2019-04-01

Exploring Fit for Nonlinear Structural Equation Models

Phillip Isaac Pfleger
Brigham Young University

Exploring Fit for Nonlinear Structural Equation Models

Phillip Isaac Pfleger

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Instructional Psychology and Technology

Brigham Young University

ABSTRACT

Exploring Fit for Nonlinear Structural Equation Models

Phillip Isaac Pfleger
Department of Instructional Psychology and Technology, BYU
Master of Science

Fit indices and fit measures commonly used to determine the accuracy and desirability of structural equation models are expected to be insensitive to nonlinearity in the data. This includes measures as ubiquitous as the CFI, TLI, RMSEA, SRMR, AIC, and BIC. Despite this, some software will report these measures when certain models are used. Consequently, some researchers may be led to use these fit measures without realizing the impropriety of the act. Alternative fit measures have been proposed, but these measures require further testing.

As part of this thesis, a large simulation study was carried out to investigate alternative fit measures and to confirm whether the traditional measures are practically blind to nonlinearity in the data. The results of the simulation provide conclusive evidence that fit statistics and fit indices based on the chi-square (χ^2) distribution or the residual covariance matrix are entirely insensitive to nonlinearity. The posterior predictive p-value was also insensitive to nonlinearity. Only fit measures based on the structural residuals (i.e., HFI and R^2) showed any sensitivity to nonlinearity. Of these, the R^2 was the only reliable measure of nonlinear model misspecification.

This thesis shows that an effective strategy for determining whether a nonlinear model is preferable to a linear one involves using the R^2 to compare models that have been fit to the same data. An R^2 that is much larger for the nonlinear model than the linear model suggests that the linear model may be less desirable than the nonlinear model.

The proposed method is intended to be supplementary to substantive theory. It is argued that any dependence on fit indices or fit statistics that places these measures on a higher pedestal than substantive theory will invariably lead to blindness on the part of the researcher. In other words, unwavering adherence to goodness-of-fit measures limits the researcher's vision to what the measures themselves can detect.

Keywords: structural equation modeling, goodness-of-fit, nonlinear statistical models, Bayesian analysis

ACKNOWLEDGMENTS

Thank you to my dear, sweet wife, Brenna, who put up with hours and hours of talk about my topic. Thank you to all of my family. Thank you to my chair, Ross Larsen, his questions were the foundation of my study. Thank you to Richard Sudweeks and Randy Davies who served on my committee. Thank you to the wonderful individuals who helped answer my questions regarding certain coding problems, Levi Morrison, Rebecca Buchner, and Ed Merkle.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

DESCRIPTION OF THESIS STRUCTURE

Exploring Fit for Nonlinear Structural Equation Models is written in a journal-ready format with a structured annotated bibliography. The heart of the thesis is represented in a journal-ready article, while an annotated bibliography ensures the additional requirements of the traditional thesis are met. The complete article will be submitted to journals that deal primarily with measurement in the social sciences, such as *Structural Equation Modeling* and *Psychometrika*.

The formal outline of this document is intended to meet the requirements of the university. Prior to this description is the cover page, abstract, acknowledgments, the tables of contents, and the lists of tables and figures. The article is presented immediately following this section. After the article, the reader will find two appendices. Appendix A is the structured annotated bibliography, and Appendix B is the replication code. Appendix B represents several different files that are required to run the simulation.

Exploring Fit for Nonlinear Structural Equation Models

Phillip Isaac Pfleger

Brigham Young University

Introduction

Fit indices and fit statistics are often used in research to determine if a mathematical model adequately represents the data. A fit index differs from a fit statistic in that a fit statistic has a known distribution and can be used to make probability statements (Maydeu-Olivares, 2013). Both fit indices and fit statistics will be referred to generically as fit measures throughout this article.

Researchers use fit measures to identify the best model for understanding their data and testing their hypotheses. For example, a researcher who would like to make the case that regression model 1 fits or predicts the data well, might examine a p-value-based on an F test. Regression model 1 is a simple case of two continuous independent variables (X_{1i}, X_{2i}) regressed on one continuous dependent variable (Y_i),

$$Y_i = b_0 + b_1X_{1i} + b_2X_{2i} + e_i, \tag{1}$$

where b_0 is the intercept, b_1 and b_2 are the slopes for X_1 and X_2 respectively, and e is the error term for the ith subject. Our knowledge of the F distribution allows us to obtain a p-value for the given F statistic. Larger values of the F statistic result in lower p-values (Kutner, Nachtsheim, & Neter, 2004). Often a cutoff criterion of $p < 0.05$ is used to determine good fit. In other words, if the model had a p-value less than 0.05 the researcher might conclude that the model fits the data well. This kind of fit is called absolute fit. Researchers, however, may also be interested in relative fit.

One example of a relative fit index, which many readers will be familiar with, is the Bayesian Information Criterion (BIC). The BIC is one of several fit measures that can be used to determine whether one model is a better fit of the data than another. A researcher interested in determining if model 1 fits the data better than a model including a nonlinear term, such as model 2, could compare the models using the BIC. Model 2 differs from model 1 by the inclusion of an interaction term (X_1X_2),

$$Y_i = b_0 + b_1X_1 + b_2X_2 + b_3X_1X_2 + e_i. \tag{2}$$

The interaction term is created by multiplying X_1 and X_2, which is why it is written as X_1X_2; it also has its own regression coefficient, b_3. To determine which model is better fitting, the researcher may examine the BIC values, declaring the model with the smallest BIC the best fitting model.

The process is similar in structural equation modeling (SEM). In SEM there are many fit measures used to test both model fit (absolute fit) and to help with model selection (relative fit). Some relative measures include the chi-square (χ^2), comparative fit index (CFI), Tucker-Lewis Index (TLI), root mean square error of approximation (RMSEA), and standardized root mean square residual (SRMR). BIC and Akaike's Information Criterion (AIC) (Wang & Wang, 2012) are relative measures. Even the maximum likelihood can be compared between models in the form of a likelihood ratio test (LRT), or for non-nested models a Vuong test (Merkle, You, Schneider, Bae & Merkle, 2018; Vuong, 1989), thus it can be considered a relative measure. Each of these fit measures has different strengths, uses, and limitations. It is common to see a number of these absolute fit measures reported together, along with predetermined cutoffs, such as the ones set by Hu and Bentler (1999).

Fit Measures

In structural equation modeling (SEM) most fit measures are based off the χ^2 distribution,

$$\chi^2 = F_{ML}(N - 1), \tag{3}$$

where N is the sample size and F_{ML} is the maximum likelihood discrepancy function used to generate the model.

The formula for F_{ML} is

$$F_{ML} = ln|\hat{\Sigma}| + tr(S\hat{\Sigma}^{-1}) - ln|S| - (p + q), \tag{4}$$

where F_{ML} is equal to the natural log of the determinant of the model implied covariance matrix ($\hat{\Sigma}$) plus the trace of the observed covariance matrix (S) times the inverse of $\hat{\Sigma}$ minus the natural log of the determinant of S minus the total number of independent and dependent indicators (p and q respectively). If the model fit the data perfectly than all the terms would cancel out and the likelihood would equal zero (Wang & Wang, 2012).

In practice the χ^2 is used to calculate a p-value for model fit, a model is often considered good fitting when $p > 0.05$. The χ^2 statistic has several known limitations; for example, the χ^2 statistic is less reliable in situations with very large sample sizes, very small sample sizes, or with skewed distributions (Wang & Wang, 2012). These limitations gave rise to a plethora of fit measures that attempt to make up for the weaknesses of the χ^2. Among these is the comparative fit index (CFI),

$$CFI = \frac{d_{null} - d_{specified}}{d_{null}}, \tag{5}$$

where $d = \chi^2 - df$, specified refers to the specified model, and null refers to the worst possible model, and *df* is the degrees of freedom for appropriate model (Bentler, 1990; Wang & Wang, 2012). The CFI is used to test absolute fit or goodness of fit. The CFI ranges from 0 to 1, with a value of 1 representing perfect fit. Hu and Bentler (1999) proposed the models with a CFI value above 0.95 could be considered good fitting models. One major weakness of the CFI is that it will have a low value if the correlations between items is low.

Another index is the Tucker-Lewis Index (TLI),

$$TLI = \frac{\chi^2_{null} df_{null} - \chi^2_{specified} df_{specified}}{\chi^2_{null} df_{null} - 1}, \tag{6}$$

which is often reported in tandem with the CFI. The TLI shares a cutoff value of 0.95 with the CFI (Hu & Bentler, 1999). The TLI is also sensitive to the correlations among the items. The TLI may have values above 1, but if this occurs the TLI is set to 1.

The root mean square error of approximation (RMSEA),

$$RMSEA = \sqrt{\frac{\left(\frac{\chi^2_s}{df_s}\right) - 1}{N}}, \tag{7}$$

has, a known distribution and therefore allows for the calculation of a *p*-value and a confidence interval. The RMSEA ranges between 0 and 1, but unlike the CFI and TLI a 0 is considered perfect fit. Hu and Bentler (1999) declare a model with RMSEA < 0.06 as a good fit of the data. Often, 90% confidence intervals are reported for the RMSEA and a *p*-value is included.

While the standardized root mean square residual (SRMR) is not calculated based on the χ^2 distribution, it is calculated based on the sample and model implied covariance matrices,

$$SRMR = \sqrt{\frac{\left(\sum_j \sum_k r^2_{jk}\right)}{e}}, \tag{8}$$

where

$$r_{jk} = \frac{S_{jk}}{S^2_{jj} S^2_{kk}} - \frac{\Sigma_{jk}}{\Sigma^2_{jj} \Sigma^2_{kk}}, \tag{9}$$

S_{jk} is the element in the j^{th} row and k^{th} column of the sample covariance matrix, and Σ_{jk} is the element in the j^{th} row and k^{th} column of the model implied covariance matrix (Wang & Wang, 2012). The elements S^2_{jj}, S^2_{kk}, Σ^2_{jj}, and Σ^2_{kk} represent the variances along the diagonal of the respective covariance matrices. The SRMR is similar in interpretation to the RMSEA. With values ranging from 0 to 1, a SRMR < 0.08 is considered good fit.

Table 1 displays the criteria commonly used to determine the quality of a model based on these measures. It is rare to see research employing SEM that does not include a threshold to aid in the analysis of goodness of fit. While Hu and Bentler's (1999) values are commonly used, there are others to be used in the same and different situations (Chen, 2007; Wang & Wang, 2012). Usually, a researcher will decide beforehand which fit measures and cutoffs to base their decisions on.

Table 1

Hu and Bentler (1999) Cutoffs for Fit Measures

Measure	Cutoff
CFI	> 0.95
TLI	> 0.95
RMSEA	< 0.06
SRMR	< 0.08

Other fit measures common in SEM include AIC and the BIC,

$$BIC = -2\ln(L) + \ln(N)\,m, \tag{10}$$

$$AIC = -2\ln(L) + 2m, \tag{11}$$

where, *L* is the maximum likelihood, *N* is the sample size, and *m* is the number of free parameters in the model (Kutner, et al., 2004). The AIC and BIC are both relative fit measures and usually obtain very similar results, though the BIC gives a bigger penalty for complexity and thus favors more parsimonious models (Wang & Wang, 2012).

Each of the commonly used fit measures is based on either the χ^2 distribution or the residual covariance matrix. Those that are based on the χ^2 distribution are mathematically blind to nonlinear terms when only the covariances are used to estimate the model (Mooijaart & Satorra, 2009). Considering that the covariance is a measure of the linear relationship between two variables, it seems doubtful that those that are based off the observed and model implied covariance matrices will be any less blind than the χ^2-based indices.

The homoscedastic fit index (HFI) was created to account for the limitations of all the χ^2 and covariance-based measures. The HFI is a refinement of the h_{het} index (Klein, Gerhard, Buchner, Diestel, & Schermelleh-Engel, 2016),

$$h_{het} = \sqrt{\frac{n}{24}}\left(\frac{n^{-1}\sum e_i^4}{(n^{-1}\sum e_i^2)^2} - 3\right) \tag{12}$$

where, n is the sample size and $\sum e_i$ is the sum of all the residuals from the structural part of the model. The HFI puts the h_{het} on a more interpretable scale (Gerhard, Buchner, Klein, & Schermelleh-Engel, 2017).

$$HFI = \begin{cases} 1 & \text{if } h_{het} \leq 0 \\ \dfrac{1}{0.032h_{het} + 1} & \text{if } h_{het} > 0 \end{cases} \tag{13}$$

The HFI tries to identify omitted nonlinear terms by investigating the residuals for heteroscedasticity. Values of the HFI range from 0 to 1 and a value of HFI > .95 suggests that a model has not omitted any nonlinear terms. The cutoff for HFI should be decreased by .01 for every additional nonlinear term in the model beyond the first. Klein et al. (2016) found that when the model is estimated using the latent moderated structural equation modeling (LMS) approach in the software Mplus (Muthén, & Muthén, 2011) that the HFI performed reasonably well (Gerhard et al., 2017).

Another measure, more well known than the HFI, is the coefficient of determination (R^2). The R^2 in SEM is like the HFI in the way it is calculated from the structural part of the model. R^2 is a measure of the variance explained by a regression model,

$$R^2 = \frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{n}(Y_i - \overline{Y}_i)^2}. \tag{14}$$

A model with a higher R^2 is often considered a better fitting model. However, R^2 is a greedy index, it will always increase when another predictor is added to the model. Consequently, the change in R^2 between models is considered. If the change is large, then the more complex model is often chosen.

Another alternative is to test model fit for the linear model in a two-step LRTprocedure outlined in Maslowsky, Jager, and Hemken (2015). In the first step a linear model is fit to the data. If the chosen fit measures meet the thresholds for good fit, then a model including the nonlinear term is selected. The two models are then compared using a LRT (Klein & Moosbrugger, 2000). This approach is dependent upon the accuracy of the fit decision in the first step (Gerhard et al., 2017).

Finally, Bayesian methods provide several alternative fit measures (Spiegelhalter, Best, Carlin, Van Der Linde, 2002). Most of these are difficult to calculate and require integrating out latent variables, which is not done by most software (Merkle, Furr, & Rabe-Hesketh, 2018). The posterior predictive checks are one option that is simple to implement. Posterior predictive checks are performed by simulating data based on the model. If the model fits the data, then the new data will be similar to the observed data (Kruschke, 2014). Any systematic differences between the data are evidence that the model doesn’t fit the observed data. While this is often done informally, the process can be formalized in the creation of a posterior predictive *p*-value (PPP) (Gelman, Stern, Carlin, Dunson, Vehtari, & Rubin, 2013). A PPP is the probability that data simulated from the posterior distribution will have a value on some discrepancy function less than the observed data,

$$PPP = P\left(f(Y, \theta) < f(\hat{Y}, \theta)\right), \tag{15}$$

where $\hat{Y}$ is data that are simulated according to the posterior distribution of the Y (Asparouhov & Muthén, 2017). The PPP most commonly used (e.g., in Mplus) is based on the χ^2 discrepancy function (Asparouhov & Muthén, 2017). This function is χ^2 distributed, and as such may not be able to detect omitted nonlinear terms. The PPP does not have to be based on the χ^2 discrepancy function and can be based on any discrepancy function (e.g., the mean square error used in ordinary least squares regression) (Asparouhov & Muthén, 2017).

Estimation Approaches

Research on estimating nonlinear models has far outpaced research on determining fit for those models (Gerhard et al., 2017), but no best method has been agreed upon. An ideal fit measure will perform well regardless of the estimation method used. Estimation approaches can be categorized as product indicator (PI) approaches, distributional approaches, method of moments approaches, or Bayesian approaches.

The PI approaches were the first type of nonlinear SEM approaches (Jöreskog & Yang, 1996; Kelava & Nagengast, 2012; Kenny & Judd, 1984; Little, Bovaird, & Widman, 2006; Marsh, Wen, & Hau, 2004; Moulder & Algina, 2002; Ping, 1995). PI approaches model the latent nonlinear variables by multiplying various combinations of the indicators and using these indicators as the observed variables for the nonlinear terms. χ^2-based fit statistics are included in the output of most statistical software when any PI approach model is estimated (Mooijaart & Satorra, 2009).

Some authors are quick to point out that none of the PI approaches consider the inherent non-normality that occurs even when two normally distributed variables are multiplied (Dimitruk, Schermelleh-Engel, Kelava, & Moosbrugger, 2007; Moosbrugger, Schermelleh-Engel, & Klein, 1997). These authors support distributional approaches such as the quasi-maximum likelihood (QML) approach (Klein & Muthén, 2007) and the latent moderated structural equations models (LMS) approach (Klein & Moosbrugger, 2000; Moosbrugger, et al., 2009). These models utilize an EM algorithm to directly model the distribution of the latent interaction from the data and do not have to rely on product indicants for the interaction (Klein & Moosbrugger, 2000; Klein & Muthén, 2007; Moosbrugger, et al., 1997). The QML and LMS are very similar, though the QML is more efficient and can handle multiple nonlinear terms at once (Klein & Muthén, 2007). One downside of the QML is that it is sensitive to starting values and has convergence problems (Umbach, Naumann, Brandt, & Kelava, 2017).

Method of moment approaches incorporate higher order moments in addition to the covariance matrix (Wall & Amemiya, 2003). They come with their own unique set of fit measures to test goodness of fit and generally perform well (Mooijaart & Bentler, 2010). While these methods show great promise in terms of accurately estimating parameters (Wall & Amemiya, 2003), they proved difficult to incorporate into the present simulation study.

Bayesian approaches have some advantages over the other (frequentist) approaches. For example, Bayesian methods do not require normality or large sample sizes, and they provide more information due to their focus on distributions instead of point estimates (Lee, Song, & Tang, 2007). For a summary of the advantages of Bayesian approaches over other frequentist methods read Muthén and Asparouhov (2012) and Bayarri and Berger (2004). Bayesian analysis is based on Bayes' Theorem,

$$P(B|A) = P(A|B)P(B)P(A), \tag{16}$$

where A is the data and B is the parameter values. This is applied in analysis in the form of

$$posterior \propto likelihood * prior \tag{17}$$

(Muthén & Asparouhov, 2012). The idea is to include information that is already known into the analysis in the form of a prior distribution. The Bayesian approach also does not rely on product indicants (Lee, et al, 2007). The Bayesian method performs well compared to the popular LMS and unconstrained approaches (Harring, Weiss, & Hsu, 2012; Kelava & Nagengast, 2012).

When PI approaches are used, the traditional fit measures are often automatically contained in the output. When LMS or QML are used, there are no fit measures contained in the output. When Bayesian models are used, if fit measures are contained in the output, they are often incorrect due to not integrating out the latent variables. This suggests four important questions for study:

1. How often will decisions based on traditional cutoffs be incorrect for PI models?
2. Are χ^2-based statistics blind to the extent that they are not useful at all, or is there some recognizable pattern among them that could be helpful?
3. How well do alternative approaches, such as the HFI, PPP, LRT, and R^2, perform in the PI approach?
4. Will the performance of these measures generalize to estimation methods other than the PI approach?

Method

Software

This study was performed using the statistical software R (R Core Team, 2013) in the IDE RStudio (RStudio Team, 2015), with graphs created using *ggplot2* (Wickham, 2016). Data were simulated using the *nlsem* (Umbach, et al., 2017) package, which was also used to estimate the QML models. The package *lavaan* (Rosseel, 2012) was used to estimate the PI approaches. Analysis of the nonlinear Bayesian models was done using *runjags* (Denwood, 2016) to interface with the software JAGS (Plummer, 2003). The package *blavaan* (Merkle & Rosseel, 2015) was used to estimate the linear Bayesian model, which also does its calculations in JAGS. The package *blavaan* integrates out the latent variables in the fit indices for this model as required. However, *blavaan* is not currently able to estimate the nonlinear models, though it is on the radar of the developer (E. Merkle, personal communication, May 29, 2018).

Simulation Design

All variables, latent and observed, were simulated as continuous variables with means of 0 and variances of 1. As shown in Table 2, this study is a 4 x 3 x 3 x 3 x 3 x 3 design. Data were simulated with varying conditions of strength of the nonlinear term (ω = 0.02, 0.13, 0.26, 0.4); sample size (n = 100, 500, 1000); and reliability of indicators (r_{xy} = 0.45, 0.65, 0.85). Three conditions of nonlinearity were used to simulate the data. In one condition there were no nonlinear terms, in another there was one interaction, and in the final there were three nonlinear terms. A linear, interaction, and full model were specified for each of the three estimation conditions (PI, QML, and Bayesian). Bandalos (2006) suggests that 500 iterations is sufficiently large when doing complex SEM simulations.

Data were simulated under three conditions of linearity, a linear model, or a model with zero nonlinear terms (Figure 1); an interaction model, or a model with one nonlinear term (Figure 2); and a full model, with three nonlinear terms (Figure 3). The linear model is

$$\eta = \alpha + \Gamma_1 \xi_1 + \Gamma_2 \xi_2 + \zeta, \tag{18}$$

where there are three latent variables, two of which (ξ_1, ξ_2) are regressed onto the third (η). α is the intercept, Γ_1 and Γ_2 are regression coefficients, and ζ represents the residuals from the structural part of the model. All the latent variables have three observed indicators. The interaction model is identical to the linear model, except that the independent latent variables were multiplied to create an interaction term; ω_{12} is the coefficient for the interction term $\xi_1\xi_2$.

$$\eta = \alpha + \Gamma_1\xi_1 + \Gamma_2\xi_2 + \omega_{12}\xi_1\xi_2 + \zeta \tag{19}$$

As seen in Figures 2 and 3, grey lines are used here to suggest terms that have been multiplied to form nonlinear latent terms.

The full model was the same as the interaction except for the inclusion of a quadratic term for each of the latent linear independent variables,

$$\eta = \alpha + \Gamma_1\xi_1 + \Gamma_2\xi_2 + \omega_{11}\xi_1\xi_1 + \omega_{12}\xi_1\xi_2 + \omega_{22}\xi_2\xi_2 + \zeta, \tag{20}$$

ω_{11} and ω_{22} are the coefficients for the quadratic terms $\xi_1\xi_1$ and $\xi_2\xi_2$ respectively.

Table 2

Levels of Variables in Simulation

Variable	Values
Nonlinear Coefficient	ω = 0.02, 0.13, 0.26, 0.4
Sample Size	n = 100, 500, 1000
Reliability of Indicators	r = .45, .65, .85
Number of Nonlinear Terms	0 (linear), 1 (interaction), 3 (full)
Estimation Model	Linear, Interaction, Full
Estimation Method	PI, QML, Bayesian

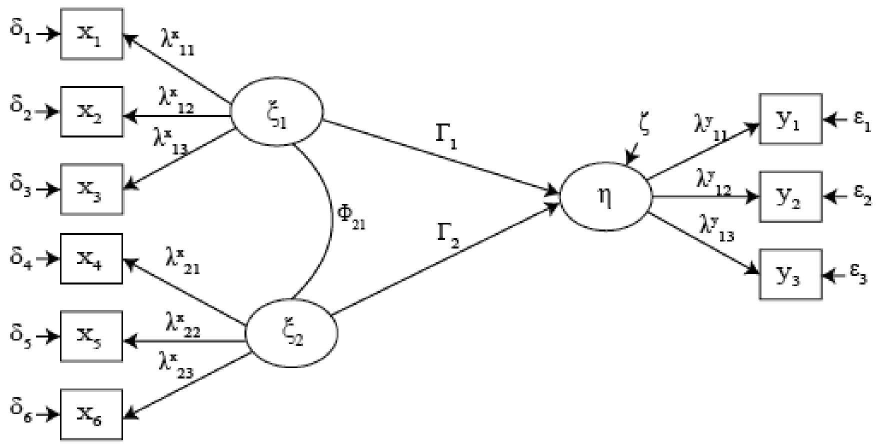

Figure 1. The linear model. The model used to simulate the linear data as well as estimate the linear models.

