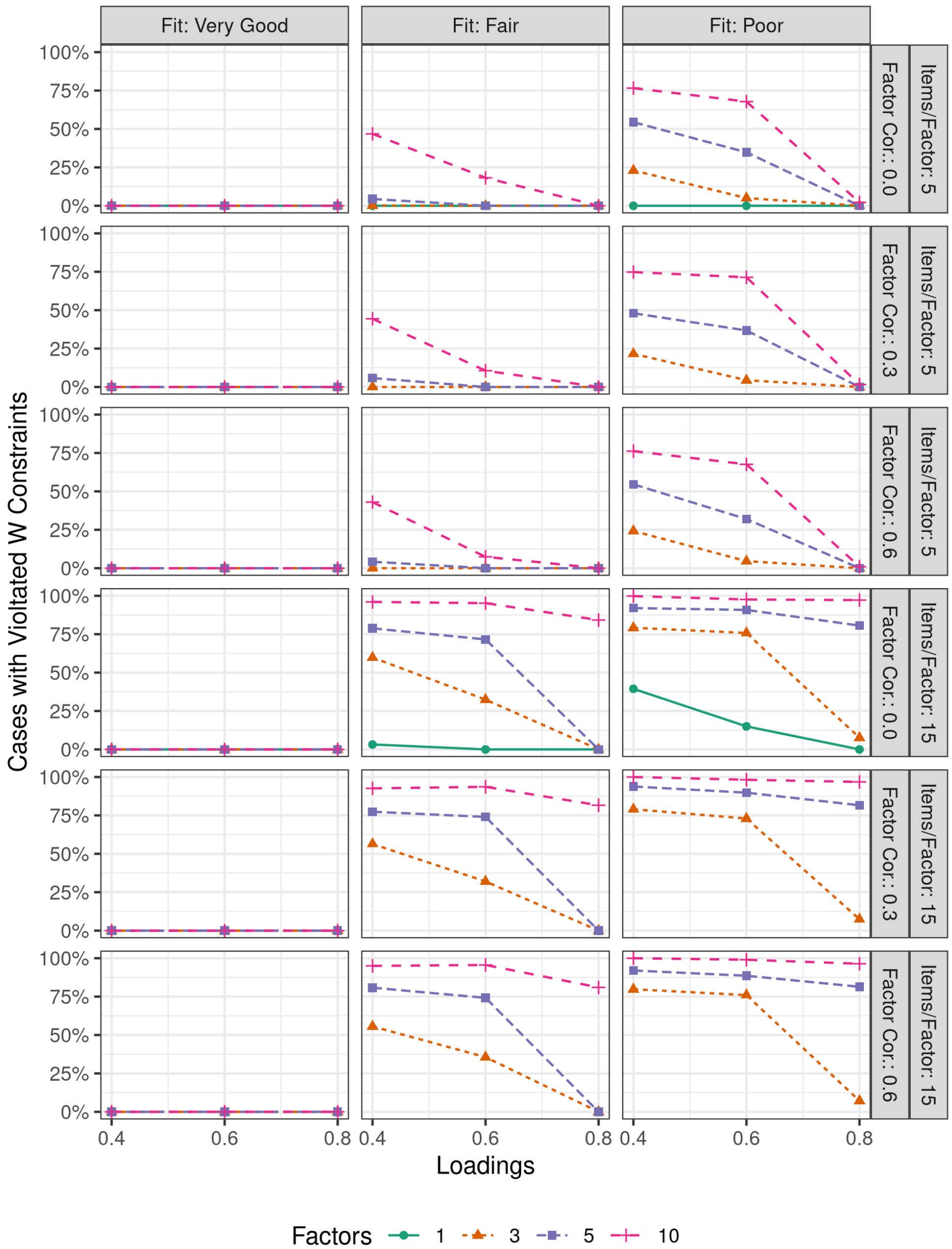


Figure B.2. The percent of cases where the constraints on **W** were violated when the $\text{TKL}_{\text{RMSEA}}$ model-error method was used, conditioned on number of factors, number of items per factor, factor loading, factor correlation, and model fit.

B.2.3 Values of ϵ and ν_e

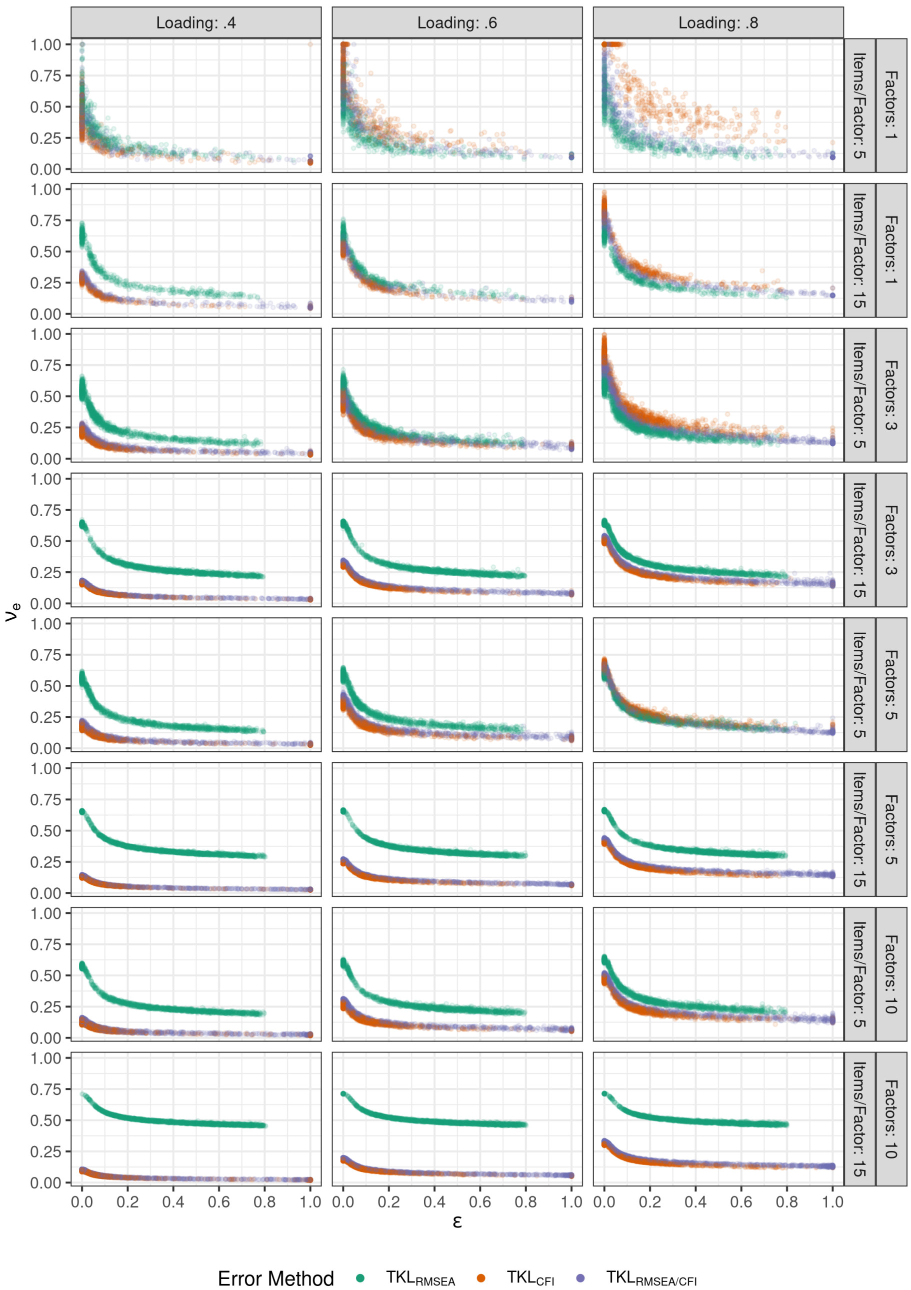


Figure B.3. Values of the TKL parameters (ϵ and ν_e) by model-error method, number of factors, number of items per factor, and factor loading strength when model fit was Poor. TKL = Tucker, Koopman, and Linn.

B.2.4 Observed RMSEA Values

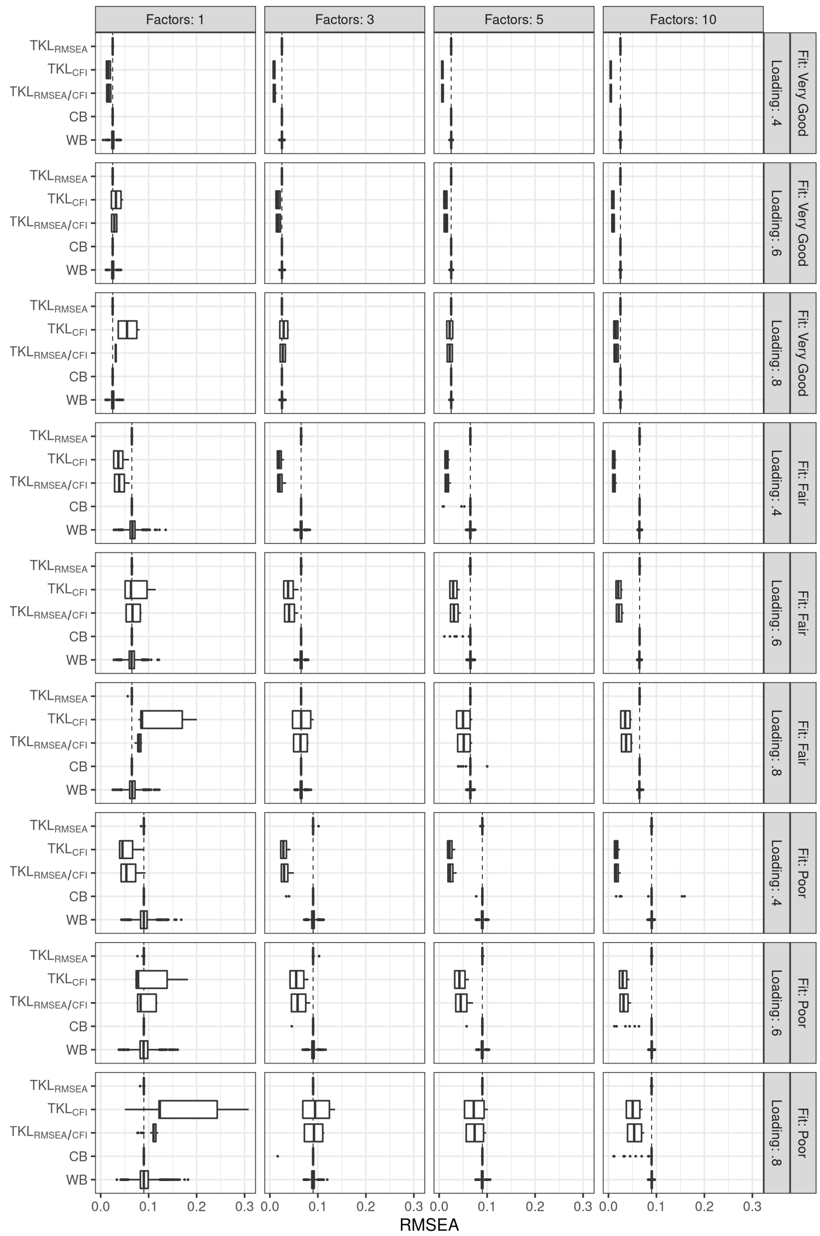


Figure B.4. Distributions of the RMSEA values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. The dashed lines indicate the target RMSEA value for each condition. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.5 Observed CFI Values