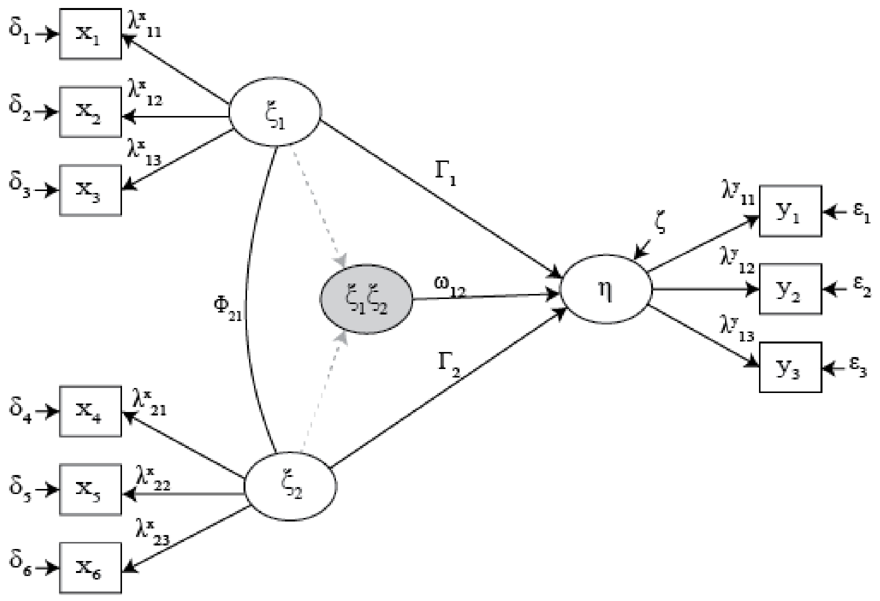

Figure 2. The interaction model. The model used in simulating the data when one nonlinear term was present (interaction data), and the model used to estimate the interaction model for the QML and Bayesian approaches.

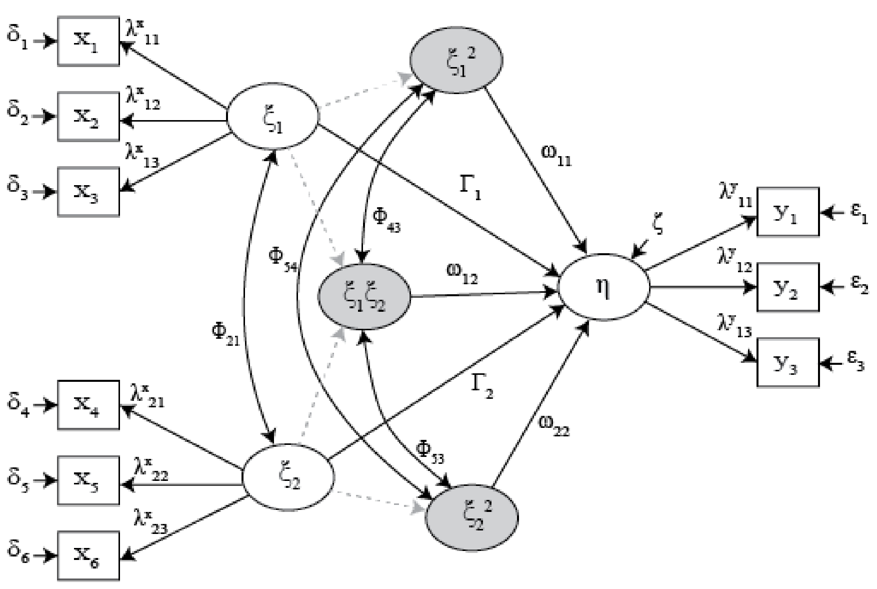

Figure 3. The full model. The model used to simulate the data with three nonlinear terms. Used to estimate the QML and Bayesian full models. The gray lines reflect multiplied terms.

The omega weights for the nonlinear terms were chosen based on the effect sizes for interactions found in Cohen (1988), and Aguinis, Beaty, Boik, and Pierce (2005), as well as one larger omega, .4, to see how the indices performed when the nonlinear terms were very large. The range of sample sizes was chosen to represent large (1000), medium (500), and small (100) sample sizes. The number of nonlinear terms was chosen according to the recommendations of Gerhard et al. (2017) and Fan, Thompson, and Wang (1999), who suggest testing fit statistics according to varying levels of misspecification. The reliability of the indicators were the conditions used by Harring, Weiss, and Hsu (2012) to compare estimation procedures. The values of 0.65 and 0.85 had previously been used by Moulder and Algina (2001) and Jaccard and Wan (1995).

Q1: Testing the traditional cutoffs. The first research question might be reworded to address the situation from the point of view of the practitioner. “If I use traditional fit statistics and fit indices (χ^2 p-value, CFI, TLI, RMSEA, and SRMR) according to their usual cutoffs, will I consistently and accurately detect any omitted nonlinear terms? Will I consistently and accurately detect any nonlinear over specifications (i.e., including nonlinear terms in the model when the underlying data were linear)?" To address these questions three maximum likelihood models were fit to the data.

The first was a linear model (Figure 1), the second was an interaction model estimated with the unconstrained approach (Figure 4), and the third was a full model estimated under the extended unconstrained approach (Figure 5). Table 2 shows all conditions and the levels of those conditions for this study.

Under the nonlinear PI models (unconstrained and extended unconstrained models), the indicators for the nonlinear variables are created by multiplying combinations of other indicators. To create a quadratic variable, each indicator was squared and then loaded onto a new quadratic latent variable. To create an interaction variable each indicator was used exactly once in combination with an indicator from the other factor participating in the interaction. The goal in using the PI approach is to replicate the conditions in which a practitioner might accidentally use the blind measures without being aware of their deficiencies, and these are the models that are in use when the software automatically reports the χ^2-based fit measures.

Mooijaart and Satorra (2009) demonstrated mathematically that the χ^2-based fit measures will be blind to nonlinearity. The consequential hypothesis is that I will not be able to detect nonlinear misspecification using traditional, χ^2-based fit measures. To test this, I will calculate the power and type 1 error rates for each fit measure used for each cutoff (Table 1) across all conditions (Table 2) using the PI models.

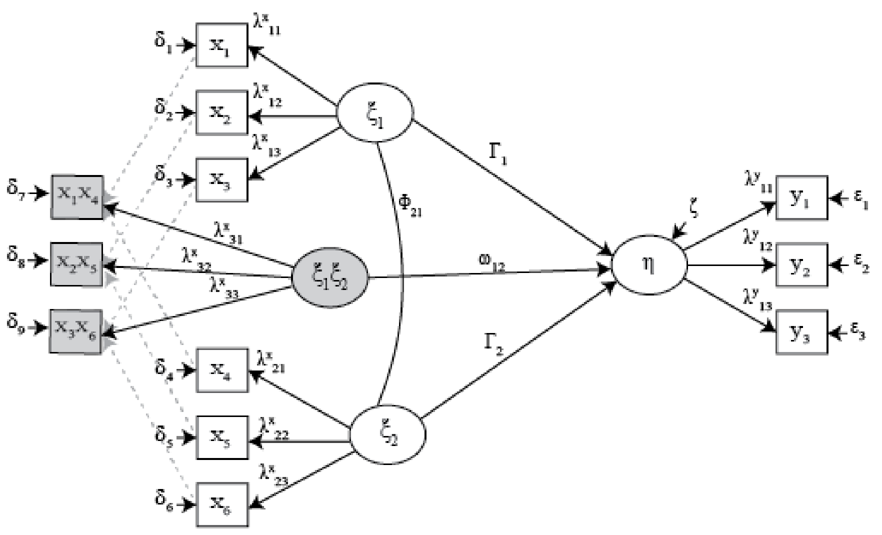

Figure 4. The unconstrained model. The interaction model for the PI approach. Observed indicators are multiplied to create indicators for the nonlinear term.

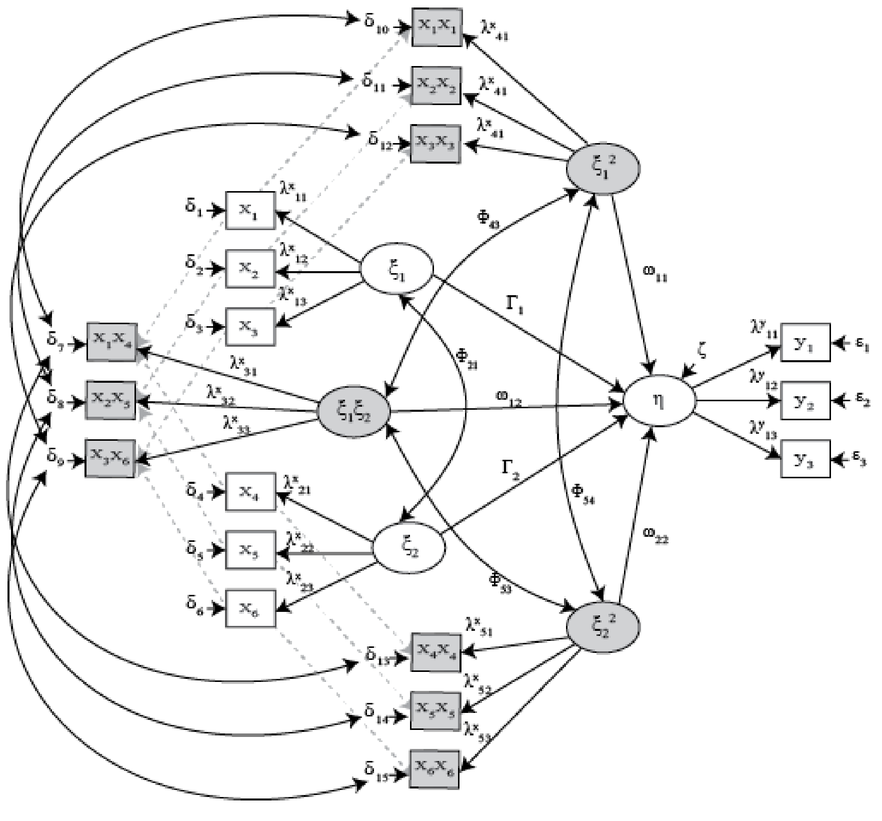

Figure 5. The extended unconstrained model. The full model for the PI approach. Observed indicators are multiplied to create indicators for the nonlinear terms.

Q2: Testing the fit measures for sensitivity. If these indices are blind in combination with their cutoffs, the next question is whether they are sensitive to nonlinearity at all. A fit measure is sensitive if the value of the fit measure is consistently more desirable for the model that matches the data then for other models. When the data are simulated with no nonlinear terms, the fit measures should favor the linear model; when the data are simulated with an interaction term, the fit measure should favor the interaction models, and so on for the full model. In this study, I measured this in two ways. First, type 1 error rate and power were calculated for the χ^2 p-value, CFI, TLI, RMSEA, and SRMR-based on the following set of hypotheses:

$$H_0 \colon \omega_{interaction} = 0 \tag{21}$$

$$H_a \colon \omega_{interaction} \neq 0 \tag{22}$$

where if

$$FM_{linear} \geq FM_{interaction} \colon \text{fail to reject the null} \tag{23}$$

$$FM_{linear} < FM_{interaction} \colon \text{reject the null} \tag{24}$$

where FM represents a fit measure and $<$ represents favorability instead of numerical direction (i.e., $RMSEA_1 = 0.01$ is more favorable than $RMSEA_2 = 0.2$, therefore $RMSEA_1 > RMSEA_2$).

The direction of the greater than sign will be changed to reflect increasing fit on the part of the index. Said more plainly, the null hypothesis will be rejected if the value of the fit index for the interaction model is more desirable than the same value for the linear model. For example, if the value of the CFI in a linear model is 0.94, and the value of the CFI for the interaction model is 0.95, then the CFI prefers the interaction model. If the underlying data are linear, then the CFI has committed a type 1 error. A type 2 error would be committed if the data were simulated with an interaction, but the linear model was favored. Power is calculated as 1 minus the type 2 error rate. This hypothesis test will be referred to throughout the remainder of the text as the comparison hypothesis.

The second way to test whether the traditional measures have any ability to identify nonlinear misspecifications will be to compare the average values of the fit indices across data types (simulated with no nonlinear terms, one nonlinear term, three nonlinear terms) and model specification (linear, interaction, full). If the values of the fit measures display systematic change across conditions of linearity, then the fit measure is, in some degree, sensitive to nonlinearity.

Q3: Exploring alternative fit measures. The third question is, "How do alternative fit measures perform?" The HFI, two-step LRT, R^2, and the PPP are the focus of this section of the article. Testing for these measures followed the pattern used to explore the χ^2-based and covariance-based fit measures under the second question. Specifically, a linear and an interaction model were fit to the data under the PI approach and these fit measures were calculated. Each model was also used as a discrepancy function for the PPP.

The HFI, R^2, and each of the PPPs were then tested according to the comparison hypothesis test (Equations 21-24). Type 1 error was calculated as the proportion of times the fit measure favored the nonlinear model when the data were linear, and the power was calculated as 1 minus the proportion of times the data were nonlinear and the fit measure favored the linear model. The residuals from the structural part of the model needed to calculate the HFI were obtained by treating the factor scores as observed and putting them into a regression. R^2 was included as part of the output from this function. The LRT for the PI models was a Vuong LRT for non-nested models.

Means and standard deviations were then compared between model specifications and number of nonlinear terms in the data. A fit measure is sensitive to nonlinearity if there is consistent variation in the means across condition of nonlinearity. For example, if the HFI for the linear model is .99 when the data are linear, but is .92 when the data are nonlinear, then we might say that the HFI is sensitive to nonlinearity. However, if the HFI does not change, then it is not sensitive to nonlinearity.

Q4: Generalizing to other estimation approaches. The final question looks to see if any of the alternative fit measures were sensitive under other approaches to estimation, specifically the Bayesian and QML approaches. The QML and Bayesian interaction models and full models are reflected in equation 19 and 20 respectively, where $\xi_1\xi_2$, $\xi_1\xi_1$, and $\xi_2\xi_2$ are estimated directly. This contrasts with the PI models, which multiply observed variables and treat them as indicators for the nonlinear latent variables. The linear model for the Bayesian condition has the same equation as the PI linear model, though it is estimated with Bayes. The QML approach had no linear model of its own, borrowing the PI linear model instead; this decision was made under the assumption that a practitioner would likely not estimate a linear model with the QML estimator, and would be more likely to use an ML approach. Thus, this slightly confounds the original question in order to investigate a potentially convenient comparison.

Though the LMS is more frequently used than the QML, the QML gives similar results and is more computationally efficient. Furthermore, it can handle more nonlinear terms than just one. For these reasons, the QML was used to estimate the interaction model (Figure 2) and full model (Figure 3) instead of the LMS. One notable downside of the QML approach is that it is sensitive to the specification of starting values, often failing to converge due to the selected starting values. To help with this, the starting values were taken from the PI approaches when those models converged. When those models did not converge, the starting values were randomly selected according to a normal distribution *N*(0,1). A diffuse prior was used in all

Bayesian conditions with starting values taken from the PI approaches (Kelava & Nagengast, 2012).

Results

None of the χ^2-based, covariance-based, or PPP-style fit measures showed any signs of nonlinear sensitivity in any condition. The HFI and R^2 were the only measures to show signs of nonlinear sensitivity, and that sensitivity varied depending on sample size, strength of the nonlinear terms, and reliability of the indicators. Since most of the measures did not vary across these three variables, and the evaluation of the HFI and R^2 is generally consistent across conditions, the results are aggregated across levels of sample size, strength of nonlinear terms, and reliability of indicators.

Q1: How Often Are Decisions Based on Common Cutoffs Incorrect?

Question 1 asks how often decisions based on frequently used cutoffs for absolute fit measures will be correct. The CFI, TLI, RMSEA, SRMR, and χ^2 were used to determine goodness of fit for models estimated under a PI approach. The thresholds proposed by Hu and Bentler (1999) were used for the CFI, TLI, RMSEA, and SRMR, while a cutoff of 0.05 was used for the χ^2 *p*-value. As Mooijaart and Satorra (2009) predicted, these measures were entirely blind to any changes to the underlying linearity of the data. In almost every instance the fit measures declared good fit, even when the data did not match the specified model. This led to high type 1 error rates (Table 3) in addition to high power (Table 4).

Table 3 shows the type 1 error rate for the fit measures CFI, TLI, RMSEA, SRMR, and of the χ^2 *p*-value when using traditional cutoffs. When no nonlinear terms were used to simulate the data, the CFI declared the interaction models and full models as having good fit 60% of the time. When there was only one nonlinear term, the CFI declared the linear models and full models as having good fit 75% of the time. In general, as the number of nonlinear terms increased the type 1 error rose; at the same time, power tended to decrease (Table 4).

Table 3

Type 1 Error Rates for CFI, TLI, RMSEA, SRMR, and χ^2 p-value

Number of nonlinear terms	CFI	TLI	RMSEA	SRMR	CHISQP
No Terms	.60	.58	.63	.64	.54
One Term	.65	.65	.66	.66	.64
Three Terms	.75	.77	.71	.71	.83

Table 4 shows the power for CFI, TLI, RMSEA, SRMR and the χ^2 *p*-value. When the data were simulated without any nonlinear terms the linear model was declared as a good fitting model 96% of the time according to the CFI. When the data were simulated with an interaction term the interaction model was shown to be a good fitting model 85% of the time according to the CFI.

Table 4

Power for CFI, TLI, RMSEA, SRMR, and χ^2 p-value

Number of nonlinear terms	CFI	TLI	RMSEA	SRMR	CHISQP
No Terms	.96	.92	.92	1.00	.93
One Term	.85	.79	.87	.99	.74
Three Terms	.68	.59	.83	.93	.39

The combined results of Table 3 and Table 4 seem to suggest that these measures are sensitive to nonlinearity (i.e., the change in type 1 error rates and power as the simulated data includes more nonlinear terms). This interpretation is a mistake. In both of these tables, the only correct model was the model specified to match the simulated data exactly (i.e., a linear model was the correct model when the data were simulated with no nolinear terms, an interaction model was correct when the data were simulated with one nonlinear term, and the full model was correct when simulated with three nonlinear terms).

Table 5 shows the proportion of times that a model was identified as fitting well according to each fit measure. This shows that, according to the CFI, the linear model fit around 96% of the time regardless of the underlying structure of the data. Likewise, the interaction and full models fit the data exactly as often when the model matched the data as when it didn’t.

The TLI, RMSEA, SRMR, and χ^2 p-value likewise did not vary across conditions of nonlinearity in the data. In all cases the changes in type 1 error and power are due to a change in the specified model. This variation is entirely independent of the number of nonlinear terms. In this way Table 3 and Table 4 are misleading when taken without the information provided by Table 5. Additionally, it appears that the commonly used cutoff values are insufficient for diagnosing nonlinear misspecifications in nonlinear models.

Table 5

Proportion of Replications When the Fit Measure Met the Threshold.

		Estimated Model	
Simulating Model	Linear	Interaction	Full
CFI			
0 Nonlinear Terms	.96	.86	.70
1 Nonlinear Term	.95	.85	.68
3 Nonlinear Terms	.96	.86	.68
TLI			
0 Nonlinear Terms	.92	.80	.61
1 Nonlinear Term	.92	.79	.59
3 Nonlinear Terms	.92	.79	.59
RMSEA			
0 Nonlinear Terms	.92	.87	.84
1 Nonlinear Term	.92	.87	.83
3 Nonlinear Terms	.92	.87	.83
SRMR			
0 Nonlinear Terms	1.00	.99	.95
1 Nonlinear Term	1.00	.99	.94
3 Nonlinear Terms	1.00	.99	.93
CHISQP			
0 Nonlinear Terms	.93	.76	.42
1 Nonlinear Term	.93	.74	.40
3 Nonlinear Terms	.93	.74	.39

Q2: How Blind Are the Traditional Fit Measures?

Question 2 asks “Are χ^2-based and covariance-based fit measures (CFI, TLI, SRMR, RMSEA, χ^2, AIC, and BIC) blind to the extent that they are not useful at all? Or, is there some recognizable pattern among them that could be helpful?” This was explored in two ways; the first, comparing the means and standard deviations for the fit measures in each condition, and the second, comparing the power and type 1 error rate according the the comparison hypothesis test (equations 21-24). Table 6 displays the mean values of the indices for each estimation model and each simulation model. The standard deviations were very small for the absolute measures (~0.01). The relative fit measures had proportionately small standard deviations, and a more complex model never had a more desirable value than that of a simpler model (e.g., AIC for an interaction model was never less than the AIC for a linear model.) Table 6 shows that any variation in the means of the CFI, TLI, RMSEA, SRMR, χ^2, AIC, and BIC indices the product of model specification. Linearity or nonlinearity of the data had no effect on the average value of any of these fit measures. In other words, even extreme nonlinearity did not affect the average values of these fit measures.

Table 7 shows the type 1 error rates and power for the CFI, TLI, RMSEA, and SRMR according to the comparison hypothesis test. The null hypothesis for this test is that the coefficient for the nonlinear term is equal to zero. An interaction model is then compared to a linear model on a fit measure. If the fit measure favors the interaction model, then the null hypothesis is rejected. While according to this definition there is a low type 1 error rate, there is also very low power (essentially identical to the type 1 error). This further ratifies that these measures are not useful when comparing them between models.

Table 6

Means for Fit Measures by Linearity of Data and Estimating Model

Model	CFI	TLI	RMSEA	SRMR	CHISQP	AIC	BIC
0 Nonlinear Terms							
Linear	.99	1.00	.02	.03	.47	7393	7495
Interaction	.98	.98	.03	.04	.29	10529	10674
Full	.96	.95	.03	.05	.12	18864	19129
1 Nonlinear Term							
Linear	.99	1.00	.02	.03	.47	7391	7493
Interaction	.98	.97	.03	.04	.28	10476	10620
Full	.96	.94	.04	.05	.11	18628	18893
3 Nonlinear Terms							
Linear	.99	1.00	.02	.03	.46	7389	7491
Interaction	.98	.97	.03	.04	.27	10509	10654
Full	.96	.94	.03	.05	.11	18631	18895

Table 7

Type 1 Error Rate and Power for Fit Measures According to the Comparison Hypothesis Test

Measure	Type 1	Power
CFI	.12	.12
TLI	.22	.22
RMSEA	.23	.22
SRMR	.04	.04

Note: The type 1 error rates and power are essentially identical; this is not a computation error.

Q3: Are There Any Useful Alternative Fit Measures?