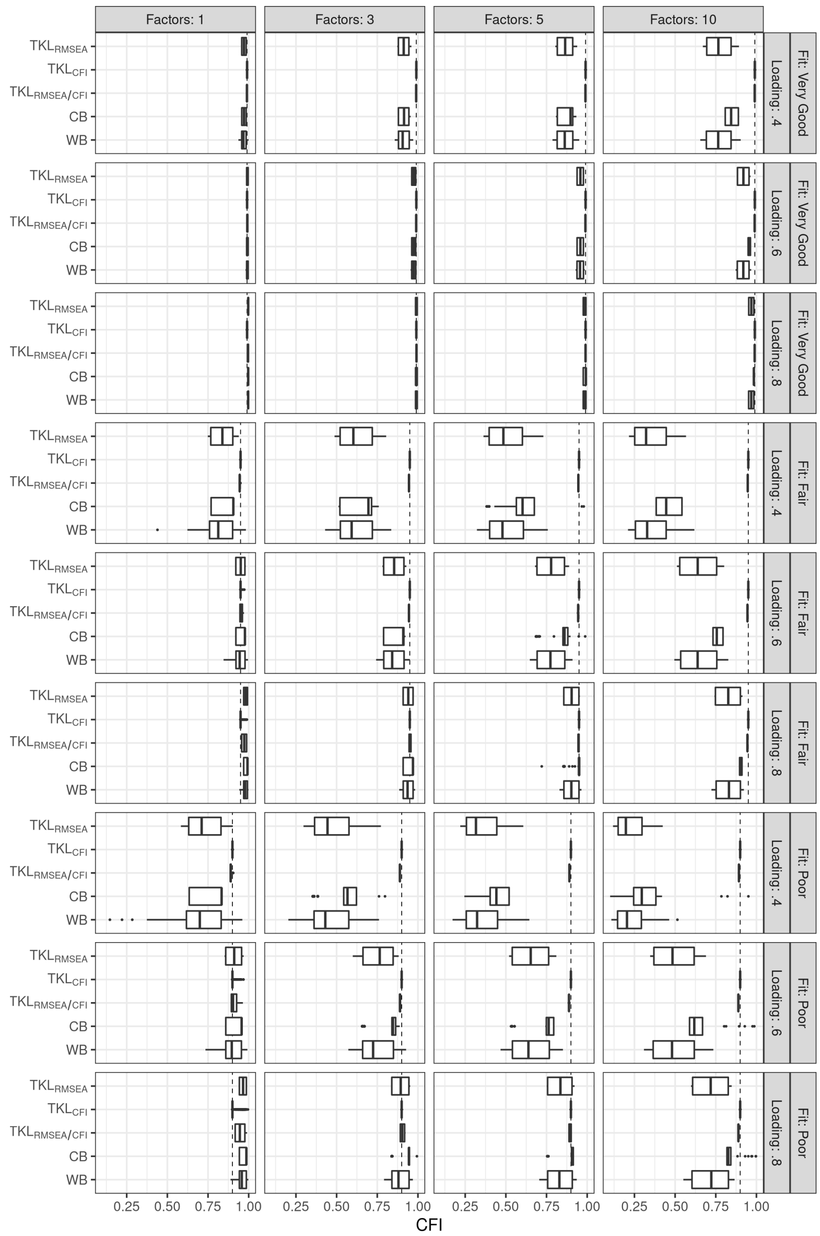


Figure B.5. Distributions of the CFI values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. The dashed lines indicate the target CFI value for each condition. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.6 Observed TLI Values

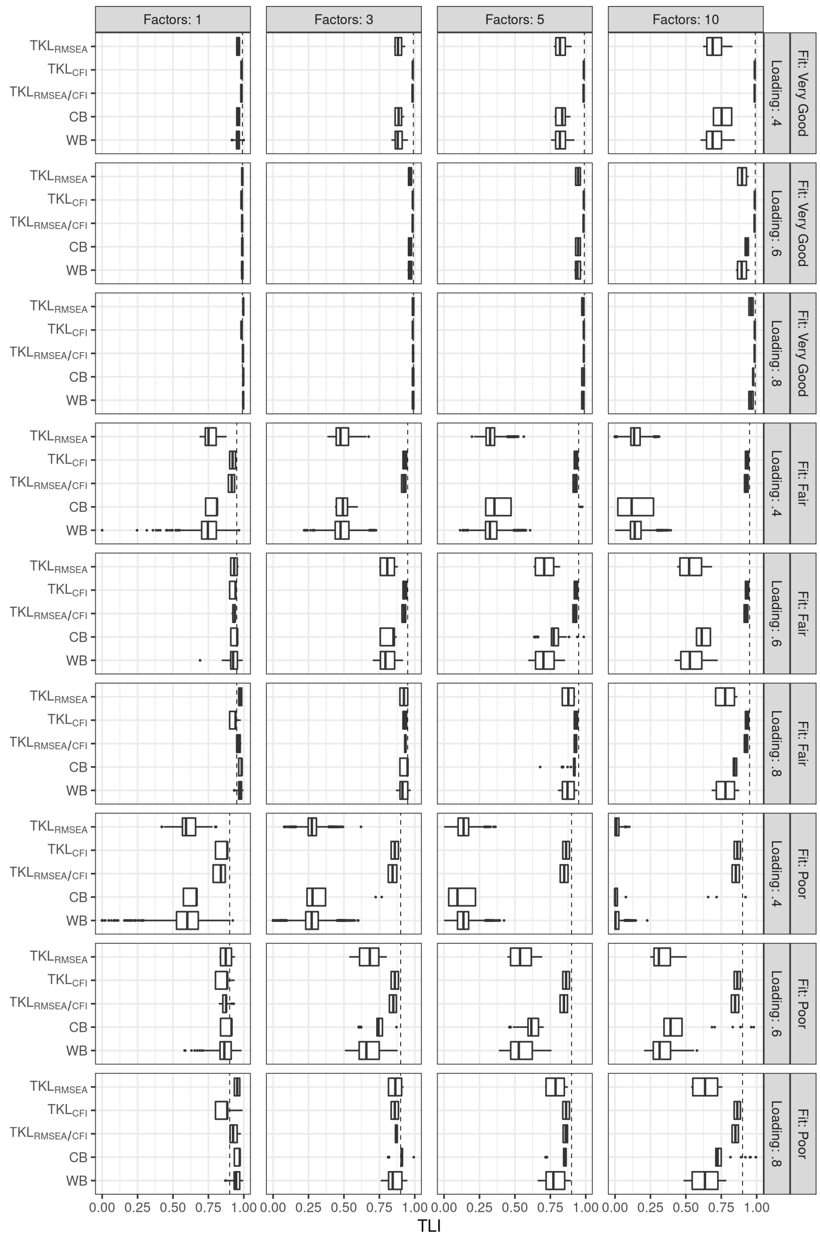


Figure B.6. Distributions of the TLI values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. The dashed lines indicate the threshold values of TLI that correspond to the targeted levels of model fit, according to Hu and Bentler (1999). TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.7 Observed CRMR Values

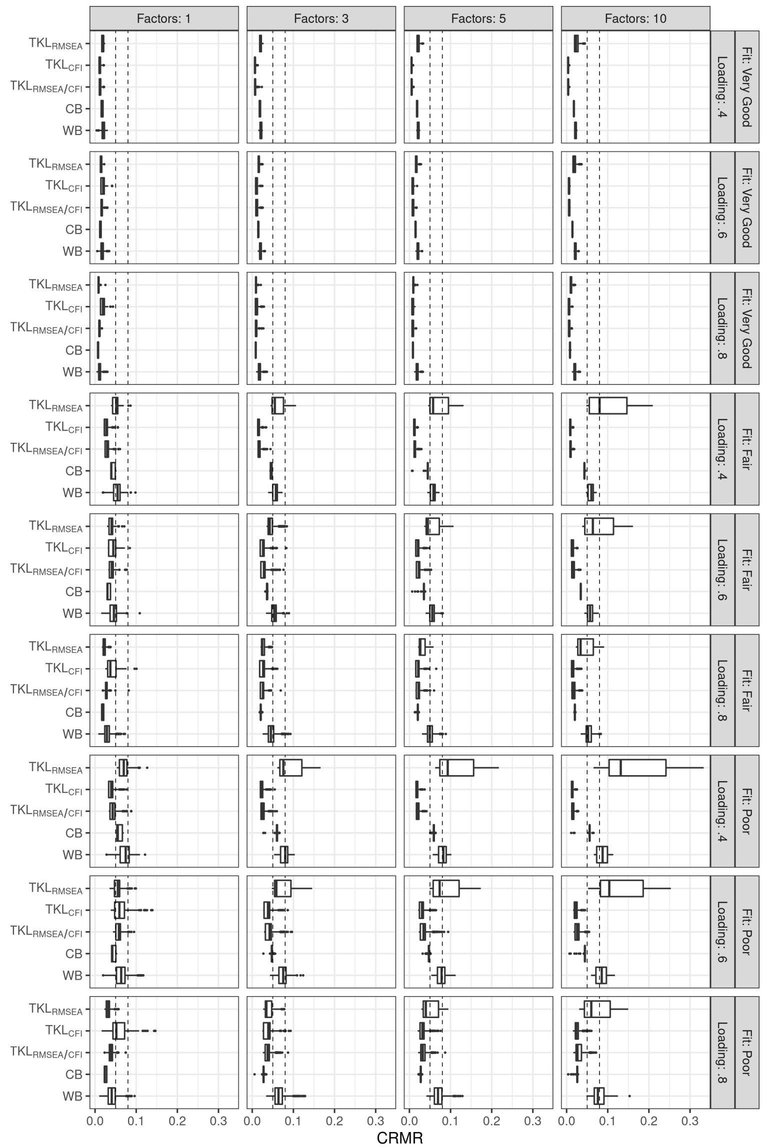


Figure B.7. Distributions of the CRMR values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. Recommendations for threshold values of SRMR/CRMR are not as fine-grained as for some other fit indices, but threshold values of 0.05 and 0.08 were proposed by Hu and Bentler (1999) as more and less conservative upper-bounds for acceptable SRMR/CRMR values. The dashed lines indicate these two threshold values. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.8 Fit Indices Based on $\hat{\Omega}$

B.2.8.1 $\text{RMSEA}_{\hat{\Omega}}$

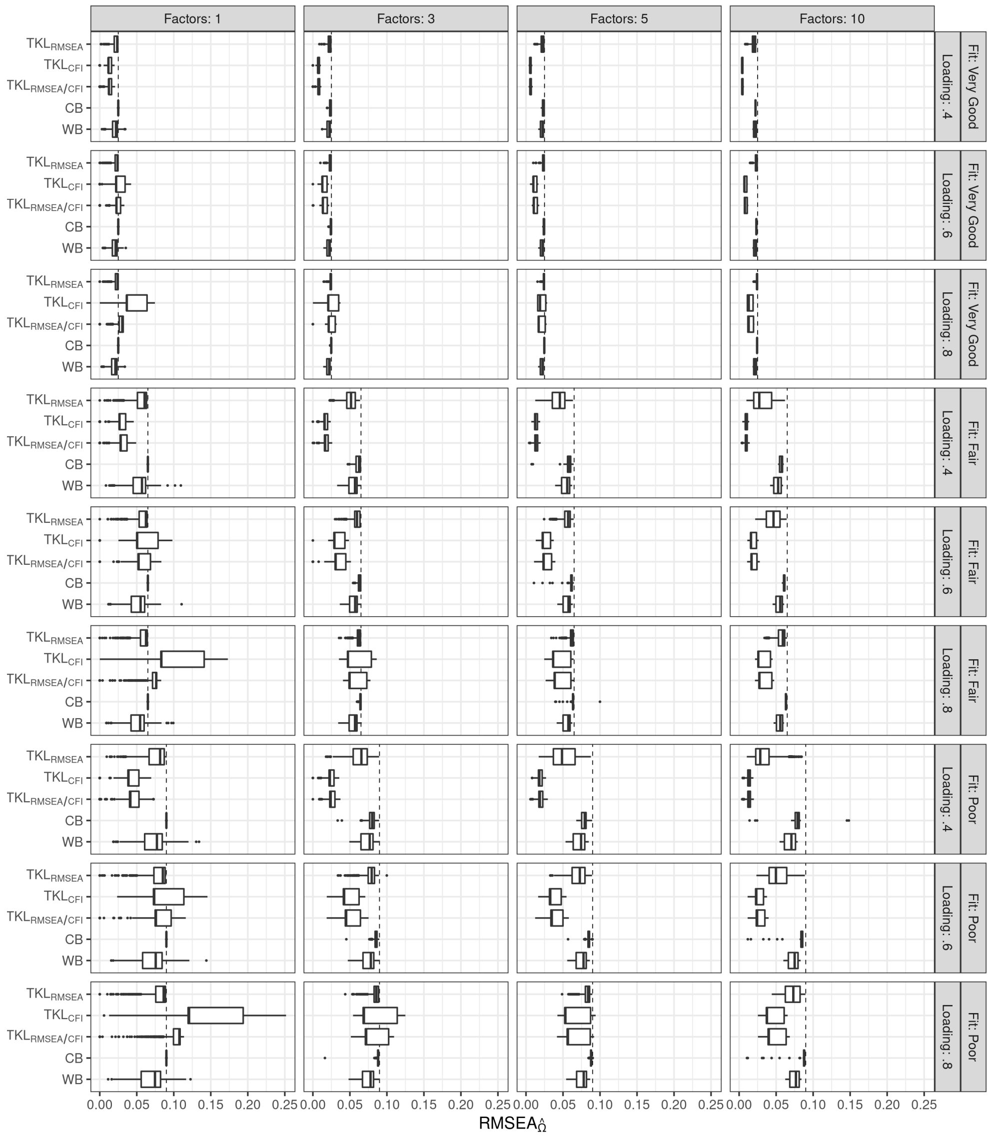


Figure B.8. Distributions of the $\text{RMSEA}_{\hat{\Omega}}$ values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. The dashed lines indicate the target RMSEA value for each condition. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.8.2 $\mathrm{CFI}_{\hat{\Omega}}$