Question 3 asks if there are any viable alternatives to the absolute and relative fit measures commonly used for PI models. This included using LRT, HFI, R^2, and PPP. When Vuong LRTs for non-nested models were conducted between linear and interaction models, linear and full models, and interaction and full models, the simpler model was always chosen regardless of whether the data were linear or nonlinear, even in the conditions of extreme nonlinearity (terms = 3, ω = 0.4, r_{xy} = 0.85).

Table 8 shows the means and standard deviations of the HFI, PPP, and R^2 for the PI models. The PPP displayed in this table specifically represents the PPP based on the likelihood function; however, the performance of this version of the PPP was identical to the others. The PPP was entirely insensitive to nonlinearity in any condition. The PPP made no change across PI models.

The HFI did not do much better than any of the other measures on average. The mean value of the HFI does change when there are multiple nonlinear terms in the data, but this is a small change. In conditions of extreme nonlinearity and with a sample size of 1000, the performance of the HFI improved. This improvement was not enough to be useful, especially since the conditions of nonlinearity were so extreme (ω = 0.4, r_{xy} = 0.85). However, it was clear that the HFI was at least sensitive to nonlinearity.

By far, the measure that was the most sensitive to nonlinearity was the R^2. When there are no nonlinear terms in the data, R^2 does, in fact, increase as expected. This increase is small (~.015). As the number of nonlinear terms increases, this difference between models becomes much larger. The average change in R^2 between the linear model and the interaction model grows to .08 when there is none nonlinear term present and .1 when there are three present. This is a very large difference.

Figure 6 displays boxplots for the change in R^2 between the PI interaction model and the PI linear model. The x-axis represents the four levels for the nonlinear coefficient, and the groups represent the conditions of linearity for the data. As the strength of the nonlinear term grew, so did the difference between the R^2 for the linear and interaction models. There are a handful of outliers in the boxplot; all of these outliers occurred when the sample size was small (100).

Table 8

Mean and Standard Deviation of the HFI, R^2, and PPP *for the PI Models*

	Estimated Model	HFI	$\boldsymbol{R^2}$	PPP
0 Nonlinear Terms				
	Linear Model	.99 (.02)	.24 (.09)	.06 (.12)
	Interaction Model	.99 (.02)	.25 (.05)	.06 (.1)
	Full Model	.99 (.02)	.27 (.05)	.06 (0)
1 Nonlinear Term				
	Linear Model	.99 (.02)	.25 (.09)	.06 (.11)
	Interaction Model	.99 (.02)	.33 (.06)	.06 (.16)
	Full Model	.99 (.02)	.35 (.07)	.06 (0)
3 Nonlinear Terms				
	Linear Model	.97 (.05)	.25 (.09)	.06 (.11)
	Interaction Model	.98 (.03)	.33 (.08)	.06 (.15)
	Full Model	.98 (.03)	.4 (.08)	.06 (0)

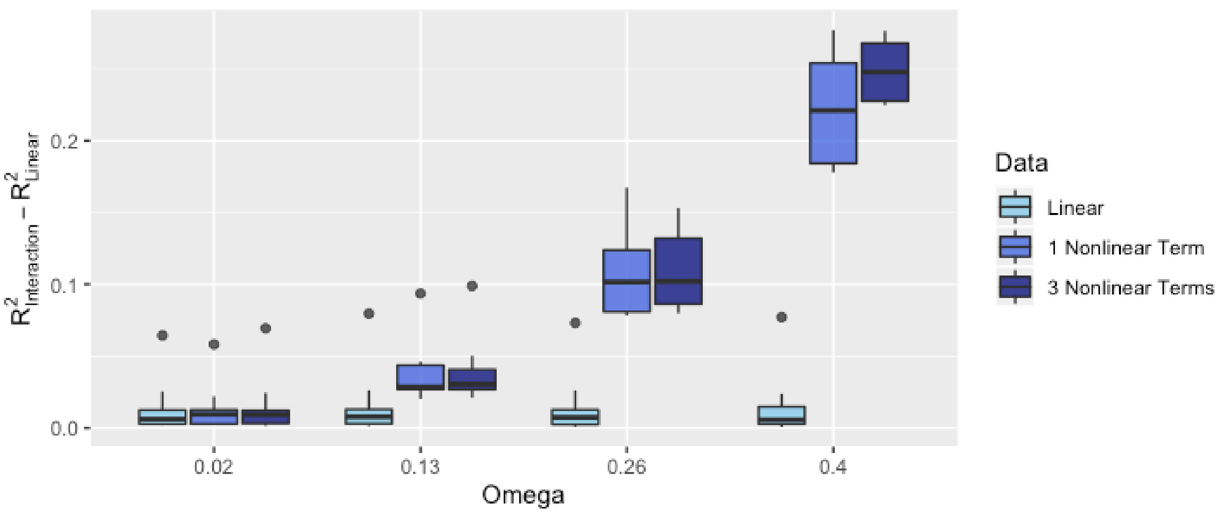

Figure 6. Boxplot of the differences in R-squared between the linear and interaction PI models.

Q4: Do the Fit Measures Work When Other Estimation Approaches Are Used?

Question 4 asks if the alternative fit measures work when other estimation approaches are used. This is a very desirable trait, one that would undoubtedly make for a useful fit measure. The two estimation approaches investigated in this condition were the QML and Bayesian methods.

QML models. When the QML method was used, the traditional fit measures (referring in this instance to the likelihood, AIC, BIC, χ^2, χ^2 *p*-value, RMSEA, and SRMR) continued to demonstrate nonlinear insensitivity. LRTs in this condition always chose the simplest model. The last remaining fit measures were the HFI, the R^2, and the PPP. Analysis of these measures was approached in the same way as the traditional measures in question 2. Table 9 shows the means and standard deviations for these measures across all conditions for the QML models.

Table 9

Mean and Standard Deviation for the HFI, $\boldsymbol{R^2}$, and PPP for the QML Models

Estimated Model	HFI	R2	PPP
Linear Data			
Linear Model	.99 (.02)	.24 (.09)	.06 (.12)
Interaction Model	.99 (.02)	.14 (.05)	.06 (.11)
Full Model	.99 (.02)	.15 (.05)	.06 (.17)
Interaction Data			
Linear Model	.99 (.02)	.25 (.09)	.06 (.11)
Interaction Model	.99 (.02)	.16 (.06)	.06 (.11)
Full Model	.99 (.02)	.17 (.07)	.06 (.17)
Full Data			
Linear Model	.97 (.05)	.25 (.09)	.06 (.11)
Interaction Model	.99 (.02)	.18 (.08)	.06 (.11)
Full Model	.99 (.02)	.19 (.08)	.06 (.13)

The linear model shown in Table 9 is the same linear model from the PI estimation approach. The nonlinear QML models were compared against the PI linear model in the hopes that a successful comparison here might be more convenient for researchers later. Unfortunately, the R^2 for the QML models was generally much lower than the R^2 for the linear PI models, perhaps due to a more lenient convergence criterion in the QML condition than in the PI condition. This makes it difficult to tell if theR^2 is sensitive to nonlinearity in the QML condition. These results also suggest that R^2 should not be compared between models estimated under different approaches.

The PPP based off the χ^2 discrepancy function was not sensitive to nonlinearity either. None of the tested PPPs were sensitive. While the PPP in Table 6 refers explicitly to the χ^2 discrepancy function-based PPP, the pattern of insensitivity was consistent across different PPPs using other discrepancy functions. The HFI did not appear to be practically sensitive to nonlinearity. However, HFI for linear models tended to decrease when multiple nonlinear terms were present, but this was a very small, practically insignificant change.

When viewed under conditions with strong nonlinear terms and a large sample size, the HFI did better (Table 10). When the data were linear, all models had the same means for the HFI. When the data were simulated with an interaction model, the average HFI for the linear model was smaller than the average HFI for the interaction model and the full model. When the data were simulated with the full model, the linear model had the smallest HFI and the full model had the largest average HFI. The PPP did not improve in this case of extreme nonlinearity. The R^2 likewise showed improved sensitivity in this case of strong linearity while the PPP showed no change. Table 10 shows the same information as Table 8 but represents only a subset of the data when the nonlinear terms were strongest.

Table 10

Mean and Standard Deviation of the HFI, R^2, and PPP for the QML Model with Very Strong Nonlinear Terms

Estimated Model	HFI	R2	PPP
Linear Data			
Linear Model	.99 (.02)	.24 (.09)	.06 (.07)
Interaction Model	.99 (.02)	.12 (.05)	.06 (.08)
Full Model	.99 (.02)	.12 (.05)	.06 (.08)
Interaction Data			
Linear Model	.97 (.03)	.24 (.09)	.06 (.08)
Interaction Model	.98 (.03)	.18 (.06)	.06 (.08)
Full Model	.98 (.02)	.18 (.07)	.06 (.08)
Full Data			
Linear Model	.83 (.08)	.24 (.09)	.06 (.07)
Interaction Model	.97 (.03)	.29 (.08)	.06 (.09)
Full Model	.98 (.03)	.3 (.08)	.06 (.08)

This table shows that if the R^2 for a QML interaction model is larger than or close to a maximum likelihood linear model fit to the same data, then there is an unmistakably large nonlinear term or terms in the data. The HFI in this condition was much more sensitive to nonlinearity, though only when there were multiple nonlinear terms.

Bayesian models. It should be noted that most of the Bayesian models did not converge, particularly the nonlinear ones. Convergence was defined as all parameters having a PRSF less than 1.05. There were 14 parameters that had not converged in the nonlinear models, though the omegas had very strong correlations with the omegas from the PI and QML methods (r > .9).

The results for the Bayesian estimation approach were consistent with the PI and QML conditions. The χ^2-based indices were not sensitive to nonlinearity. Neither were the likelihood-based indices. Table 11 shows the mean values and standard deviations of the HFI, R^2, and PPP. The PPP showed no change across conditions of nonlinearity. Neither did the HFI. Once again, the only fit measure to show any sign of nonlinear sensitivity was the R^2. As the data became increasingly nonlinear the difference between the R^2 for the linear and nonlinear models increases. This difference, unfortunately, is not very large (~.07). These results are probably affected by the convergence problems.

Table 11

Mean and Standard Deviation of the HFI, R^2 and PPP for the Bayesian Models

Estimated Model	HFI	$\boldsymbol{R^2}$	PPP
Linear Data			
Linear Model	.99 (.02)	.13 (.09)	.06 (.11)
Interaction Model	.99 (.02)	.15 (.05)	.06 (.03)
Full Model	.99 (.02)	.16 (.05)	.06 (.08)
Interaction Data			
Linear Model	.99 (.02)	.13 (.09)	.06 (.11)
Interaction Model	.99 (.02)	.17 (.06)	.06 (.02)
Full Model	.99 (.02)	.18 (.07)	.06 (.07)
Full Data			
Linear Model	.97 (.05)	.13 (.09)	.06 (.1)
Interaction Model	.99 (.02)	.2 (.08)	.06 (.03)
Full Model	.99 (.02)	.21 (.08)	.06 (.08)

Discussion

This article attempts to provide a guideline for practitioners to use when making nonlinear modeling decisions. Four questions have been addressed in this pursuit. First, how frequently are nonlinear modeling decisions based on χ^2-based indices incorrect? Second, do the traditional χ^2-based and covariance-based measures provide any nonlinear information? Third, are there any viable alternatives to traditional fit measures? Fourth and finally, do any of the alternative measures work in the QML and Bayesian context?

This study demonstrated that the absolute fit measures CFI, TLI, RMSEA, SRMR, and χ^2 p-value provide no useful nonlinear information at their cutoffs. Not only were these absolute fit measures insensitive to nonlinearity at their cutoffs, but they were also insensitive when a general test of sensitivity was used. Namely, an interaction model and a linear model were fit to the data, and the proportion of times that each fit measure favored each model in each condition was calculated. These measures showed no indication of discrimination between models.

Researchers using these absolute fit measures are essentially making uninformed decisions in regards to nonlinear terms. The relative fit measures did no better. The AIC, BIC, and two-step procedure using the Vuong test and LRT always favored the simpler model. Thus, practitioners relying on these relative measures will be no more successful than when relying on the traditional absolute fit measures.

Of the alternative fit measures explored—R^2, HFI, and the different versions of the PPP—only the R^2 showed any useful degree of sensitivity to nonlinearity. Across all conditions, no version of the PPP showed any level of nonlinear sensitivity, while the HFI was sensitive only when the data were significantly nonlinear. R^2 was most consistently accurate and grew more accurate as the data became more nonlinear. While R^2 did not perform as well in the Bayesian and QML conditions, the same pattern was apparent. As the data became more nonlinear, the change in $\boldsymbol{R^2}$ grew.

Since $\boldsymbol{R^2}$ is a greedy algorithm it is not entirely clear when a difference in R^2 should be considered evidence of nonlinearity. A “large” $\boldsymbol{R^2}$ difference in this study was almost entirely exclusive to conditions of nonlinearity. Furthermore, R^2 is much more useful in the PI approach than when the other approaches were used. The suggested guideline for practitioners is therefore as follows:

- Fit a linear and a nonlinear PI model and calculate R^2
- Ignore any χ^2 or covariance-based fit measures
- Compare the two models by the R^2

If the change in R^2 between the linear and nonlinear model is large, then use the nonlinear model. If the change is not very large, then the researcher may be justified in using the linear model. After deciding on the appropriate model, the researcher might then estimate their model using the estimation approach of their choice. Obviously, when using the R^2 as a fit measure there must be a dependent variable involved.

This procedure should not be used in the place of substantive theory. Experience shows that many practitioners tend to blindly trust the software with many decisions. The software is a black box; the model goes in, and when it comes out practitioners check to see if it converged, has standard errors, etc. This may not be enough to ensure that estimations are appropriate. For example, practitioners who were unaware of the literature on fit measures in the context of nonlinearity might assume because their model converged and has the desired output suggesting the estimation went well, that they can rely on the fit measures to decide whether to include an interaction.

Part of the allure of fit measures is that they provide greater certainty than simple theory. However, while fit measures can summarize otherwise unobservable patterns in data, they are also subject to greater limitations than researchers are. Allowing fit measures to have the loudest voice in modeling decisions is, in a way, throwing out any other experience, expertise, and literature on a given topic. If fit measures were the only thing required to understand the complexity of the human organism, then research could one day become entirely automated.

Another potential reason that fit measures are so tempting to use as the primary decision-making tool, is the hope of finding the best model. This puts pressure on researchers to be perfect, while in all acutality there may not be a best model. Maybe what researchers really need to find are better, or good-enough models. With this mindset, fit measures become less important as a dominating factor; substantive theory will lead the conversation. While fit measures are healthy, contributing members to the conversation, it is important that other evidences, such as experience and previous literature, are not excluded.

The importance of researchers and practitioners deepening their understanding of statistical principles has obvious application to fite measures. Research on fit measures for nonlinear SEM and statistical models are constantly under development. While it is nearly impossible to know everything related to statistical modeling in the social sciences, keeping as up to date as possible can serve to protect against issues, such as the inappropriate use of common fit measures. This serves as another reason why relying on theory is important. The future may find that current statistics are inadequate; models that were deemed inaccurate now due to an overreliance on current statistics, might be found to be good models with future statistics.

Finally, researchers should become familiar with the defaults of their software. This is an essential, and sometimes over looked practice. This is important to consider because defaults were chosen according to assumptions and theories that may not be appropriate for the research at hand, such as the inclusion of traditional fit measures when PI approaches are used. The software is unaware that a PI model has been used, and usually the default is to include those measures. Hence, an unsuspecting researcher may be unintentionally misled by defaults. In summation, to reduce the occurrence of these kind of errors, researchers and practicioners might (a) rely upon substantive theory, (b) be aware of the limitations of fit measures, and (c) make sure they understand each of the defaults in their chosen estimating software.

As was mentioned previously, R^2 was less useful when the QML and Bayesian approaches were used to estimate models. For the Bayesian approach, this may be because most of the models did not converge. Future research may test different priors, more thinning, or more time allotted for the models to converge; perhaps the R^2 might be more useful in identifying nonlinear misspecifications under those conditions. Additionally, this suggested future research might also investigate the performance of the Bayesian specific fit measures if the software becomes available to appropriately integrate out the latent terms in nonlinear models.

The R^2 did not perform as well when QML models were used. One likely reason for this is that the QML approach borrowed its linear condition from the PI approach. The original intent for this design decision was that, if it worked, it would be more convenient for the practitioner to compare models in this way. However, the models do not seem to be comparable. It should also be noted that R^2 for the QML models are much lower than for the PI models. Perhaps making the convergence criterion stricter would make the QML and PI approaches more comparable.

Future research might also explore the method of moments approach in association with these fit measures and the fit measures related to it. This approach was excluded from this study due to the difficulty in incorporating this approach into the simulation, but previous research suggests that it might be a productive avenue to explore.

This study showed conclusively that the residuals from the structural part of the model are more beneficial for exploring nonlinear misspecification than the residual covariance matrix. The HFI and R^2 were the only measures that were connected to the structural part of the model, and they were the only measures sensitive to nonlinearity. Future research should further explore the structural part of the model. Perhaps a variation on the HFI or one of its parent measures would prove to be more sensitive than the HFI. Perhaps future research might derive a standardized fit measure from R^2. It seems that the future of nonlinear fit measures, at least for PI approaches, should be built on the structural part of the model.

Conclusion

Mooijaart and Satorra (2009) asserted the mathematical blindness of traditional χ^2-based fit measures. This article demonstrated the practical blindness of not only absolute fit measures like the CFI, TLI, RMSEA, SRMR, and χ^2 p-value, but also relative fit measures such as the AIC, BIC, and the two-step comparison using the Vuong test and the traditional LRT. The HFI was shown to be practically insensitive to nonlinearity except in cases of extreme nonlinearity. R^2 was consistently sensitive to nonlinearity across conditions, though this sensitivity was most pronounced with PI models.

Beyond the fit measures themselves, the topic of blind fit measures raises the concern of “the blind leading the blind.” If researchers become too dependent upon fit measures that are only reliable in certain situations, they will inevitably, at some point, be blindly led to false conclusions. Rather, researchers and practitioners should allow substantive theory to guide modeling decisions and let the fit measures take a more supportive role. This emphasis on theory-centered modeling will reduce the blindness on the part of the researcher.

References

Aguinis, H., Beaty, J. C., Boik, R. J., & Pierce, C. A. (2005). Effect size and power in assessing moderating effects of categorical variables using multiple regression: A 30-year review. *Journal of Applied Psychology, 90*(1), 94–107.

Asparouhov, T., & Muthén, B. (2017). *Prior-posterior predictive p-values*. Retrieved from https://www.statmodel.com/download/PPPP.pdf.

Bandalos, D. L. (2006). The use of Monte Carlo studies in structural equation modeling research. In G. Hancock & R. Mueller (Eds.), *Structural equation modeling: A second course* (pp. 385–426). Greenwich, CT: Information Age.

Bayarri, M. J., & Berger, J. O. (2004). The interplay of Bayesian and frequentist analysis. *Statistical Science, 14*(1), 58-80.

Bentler, P. M. (1990). Comparative fit indexes in structural models. *Psychological Bulletin, 107*(2), 238-246.

Chen, F. F. (2007). Sensitivity of goodness of fit indexes to lack of measurement invariance. *Structural Equation Modeling: A Multidisciplinary Journal, 14*(3), 464-504.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Hillsdale, NJ: Erlbaum.

Denwood, M. J. (2016). runjags: An R package providing interface utilities, model templates, parallel computing methods and additional distributions for MCMC models in JAGS. *Journal of Statistical Software, 71*(9), 1-25.

Dimitruk, P., Schermelleh-Engel, K., Kelava, A., & Moosbrugger, H. (2007). Challenges in nonlinear structural equation modeling. *Methodology: European Journal of Research Methods for the Behavioral and Social Sciences, 3*(3), 100-114.

Fan, X., Thompson, B., & Wang, L. (1999). Effects of sample size, estimation methods, and model specification on structural equation modeling fit indexes. *Structural Equation Modeling: A Multidisciplinary Journal, 6*(1), 56-83.

Gelman, A., Stern, H. S., Carlin, J. B., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). *Bayesian data analysis*. New York, NY: Chapman and Hall/CRC.

Gerhard, C., Büchner, R. D., Klein, A. G., & Schermelleh-Engel, K. (2017). A fit index to assess model fit and detect omitted terms in nonlinear SEM. *Structural Equation Modeling: A Multidisciplinary Journal, 24*(3), 414-427.

Harring, J. R., Weiss, B. A., & Hsu, J. C. (2012). A comparison of methods for estimating quadratic effects in nonlinear structural equation models. *Psychological Methods, 17*(2), 193-214.

Hu, L. T., & 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.

Jöreskog, K. G., & Yang, F. (1996). Nonlinear structural equation models: The Kenny-Judd model with interaction effects. In G. Marcoulides & R. Schumacker (Eds.), *Advanced structural equation modeling: Issues and techniques*, (pp. 57-88). Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Kelava, A., & Nagengast, B. (2012). A Bayesian model for the estimation of latent interaction and quadratic effects when latent variables are non-normally distributed. *Multivariate Behavioral Research, 47*(5), 717-742. DOI: 10.1080/00273171.2012.715560

Kenny, D. A., & Judd, C. M. (1984). Estimating the nonlinear and interactive effects of latent variables. *Psychological Bulletin, 96*(1), 201-210.

Klein, A. G., Gerhard, C., Büchner, R. D., Diestel, S., & Schermelleh-Engel, K. (2016). The detection of heteroscedasticity in regression models for psychological data. *Psychological Test and Assessment Modeling, 58*(4), 567–592.

Klein, A., & Moosbrugger, H. (2000). Maximum likelihood estimation of latent interaction effects with the LMS method. *Psychometrika, 65*(4), 457-474.

Klein, A. G., & Muthén, B. O. (2007). Quasi-maximum likelihood estimation of structural equation models with multiple interaction and quadratic effects. *Multivariate Behavioral Research, 42*(4), 647-673.

Kruschke, J. (2014). *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan.* New York, NY: Academic Press.

Kutner, M. H., Nachtsheim, C., & Neter, J. (2004). *Applied linear regression models*. Boston, MA: McGraw-Hill/Irwin.

Lee, S. Y., Song, X. Y., & Tang, N. S. (2007). Bayesian methods for analyzing structural equation models with covariates, interaction, and quadratic latent variables. *Structural Equation Modeling, 14*(3), 404-434.

Little, T. D., Bovaird, J. A., & Widaman, K. F. (2006). On the merits of orthogonalizing powered and product terms: Implications for modeling interactions among latent variables. *Structural Equation Modeling, 13*(4), 497-519. DOI: 10.1207/s15328007sem1304_1

Marsh, H. W., Wen, Z., & Hau, K. T. (2004). Structural equation models of latent interactions: evaluation of alternative estimation strategies and indicator construction. *Psychological Methods, 9*(3), 275-300.

Maslowsky, J., Jager, J., & Hemken, D. (2015). Estimating and interpreting latent variable interactions: A tutorial for applying the latent moderated structural equations method. *International Journal of Behavioral Development, 39*(1), 87-96.

Maydeu-Olivares, A. (2013). Goodness-of-fit assessment of item response theory models. *Measurement: Interdisciplinary Research and Perspectives, 11*(3), 71-101.

Merkle, E., Furr, D., & Rabe-Hesketh, S. (2018). Bayesian model assessment: Use of conditional vs marginal likelihoods. Retrieved from https://arxiv.org/abs/1802.04452.