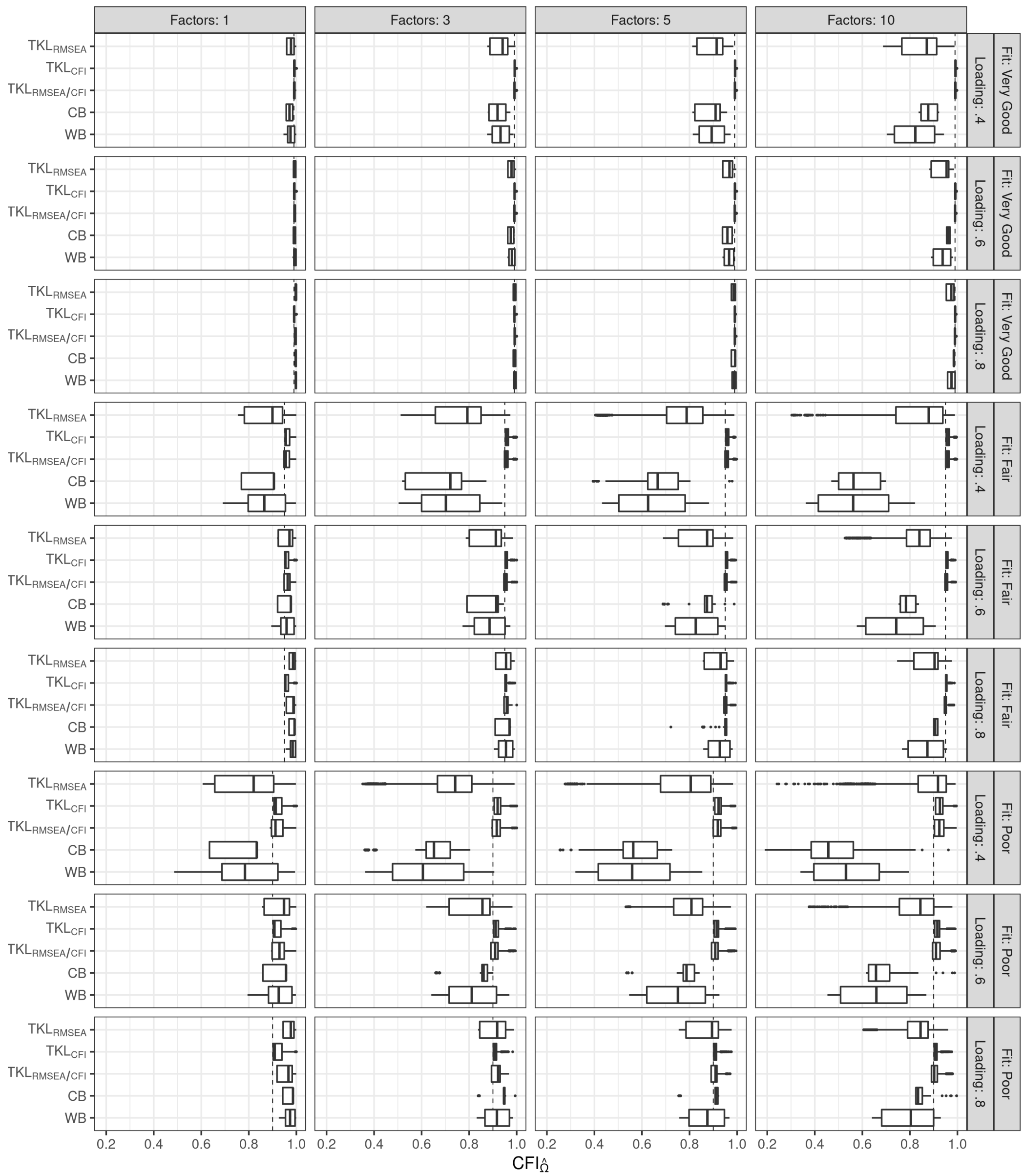


Figure B.9. Distributions of the $\mathrm{CFI}_{\hat{\Omega}}$ values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. The dashed lines indicate the target CFI value for each condition. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.8.3 $\text{TLI}_{\hat{\Omega}}$

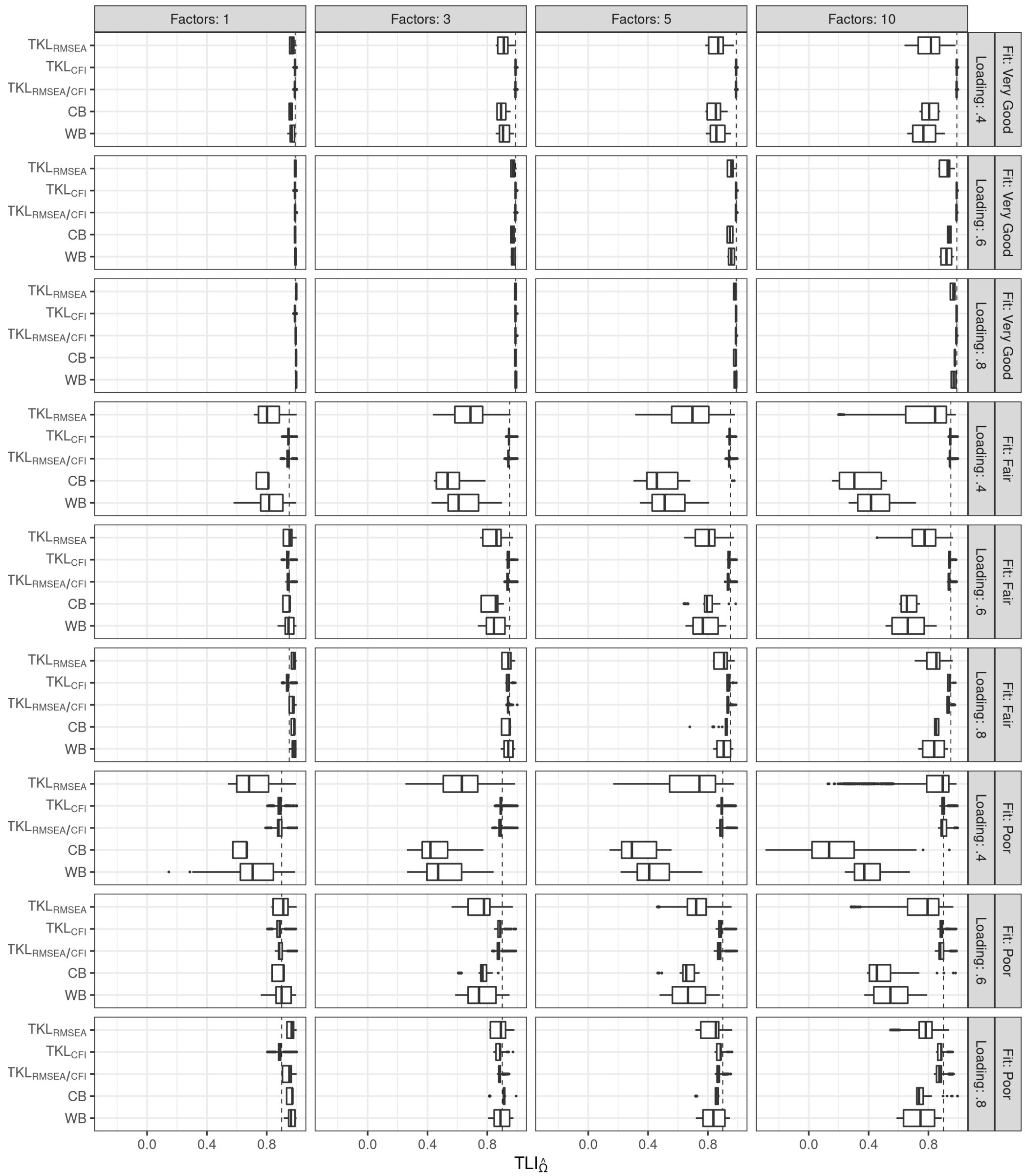


Figure B.10. Distributions of the $\text{TLI}_{\hat{\Omega}}$ values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. The dashed lines indicate the threshold values of TLI that correspond to the targeted levels of model fit, according to Hu and Bentler (1999). TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.8.4 $\mathrm{CRMR}_{\hat{\Omega}}$

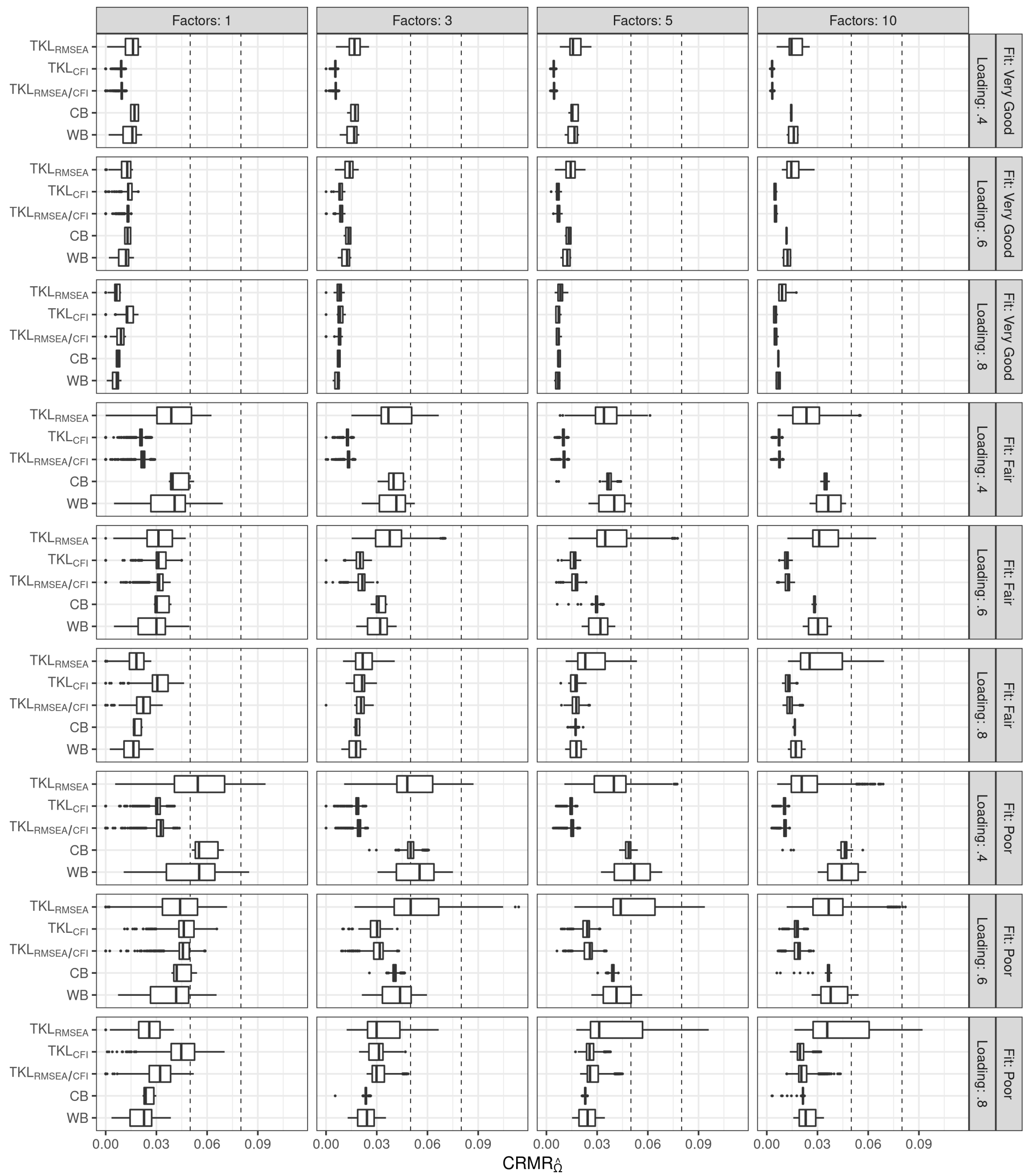


Figure B.11. Distributions of the $\mathrm{TLI}_{\hat{\Omega}}$ values for solutions produced by each of the model-error methods, conditioned on number of factors, model fit, and factor loading strength. Recommendations for threshold values of SRMR/CRMR are not as fine-grained as for some other fit indices, but threshold values of 0.05 and 0.08 were proposed by Hu and Bentler (1999) as more and less conservative upper-bounds for acceptable SRMR/CRMR values. The dashed lines indicate these two threshold values. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.9 Fit Index Agreement (D)