Merkle, E., You, D., Schneider, L., Bae, S., & Merkle, M. E. (2018). Package 'nonnest2'. Retrieved from https://cran.r-project.org/web/packages/nonnest2/nonnest2.pdf.

Mooijaart, A., & Bentler, P. M. (2010). An alternative approach for nonlinear latent variable models. *Structural Equation Modeling, 17*(3), 357-373.

Mooijaart, A., & Satorra, A. (2009). On insensitivity of the chi-square model test to nonlinear misspecification in structural equation models. *Psychometrika, 74*(3), 443-455. https://doi.org/10.1007/s11336-009-9112-5

Moosbrugger, H., Schermelleh-Engel, K., Kelava, A., & Klein, A. G. (2009). Testing multiple nonlinear effects in structural equation modeling: A comparison of alternative estimation approaches. In T. Teo & M. Khine (Eds.), S*tructural equation modelling in educational research: Concepts and applications* (pp. 103-136). Rotterdam, NL: Sense Publishers.

Moosbrugger, H., Schermelleh-Engel, K., & Klein, A. (1997). Methodological problems of estimating latent interaction effects. *Methods of Psychological Research Online, 2*(2), 95-111.

Moulder, B. C., & Algina, J. (2002). Comparison of methods for estimating and testing latent variable interactions. *Structural Equation Modeling, 9*(1), 1-19.

Muthén, B. O., & Asparouhov, T. (2012). Bayesian structural equation modeling: A more flexible representation of substantive theory. *Psychological Methods, 17*(3), 313-335.

Muthén, L. K., & Muthén, B. O. (2011). *Mplus user's guide* (6th ed.). Los Angeles, CA: Muthén & Muthén.

Ping, R. A., Jr. (1995). A parsimonious estimating technique for interaction and quadratic latent variables. *Journal of Marketing Research 32*(3), 336-347.

Plummer, M. (2003). Jags: A program for analysis of Bayesian graphical models using Gibbs sampling. Retreived from http://citeseer.ist.psu.edu/plummer03jags.html.

R Core Team (2013). R: A language and environment for statistical computing. Retrieved from http://www.R-project.org/.

Rosseel, Y. (2012). lavaan: An R package for structural equation modeling. *Journal of Statistical Software, 48*(2), 1-36. Retrieved from http://www.jstatsoft.org/v48/i02/.

RStudio Team (2015). *RStudio: Integrated Development for R*. Retrieved from http://www.rstudio.com/.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., & Van Der Linde, A. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society: Series B (Statistical Methodology), 64*(4), 583-639.

Umbach, N., Naumann, K., Brandt, H., & Kelava, A. (2017). Fitting nonlinear structural equation models in R with package nlsem. *Journal of Statistical Software*, *77*(1), 1-20.

Vuong, Q. (1989). Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica: Journal of the Econometric Society 57*(2), 307-333.

Wall, M. M., & Amemiya, Y. (2003). A method of moments technique for fitting interaction effects in structural equation models. *British Journal of Mathematical and Statistical Psychology, 56*(1), 47-63.

Wang, J., & Wang, X. (2012). *Structural equation modeling: Applications using Mplus*. Chicester, UK: John Wiley & Sons.

Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis*. New York, NY: Springer-Verlag New York, Inc.

APPENDIX A

Annotated Bibliography

Statistical models are often developed with the help of goodness-of-fit measures. These are statistics that measure how well a model predicts the data. In the special case of nonlinear structural equation modeling, no fit indices are widely accepted as appropriate. While a lot of work has explored different estimation methods, little work has been done to develop fit indices in this case, and what work has been done requires further testing. This annotated bibliography sets the literary foundation for the accompanying thesis, briefly introducing the important topics and articles.

Fit Measures, Their Blindness, and Alternatives

This section covers references related to model selection and goodness of fit, the blindness of traditional measures, and alternative fit measures.

Fit measures. The traditional fit measures that are regularly referred to in the article are the chi-square (χ^2) or covariance-based fit measures. Specifically, the χ^2 *p*-value, CFI, TLI, RMSEA, SRMR, AIC, and BIC. These measures are blind to nonlinearity.

Bentler, P. M. (1990). Comparative fit indexes in structural models. *Psychological Bulletin, 107*(2), 238-246.

This is the introduction of the CFI as a fit index. The CFI was created with a number of other indices, though the author concludes that the CFI is the best. The author concludes that a new way of thinking about model fit may be in order, and suggests exploring new distributions outside of the χ^2. The NNFI (TLI) was compared to these measures and differences were discussed. The author favored the CFI because the TLI had a wider

range of values, so that in some circumstances it led to overfitting and in others underfitting. The author did not consider nonlinearity in their test of fit measures.

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.

The χ^2 is very sensitive to sample size. Bentler and Bonett demonstrate that results drawn from the χ^2 *p*-value are heavily influenced by sample size to the point that a good fitting model may be less desirable than another model, it is simply that the sample size was small enough that many models would have been accepted. Conversely, poor fitting models may be close to the true model and have poor fit because the statistic is too sensitive. It is this weakness specifically that will lead to the development of many other fit measures. Most notably, from one of the authors, Bentler, is the creation of the CFI.

Chen, F. F. (2007). Sensitivity of goodness of fit indexes to lack of measurement invariance. *Structural Equation Modeling: A Multidisciplinary Journal*, *14*(3), 464-504.

This article shows one example of how χ^2-based fit statistics can be used for testing model invariance. This is also an example of establishing cutoff values, something that I hope to do if the χ^2-based fit statistics perform reasonably well.

Hu, L. T., & 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.

The authors describe a number of common fit measures, including those discussed as part of this thesis. Following this, a simulation study was conducted and different cutoffs were suggested. Considering that these fit statistics are included in the output when a product indicator approach is used despite being mathematically blind to nonlinearity, it is needful to test these cutoffs in a simulation study to see if there is any merit to them in terms of detecting nonlinear related misspecifications. The authors suggest not only cutoffs for single fit measures, but also for specific combinations of fit measures. This latter usage seems to be less commonly used and was not focused on as part of this thesis.

Kutner, M. H., Nachtsheim, C., & Neter, J. (2004). *Applied linear regression models*. Boston, MA: McGraw-Hill/Irwin.

This text is an excellent introductory work on applied linear regression. The authors present particularly useful discussions of some of the statistics discussed in this thesis, including the F test and the AIC and BIC.

Mooijaart, A., & Satorra, A. (2009). On insensitivity of the chi-square model test to nonlinear misspecification in structural equation models. *Psychometrika*, *74*(3), 443-455. https://doi.org/10.1007/s11336-009-9112-5

This is one of the most important articles relating to my thesis. Here Mooijaart and Satorra explain why fit statistics are essentially useless when applied to nonlinear latent variable modeling. They are all extensions of the χ^2 statistic, which is blind to the sort of nonnormality that results in nonlinear situations. This article shows mathematically that the χ^2 statistic is blind to nonlinearity, which makes me think that there will be no helpful pattern in the fit statistics.

Wang, J., & Wang, X. (2012). *Structural Equation Modeling: Applications Using Mplus*. Hoboken, NJ: Wiley.

This text covers many related topics in SEM and how to do them in Mplus. This text also served as an introduction to the traditional meausres described in the article portion of this thesis. The general procedure for how to use these measures was also taken from this

book, though the cutoffs were Hu and Bentler's contribution. In terms of nonlinear estimation, this text only mentions that Mplus is capable of estimating these models. It does not delve into a discussion of estimation approaches or any related topic. No mention is made of the traditional measures' blindness to nonlinearity.

Alternative fit measures. Little research has been conducted to see what other options there are for fit statistics to aid in model building. Many researchers simply decide which model to accept based on substantive theory, or they try the two-step comparison approach. The homoscedastic fit index (HFI) was developed specifically to detect omitted nonlinear terms. Bayesian fit measures also exist which are not built on the same assumptions as the frequentist fit statistics, though they too have troubles that arise when estimating nonlinear models. Posterior predictive *p*-values (PPP) are Bayesian fit statistics that are easy to implement and adapt.

Asparouhov, T., & Muthén, B. (2017). *Prior-posterior predictive p-values*. Retrieved from https://www.statmodel.com/download/PPPP.pdf.

This article helped me understand more about posterior predictive *p*-values. The most interesting thing to come out of this article was the fact that you could use any discrepancy function to calculate a PPP. This concept was combined with the other fit measures to create new kinds of PPPs.

Gerhard, C., Büchner, R. D., Klein, A. G., & Schermelleh-Engel, K. (2017). A fit index to assess model fit and detect omitted terms in nonlinear SEM. *Structural Equation Modeling: A Multidisciplinary Journal*, 24(3), 414-427.

This article develops the homoscedastic fit index, an index applicable in either regression or structural equation modeling, and specifically recommended cutoffs. The HFI is just a transformation of the hhet statistic (Klein et al., 2016). It also covers simulations testing

the HFI. It is included in the article for testing outside of the LMS method, which was used in this article. The size of the interaction terms they tried in this article seem large compared to the effect sizes that were given in Cohen (1988) and Aguinis, Beaty, Boik, and Pierce (2005). An unofficial, yet preliminary analysis adds doubt as to the effectiveness of the statistic with smaller effect sizes. Another important concept from this article is the fact that, as a 2017 article, they point out that there is very little research on proposing fit in this context.

Kruschke, J. (2014). *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan.* New York, NY: Academic Press.

This book explains posterior predictive checks and how to do them. It is a good introduction into Bayes used in a non-SEM context and helps to more fully comprehend Bayes. It also provides examples of many Bayesian analyses in R.

Maslowsky, J., Jager, J., & Hemken, D. (2015). Estimating and interpreting latent variable interactions: A tutorial for applying the latent moderated structural equations method. *International Journal of Behavioral Development*, *39*(1), 87-96.

This article outlines what I have termed the two-step comparison approach to model fitting as the authors gave it no name. In the first step, a linear model is fit to the data. If good fit is proposeed using χ^2 based fit statistics, then a model including the nonlinear term is fit. In the second step, the two are then compared using a LRT.

Merkle, E., Furr, D., & Rabe-Hesketh, S. (2018). Bayesian model assessment: Use of conditional vs marginal likelihoods. Retrieved from https://arxiv.org/abs/1802.04452.

When estimating Bayesian fit measures, it is necessary to integrate out latent terms. Unfortunately, most software does not do this, and the software that does do it will not run nonlinear Bayesian models. For this reason, most Bayesian fit measures were not used in this study, though future research should explore them.

Mooijaart, A., & Bentler, P. M. (2010). An alternative approach for nonlinear latent variable models. *Structural Equation Modeling: A Multidisciplinary Journal*, *17*(3), 357-373.

In this article Mooijaart and Bentler build on the work of Mooijaart & Satorra (2009). In the previous article the authors pointed out that when only the covariances were used to estimate the model, then tests based on the χ^2 will be blind to non-linearity. In this article the authors try a method of moments approach that compares well to a maximum likelihood approach. They also explain that the LRT the χ^2test is built on is not well understood when comparing a linear model with a nonlinear model.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., & Van Der Linde, A. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society: Series B (Statistical Methodology), 64*(4), 583-639.

This article introduces the deviance information criterion (DIC). The conceptual definition is especially useful, since the DIC does not have the same distributional assumptions as the χ^2-based tests, I think it might be able to detect misspecification in regard to nonlinear terms. The DIC was not used in this article because there is currently no software available that both integrates out the latent variables as appropriate and estimates nonlinear models.

Vuong, Q. (1989). Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica: Journal of the Econometric Society 57*(2), 307-333.

The Vuong LRT can be used for nested and non-nested models. It was used here to compare the PI models.

Estimation Approaches

Many methods exist to estimate nonlinear models. The numerous methods generally fit into categories of product indicator approaches, distributional approaches, method of moment approaches, and a Bayesian approach. Many comparisons have been done to demonstrate that each of these types of approaches excels in one area or another. No one approach has been shown to be the best, though the most frequently used appear to be the Extended Unconstrained approach, the Latent Moderated Structural Equation Modeling approach, and the Bayesian approach.

Bayarri, M. J., & Berger, J. O. (2004). The interplay of Bayesian and frequentist analysis. *Statistical Science*, *19*(1), 58-80.

This article is a resource summarizing the Bayesian versus frequentist debate. Essentially, Bayesian statistics are more flexible because they are not built on previous assumptions, they are more accurate with large and small sample sizes, and they allow the incorporation of previous knowledge. At the same time, the incorporation of a prior distribution is a double-edged sword, as it has the potential to bias the results.

Dimitruk, P., Schermelleh-Engel, K., Kelava, A., & Moosbrugger, H. (2007). Challenges in nonlinear structural equation modeling. *Methodology: European Journal of Research Methods for the Behavioral and Social Sciences*, *3*(3), 100-114.

This article builds off of the Moosbrugger, Schermelleh-Engel, and Klein (1997) article by reiterating some of their points and running a simulation study demonstrating the advantages of the LMS and QML methods.

Gelman, A., Stern, H. S., Carlin, J. B., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). *Bayesian data analysis.* New York, NY: Chapman and Hall/CRC.

This book covers many Bayesian data analyses, though it does not include SEM. I used it in my paper primarily to explain how PPP and posterior predictive checks are done. Notably, one difference between Kruschke's work and this book is the exclusion of *p*-value in Kruschke's work. While this is outside the scope of the research in this thesis, I think it is important to explain that Kruschke rejects the idea of doing something to resemble NHST while this work does not hesitate to go that direction.

Harring, J. R., Weiss, B. A., & Hsu, J. C. (2012). A comparison of methods for estimating quadratic effects in nonlinear structural equation models. *Psychological Methods*, *17*(2), 193-214.

This was one of the more recent comparisons between estimation methods, and it focused on estimating latent quadratic terms rather than the latent interactions of the other comparison studies I looked at. The authors compared the two-stage moderated regression approach (latent variable scores), the unconstrained approach, the LMS approach, the Bayesian approach, and the marginal maximum likelihood approach, they found that the models were comparable in most aspects. The determination of which estimation procedure to use may ultimately be decided by the complexity of the model as opposed to any other criteria, with the Bayesian approach and latent variables scores approach handling the most general models, and the LMS the least. It should be noted that this inflexibility is in part what led to the creation of the QML. The latent variable scores approach performed less well than the other methods.

Jöreskog, K. G., & Yang, F. (1996). Nonlinear structural equation models: The Kenny-Judd model with interaction effects. In G. Marcoulides & R. Schumacker (Eds.), *Advanced* *structural equation modeling: Issues and techniques*, (pp. 57-88). Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

In this chapter, Joreskog and Yang propose some problems with the Kenny and Judd model. Particularly, that the intercepts cannot be thought of as zero. They also provide a simpler model than Kenny and Judd and demonstrate that the estimates are unbiased and recoverable.

Kelava, A., & Nagengast, B. (2012). A Bayesian model for the estimation of latent interaction and quadratic effects when latent variables are non-normally distributed. *Multivariate Behavioral Research, 47*(5), 717-742.

This article covers how to do the Bayesian procedure in the case of multiple nonlinear terms. They also suggest getting the priors from the estimates generated from the extended unconstrained approach, which they also develop in this article. This is a useful article that will help me implement the approaches I will be using.

Kenny, D. A., & Judd, C. M. (1984). Estimating the nonlinear and interactive effects of latent variables. *Psychological Bulletin*, *96*(1), 201-210.

Kenny and Judd are the first to propose a solution to the problem of how to deal with nonlinear estimation in latent variable modeling. Interactions and quadratic terms in this situation lead to severe non-normality. They propose a model for estimation that results in unbiased estimators. This model, though not perfect, set the stage for dozens of other models and decades of discussion. This model is the first of the product indicator approaches.

Klein, A. G., & Moosbrugger, H. (2000). Maximum likelihood estimation of latent interaction effects with the LMS method. *Psychometrika*, *65*(4), 457-474.

The authors develop a more general interaction model that allows multiple nonlinear terms to be estimated by the LMS method. Not incredibly important in regards to my thesis, though it shows more on the development of the LMS approach.

Klein, A. G., & Muthén, B. O. (2007). Quasi-maximum likelihood estimation of structural equation models with multiple interaction and quadratic effects. *Multivariate Behavioral Research*, *42*(4), 647-673.

This article describes the development of a Quasi-Maximum Likelihood (QML) approach for estimating multiple nonlinear terms in SEM. The authors attempt to replicate Marsh et al.'s (2004) study comparing the QML to other methods. They were able to replicate the positive results, but not the strong biases that Marsh et al reported.

Lee, S. Y., Song, X. Y., & Tang, N. S. (2007). Bayesian methods for analyzing structural equation models with covariates, interaction, and quadratic latent variables. *Structural Equation Modeling: A Multidisciplinary Journal, 14*(3), 404-434.

This article explains the Bayesian approach. Harring et al. (2012) found this model to be preferable to other models in terms of estimation. The quick references to the Bayes Factor and the deviance information criterion give me hope that these two fit statistics will perform well under all conditions.

Little, T. D., Bovaird, J. A., & Widaman, K. F. (2006). On the merits of orthogonalizing powered and product terms: Implications for modeling interactions among latent variables. *Structural Equation Modeling: A Multidisciplinary Journal*, *13*(4), 497-519. http://dx.doi.org/10.1207/s15328007sem1304_1

This article talks about another method for estimation. It is only referenced as an alternative estimation strategy in this paper.

Marsh, H. W., Wen, Z., & Hau, K. T. (2004). Structural equation models of latent interactions: Evaluation of alternative estimation strategies and indicator construction. *Psychological Methods*, *9*(3), 275-300.

This article is one of the more prominent comparisons between methods. It compares the constrained approach, the two-stage least squares, the generalized appended product indicator approach, the QML, and the unconstrained approaches. They develop the unconstrained approach in this article, and found it to be easier to implement than the others. It is also in this article where it is claimed that the central limit theorem does not help with the normality issue of the latent interactions. The unconstrained approach is robust to this problem though, because no constraints are placed based on the assumption of normality.

Moosbrugger, H., Schermelleh-Engel, K., Kelava, A., & Klein, A. G. (2009). Testing multiple nonlinear effects in structural equation modeling: A comparison of alternative estimation approaches. In T. Teo & M. Khine (Eds.), *Structural equation modelling in educational research: Concepts and applications* (pp. 103-136). Rotterdam, NL: Sense Publishers.

This is another article in support of the QML and LMS approaches. The idea here though is that the QML, while very similar to the LMS method, is less computationally intensive and more flexible. They compared the LMS, QML, extended constrained, and extended unconstrained approaches and found the LMS and QML to be much more efficient and to have more accurate standard errors than the other approaches.

Moosbrugger, H., Schermelleh-Engel, K., & Klein, A. (1997). Methodological problems of estimating latent interaction effects. *Methods of Psychological Research Online, 2*(2), 95-111.

These authors outline what they claim are the five issues that impede estimation of latent nonlinear models:

1. Measurement error of indicator variables
2. Nonlinearity of parameters
3. Mean structures
4. Variable transformation problems
5. Non-normality of variables

The authors also make the case that the latent moderated structural equations (LMS) approach is the best estimation method, as it takes the non-normality of the interaction term explicitly into account. It compares the LMS to other models on a theoretical basis.

Moulder, B. C., & Algina, J. (2002). Comparison of methods for estimating and testing latent variable interactions. *Structural Equation Modeling*, *9*(1), 1-19.

This article compares a number of product indicator approaches, including the authors' variation of Joreskog and Yang's procedure. This is later developed further into the unconstrained approach. More importantly, this article compares these methods using the χ^2-based fit statistics since they were all product indicator approaches. This is an inappropriate activity according to Mooijaart and Satorra (2009). I wonder if we wouldn't see more inappropriate use of χ^2 based fit indices if the other approaches also gave us these fit statistics. More importantly, is this portion of the evidence Algina and Moulder cite to support their findings valid? The authors provide other evidence to support their conclusions, so if their method is not mathematically valid, why were their results consistent?

Muthén, B. O., & Asparouhov, T. (2012). Bayesian structural equation modeling: A more flexible representation of substantive theory. *Psychological Methods*, *17*(3), 313-335.

This article has a succinct explanation of the advantages of Bayesian analysis. Bayesian analyses do not rely on normality or large sample sizes, have more information available, can be less computationally demanding, and enable new questions to be explored. The advantages of the Bayesian methodologies suggest that this approach might be the most accurate for estimation of latent nonlinear terms. Harring, Weiss, and Hsu (2012) found the Bayesian approach to achieve more accurate estimates and to handle more general models than other approaches.

Ping, R. A., Jr. (1995). A parsimonious estimating technique for interaction and quadratic latent variables. *Journal of Marketing Research 32*(3), 336-347.

Ping establishes a two-stage least squares. Later research found that this method was plagued with many of the problems of the other product indicator approaches; particularly, unreliable standard error estimates.

Wall, M. M., & Amemiya, Y. (2003). A method of moments technique for fitting interaction effects in structural equation models. *British Journal of Mathematical and Statistical Psychology*, *56*(1), 47-63

In this article the authors give a summary of method of moment approaches and demonstrate the advantages of the two-stage method of moments approach.

Simulation and Coding Details

This thesis stands on the shoulders of giants. While the vast majority of the coding was done personally, many aspects were borrowed or adapted from other articles. The design details were in large part taken from other sources as well. A number of softwares were used in this simulation study including a significant number of R packages and the program JAGS. The data were simulated on BYU's Mary Lou Fulton Supercomputer. Other options for the analysis were considered and are included here as a reference. Additionally, some software was under development at the time this thesis was completed that might make future efforts simpler.

Bandalos, D. L., & Leite, W. L. (2006). The use of Monte Carlo studies in structural equation modeling research. In G. Hancock & R. Mueller (Eds.), S*tructural equation modeling: A second course* (p. 385- 426). Greenwich, CT: Information Age.

This research suggests that 500 replications are sufficient for large simulations when it comes to SEM simulation studies. This chapter helped me decide to do 500 iterations instead of the 1000 iterations I had been considering previously. When certain conditions failed to reach the appropriate number of estimated conditions, this standard encouraged me to run more.

Denwood, M. J. (2016). runjags: An R package providing interface utilities, model templates, parallel computing methods and additional distributions for MCMC models in JAGS. *Journal of Statistical Software, 71*(9), 1-25.

This package was used to interface with JAGS for the nonlinear Bayesian models. This package does not output the corrected fit measures for Bayesian analysis, but blavaan does. However, blavaan cannot yet implement nonlinear models.

Fan, X., Thompson, B., & Wang, L. (1999). Effects of sample size, estimation methods, and model specification on structural equation modeling fit indexes. *Structural Equation Modeling: A Multidisciplinary Journal*, *6*(1), 56-83.

This article is an example of a simulation study testing the effects of varying conditions on χ^2-based fit statistics. They suggest investigating various levels of model misspecification in future research on fit indexes.

Merkle, E., You, D., Schneider, L., Bae, S., & Merkle, M. E. (2018). Package ‘nonnest2’. Retrieved from https://cran.r-project.org/web/packages/nonnest2/nonnest2.pdf.

This package was used to calculate the Vuong LRT for nonnested models. This was done between all possible combinations of linear, interaction, and full models in the product indicator condition.

Plummer, M. (2003). Jags: A program for analysis of Bayesian graphical models using Gibbs sampling. Retreived from http://citeseer.ist.psu.edu/plummer03jags.html.