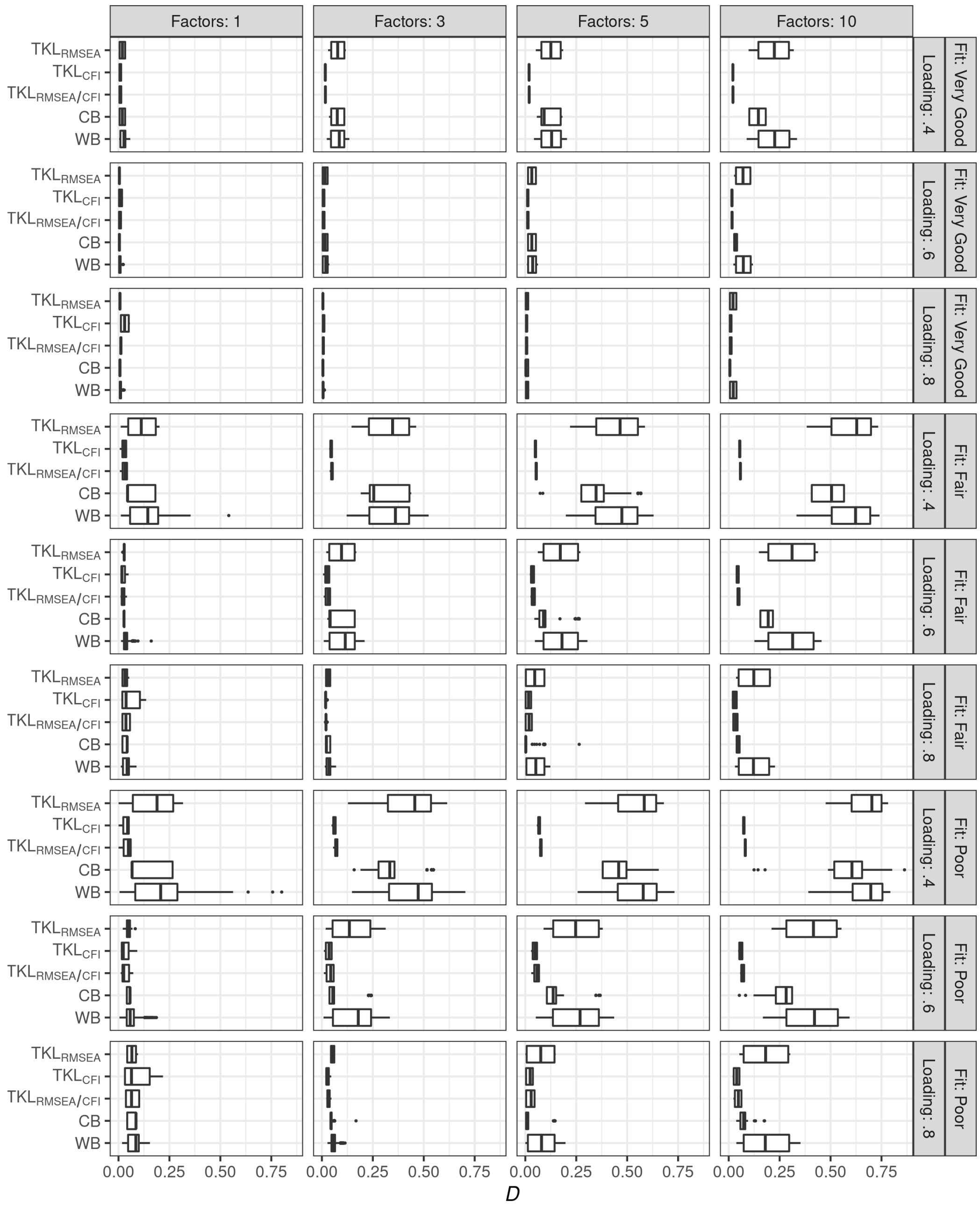


Figure B.12. The sum of the absolute differences between the observed and target RMSEA and CFI values (D), conditioned on number of factors, model fit, and factor loading strength. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.10 Conditional Distributions of CFI at Fixed RMSEA Values

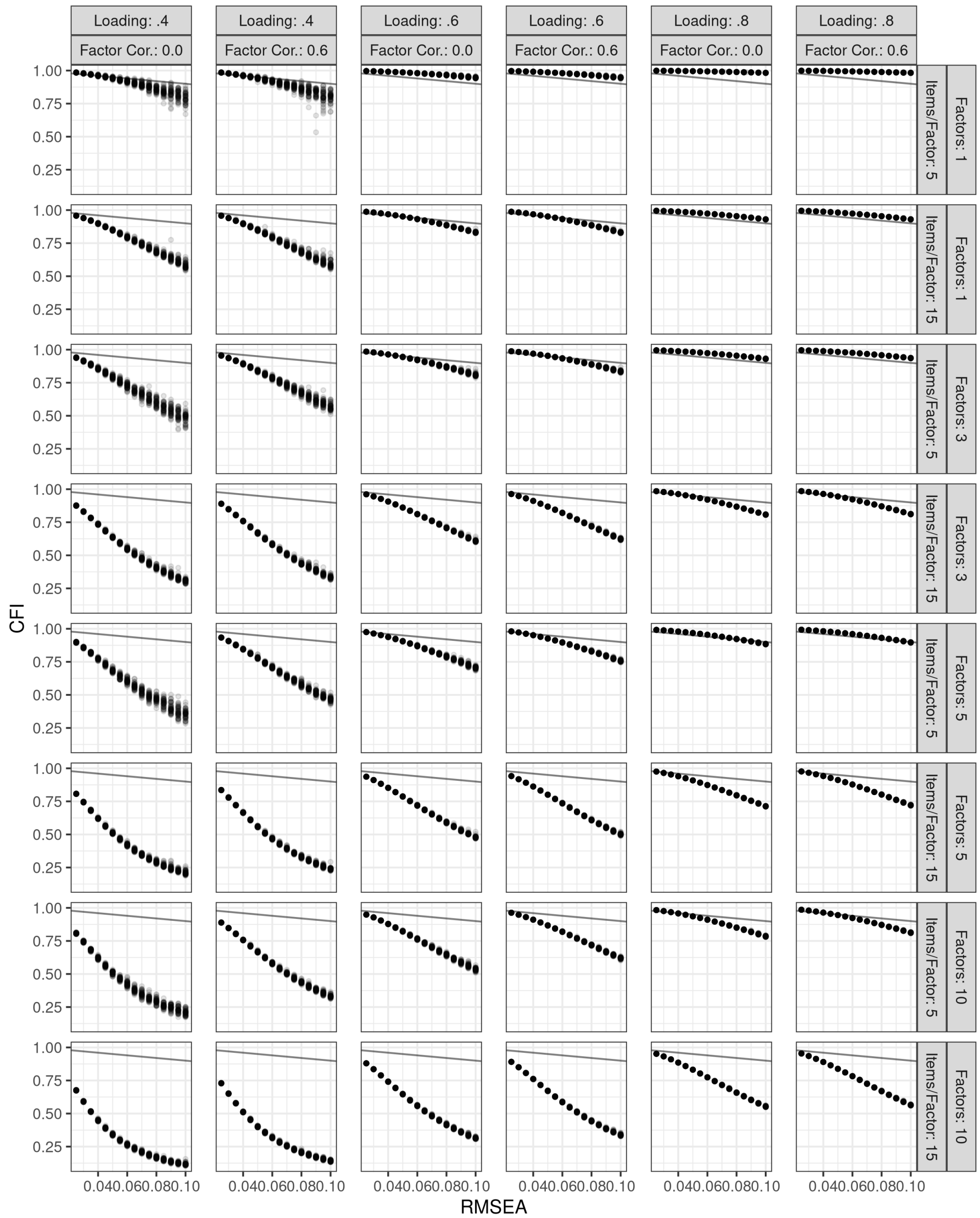


Figure B.13. Observed CFI and RMSEA values for solutions generated using the $\text{TKL}_{\text{RMSEA}}$ method. For each combination of number of factors, number of items per factor, factor loading strength, and factor correlation, 50 solutions were generated for each of 16 target RMSEA values equally-spaced between 0.025 and 0.100. The solid black line indicates where RMSEA and $1 - \text{CFI}$ were equal.