JAGS is software used for estimating Bayesian models. It uses the same syntax as BUGS and can be confusing to use. Some packages in R allow for an easier model specification into JAGS.

R Core Team (2013). R: A language and environment for statistical computing. Retrieved from http://www.R-project.org/.

R is a free, open-source software for statistical computing. A number of R packages were used in the simulation and analysis. The entire study was done in R except for the estimation of the Bayesian models, which were estimated in JAGS.

Rosseel, Y. (2012). lavaan: An R package for structural equation modeling. *Journal of Statistical Software, 48*(2), 1-36. Retrieved from http://www.jstatsoft.org/v48/i02/.

The product indicator approaches were estimated using lavaan, which is an excellent package in R for structural equation modeling.

RStudio Team (2015). *RStudio: Integrated Development for R*. Retrieved from http://www.rstudio.com/.

Rstudio is an integrated development environment (IDE) that is convenient for data analysis. It was used when writing the simulation code, testing it, and analyzing the data.

Umbach, N., Naumann, K., Brandt, H., & Kelava, A. (2017). Fitting nonlinear structural equation models in R with package nlsem. *Journal of Statistical Software*, *77*(1), 1-20.

This article introduces the package nlsem. This package was used in my thesis to simulate the data, then to analyze it using the QML approach. This package is capable of QML, LMS, and NSEMM approaches.

Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis*. New York, NY: Springer-Verlag New York, Inc.

The ggplot package was used to create the boxplot that is contained in the thesis as well as all other plots that were used in analysis but not contained in the text. This package creates nice images easily.

APPENDIX B

Replication Code

Data Collection Code

```
Start = Sys.time()

library(runjags)

library(nlsem)

library(lavaan)

library(blavaan)

library(Matrix)

library(psych)

runjags.options(jagspath = "/fslhome/pip89/opt/jags/4.3.0/bin/jags")

condition <-as.integer(Sys.getenv("SLURM_ARRAY_TASK_ID"))

model <- as.integer(Sys.getenv("SLURM_PROCID"))

filename <- paste(condition,model, "csv", sep = ".")

directory <- file.path(Sys.getenv("SLURM_ARRAY_JOB_ID"))

dir.create(directory, showWarnings=FALSE, recursive=TRUE)

fullpath <- file.path(directory, filename)

source("/fslhome/pip89/compute/ZZZ/all_Functions_ML.r")

if(is.na(condition)){

condition=0

model=0

directory = 'test'

dir.create(directory, showWarnings=FALSE, recursive=TRUE)
```

```
fullpath <- file.path(directory, filename)
}

condition = condition + 1
# Variables manipulated
es = c(.02, .13, .26, .4)
rxy = c(.45, .65, .85)
terms = c(0,1,3)
ss = c(100, 500, 1000)
conditionMarker = expand.grid(es,rxy,terms,ss)
colnames(conditionMarker) = c("es", "rxy", "terms", "ss")
conditionMarker$condie = 1:nrow(conditionMarker)
paste0("The condition is: ", condition)
paste0("The model is: ", model)
ct = condition %% 36
# ct = ifelse(condition > nrow(conditionMarker) & condition <= nrow(conditionMarker)*2,
#           condition -nrow(conditionMarker),
#           ifelse(condition > nrow(conditionMarker)*2 & condition <= nrow(conditionMarker)*3,
#                 condition -nrow(conditionMarker)*2,
#                 ifelse(condition > nrow(conditionMarker)*3 & condition <=
nrow(conditionMarker)*4,
#                       condition -nrow(conditionMarker)*3,
#                       ifelse(condition > nrow(conditionMarker)*4 & condition <=
nrow(conditionMarker)*5,
#                               condition -nrow(conditionMarker)*4,
#                               condition
#                       ))))
# How many iterations?
iters = 3 * 3


# Create a place for the results
summaryTable = data.frame(method= rep(NA, iters*8), converged= NA,
                       terms= NA, seed= NA,
                       time= NA, likl= NA,
                       aic= NA, bic= NA,
                       chisq= NA, chisqp= NA,
                       rmsea= NA, srmr= NA,
                       hfi= NA, hfir= NA,
                       R2= NA, R2r= NA,
                       hfi0= NA, hfi1= NA,
                       hfi3= NA, PPP_likl= NA,
                       PPP_aic= NA, PPP_bic= NA,
                       PPP_chisq= NA, PPP_chisqp= NA,
                       PPP_rmsea= NA, PPP_srmr= NA,
                       PPP_hfi= NA, PPP_hfir= NA,
                       PPP_R2= NA, PPP_R2r= NA, PPP_cfi = NA, PPP_tli = NA, PPP_rmseap=NA,
PPP_logl = NA,
                       es= NA, ss= NA, rxy= NA,lambda.x2= NA,
                       lambda.x3= NA, lambda.x5= NA,
                       lambda.x6= NA, lambda.y2= NA,
                       lambda.y3= NA, nu.x1= NA,
                       nu.x2= NA, nu.x3= NA,
                       nu.x4= NA, nu.x5= NA,
                       nu.x6= NA, nu.y1= NA, nu.y2= NA,
                       nu.y3= NA, phi1= NA,
                       phi2= NA, phi4= NA,
                       psi= NA, gamma1= NA, gamma2= NA,
                       omegai= NA, omega1= NA,
                       omega2= NA, theta.y1= NA, theta.y2= NA,
                       theta.y3= NA, theta.x1= NA,
                       theta.x2= NA, theta.x3= NA,
                       theta.x4= NA, theta.x5= NA,
                       theta.x6= NA)

# Set up the clock checker
v = 1
# Set up the iteration loop
for(t in 1:iters){
  print(t)
ct = ifelse(t %% 3 == 0, condition %% 36 + 72, ifelse(t %% 3 == 2, condition %% 36 +
36,  condition %% 36))
print(ct)
  # Simulate the data
  offset =as.numeric(paste0(condition,model,t, 999))
  set.seed = offset



  # Set the parameters


  es =conditionMarker$es[conditionMarker$condie==ct]
  ss = conditionMarker$ss[conditionMarker$condie==ct]
  rxy = conditionMarker$rxy[conditionMarker$condie==ct]
  terms = conditionMarker$terms[conditionMarker$condie==ct]
  intsim = ifelse(terms == 3, "eta1~xi1:xi2,eta1~xi1:xi1,eta1~xi2:xi2", "eta1~xi1:xi2")

  if(terms == 3){
    #es/2 because of relationship between correlation and omega. This is only true when there is
only one omega.
    meg = matrix(c(es/2,0,es,es/2),nrow=2)
    gamma = c(.3,.3)
  }else if(terms==1){
```

```
  meg = matrix(c(0,0,es,0),nrow=2)
  gamma = c(.3,.3)

 }else{
  meg = matrix(c(0,0,0.000000000000000000000000001,0),nrow=2)
  gamma = c(.3,.3)
 }
 theta = matrix(0, ncol = 9, nrow = 9)
 diag(theta) = NA



 data = simulate_nlsem(N = ss, m = 1, interaction_simulation = intsim, gamma = gamma,
Omega = meg,
            covariance = 0, rel_x = rep(rxy,6), rel_y = rep(rxy,3), num.x_sim = 6,
            num.y_sim = 3, num.xi_sim = 2, xi_sim = "x1-x3,x4-x6", eta_sim = "y1-y3")
 dat = data.frame(data[,,1])

 colnames(dat) = c(paste0("X", 1:6), paste0("Y",1:3))
 db = list("X1" = dat$X1, "X2" = dat$X1,"X3" = dat$X1,
      "X4" = dat$X1, "X5" = dat$X1, "X6" = dat$X1
      ,"Y1" = dat$X1, "Y2" = dat$X1, "Y3" = dat$X1,
      N = ss,
      alpha = matrix(NA, nrow=3, ncol=1),
          omega = matrix(c(0,0,NA, 0,0,NA), ncol=2),
          lambda = matrix(c(rep(NA, 3), rep(0,9), rep(NA,3), rep(0,9),rep(NA,3)), ncol=3),
          nu = matrix(rep(NA, 9), ncol=1),
          psi = matrix(c(NA, 0,0, NA, NA, 0, 0, 0, NA), ncol=3),
          theta = theta,
          omega = matrix(c(0, NA, 0, 0), nrow=2))


  monitors = c("lambda[1,1]", "lambda[2,1]", "lambda[3,1]", "lambda[4,2]", "lambda[5,2]",
"lambda[6,2]", "lambda[7,3]",
            "lambda[8,3]", "lambda[9,3]", "omega[3,1]",   "omega[3,2]",   "theta[1,1]",
"theta[2,2]",  "theta[3,3]",
            "theta[4,4]",  "theta[5,5]" , "theta[6,6]",  "theta[7,7]",  "theta[8,8]" , "theta[9,9]",
"psi[1,1]" ,
            "psi[2,2]",    "psi[3,3]" ,   "psi[1,2]" ,   "nu[1,1]" ,    "nu[2,1]",     "nu[3,1]" ,
"nu[4,1]" ,
            "nu[5,1]",     "nu[6,1]",     "nu[7,1]" ,    "nu[8,1]" ,    "nu[9,1]",     "alpha[1,1]",
"alpha[2,1]",
            "alpha[3,1]", "omega[2,1]", "omega[1,1]", "omega[2,2]")
  # Estimate the variable


   # mlin, blin
   # mlin
   mod <- '
```

```
KSI1 =~ 1*X1 + X2 + X3

KSI2 =~ 1*X4 + X5 + X6

ETA =~ 1*Y1 + Y2 + Y3

Y1 ~ 1

Y2 ~ 1

Y3 ~ 1

Y1 ~~ Y1

Y2 ~~ Y2

Y3 ~~ Y3

X1 ~~ X1

X2 ~~ X2

X3 ~~ X3

X4 ~~ X4

X5 ~~ X5

X6 ~~ X6

ETA ~ KSI1

ETA ~ KSI2

KSI1 ~~ KSI1

KSI1 ~~ KSI2

KSI2 ~~ KSI2

ETA ~~ ETA

'
```

```
a = Sys.time()
mlin <- lavaan::sem(mod, data=dat, control = list(iter.max=3000))
b = Sys.time()

summaryTable$method[v] = c("ML")
summaryTable$converged[v] = mlin@optim$converged*1
summaryTable$es[v] = c(es)
summaryTable$ss[v] = c(ss)
summaryTable$rxy[v] = c(rxy)
summaryTable$terms[v] = c(terms)
summaryTable$seed[v] = offset
summaryTable$time[v] = as.numeric(difftime(b, a, units = "secs"))

 summaryTable$lambda.x2[v] = coef(mlin)[grep("KSI1=~X2", names(coef(mlin)))]
 summaryTable$lambda.x3[v] = coef(mlin)[grep("KSI1=~X3", names(coef(mlin)))]
 summaryTable$lambda.x5[v] = coef(mlin)[grep("KSI2=~X5", names(coef(mlin)))]
 summaryTable$lambda.x6[v] = coef(mlin)[grep("KSI2=~X6", names(coef(mlin)))]
 summaryTable$lambda.y2[v] = coef(mlin)[grep("ETA=~Y2", names(coef(mlin)))]
 summaryTable$lambda.y3[v] = coef(mlin)[grep("ETA=~Y3", names(coef(mlin)))]

 summaryTable$nu.x1[v] = coef(mlin)[grep("X1~1", names(coef(mlin)))]
 summaryTable$nu.x2[v] = coef(mlin)[grep("X2~1", names(coef(mlin)))]
 summaryTable$nu.x3[v] = coef(mlin)[grep("X3~1", names(coef(mlin)))]
```

```
summaryTable$nu.x4[v] = coef(mlin)[grep("X4~1", names(coef(mlin)))]
summaryTable$nu.x5[v] = coef(mlin)[grep("X5~1", names(coef(mlin)))]
summaryTable$nu.x6[v] = coef(mlin)[grep("X6~1", names(coef(mlin)))]
summaryTable$nu.y1[v] = coef(mlin)[grep("Y1~1", names(coef(mlin)))]
summaryTable$nu.y2[v] = coef(mlin)[grep("Y2~1", names(coef(mlin)))]
summaryTable$nu.y3[v] = coef(mlin)[grep("Y3~1", names(coef(mlin)))]

summaryTable$theta.x1[v] = coef(mlin)[grep("X1~~X1", names(coef(mlin)))]
summaryTable$theta.x2[v] = coef(mlin)[grep("X2~~X2", names(coef(mlin)))]
summaryTable$theta.x3[v] = coef(mlin)[grep("X3~~X3", names(coef(mlin)))]
summaryTable$theta.x4[v] = coef(mlin)[grep("X4~~X4", names(coef(mlin)))]
summaryTable$theta.x5[v] = coef(mlin)[grep("X5~~X5", names(coef(mlin)))]
summaryTable$theta.x6[v] = coef(mlin)[grep("X6~~X6", names(coef(mlin)))]
summaryTable$theta.y1[v] = coef(mlin)[grep("Y1~~Y1", names(coef(mlin)))]
summaryTable$theta.y2[v] = coef(mlin)[grep("Y2~~Y2", names(coef(mlin)))]
summaryTable$theta.y3[v] = coef(mlin)[grep("Y3~~Y3", names(coef(mlin)))]

summaryTable$gamma1[v] = coef(mlin)[grep("ETA~KSI1", names(coef(mlin)))]
summaryTable$gamma2[v] = coef(mlin)[grep("ETA~KSI2", names(coef(mlin)))]

summaryTable$phi1[v] = coef(mlin)[grep("KSI1~~KSI1", names(coef(mlin)))]
#phi2 is the covariance
summaryTable$phi2[v] = coef(mlin)[grep("KSI1~~KSI2", names(coef(mlin)))]
```

```
#phi4 is the var of 2
summaryTable$phi4[v] = coef(mlin)[grep("KSI2~~KSI2", names(coef(mlin)))]
summaryTable$psi[v] = coef(mlin)[grep("ETA~~ETA", names(coef(mlin)))]
summaryTable$omegai[v] = 0
summaryTable$omega1[v] = 0
summaryTable$omega2[v] = 0

fit = establishFit(obs.data = dat[1:9], model.obj = mlin, m = 300, nfac = 0, model = mod, df =
fitMeasures(mlin)[4])
summaryTable$likl[v] = fit[1]
summaryTable$aic[v] = fit[2]
summaryTable$bic[v] = fit[3]
summaryTable$chisq[v] = fit[4]
summaryTable$chisqp[v] = fit[5]
summaryTable$rmsea[v] = fit[6]
summaryTable$srmr[v] = fit[7]
summaryTable$hfi[v] = fit[8]
summaryTable$hfir[v] = fit[9]
summaryTable$R2[v] = fit[10]
summaryTable$R2r[v] = fit[11]
summaryTable$hfi0[v] = fit[12]
summaryTable$hfi1[v] = fit[13]
summaryTable$hfi3[v] = fit[14]
```

```
 summaryTable$PPP_likl[v] = fit[15]
 summaryTable$PPP_aic[v] = fit[16]
 summaryTable$PPP_bic[v] = fit[17]
 summaryTable$PPP_chisq[v] = fit[18]
 summaryTable$PPP_chisqp[v] = fit[19]
 summaryTable$PPP_rmsea[v] = fit[20]
 summaryTable$PPP_srmr[v] = fit[21]
 summaryTable$PPP_hfi[v] = fit[22]
 summaryTable$PPP_hfir[v] = fit[23]
 summaryTable$PPP_R2r[v] = fit[24]
 summaryTable$PPP_R2[v] = fit[25]
 summaryTable$PPP_logl[v] = fit[26]
 summaryTable$PPP_cfi[v] = fit[27]
 summaryTable$PPP_tli[v] = fit[28]
 summaryTable$PPP_rmseap[v] = fit[29]

v = v+1
write.csv(summaryTable, file = fullpath, row.names=F)


 # blin
 a = Sys.time()
 blin <- bsem(mod, data=dat, n.chains = 3, inits = parameterEstimates(mlin))
```

```
    #load("~/Desktop/OmegaFolder/semjags.rda")
    #blin <- run.jags("sem.jag", monitor = jagtrans$monitors, data = jagtrans$data, inits =
parameterEstimates(mlin))
    b = Sys.time()

    summaryTable$method[v] = "blin"
    summaryTable$converged[v] = mean(na.omit(blin@ParTable$psrf<1.005)*1)
    summaryTable$es[v] = c(es)
    summaryTable$ss[v] = c(ss)
    summaryTable$rxy[v] = c(rxy)
    summaryTable$terms[v] = c(terms)
    summaryTable$seed[v] = offset
    summaryTable$time[v] = as.numeric(difftime(b, a, units = "secs"))

     summaryTable$lambda.x2[v] = coef(blin)[grep("KSI1=~X2", names(coef(blin)))]
     summaryTable$lambda.x3[v] = coef(blin)[grep("KSI1=~X3", names(coef(blin)))]
     summaryTable$lambda.x5[v] = coef(blin)[grep("KSI2=~X5", names(coef(blin)))]
     summaryTable$lambda.x6[v] = coef(blin)[grep("KSI2=~X6", names(coef(blin)))]
     summaryTable$lambda.y2[v] = coef(blin)[grep("ETA=~Y2", names(coef(blin)))]
     summaryTable$lambda.y3[v] = coef(blin)[grep("ETA=~Y3", names(coef(blin)))]

     summaryTable$nu.x1[v] = coef(blin)[grep("X1~1", names(coef(blin)))]
     summaryTable$nu.x2[v] = coef(blin)[grep("X2~1", names(coef(blin)))]
```

```
summaryTable$nu.x3[v] = coef(blin)[grep("X3~1", names(coef(blin)))]

summaryTable$nu.x4[v] = coef(blin)[grep("X4~1", names(coef(blin)))]

summaryTable$nu.x5[v] = coef(blin)[grep("X5~1", names(coef(blin)))]

summaryTable$nu.x6[v] = coef(blin)[grep("X6~1", names(coef(blin)))]

summaryTable$nu.y1[v] = coef(blin)[grep("Y1~1", names(coef(blin)))]

summaryTable$nu.y2[v] = coef(blin)[grep("Y2~1", names(coef(blin)))]

summaryTable$nu.y3[v] = coef(blin)[grep("Y3~1", names(coef(blin)))]


summaryTable$theta.x1[v] = coef(blin)[grep("X1~~X1", names(coef(blin)))]

summaryTable$theta.x2[v] = coef(blin)[grep("X2~~X2", names(coef(blin)))]

summaryTable$theta.x3[v] = coef(blin)[grep("X3~~X3", names(coef(blin)))]

summaryTable$theta.x4[v] = coef(blin)[grep("X4~~X4", names(coef(blin)))]

summaryTable$theta.x5[v] = coef(blin)[grep("X5~~X5", names(coef(blin)))]

summaryTable$theta.x6[v] = coef(blin)[grep("X6~~X6", names(coef(blin)))]

summaryTable$theta.y1[v] = coef(blin)[grep("Y1~~Y1", names(coef(blin)))]

summaryTable$theta.y2[v] = coef(blin)[grep("Y2~~Y2", names(coef(blin)))]

summaryTable$theta.y3[v] = coef(blin)[grep("Y3~~Y3", names(coef(blin)))]


summaryTable$gamma1[v] = coef(blin)[grep("ETA~KSI1", names(coef(blin)))]

summaryTable$gamma2[v] = coef(blin)[grep("ETA~KSI2", names(coef(blin)))]


summaryTable$phi1[v] = coef(blin)[grep("KSI1~~KSI1", names(coef(blin)))]

#phi2 is the covariance
```

```
summaryTable$phi2[v] = coef(blin)[grep("KSI1~~KSI2", names(coef(blin)))]

#phi4 is the var of 2

summaryTable$phi4[v] = coef(blin)[grep("KSI2~~KSI2", names(coef(blin)))]

summaryTable$psi[v] = coef(blin)[grep("ETA~~ETA", names(coef(blin)))]

summaryTable$omegai[v] = 0

summaryTable$omega1[v] = 0

summaryTable$omega2[v] = 0


fit = establishFit(obs.data = dat[1:9], model.obj = blin, m = 300, nfac = 0, model = mod)

summaryTable$likl[v] = fit[1]

summaryTable$aic[v] = fit[2]

summaryTable$bic[v] = fit[3]

summaryTable$chisq[v] = fit[4]

summaryTable$chisqp[v] = fit[5]

summaryTable$rmsea[v] = fit[6]

summaryTable$srmr[v] = fit[7]

summaryTable$hfi[v] = fit[8]

summaryTable$hfir[v] = fit[9]

summaryTable$R2[v] = fit[10]

summaryTable$R2r[v] = fit[11]

summaryTable$hfi0[v] = fit[12]

summaryTable$hfi1[v] = fit[13]

summaryTable$hfi3[v] = fit[14]
```

```
    summaryTable$PPP_likl[v] = fit[15]

    summaryTable$PPP_aic[v] = fit[16]

    summaryTable$PPP_bic[v] = fit[17]

    summaryTable$PPP_chisq[v] = fit[18]

    summaryTable$PPP_chisqp[v] = fit[19]

    summaryTable$PPP_rmsea[v] = fit[20]

    summaryTable$PPP_srmr[v] = fit[21]

    summaryTable$PPP_hfi[v] = fit[22]

    summaryTable$PPP_hfir[v] = fit[23]

    summaryTable$PPP_R2r[v] = fit[24]

    summaryTable$PPP_R2[v] = fit[25]


  v = v+1


  write.csv(summaryTable, file = fullpath, row.names=F)



# qml3, ExUC, bnl3

# Extended Unconstrained approach

dat$X1X4 <- dat$X1 * dat$X4

dat$X2X5 <- dat$X2 * dat$X5

dat$X3X6 <- dat$X3 * dat$X6

dat$X1X1 <- dat$X1 * dat$X1
```

```
dat$X2X2 <- dat$X2 * dat$X2

dat$X3X3 <- dat$X3 * dat$X3

dat$X4X4 <- dat$X4 * dat$X4

dat$X5X5 <- dat$X5 * dat$X5

dat$X6X6 <- dat$X6 * dat$X6


mod.unc <- '

KSI1 =~ 1*X1 + X2 + X3

KSI2 =~ 1*X4 + X5 + X6

ETA =~ 1*Y1 + Y2 + Y3

KSI1KSI2 =~ 1*X1X4 + X2X5 + X3X6

KSI1KSI1 =~ 1*X1X1 + X2X2 + X3X3

KSI2KSI2 =~ 1*X4X4 + X5X5 + X6X6

Y1 ~ 1

Y2 ~ 1

Y3 ~ 1

X1X4 ~ 1

X2X5 ~ 1

X3X6 ~ 1

X1X1 ~ 1

X2X2 ~ 1

X3X3 ~ 1

X4X4 ~ 1
```

```
X5X5 ~ 1

X6X6 ~ 1

Y1 ~~ Y1

Y2 ~~ Y2

Y3 ~~ Y3

X1 ~~ X1

X2 ~~ X2

X3 ~~ X3

X4 ~~ X4

X5 ~~ X5

X6 ~~ X6

X1X4 ~~ X1X4

X2X5 ~~ X2X5

X3X6 ~~ X3X6

X1X1 ~~ X1X1

X2X2 ~~ X2X2

X3X3 ~~ X3X3

X4X4 ~~ X4X4

X5X5 ~~ X5X5

X6X6 ~~ X6X6

X1X1 ~~ X1X4

X2X2 ~~ X2X5

X3X3 ~~ X3X6
```

```
X4X4 ~~ X1X4

X5X5 ~~ X2X5

X6X6 ~~ X3X6

ETA ~ KSI1

ETA ~ KSI2

ETA ~ KSI1KSI2

ETA ~ KSI1KSI1

ETA ~ KSI2KSI2

KSI1 ~~ KSI1

KSI1 ~~ KSI2

KSI2 ~~ KSI2

KSI1KSI2 ~~ KSI1KSI2

KSI1KSI1 ~~ KSI1KSI2

KSI1KSI1 ~~ KSI1KSI1

KSI2KSI2 ~~ KSI2KSI2

KSI2KSI2 ~~ KSI1KSI1

KSI2KSI2 ~~ KSI1KSI2

ETA ~~ ETA

'

a = Sys.time()

ExUn <- lavaan::sem(mod.unc, data=dat, control = list(iter.max=30000))

b = Sys.time()
```

```
summaryTable$method[v] = c("ExUC")
summaryTable$converged[v] = ExUn@optim$converged*1
summaryTable$es[v] = c(es)
summaryTable$ss[v] = c(ss)
summaryTable$rxy[v] = c(rxy)
summaryTable$terms[v] = c(terms)
summaryTable$seed[v] = offset
summaryTable$time[v] = as.numeric(difftime(b, a, units = "secs"))
if(ExUn@optim$converged){
  summaryTable$lambda.x2[v] = coef(ExUn)[grep("KSI1=~X2", names(coef(ExUn)))][1]
  summaryTable$lambda.x3[v] = coef(ExUn)[grep("KSI1=~X3", names(coef(ExUn)))][1]
  summaryTable$lambda.x5[v] = coef(ExUn)[grep("KSI2=~X5", names(coef(ExUn)))][1]
  summaryTable$lambda.x6[v] = coef(ExUn)[grep("KSI2=~X6", names(coef(ExUn)))][1]
  summaryTable$lambda.y2[v] = coef(ExUn)[grep("ETA=~Y2", names(coef(ExUn)))][1]
  summaryTable$lambda.y3[v] = coef(ExUn)[grep("ETA=~Y3", names(coef(ExUn)))][1]

  summaryTable$nu.x1[v] = coef(ExUn)[grep("X1~1", names(coef(ExUn)))][1]
  summaryTable$nu.x2[v] = coef(ExUn)[grep("X2~1", names(coef(ExUn)))][1]
  summaryTable$nu.x3[v] = coef(ExUn)[grep("X3~1", names(coef(ExUn)))][1]
  summaryTable$nu.x4[v] = coef(ExUn)[grep("X4~1", names(coef(ExUn)))][1]
  summaryTable$nu.x5[v] = coef(ExUn)[grep("X5~1", names(coef(ExUn)))][1]
  summaryTable$nu.x6[v] = coef(ExUn)[grep("X6~1", names(coef(ExUn)))][1]
  summaryTable$nu.y1[v] = coef(ExUn)[grep("Y1~1", names(coef(ExUn)))][1]
```

```
summaryTable$nu.y2[v] = coef(ExUn)[grep("Y2~1", names(coef(ExUn)))][1]

summaryTable$nu.y3[v] = coef(ExUn)[grep("Y3~1", names(coef(ExUn)))][1]


summaryTable$theta.x1[v] = coef(ExUn)[grep("X1~~X1", names(coef(ExUn)))][1]

summaryTable$theta.x2[v] = coef(ExUn)[grep("X2~~X2", names(coef(ExUn)))][1]

summaryTable$theta.x3[v] = coef(ExUn)[grep("X3~~X3", names(coef(ExUn)))][1]

summaryTable$theta.x4[v] = coef(ExUn)[grep("X4~~X4", names(coef(ExUn)))][1]

summaryTable$theta.x5[v] = coef(ExUn)[grep("X5~~X5", names(coef(ExUn)))][1]

summaryTable$theta.x6[v] = coef(ExUn)[grep("X6~~X6", names(coef(ExUn)))][1]

summaryTable$theta.y1[v] = coef(ExUn)[grep("Y1~~Y1", names(coef(ExUn)))][1]

summaryTable$theta.y2[v] = coef(ExUn)[grep("Y2~~Y2", names(coef(ExUn)))][1]

summaryTable$theta.y3[v] = coef(ExUn)[grep("Y3~~Y3", names(coef(ExUn)))][1]


summaryTable$gamma1[v] = coef(ExUn)[grep("ETA~KSI1", names(coef(ExUn)))][1]

summaryTable$gamma2[v] = coef(ExUn)[grep("ETA~KSI2", names(coef(ExUn)))][1]


summaryTable$phi1[v] = coef(ExUn)[grep("KSI1~~KSI1", names(coef(ExUn)))][1]

#phi2 is the covariance

summaryTable$phi2[v] = coef(ExUn)[grep("KSI1~~KSI2", names(coef(ExUn)))][1]

#phi4 is the var of 2

summaryTable$phi4[v] = coef(ExUn)[grep("KSI2~~KSI2", names(coef(ExUn)))][1]

summaryTable$psi[v] = coef(ExUn)[grep("ETA~~ETA", names(coef(ExUn)))][1]

summaryTable$omegai[v] = coef(ExUn)[grep("ETA~KSI1KSI2", names(coef(ExUn)))][1]
```

```
summaryTable$omega1[v] = coef(ExUn)[grep("ETA~KSI1KSI1", names(coef(ExUn)))][1]
summaryTable$omega2[v] = coef(ExUn)[grep("ETA~KSI2KSI2", names(coef(ExUn)))][1]


fit = establishFit(obs.data = dat, model.obj = ExUn,m = 300, nfac = 3, model = mod.unc, df = 
fitMeasures(ExUn)[4])
summaryTable$likl[v] = fit[1]
summaryTable$aic[v] = fit[2]
summaryTable$bic[v] = fit[3]
summaryTable$chisq[v] = fit[4]
summaryTable$chisqp[v] = fit[5]
summaryTable$rmsea[v] = fit[6]
summaryTable$srmr[v] = fit[7]
summaryTable$hfi[v] = fit[8]
summaryTable$hfir[v] = fit[9]
summaryTable$R2[v] = fit[10]
summaryTable$R2r[v] = fit[11]
summaryTable$hfi0[v] = fit[12]
summaryTable$hfi1[v] = fit[13]
summaryTable$hfi3[v] = fit[14]
summaryTable$PPP_likl[v] = fit[15]
summaryTable$PPP_aic[v] = fit[16]
summaryTable$PPP_bic[v] = fit[17]
summaryTable$PPP_chisq[v] = fit[18]
```

```
  summaryTable$PPP_chisqp[v] = fit[19]
  summaryTable$PPP_rmsea[v] = fit[20]
  summaryTable$PPP_srmr[v] = fit[21]
  summaryTable$PPP_hfi[v] = fit[22]
  summaryTable$PPP_hfir[v] = fit[23]
  summaryTable$PPP_R2r[v] = fit[24]
  summaryTable$PPP_R2[v] = fit[25]
  summaryTable$PPP_logl[v] = fit[26]
  summaryTable$PPP_cfi[v] = fit[27]
  summaryTable$PPP_tli[v] = fit[28]
  summaryTable$PPP_rmseap[v] = fit[29]
}
v = v+1
write.csv(summaryTable, file = fullpath, row.names=F)


  mod =
    'xi1 =~ x1 + x2 + x3
  xi2 =~ x4 + x5 + x6
  eta =~ y1 + y2 + y3
  eta ~ xi1 + xi2 + xi1:xi2 + xi1:xi1 + xi2:xi2'


  m2=lav2nlsem(mod)
```

```
    a = Sys.time()
        if(1 / summaryTable$gamma1[(v-1)] < 1 | summaryTable$converged[(v-1)] < 1){
      start = abs(rnorm(33, .3, .1))
      start = start *
sign(ExUn@ParTable$est[c(2,3,5,6,8,9,55,56,31:39,69,60,61,62,77,78,80,81,19,20,84,82,83,58,
57,59)])
    }else{
      start =
ExUn@ParTable$est[c(2,3,5,6,8,9,55,56,31:39,69,60,61,62,77,78,80,81,19,20,84,82,83,58,57,59
)]
    }

    QML3 <- tryCatch({em(m2, data.frame(data[,,1]), start, qml=TRUE, verbose= TRUE,
convergence = .1, neg.hessian = FALSE)
    },
error = function(e){
"Failed to Converge"
})

    b = Sys.time()

    summaryTable$method[v] = c("QML3")
    summaryTable$es[v] = c(es)
    summaryTable$ss[v] = c(ss)
    summaryTable$rxy[v] = c(rxy)
    summaryTable$terms[v] = c(terms)
    summaryTable$seed[v] = offset
    summaryTable$time[v] = as.numeric(difftime(b, a, units = "secs"))
if("Failed to Converge" != QML3){
    summaryTable$converged[v] = (QML3$em.convergence=="yes")*1

     summaryTable$lambda.x2[v] = coef(QML3)[grep("Lambda.x2", names(coef(QML3)))]
     summaryTable$lambda.x3[v] = coef(QML3)[grep("Lambda.x3", names(coef(QML3)))]
     summaryTable$lambda.x5[v] = coef(QML3)[grep("Lambda.x11", names(coef(QML3)))]
     summaryTable$lambda.x6[v] = coef(QML3)[grep("Lambda.x12", names(coef(QML3)))]
     summaryTable$lambda.y2[v] = coef(QML3)[grep("Lambda.y2", names(coef(QML3)))]
     summaryTable$lambda.y3[v] = coef(QML3)[grep("Lambda.y3", names(coef(QML3)))]

     summaryTable$nu.x2[v] = coef(QML3)[grep("nu.x2", names(coef(QML3)))]
     summaryTable$nu.x3[v] = coef(QML3)[grep("nu.x3", names(coef(QML3)))]
     summaryTable$nu.x5[v] = coef(QML3)[grep("nu.x5", names(coef(QML3)))]
     summaryTable$nu.x6[v] = coef(QML3)[grep("nu.x6", names(coef(QML3)))]
     summaryTable$nu.y2[v] = coef(QML3)[grep("nu.y2", names(coef(QML3)))]
     summaryTable$nu.y3[v] = coef(QML3)[grep("nu.y3", names(coef(QML3)))]

     summaryTable$theta.x1[v] = coef(QML3)[grep("Theta.d", names(coef(QML3)))][1]
```

```
summaryTable$theta.x2[v] = coef(QML3)[grep("Theta.d", names(coef(QML3)))][2]

summaryTable$theta.x3[v] = coef(QML3)[grep("Theta.d", names(coef(QML3)))][3]

summaryTable$theta.x4[v] = coef(QML3)[grep("Theta.d", names(coef(QML3)))][4]

summaryTable$theta.x5[v] = coef(QML3)[grep("Theta.d", names(coef(QML3)))][5]

summaryTable$theta.x6[v] = coef(QML3)[grep("Theta.d", names(coef(QML3)))][6]

summaryTable$theta.y1[v] = coef(QML3)[grep("Theta.e", names(coef(QML3)))][1]

summaryTable$theta.y2[v] = coef(QML3)[grep("Theta.e", names(coef(QML3)))][2]

summaryTable$theta.y3[v] = coef(QML3)[grep("Theta.e", names(coef(QML3)))][3]


summaryTable$gamma1[v] = coef(QML3)[grep("Gamma1", names(coef(QML3)))]

summaryTable$gamma2[v] = coef(QML3)[grep("Gamma2", names(coef(QML3)))]


summaryTable$phi1[v] = coef(QML3)[grep("Phi1", names(coef(QML3)))]

#phi2 is the covariance

summaryTable$phi2[v] = coef(QML3)[grep("Phi2", names(coef(QML3)))]

#phi4 is the var of 2

summaryTable$phi4[v] = coef(QML3)[grep("Phi4", names(coef(QML3)))]

summaryTable$psi[v] = coef(QML3)[grep("Psi", names(coef(QML3)))]

summaryTable$omegai[v] = coef(QML3)[grep("Omega3", names(coef(QML3)))]

summaryTable$omega1[v] = coef(QML3)[grep("Omega1", names(coef(QML3)))]

summaryTable$omega2[v] = coef(QML3)[grep("Omega4", names(coef(QML3)))]


fit = establishFit(obs.data = dat[1:9], model.obj = QML3, m = 300, nfac = 3, model = mod)
```

```
summaryTable$likl[v] = fit[1]

summaryTable$aic[v] = fit[2]

summaryTable$bic[v] = fit[3]

summaryTable$chisq[v] = fit[4]

summaryTable$chisqp[v] = fit[5]

summaryTable$rmsea[v] = fit[6]

summaryTable$srmr[v] = fit[7]

summaryTable$hfi[v] = fit[8]

summaryTable$hfir[v] = fit[9]

summaryTable$R2[v] = fit[10]

summaryTable$R2r[v] = fit[11]

summaryTable$hfi0[v] = fit[12]

summaryTable$hfi1[v] = fit[13]

summaryTable$hfi3[v] = fit[14]

summaryTable$PPP_likl[v] = fit[15]

summaryTable$PPP_aic[v] = fit[16]

summaryTable$PPP_bic[v] = fit[17]

summaryTable$PPP_chisq[v] = fit[18]

summaryTable$PPP_chisqp[v] = fit[19]

summaryTable$PPP_rmsea[v] = fit[20]

summaryTable$PPP_srmr[v] = fit[21]

summaryTable$PPP_hfi[v] = fit[22]

summaryTable$PPP_hfir[v] = fit[23]
```

```
    summaryTable$PPP_R2r[v] = fit[24]
    summaryTable$PPP_R2[v] = fit[25]
}
   v = v+1
   write.csv(summaryTable, file = fullpath, row.names=F)

   # bnl3
   rasp = coef(ExUn)[c(1:6,49,
                  50,28:33,25:27,
                  63,70,71,72,73,74,75,13,
                  14,15, 54,56,55,51,52,53)]
   rasp[grep("\\d~~", names(rasp))[which(rasp[grep("\\d~~", names(rasp))]<0)]] = NA





   db$omega = matrix(c(NA, NA, 0, NA), nrow=2)
   inits = list(c1 = list(parvec= c(rasp)),
             c2 = list(parvec= c(rasp)),
             c3 = list(parvec= c(rasp)))

   a = Sys.time()
```

```
    bnl3 <-tryCatch({run.jags("/fslhome/pip89/compute/Omega/full.jag", monitor = monitors,
data = db, n.chains = 3, thin = 1, inits = inits)
    },
error = function(e){
"Failed to Converge"
})


    b = Sys.time()


    summaryTable$method[v] = "bnl3"
    summaryTable$es[v] = c(es)
    summaryTable$ss[v] = c(ss)
    summaryTable$rxy[v] = c(rxy)
    summaryTable$terms[v] = c(terms)
    summaryTable$seed[v] = offset
    summaryTable$time[v] = as.numeric(difftime(b, a, units = "secs"))
if("Failed to Converge" != bnl3){
      summaryTable$converged[v] =
mean(na.omit(data.frame(summary(bnl3))$psrf[1:39]<1.005))
      summaryTable$lambda.x2[v] = data.frame(summary(bnl3))[grep("lambda",
rownames(data.frame(summary(bnl3)))),][2,2]
      summaryTable$lambda.x3[v] = data.frame(summary(bnl3))[grep("lambda",
rownames(data.frame(summary(bnl3)))),][3,2]
```

```
	summaryTable$lambda.x5[v] = data.frame(summary(bnl3))[grep("lambda",
rownames(data.frame(summary(bnl3)))),][5,2]
	summaryTable$lambda.x6[v] = data.frame(summary(bnl3))[grep("lambda",
rownames(data.frame(summary(bnl3)))),][6,2]
	summaryTable$lambda.y2[v] = data.frame(summary(bnl3))[grep("lambda",
rownames(data.frame(summary(bnl3)))),][8,2]
	summaryTable$lambda.y3[v] = data.frame(summary(bnl3))[grep("lambda",
rownames(data.frame(summary(bnl3)))),][9,2]

	summaryTable$nu.x1[v] = data.frame(summary(bnl3))[grep("nu",
rownames(data.frame(summary(bnl3)))),][1,2]
	summaryTable$nu.x2[v] = data.frame(summary(bnl3))[grep("nu",
rownames(data.frame(summary(bnl3)))),][2,2]
	summaryTable$nu.x3[v] = data.frame(summary(bnl3))[grep("nu",
rownames(data.frame(summary(bnl3)))),][3,2]
	summaryTable$nu.x4[v] = data.frame(summary(bnl3))[grep("nu",
rownames(data.frame(summary(bnl3)))),][4,2]
	summaryTable$nu.x5[v] = data.frame(summary(bnl3))[grep("nu",
rownames(data.frame(summary(bnl3)))),][5,2]
	summaryTable$nu.x6[v] = data.frame(summary(bnl3))[grep("nu",
rownames(data.frame(summary(bnl3)))),][6,2]
	summaryTable$nu.y1[v] = data.frame(summary(bnl3))[grep("nu",
rownames(data.frame(summary(bnl3)))),][7,2]
```

```
    summaryTable$nu.y2[v] = data.frame(summary(bnl3))[grep("nu",
rownames(data.frame(summary(bnl3)))),][8,2]
    summaryTable$nu.y3[v] = data.frame(summary(bnl3))[grep("nu",
rownames(data.frame(summary(bnl3)))),][9,2]



    summaryTable$theta.x1[v] = data.frame(summary(bnl3))[grep("theta",
rownames(data.frame(summary(bnl3)))),][1,2]
    summaryTable$theta.x2[v] = data.frame(summary(bnl3))[grep("theta",
rownames(data.frame(summary(bnl3)))),][2,2]
    summaryTable$theta.x3[v] = data.frame(summary(bnl3))[grep("theta",
rownames(data.frame(summary(bnl3)))),][3,2]
    summaryTable$theta.x4[v] = data.frame(summary(bnl3))[grep("theta",
rownames(data.frame(summary(bnl3)))),][4,2]
    summaryTable$theta.x5[v] = data.frame(summary(bnl3))[grep("theta",
rownames(data.frame(summary(bnl3)))),][5,2]
    summaryTable$theta.x6[v] = data.frame(summary(bnl3))[grep("theta",
rownames(data.frame(summary(bnl3)))),][6,2]
    summaryTable$theta.y1[v] = data.frame(summary(bnl3))[grep("theta",
rownames(data.frame(summary(bnl3)))),][1,2]
    summaryTable$theta.y2[v] = data.frame(summary(bnl3))[grep("theta",
rownames(data.frame(summary(bnl3)))),][2,2]
```

```
    summaryTable$theta.y3[v] = data.frame(summary(bnl3))[grep("theta",
rownames(data.frame(summary(bnl3)))),][3,2]


    summaryTable$gamma1[v] = data.frame(summary(bnl3))[grep("omega",
rownames(data.frame(summary(bnl3)))),][1,2]
    summaryTable$gamma2[v] = data.frame(summary(bnl3))[grep("omega",
rownames(data.frame(summary(bnl3)))),][2,2]


    summaryTable$phi1[v] = data.frame(summary(bnl3))[grep("psi",
rownames(data.frame(summary(bnl3)))),][1,2]
    #phi2 is the covariance
    summaryTable$phi2[v] = data.frame(summary(bnl3))[grep("psi",
rownames(data.frame(summary(bnl3)))),][4,2]
    #phi4 is the var of 2
    summaryTable$phi4[v] = data.frame(summary(bnl3))[grep("psi",
rownames(data.frame(summary(bnl3)))),][2,2]
    summaryTable$psi[v] = data.frame(summary(bnl3))[grep("psi",
rownames(data.frame(summary(bnl3)))),][3,2]
    summaryTable$omegai[v] = data.frame(summary(bnl3))[grep("omega",
rownames(data.frame(summary(bnl3)))),][1,1]
    summaryTable$omega1[v] = data.frame(summary(bnl3))[grep("omega",
rownames(data.frame(summary(bnl3)))),][2,1]
```

```
    summaryTable$omega2[v] = data.frame(summary(bnl3))[grep("omega",
rownames(data.frame(summary(bnl3)))),][3,1]

    fit = establishFit(obs.data = dat[1:9], model.obj = bnl3, m = 300, nfac = 3)
    summaryTable$likl[v] = fit[1]
    summaryTable$aic[v] = fit[2]
    summaryTable$bic[v] = fit[3]
    summaryTable$chisq[v] = fit[4]
    summaryTable$chisqp[v] = fit[5]
    summaryTable$rmsea[v] = fit[6]
    summaryTable$srmr[v] = fit[7]
    summaryTable$hfi[v] = fit[8]
    summaryTable$hfir[v] = fit[9]
    summaryTable$R2[v] = fit[10]
    summaryTable$R2r[v] = fit[11]
    summaryTable$hfi0[v] = fit[12]
    summaryTable$hfi1[v] = fit[13]
    summaryTable$hfi3[v] = fit[14]
    summaryTable$PPP_likl[v] = fit[15]
    summaryTable$PPP_aic[v] = fit[16]
    summaryTable$PPP_bic[v] = fit[17]
    summaryTable$PPP_chisq[v] = fit[18]
    summaryTable$PPP_chisqp[v] = fit[19]
```

```
      summaryTable$PPP_rmsea[v] = fit[20]
      summaryTable$PPP_srmr[v] = fit[21]
      summaryTable$PPP_hfi[v] = fit[22]
      summaryTable$PPP_hfir[v] = fit[23]
      summaryTable$PPP_R2r[v] = fit[24]
      summaryTable$PPP_R2[v] = fit[25]
}
    v = v+1
    write.csv(summaryTable, file = fullpath, row.names=F)

   # Un, bnl1, qml1
   # Un
   dat = dat[1:9]
   dat$X1X4 <- dat$X1 * dat$X4
   dat$X2X5 <- dat$X2 * dat$X5
   dat$X3X6 <- dat$X3 * dat$X6
   mod.unc <- '
   ETA =~ 1*Y1
   ETA =~ Y2
   ETA =~ Y3
   KSI1 =~ 1*X1
   KSI1 =~ X2
   KSI1 =~ X3
```

```
KSI2 =~ 1*X4

KSI2 =~ X5

KSI2 =~ X6

KSI1KSI2 =~ 1*X1X4

KSI1KSI2 =~ X2X5

KSI1KSI2 =~ X3X6

Y1 ~ 1

Y2 ~ 1

Y3 ~ 1

X1X4 ~ 1

X2X5 ~ 1

X3X6 ~ 1

Y1 ~~ Y1

Y2 ~~ Y2

Y3 ~~ Y3

X1 ~~ X1

X2 ~~ X2

X3 ~~ X3

X4 ~~ X4

X5 ~~ X5

X6 ~~ X6

X1X4 ~~ X1X4

X2X5 ~~ X2X5
```

```
  X3X6 ~~ X3X6
  ETA ~ KSI1
  ETA ~ KSI2
  ETA ~ KSI1KSI2
  KSI1 ~~ KSI1
  KSI1 ~~ KSI2
  KSI2 ~~ KSI2
  KSI1KSI2 ~~ KSI1KSI2
  ETA ~~ ETA
  '
  a = Sys.time()
  Un <- lavaan::sem(mod.unc, dat, control = list(iter.max=3000))
  b = Sys.time()
  summaryTable$method[v] = c("UC")
  summaryTable$converged[v] = Un@optim$converged*1
  summaryTable$es[v] = c(es)
  summaryTable$ss[v] = c(ss)
  summaryTable$rxy[v] = c(rxy)
  summaryTable$terms[v] = c(terms)
  summaryTable$seed[v] = offset
  summaryTable$time[v] = as.numeric(difftime(b, a, units = "secs"))
if(Un@optim$converged){
   summaryTable$lambda.x2[v] = coef(Un)[grep("KSI1=~X2", names(coef(Un)))][1]
```

```
summaryTable$lambda.x3[v] = coef(Un)[grep("KSI1=~X3", names(coef(Un)))][1]

summaryTable$lambda.x5[v] = coef(Un)[grep("KSI2=~X5", names(coef(Un)))][1]

summaryTable$lambda.x6[v] = coef(Un)[grep("KSI2=~X6", names(coef(Un)))][1]

summaryTable$lambda.y2[v] = coef(Un)[grep("ETA=~Y2", names(coef(Un)))][1]

summaryTable$lambda.y3[v] = coef(Un)[grep("ETA=~Y3", names(coef(Un)))][1]


summaryTable$nu.x1[v] = coef(Un)[grep("X1~1", names(coef(Un)))][1]

summaryTable$nu.x2[v] = coef(Un)[grep("X2~1", names(coef(Un)))][1]

summaryTable$nu.x3[v] = coef(Un)[grep("X3~1", names(coef(Un)))][1]

summaryTable$nu.x4[v] = coef(Un)[grep("X4~1", names(coef(Un)))][1]

summaryTable$nu.x5[v] = coef(Un)[grep("X5~1", names(coef(Un)))][1]

summaryTable$nu.x6[v] = coef(Un)[grep("X6~1", names(coef(Un)))][1]

summaryTable$nu.y1[v] = coef(Un)[grep("Y1~1", names(coef(Un)))][1]

summaryTable$nu.y2[v] = coef(Un)[grep("Y2~1", names(coef(Un)))][1]

summaryTable$nu.y3[v] = coef(Un)[grep("Y3~1", names(coef(Un)))][1]


summaryTable$theta.x1[v] = coef(Un)[grep("X1~~X1", names(coef(Un)))][1]

summaryTable$theta.x2[v] = coef(Un)[grep("X2~~X2", names(coef(Un)))][1]

summaryTable$theta.x3[v] = coef(Un)[grep("X3~~X3", names(coef(Un)))][1]

summaryTable$theta.x4[v] = coef(Un)[grep("X4~~X4", names(coef(Un)))][1]

summaryTable$theta.x5[v] = coef(Un)[grep("X5~~X5", names(coef(Un)))][1]

summaryTable$theta.x6[v] = coef(Un)[grep("X6~~X6", names(coef(Un)))][1]

summaryTable$theta.y1[v] = coef(Un)[grep("Y1~~Y1", names(coef(Un)))][1]
```

```
    summaryTable$theta.y2[v] = coef(Un)[grep("Y2~~Y2", names(coef(Un)))][1]
    summaryTable$theta.y3[v] = coef(Un)[grep("Y3~~Y3", names(coef(Un)))][1]

    summaryTable$gamma1[v] = coef(Un)[grep("ETA~KSI1", names(coef(Un)))][1]
    summaryTable$gamma2[v] = coef(Un)[grep("ETA~KSI2", names(coef(Un)))][1]

    summaryTable$phi1[v] = coef(Un)[grep("KSI1~~KSI1", names(coef(Un)))][1]
    #phi2 is the covariance
    summaryTable$phi2[v] = coef(Un)[grep("KSI1~~KSI2", names(coef(Un)))][1]
    #phi4 is the var of 2
    summaryTable$phi4[v] = coef(Un)[grep("KSI2~~KSI2", names(coef(Un)))][1]
    summaryTable$psi[v] = coef(Un)[grep("ETA~~ETA", names(coef(Un)))][1]
    summaryTable$omegai[v] = coef(Un)[grep("ETA~KSI1KSI2", names(coef(Un)))][1]
    summaryTable$omega1[v] = 0
    summaryTable$omega2[v] = 0

    fit = establishFit(obs.data = dat, model.obj = Un, m = 300, nfac = 1, model = mod.unc, df =
fitMeasures(Un)[4])
    summaryTable$likl[v] = fit[1]
    summaryTable$aic[v] = fit[2]
    summaryTable$bic[v] = fit[3]
    summaryTable$chisq[v] = fit[4]
    summaryTable$chisqp[v] = fit[5]
```

```
summaryTable$rmsea[v] = fit[6]

summaryTable$srmr[v] = fit[7]

summaryTable$hfi[v] = fit[8]

summaryTable$hfir[v] = fit[9]

summaryTable$R2[v] = fit[10]

summaryTable$R2r[v] = fit[11]

summaryTable$hfi0[v] = fit[12]

summaryTable$hfi1[v] = fit[13]

summaryTable$hfi3[v] = fit[14]

summaryTable$PPP_likl[v] = fit[15]

summaryTable$PPP_aic[v] = fit[16]

summaryTable$PPP_bic[v] = fit[17]

summaryTable$PPP_chisq[v] = fit[18]

summaryTable$PPP_chisqp[v] = fit[19]

summaryTable$PPP_rmsea[v] = fit[20]

summaryTable$PPP_srmr[v] = fit[21]

summaryTable$PPP_hfi[v] = fit[22]

summaryTable$PPP_hfir[v] = fit[23]

summaryTable$PPP_R2r[v] = fit[24]

summaryTable$PPP_R2[v] = fit[25]

summaryTable$PPP_logl[v] = fit[26]

summaryTable$PPP_cfi[v] = fit[27]

summaryTable$PPP_tli[v] = fit[28]
```

```
     summaryTable$PPP_rmseap[v] = fit[29]
}
   v = v+1
   write.csv(summaryTable, file = fullpath, row.names=F)

     # QML1

   if(1 / summaryTable$gamma1[6] < 1){
     start = abs(rnorm(31, .3, .1))
     start = start *
sign(Un@ParTable$est[c(2,3,5,6,8,9,31,32,19:27,38,34,36,35,41:44,49,13,14,47,48,33)])
   }else{
     start = Un@ParTable$est[c(2,3,5,6,8,9,31,32,19:27,38,34,36,35,41:44,49,13,14,47,48,33)]
   }

     qml1 <- specify_sem(num.x = 6, num.y = 3, num.xi = 2, num.eta = 1, xi = "x1-x3,x4-x6",
                   eta = "y1-y3", interaction = "xi1:xi2")

     a = Sys.time()
     QML1 =tryCatch({
em(qml1, data.frame(data[,,1]), start = start, qml =T, verbose = TRUE, convergence=.1,
neg.hessian=FALSE)
},
error = function(e){
"Failed to Converge"
})
     b = Sys.time()
     summaryTable$method[v] = "QML1"
     summaryTable$es[v] = es
     summaryTable$ss[v] = ss
     summaryTable$rxy[v] = rxy
     summaryTable$terms[v] = terms
     summaryTable$seed[v] = offset
     summaryTable$time[v] = as.numeric(difftime(b, a, units = "secs"))
if("Failed to Converge" != QML1){
   summaryTable$converged[v] = (QML1$em.convergence=="yes")*1
     summaryTable$lambda.x2[v] = coef(QML1)[grep("Lambda.x2", names(coef(QML1)))]
     summaryTable$lambda.x3[v] = coef(QML1)[grep("Lambda.x3", names(coef(QML1)))]
     summaryTable$lambda.x5[v] = coef(QML1)[grep("Lambda.x11", names(coef(QML1)))]
     summaryTable$lambda.x6[v] = coef(QML1)[grep("Lambda.x12", names(coef(QML1)))]
     summaryTable$lambda.y2[v] = coef(QML1)[grep("Lambda.y2", names(coef(QML1)))]
     summaryTable$lambda.y3[v] = coef(QML1)[grep("Lambda.y3", names(coef(QML1)))]

     summaryTable$nu.x2[v] = coef(QML1)[grep("nu.x2", names(coef(QML1)))]
     summaryTable$nu.x3[v] = coef(QML1)[grep("nu.x3", names(coef(QML1)))]
     summaryTable$nu.x5[v] = coef(QML1)[grep("nu.x5", names(coef(QML1)))]
```

```
summaryTable$nu.x6[v] = coef(QML1)[grep("nu.x6", names(coef(QML1)))]
summaryTable$nu.y2[v] = coef(QML1)[grep("nu.y2", names(coef(QML1)))]
summaryTable$nu.y3[v] = coef(QML1)[grep("nu.y3", names(coef(QML1)))]

summaryTable$theta.x1[v] = coef(QML1)[grep("Theta.d", names(coef(QML1)))][1]
summaryTable$theta.x2[v] = coef(QML1)[grep("Theta.d", names(coef(QML1)))][2]
summaryTable$theta.x3[v] = coef(QML1)[grep("Theta.d", names(coef(QML1)))][3]
summaryTable$theta.x4[v] = coef(QML1)[grep("Theta.d", names(coef(QML1)))][4]
summaryTable$theta.x5[v] = coef(QML1)[grep("Theta.d", names(coef(QML1)))][5]
summaryTable$theta.x6[v] = coef(QML1)[grep("Theta.d", names(coef(QML1)))][6]
summaryTable$theta.y1[v] = coef(QML1)[grep("Theta.e", names(coef(QML1)))][1]
summaryTable$theta.y2[v] = coef(QML1)[grep("Theta.e", names(coef(QML1)))][2]
summaryTable$theta.y3[v] = coef(QML1)[grep("Theta.e", names(coef(QML1)))][3]

summaryTable$gamma1[v] = coef(QML1)[grep("Gamma1", names(coef(QML1)))]
summaryTable$gamma2[v] = coef(QML1)[grep("Gamma2", names(coef(QML1)))]

summaryTable$phi1[v] = coef(QML1)[grep("Phi1", names(coef(QML1)))]
#phi2 is the covariance
summaryTable$phi2[v] = coef(QML1)[grep("Phi2", names(coef(QML1)))]
#phi4 is the var of 2
summaryTable$phi4[v] = coef(QML1)[grep("Phi4", names(coef(QML1)))]
summaryTable$psi[v] = coef(QML1)[grep("Psi", names(coef(QML1)))]
```

```
summaryTable$omegai[v] = coef(QML1)[grep("Omega3", names(coef(QML1)))]

summaryTable$omega1[v] = 0

summaryTable$omega2[v] = 0


fit = establishFit(obs.data = dat[1:9], model.obj = QML1, m = 300, nfac = 1)

summaryTable$likl[v] = fit[1]

summaryTable$aic[v] = fit[2]

summaryTable$bic[v] = fit[3]

summaryTable$chisq[v] = fit[4]

summaryTable$chisqp[v] = fit[5]

summaryTable$rmsea[v] = fit[6]

summaryTable$srmr[v] = fit[7]

summaryTable$hfi[v] = fit[8]

summaryTable$hfir[v] = fit[9]

summaryTable$R2[v] = fit[10]

summaryTable$R2r[v] = fit[11]

summaryTable$hfi0[v] = fit[12]

summaryTable$hfi1[v] = fit[13]

summaryTable$hfi3[v] = fit[14]

summaryTable$PPP_likl[v] = fit[15]

summaryTable$PPP_aic[v] = fit[16]

summaryTable$PPP_bic[v] = fit[17]

summaryTable$PPP_chisq[v] = fit[18]
```

```
    summaryTable$PPP_chisqp[v] = fit[19]
    summaryTable$PPP_rmsea[v] = fit[20]
    summaryTable$PPP_srmr[v] = fit[21]
    summaryTable$PPP_hfi[v] = fit[22]
    summaryTable$PPP_hfir[v] = fit[23]
    summaryTable$PPP_R2r[v] = fit[24]
    summaryTable$PPP_R2[v] = fit[25]
}
   v = v+1
   write.csv(summaryTable, file = fullpath, row.names=F)

   rasp = c(coef(Un)[3:4],coef(Un)[5:6],coef(Un)[c(1,2,27,28,18:23, 15:17, 34,37:42,
9,10,11)],coef(Un)[c(30,32,31,29)]])
   rasp[grep("\\d~~", names(rasp))[which(rasp[grep("\\d~~", names(rasp))]<0)]] = NA

   # Bnl1
   db$omega = matrix(c(0, NA, 0, 0), nrow=2)
   inits = list(c1 = list(parvec= c(rasp)),
                c2 = list(parvec= c(rasp)),
                c3 = list(parvec= c(rasp)))

   a = Sys.time()
   bnl1 <- tryCatch({
```

```
run.jags("/fslhome/pip89/compute/Omega/int.jag", monitor = monitors, data = db, n.chains = 3,
thin = 1, inits = inits)
},
error = function(e){
"Failed to Converge"
})
    b = Sys.time()


    summaryTable$method[v] = "bnl1"
    summaryTable$es[v] = c(es)
    summaryTable$ss[v] = c(ss)
    summaryTable$rxy[v] = c(rxy)
    summaryTable$terms[v] = c(terms)
    summaryTable$seed[v] = offset
    summaryTable$time[v] = as.numeric(difftime(b, a, units = "secs"))
if("Failed to Converge" != bnl1){
summaryTable$converged[v] = mean(data.frame(summary(bnl1))$psrf[1:39]<1.005, na.rm=T)
      summaryTable$lambda.x2[v] = data.frame(summary(bnl1))[grep("lambda",
rownames(data.frame(summary(bnl1)))),][2,2]
      summaryTable$lambda.x3[v] = data.frame(summary(bnl1))[grep("lambda",
rownames(data.frame(summary(bnl1)))),][3,2]
      summaryTable$lambda.x5[v] = data.frame(summary(bnl1))[grep("lambda",
rownames(data.frame(summary(bnl1)))),][5,2]
```

```
    summaryTable$lambda.x6[v] = data.frame(summary(bnl1))[grep("lambda",
rownames(data.frame(summary(bnl1)))),][6,2]
    summaryTable$lambda.y2[v] = data.frame(summary(bnl1))[grep("lambda",
rownames(data.frame(summary(bnl1)))),][8,2]
    summaryTable$lambda.y3[v] = data.frame(summary(bnl1))[grep("lambda",
rownames(data.frame(summary(bnl1)))),][9,2]

    summaryTable$nu.x1[v] = data.frame(summary(bnl1))[grep("nu",
rownames(data.frame(summary(bnl1)))),][1,2]
    summaryTable$nu.x2[v] = data.frame(summary(bnl1))[grep("nu",
rownames(data.frame(summary(bnl1)))),][2,2]
    summaryTable$nu.x3[v] = data.frame(summary(bnl1))[grep("nu",
rownames(data.frame(summary(bnl1)))),][3,2]
    summaryTable$nu.x4[v] = data.frame(summary(bnl1))[grep("nu",
rownames(data.frame(summary(bnl1)))),][4,2]
    summaryTable$nu.x5[v] = data.frame(summary(bnl1))[grep("nu",
rownames(data.frame(summary(bnl1)))),][5,2]
    summaryTable$nu.x6[v] = data.frame(summary(bnl1))[grep("nu",
rownames(data.frame(summary(bnl1)))),][6,2]
    summaryTable$nu.y1[v] = data.frame(summary(bnl1))[grep("nu",
rownames(data.frame(summary(bnl1)))),][7,2]
    summaryTable$nu.y2[v] = data.frame(summary(bnl1))[grep("nu",
rownames(data.frame(summary(bnl1)))),][8,2]
```

```
    summaryTable$nu.y3[v] = data.frame(summary(bnl1))[grep("nu",
rownames(data.frame(summary(bnl1)))),][9,2]



    summaryTable$theta.x1[v] = data.frame(summary(bnl1))[grep("theta",
rownames(data.frame(summary(bnl1)))),][1,2]
    summaryTable$theta.x2[v] = data.frame(summary(bnl1))[grep("theta",
rownames(data.frame(summary(bnl1)))),][2,2]
    summaryTable$theta.x3[v] = data.frame(summary(bnl1))[grep("theta",
rownames(data.frame(summary(bnl1)))),][3,2]
    summaryTable$theta.x4[v] = data.frame(summary(bnl1))[grep("theta",
rownames(data.frame(summary(bnl1)))),][4,2]
    summaryTable$theta.x5[v] = data.frame(summary(bnl1))[grep("theta",
rownames(data.frame(summary(bnl1)))),][5,2]
    summaryTable$theta.x6[v] = data.frame(summary(bnl1))[grep("theta",
rownames(data.frame(summary(bnl1)))),][6,2]
    summaryTable$theta.y1[v] = data.frame(summary(bnl1))[grep("theta",
rownames(data.frame(summary(bnl1)))),][1,2]
    summaryTable$theta.y2[v] = data.frame(summary(bnl1))[grep("theta",
rownames(data.frame(summary(bnl1)))),][2,2]
    summaryTable$theta.y3[v] = data.frame(summary(bnl1))[grep("theta",
rownames(data.frame(summary(bnl1)))),][3,2]
```

```
        summaryTable$gamma1[v] = data.frame(summary(bnl1))[grep("omega",
rownames(data.frame(summary(bnl1)))),][1,2]
        summaryTable$gamma2[v] = data.frame(summary(bnl1))[grep("omega",
rownames(data.frame(summary(bnl1)))),][2,2]


        summaryTable$phi1[v] = data.frame(summary(bnl1))[grep("psi",
rownames(data.frame(summary(bnl1)))),][1,2]
        #phi2 is the covariance
        summaryTable$phi2[v] = data.frame(summary(bnl1))[grep("psi",
rownames(data.frame(summary(bnl1)))),][4,2]
        #phi4 is the var of 2
        summaryTable$phi4[v] = data.frame(summary(bnl1))[grep("psi",
rownames(data.frame(summary(bnl1)))),][2,2]
        summaryTable$psi[v] = data.frame(summary(bnl1))[grep("psi",
rownames(data.frame(summary(bnl1)))),][3,2]
        summaryTable$omegai[v] = data.frame(summary(bnl1))[grep("omega",
rownames(data.frame(summary(bnl1)))),][1,2]
        summaryTable$omega1[v] = data.frame(summary(bnl1))[grep("omega",
rownames(data.frame(summary(bnl1)))),][2,2]
        summaryTable$omega2[v] = data.frame(summary(bnl1))[grep("omega",
rownames(data.frame(summary(bnl1)))),][3,2]


        fit = establishFit(obs.data = dat[1:9], model.obj = bnl1, m = 300, nfac = 1)
```

```
summaryTable$likl[v] = fit[1]

summaryTable$aic[v] = fit[2]

summaryTable$bic[v] = fit[3]

summaryTable$chisq[v] = fit[4]

summaryTable$chisqp[v] = fit[5]

summaryTable$rmsea[v] = fit[6]

summaryTable$srmr[v] = fit[7]

summaryTable$hfi[v] = fit[8]

summaryTable$hfir[v] = fit[9]

summaryTable$R2[v] = fit[10]

summaryTable$R2r[v] = fit[11]

summaryTable$hfi0[v] = fit[12]

summaryTable$hfi1[v] = fit[13]

summaryTable$hfi3[v] = fit[14]

summaryTable$PPP_likl[v] = fit[15]

summaryTable$PPP_aic[v] = fit[16]

summaryTable$PPP_bic[v] = fit[17]

summaryTable$PPP_chisq[v] = fit[18]

summaryTable$PPP_chisqp[v] = fit[19]

summaryTable$PPP_rmsea[v] = fit[20]

summaryTable$PPP_srmr[v] = fit[21]

summaryTable$PPP_hfi[v] = fit[22]

summaryTable$PPP_hfir[v] = fit[23]
```

```
        summaryTable$PPP_R2r[v] = fit[24]
        summaryTable$PPP_R2[v] = fit[25]
}
      v = v+1
      write.csv(summaryTable, file = fullpath, row.names=F)
}
write.csv(summaryTable, file = fullpath, row.names=F)
print(summaryTable)
print(Sys.time()-Start)
```