B.2.11 Conditional Distributions of RMSEA at Fixed CFI Values

B.2.12 Heatmap Representations of $\mathbf{\Sigma}$ and $\hat{\mathbf{\Omega}}$

B.2.12.1 Heatmap Representations of $\mathbf{\Sigma}$ and $\hat{\mathbf{\Omega}}$ for the Condition with Ten Orthogonal Factors, Five Items per Factor, Weak Factor Loadings, and Poor Model Fit

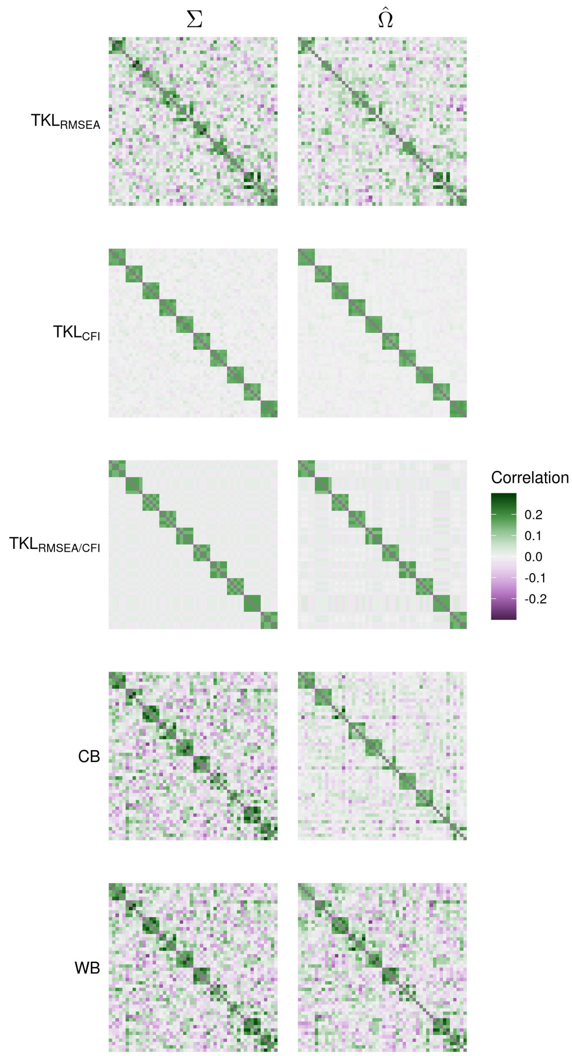


Figure B.14. Correlation matrices with model error ($\mathbf{\Sigma}$) and the implied correlation matrix obtained by fitting the population major-factor model to $\mathbf{\Sigma}$ ($\hat{\mathbf{\Omega}}$) for each model-error method. Results are shown for a set of solution matrices from the condition with ten orthogonal factors, five items per factor, weak factor loadings of 0.3, and Poor model fit. TKL = Tucker, Koopman, and Linn; CB = Cudeck and Browne; WB = Wu and Browne.

B.2.12.2 Heatmap Representations of $\boldsymbol{\Sigma}$ and $\hat{\boldsymbol{\Omega}}$ for the Condition with Ten Orthogonal Factors, Ten Items per Factor, Weak Factor Loadings, and Poor Model Fit

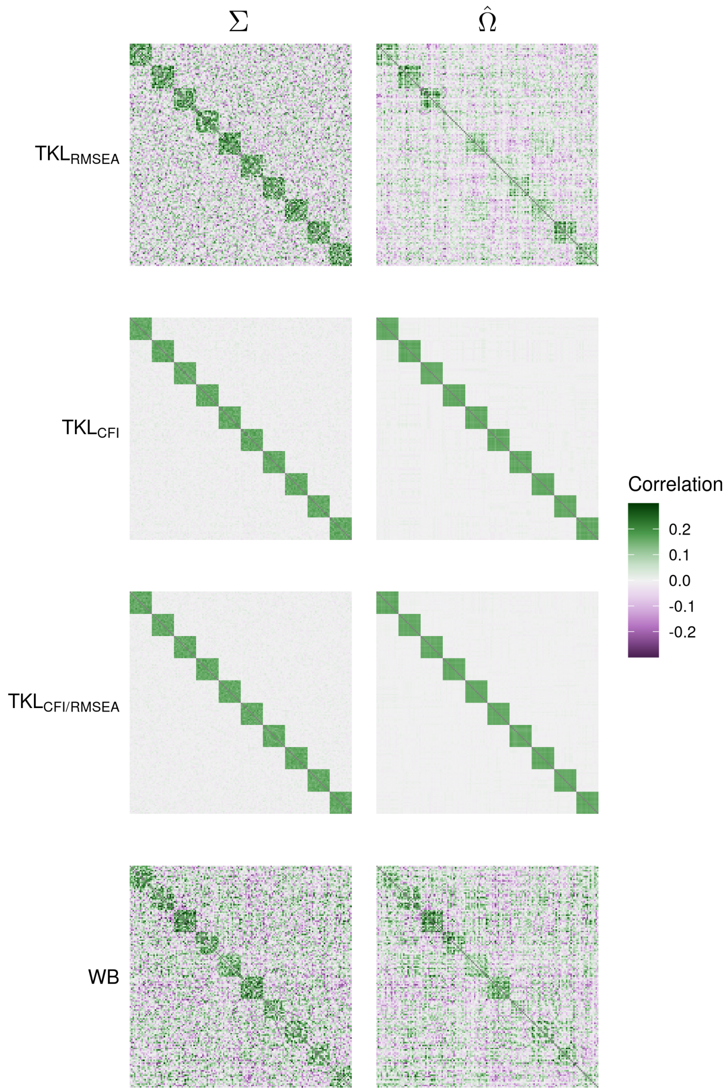


Figure B.15. Correlation matrices with model error ($\boldsymbol{\Sigma}$) and the implied correlation matrix obtained by fitting the population major-factor model to $\boldsymbol{\Sigma}$ ($\hat{\boldsymbol{\Omega}}$) for each model-error method. Results are shown for a set of solution matrices from the condition with ten orthogonal factors, ten items per factor, weak factor loadings of 0.4, and Poor model fit. Note that the Cudeck and Browne method is omitted because it was not used for conditions with ten factors and 15 items per factor. TKL = Tucker, Koopman, and Linn; WB = Wu and Browne.

Notes

24-1 Jul 21, 2022 at 2:46 PM, NW

This