C1.1: Jags code for the interaction model

```
model{
for(i in 1:N) {

   X1[i] ~ dnorm(mu[i,1], 1/theta[1,1])
   X2[i] ~ dnorm(mu[i,2], 1/theta[2,2])
   X3[i] ~ dnorm(mu[i,3], 1/theta[3,3])

   X4[i] ~ dnorm(mu[i,4], 1/theta[4,4])
   X5[i] ~ dnorm(mu[i,5], 1/theta[5,5])
   X6[i] ~ dnorm(mu[i,6], 1/theta[6,6])

   Y1[i] ~ dnorm(mu[i,7], 1/theta[7,7])
```

```
  Y2[i] ~ dnorm(mu[i,8], 1/theta[8,8])
  Y3[i] ~ dnorm(mu[i,9], 1/theta[9,9])

  eta[i,1] ~ dnorm(mu_eta[i,1], 1/psi[1,1])
  eta[i,2] ~ dnorm(mu_eta[i,2], 1/psi[2,2])
  eta[i,3] ~ dnorm(mu_eta[i,3], 1/psi[3,3])
}

for(i in 1:N) {
  mu[i,1] <- nu[1,1] + lambda[1,1]*eta[i,1]
  mu[i,2] <- nu[2,1] + lambda[2,1]*eta[i,1]
  mu[i,3] <- nu[3,1] + lambda[3,1]*eta[i,1]
  mu[i,4] <- nu[4,1] + lambda[4,2]*eta[i,2]
  mu[i,5] <- nu[5,1] + lambda[5,2]*eta[i,2]
  mu[i,6] <- nu[6,1] + lambda[6,2]*eta[i,2]
  mu[i,7] <- nu[7,1] + lambda[7,3]*eta[i,3]
  mu[i,8] <- nu[8,1] + lambda[8,3]*eta[i,3]
  mu[i,9] <- nu[9,1] + lambda[9,3]*eta[i,3]

  mu_eta[i,1] <- alpha[1,1]
  mu_eta[i,2] <- alpha[2,1]
  mu_eta[i,3] <- alpha[3,1] + omega[3,1]*eta[i,1] + omega[3,2]*eta[i,2] +
omega[2,1]*eta[i,1]*eta[i,2]}
```

```
lambda[1,1] <- 1

lambda[2,1] <- parvec[1]

lambda[3,1] <- parvec[2]

lambda[4,2] <- 1

lambda[5,2] <- parvec[3]

lambda[6,2] <- parvec[4]

lambda[7,3] <- 1

lambda[8,3] <- parvec[5]

lambda[9,3] <- parvec[6]

omega[3,1] <- parvec[7]

omega[3,2] <- parvec[8]

theta[1,1] <- pow(parvec[9],-1)

theta[2,2] <- pow(parvec[10],-1)

theta[3,3] <- pow(parvec[11],-1)

theta[4,4] <- pow(parvec[12],-1)

theta[5,5] <- pow(parvec[13],-1)

theta[6,6] <- pow(parvec[14],-1)

theta[7,7] <- pow(parvec[15],-1)

theta[8,8] <- pow(parvec[16],-1)

theta[9,9] <- pow(parvec[17],-1)

psi[3,3] <- pow(parvec[18],-1)

nu[1,1] <- parvec[19]

nu[2,1] <- parvec[20]
```

```
nu[3,1] <- parvec[21]

nu[4,1] <- parvec[22]

nu[5,1] <- parvec[23]

nu[6,1] <- parvec[24]

nu[7,1] <- parvec[25]

nu[8,1] <- parvec[26]

nu[9,1] <- parvec[27]

alpha[1,1] <- 0

alpha[2,1] <- 0

alpha[3,1] <- 0

psi[1,1] <- pow(parvec[28],-1)

psi[2,2] <- pow(parvec[29],-1)

psi[1,2] <- pow(parvec[30],-1)

omega[2,1] <- parvec[31]

parvec[1] ~ dnorm(0,1e-2)

parvec[2] ~ dnorm(0,1e-2)

parvec[3] ~ dnorm(0,1e-2)

parvec[4] ~ dnorm(0,1e-2)

parvec[5] ~ dnorm(0,1e-2)

parvec[6] ~ dnorm(0,1e-2)

parvec[7] ~ dnorm(0,1e-2)

parvec[8] ~ dnorm(0,1e-2)

parvec[9] ~ dgamma(1,.5)
```

```
  parvec[10] ~ dgamma(1,.5)
  parvec[11] ~ dgamma(1,.5)
  parvec[12] ~ dgamma(1,.5)
  parvec[13] ~ dgamma(1,.5)
  parvec[14] ~ dgamma(1,.5)
  parvec[15] ~ dgamma(1,.5)
  parvec[16] ~ dgamma(1,.5)
  parvec[17] ~ dgamma(1,.5)
  parvec[18] ~ dgamma(1,.5)
  parvec[19] ~ dnorm(0,1e-3)
  parvec[20] ~ dnorm(0,1e-3)
  parvec[21] ~ dnorm(0,1e-3)
  parvec[22] ~ dnorm(0,1e-3)
  parvec[23] ~ dnorm(0,1e-3)
  parvec[24] ~ dnorm(0,1e-3)
  parvec[25] ~ dnorm(0,1e-3)
  parvec[26] ~ dnorm(0,1e-3)
  parvec[27] ~ dnorm(0,1e-3)
  parvec[28] ~ dgamma(1,.5)
  parvec[29] ~ dgamma(1,.5)
  parvec[30] ~ dgamma(1,.5)
  parvec[31] ~ dnorm(0,1e-2)
}
```

C1.2: Jags code for the full model

```
model {
 for(i in 1:N){


   X1[i] ~ dnorm(mu[i,1], 1/theta[1,1])
   X2[i] ~ dnorm(mu[i,2], 1/theta[2,2])
   X3[i] ~ dnorm(mu[i,3], 1/theta[3,3])

   X4[i] ~ dnorm(mu[i,4], 1/theta[4,4])
   X5[i] ~ dnorm(mu[i,5], 1/theta[5,5])
   X6[i] ~ dnorm(mu[i,6], 1/theta[6,6])

   Y1[i] ~ dnorm(mu[i,7], 1/theta[7,7])
   Y2[i] ~ dnorm(mu[i,8], 1/theta[8,8])
   Y3[i] ~ dnorm(mu[i,9], 1/theta[9,9])

   eta[i,1] ~ dnorm(mu_eta[i,1], 1/psi[1,1])

   eta[i,2] ~ dnorm(mu_eta[i,2], 1/psi[2,2])

   eta[i,3] ~ dnorm(mu_eta[i,3], 1/psi[3,3])}
for(i in 1:N) {
   mu[i,1] <- nu[1,1] + lambda[1,1]*eta[i,1]
```

```
    mu[i,2] <- nu[2,1] + lambda[2,1]*eta[i,1]

    mu[i,3] <- nu[3,1] + lambda[3,1]*eta[i,1]

    mu[i,4] <- nu[4,1] + lambda[4,2]*eta[i,2]

    mu[i,5] <- nu[5,1] + lambda[5,2]*eta[i,2]

    mu[i,6] <- nu[6,1] + lambda[6,2]*eta[i,2]

    mu[i,7] <- nu[7,1] + lambda[7,3]*eta[i,3]

    mu[i,8] <- nu[8,1] + lambda[8,3]*eta[i,3]

    mu[i,9] <- nu[9,1] + lambda[9,3]*eta[i,3]


    mu_eta[i,1] <- alpha[1,1]

    mu_eta[i,2] <- alpha[2,1]

    mu_eta[i,3] <- alpha[3,1] + omega[3,1]*eta[i,1] + omega[3,2]*eta[i,2] +
omega[2,1]*eta[i,1]*eta[i,2] + omega[1,1]*eta[i,1]*eta[i,1] + omega[2,2]*eta[i,2]*eta[i,2]
}


  lambda[1,1] <- 1

  lambda[2,1] <- parvec[1]

  lambda[3,1] <- parvec[2]

  lambda[4,2] <- 1

  lambda[5,2] <- parvec[3]

  lambda[6,2] <- parvec[4]

  lambda[7,3] <- 1

  lambda[8,3] <- parvec[5]
```

```
lambda[9,3] <- parvec[6]

omega[3,1] <- parvec[7]

omega[3,2] <- parvec[8]

theta[1,1] <- pow(parvec[9],-1)

theta[2,2] <- pow(parvec[10],-1)

theta[3,3] <- pow(parvec[11],-1)

theta[4,4] <- pow(parvec[12],-1)

theta[5,5] <- pow(parvec[13],-1)

theta[6,6] <- pow(parvec[14],-1)

theta[7,7] <- pow(parvec[15],-1)

theta[8,8] <- pow(parvec[16],-1)

theta[9,9] <- pow(parvec[17],-1)

psi[3,3] <- pow(parvec[18],-1)

nu[1,1] <- parvec[19]

nu[2,1] <- parvec[20]

nu[3,1] <- parvec[21]

nu[4,1] <- parvec[22]

nu[5,1] <- parvec[23]

nu[6,1] <- parvec[24]

nu[7,1] <- parvec[25]

nu[8,1] <- parvec[26]

nu[9,1] <- parvec[27]

alpha[1,1] <- 0
```

```
alpha[2,1] <- 0

alpha[3,1] <- 0

psi[1,1] <- pow(parvec[28],-1)

psi[2,2] <- pow(parvec[29],-1)

psi[1,2] <- pow(parvec[30],-1)

omega[2,1] <- parvec[31]

omega[1,1] <- parvec[32]

omega[2,2] <- parvec[33]


parvec[1] ~ dnorm(0,1e-2)

parvec[2] ~ dnorm(0,1e-2)

parvec[3] ~ dnorm(0,1e-2)

parvec[4] ~ dnorm(0,1e-2)

parvec[5] ~ dnorm(0,1e-2)

parvec[6] ~ dnorm(0,1e-2)

parvec[7] ~ dnorm(0,1e-2)

parvec[8] ~ dnorm(0,1e-2)

parvec[9] ~ dgamma(1,.5)

parvec[10] ~ dgamma(1,.5)

parvec[11] ~ dgamma(1,.5)

parvec[12] ~ dgamma(1,.5)

parvec[13] ~ dgamma(1,.5)

parvec[14] ~ dgamma(1,.5)
```

```
parvec[15] ~ dgamma(1,.5)
parvec[16] ~ dgamma(1,.5)
parvec[17] ~ dgamma(1,.5)
parvec[18] ~ dgamma(1,.5)
parvec[19] ~ dnorm(0,1e-3)
parvec[20] ~ dnorm(0,1e-3)
parvec[21] ~ dnorm(0,1e-3)
parvec[22] ~ dnorm(0,1e-3)
parvec[23] ~ dnorm(0,1e-3)
parvec[24] ~ dnorm(0,1e-3)
parvec[25] ~ dnorm(0,1e-3)
parvec[26] ~ dnorm(0,1e-3)
parvec[27] ~ dnorm(0,1e-3)
parvec[28] ~ dgamma(1,.5)
parvec[29] ~ dgamma(1,.5)
parvec[30] ~ dgamma(1,.5)
parvec[31] ~ dnorm(0,1e-2)
parvec[32] ~ dnorm(0,1e-2)
parvec[33] ~ dnorm(0,1e-2)}
```

Functions Used in Data Generation and Collection

Some of the code in this section is a direct contribution by Rebecca Buchner, specifically the function used to simulate the data. This code was reproduced in the function "establish fit" to aid in the creation of PPP statistics. The function “get_factor_scores” was taken from an appendix by Nora Umbach, simply given a name to serve as a function.

```
establishFit <- function(obs.data, model.obj,nfac, N = ss, m = 3, num.x_sim = 6,
                         num.y_sim = 3, num.xi_sim = 2, xi_sim = "x1-x3,x4-x6",
                         eta_sim = "y1-y3", model = NULL, df = NULL){

  bvec = preprocess(model.obj, nfac)

  gamma = bvec[grep("Gamma", names(bvec))]
  omega = matrix( c(bvec[grep("Omega", names(bvec))][1],0,
                    bvec[grep("Omega", names(bvec))][2],
                    bvec[grep("Omega", names(bvec))][3]), nrow=2)
  nu = bvec[grep("nu", names(bvec))]
  alphatau = c(bvec[grep("alpha", names(bvec))], bvec[grep("tau", names(bvec))])
  PsiPhi = c(bvec[grep("Psi", names(bvec))],  bvec[grep("Phi", names(bvec))])
  lamda =  bvec[grep("Lambda", names(bvec))]
  theta =  bvec[grep("Theta", names(bvec))]
  if(!class(model.obj)=="lavaan" ){
    predval = data.frame(get_factor_scores(bvec, data.frame(obs.data), type = 3))
    predval$ETA = as.vector(factor.scores(x=obs.data[grep("Y", colnames(obs.data))],
f=cbind(c(1,bvec[grep("Lambda.y", names(bvec))])), Phi=1, method = "regression")$scores)
    }else if(class(model.obj)=="lavaan" ){
    predval = data.frame(predict(model.obj))
  }
  if(nfac<3){
    predval$KSI1KSI1 = predval$KSI1*predval$KSI1
    predval$KSI2KSI2 = predval$KSI2*predval$KSI2
  }
    if(nfac<1){
      predval$KSI1KSI2 = predval$KSI1*predval$KSI2
    }
  interaction_simulation =
ifelse(nfac==3,"eta1~xi1:xi2,eta1~xi1:xi1,eta1~xi2:xi2","eta1~xi1:xi2")

  model_simulation <- specify_sem(num.x=num.x_sim, num.y=num.y_sim, num.xi=num.xi_sim,
num.eta=1, xi=xi_sim,
                              eta=eta_sim,interaction = interaction_simulation)

  pars <- c(lamda,#[!lamda==1],
          gamma,
          theta,
          PsiPhi,
          nu,
          alphatau,
        t(omega)[t(omega) != 0]
  )
  if(nfac==0){
    pars=c(pars,0)
  }

  if(is.null(df)){
    df = 9*10/2 - ifelse(nfac == 0, 30,ifelse(nfac==1, 31, 33))
  }
  if(class(model.obj)=="lavaan"){
    df = ifelse(nfac==0, 24, ifelse(nfac==1, 48, 114))
  }
  # Find the true values
  print("Done preparing. Starting 'true' calculations.")
  if(nfac ==3){
    reg = lm(predval$ETA~predval$KSI1 + predval$KSI2 +
             predval$KSI1KSI2 + predval$KSI1KSI1+ predval$KSI2KSI2)
  }else if(nfac==1){
    reg = lm(predval$ETA~predval$KSI1 + predval$KSI2 +
             predval$KSI1KSI2)
  }else{
    reg = lm(predval$ETA~predval$KSI1 + predval$KSI2)
  }


  reg0 = lm(predval$ETA~predval$KSI1 + predval$KSI2)
  reg1 = lm(predval$ETA~predval$KSI1 + predval$KSI2 +
            predval$KSI1KSI2)
  reg3 = lm(predval$ETA~predval$KSI1 + predval$KSI2 +
            predval$KSI1KSI2 + predval$KSI1KSI1+ predval$KSI2KSI2)
  e0 = reg0$residuals
  e1 = reg1$residuals
  e3 = reg3$residuals

  if(nfac == 3){
    e = predval$ETA - (predval$KSI1*gamma[1] + predval$KSI2*gamma[2] +
predval$KSI1KSI2*bvec[length(bvec)-1] + bvec[length(bvec)-2]*predval$KSI1KSI1+
bvec[length(bvec)]*predval$KSI2KSI2 )
  }else if (nfac==1){
    e = predval$ETA - (predval$KSI1*gamma[1] + predval$KSI2*gamma[2] +
predval$KSI1KSI2*bvec[length(bvec)-1])
  }else{
    e = predval$ETA - (predval$KSI1*gamma[1] + predval$KSI2*gamma[2])
  }

  re = reg$residuals
  obs = cov(obs.data)
  print("Errors: Done")

  implied = implicate(bvec, model.obj = model.obj, nfac, obs.data)

  likl =  log(det(implied)) + sum(diag(obs%*%solve(implied))) - log(det(obs)) - 9
  if(is.nan(likl)){
    obs = cor(obs.data)
    implied = cor.smooth(implied)
    likl =  log(det(implied)) + sum(diag(obs%*%solve(implied))) - log(det(obs)) - 9
  }
  print("Likelihood: Done")

  bic = -2*log(likl) + log(N)*((count_free_parameters(model_simulation) - ifelse(nfac == 0, 1,
0)))
  aic = -2*log(likl) + 2*((count_free_parameters(model_simulation) - ifelse(nfac == 0, 1, 0)))
  chisq =  likl*(N-1)
  print("Chisq: Done")
  chisqp = ifelse(chisq < 0, 0, pchisq(chisq,df))
  rmsea = ifelse((chisq/(df)-1)/nrow(obs.data)<0, NA, sqrt((chisq/(df)-1)/nrow(obs.data)))
  srmr = ifelse(is.nan(SRMR(obs,implied)), 100+SRMR(abs(obs), abs(implied)), SRMR(obs,
implied))

  if(class(model.obj)=="lavaan"){
  aic = fitMeasures(model.obj)[19]
  bic = fitMeasures(model.obj)[20]
  cfi = fitMeasures(model.obj)[9]
  tli = fitMeasures(model.obj)[10]
  rmsea = fitMeasures(model.obj)[23]
  rmsea.p = fitMeasures(model.obj)[26]
  srmr = fitMeasures(model.obj)[29]
  logl = fitMeasures(model.obj)[17]
  chisq = fitMeasures(model.obj)[3]
  chisqp = fitMeasures(model.obj)[5]
  }
  hfi = 1/(.032*(sqrt(ss/24) * (var(e^2)/(var(e)^2) + 1 - 3))+1)
  hfir = 1/(.032*(sqrt(ss/24) * (var(re^2)/(var(re)^2) + 1 - 3))+1)
  R2r = summary(reg)$r.squared
  R2 = 1- sum(e^2)/(sum(predval$ETA^2)-(sum(predval$ETA)^2)/N)
  hfi0 = 1/(.032*(sqrt(ss/24) * (var(e0^2)/(var(e0)^2) + 1 - 3))+1)
  hfi1 = 1/(.032*(sqrt(ss/24) * (var(e1^2)/(var(e1)^2) + 1 - 3))+1)
  hfi3 = 1/(.032*(sqrt(ss/24) * (var(e3^2)/(var(e3)^2) + 1 - 3))+1)
  print(paste("True values calculated. Begining", m, "simulations."))
  #

  sim.data <- replicate(m, simulate(model_simulation, parameters=pars, n=N))
  parameters = list(PPP_likl = NA, PPP_aic = NA, PPP_bic = NA, PPP_chisq = NA, PPP_chisqp
= NA, PPP_rmsea = NA, PPP_srmr = NA, PPP_hfi = rep(NA, m), PPP_hfir = NA,  PPP_R2r =
NA)
  for(i in 1:m){
    print(paste("Simulation", i))
    sobs = cov(sim.data[,,i])
    if(is.nan(likl)){
      sobs = cor(sim.data[,,i])
    }
    if(class(model.obj)=="lavaan"){
      dat = data.frame(sim.data[,,i])
      colnames(dat) = c(paste0("X", 1:6), paste0("Y",1:3))

      if(nfac==1){
        dat$X1X4 <- dat$X1 * dat$X4
        dat$X2X5 <- dat$X2 * dat$X5
        dat$X3X6 <- dat$X3 * dat$X6
      }else if(nfac==3){
        dat$X1X1 <- dat$X1 * dat$X1
        dat$X2X2 <- dat$X2 * dat$X2
        dat$X3X3 <- dat$X3 * dat$X3
        dat$X4X4 <- dat$X4 * dat$X4
        dat$X5X5 <- dat$X5 * dat$X5
        dat$X6X6 <- dat$X6 * dat$X6
        dat$X1X4 <- dat$X1 * dat$X4
        dat$X2X5 <- dat$X2 * dat$X5
        dat$X3X6 <- dat$X3 * dat$X6
      }

      model.obj1 = sem(model, data=dat, control = list(iter.max=3000))
      bvec = preprocess(model.obj1, nfac)
      sobs = cov(dat)
      predval1 = data.frame(predict(model.obj1))
      if(!model.obj1@optim$converged){

next
predval1 = data.frame(ETA = NA, KSI1 = NA, KSI2 = NA, KSI1KSI2 = NA, KSI1KSI1 = NA,
KSI2KSI2 = NA)
      }}
    if(class(model.obj)!="lavaan"){
      model.obj1 = "notlavaan"
      predval1 = data.frame(get_factor_scores(bvec, data.frame(obs.data), type = 3))
      predval1$ETA = as.vector(factor.scores(x=obs.data[grep("Y", colnames(obs.data))],
f=cbind(c(1,bvec[grep("Lambda.y", names(bvec))])), Phi=1, method = "regression")$scores)
    }
    if(nfac<3){
      predval1$KSI1KSI1 = predval1$KSI1*predval1$KSI1
      predval1$KSI2KSI2 = predval1$KSI2*predval1$KSI2
    }
      if(nfac<1){
        predval1$KSI1KSI2 = predval1$KSI1*predval1$KSI2
      }


    if(nfac==3 & !is.na(predval1$KSI1)){
      sreg = lm(predval1$ETA~predval1$KSI1 + predval1$KSI2 +
               predval1$KSI1KSI2 + predval1$KSI1KSI1+ predval1$KSI2KSI2)
    }else if(nfac==1&!is.na(predval1$KSI1)){
      sreg = lm(predval1$ETA~predval1$KSI1 + predval1$KSI2 +
               predval1$KSI1KSI2)
    }else if(!is.na(predval1$KSI1)){
      sreg = lm(predval1$ETA~predval1$KSI1 + predval1$KSI2)
    }



    likl1 = log(det(implied)) + sum(diag(sobs%*%solve(implied))) - log(det(sobs)) - 9

    chisq1 =  likl1*(N-1)
    chisqp1 =  1-pchisq(chisq1,df)

    if(nfac == 3 & !is.na(predval1$KSI1)){
      se = predval1$ETA - (predval1$KSI1*gamma[1] + predval1$KSI2*gamma[2] +
predval1$KSI1KSI2*bvec[length(bvec)-1] + bvec[length(bvec)-2]*predval1$KSI1KSI1+
bvec[length(bvec)]*predval1$KSI2KSI2 )
    }else if (nfac==1 & !is.na(predval1$KSI1)){
      se = predval1$ETA - (predval1$KSI1*gamma[1] + predval1$KSI2*gamma[2] +
predval1$KSI1KSI2*bvec[length(bvec)-1])
    }else if(!is.na(predval1$KSI1)){
      se = predval1$ETA - (predval1$KSI1*gamma[1] + predval1$KSI2*gamma[2])
    }

    if(is.na(predval1$ETA)){
      se = NA
      sreg = data.frame(residuals = NA)
    }
    sre = sreg$residuals

    if(class(model.obj1)=="lavaan"){
      if(model.obj1@optim$converged){
      aic1 = fitMeasures(model.obj1)[19]
      bic1 = fitMeasures(model.obj1)[20]
      cfi1 = fitMeasures(model.obj1)[9]
      tli1 = fitMeasures(model.obj1)[10]
      rmsea1 = fitMeasures(model.obj1)[23]
      rmsea.p1 = fitMeasures(model.obj1)[26]
      srmr1 = fitMeasures(model.obj1)[29]
      logl1 = fitMeasures(model.obj1)[17]
      chisq1 = fitMeasures(model.obj1)[3]
      chisqp1 = fitMeasures(model.obj1)[5]
      }else{
        aic1 = NA
        bic1 = NA
        cfi1 = NA
        tli1 = NA
        rmsea1 = NA
        rmsea.p1 = NA
        srmr1 = NA
        logl1 = NA
        chisq1 = NA
        chisqp1 = NA
      }
      parameters$PPP_srmr[i] = srmr1 < srmr
      parameters$PPP_logl[i] = logl1 < logl
      parameters$PPP_aic[i] = aic1 < aic
      parameters$PPP_bic[i] = bic1 < bic
      parameters$PPP_cfi[i] = cfi1 < cfi
      parameters$PPP_tli[i] = tli1 < tli
      parameters$PPP_chisq[i] = chisq1 < chisq
      parameters$PPP_chisqp[i] = chisqp1 < chisqp
      parameters$PPP_rmsea[i] = rmsea1 < rmsea
      parameters$PPP_rmseap[i] = rmsea.p1 < rmsea.p

    }
    parameters$PPP_R2r[i] = summary(sreg)$r.squared < R2r
    parameters$PPP_R2[i] = 1- sum(se^2)/(sum(predval1$ETA^2)-(sum(predval1$ETA)^2)/N) <
R2
    parameters$PPP_hfi[i] = 1/(.032*(sqrt(ss/24) * (var(se^2)/(var(se)^2) + 1 - 3))+1) < hfi
    parameters$PPP_hfir[i] = 1/(.032*(sqrt(ss/24) * (var(sre^2)/(var(sre)^2) + 1 - 3))+1) < hfir
    parameters$PPP_srmr[i] = ifelse(is.nan(SRMR(sobs,implied)), 100+SRMR(abs(sobs),
abs(implied)), SRMR(sobs, implied)) < srmr

    parameters$PPP_likl[i] = likl1 < likl
    parameters$PPP_aic[i] = -2*log(likl1) + 2*((count_free_parameters(model_simulation) -
ifelse(nfac == 0, 1, 0))) < aic
    parameters$PPP_bic[i] = -2*log(likl1) + log(N)*((count_free_parameters(model_simulation) -
ifelse(nfac == 0, 1, 0))) < bic
```

```
    parameters$PPP_chisq[i] = chisq1 < chisq
    parameters$PPP_chisqp[i] = chisqp1 < chisqp
    parameters$PPP_rmsea[i] = sqrt((chisq1/(df)-1)/nrow(sim.data[,,i])) < rmsea
    parameters$PPP_likl[i] = likl1 < likl
  }

  return(c(likl = likl,aic = aic, bic = bic, chisq = chisq, chisqp = chisqp, rmsea = rmsea,srmr =
srmr, hfi = hfi, hfir=hfir, R2 = R2, R2r = R2r , hfi0 = hfi0, hfi1=hfi1, hfi3=hfi3,
colMeans(data.frame(parameters), na.rm=TRUE)))

}

##########################################################################################
################################

preprocess <- function(model.obj, nfac){
  if(class(model.obj) =='runjags'){
    bvec = data.frame(summary(model.obj))[c(2,3,5,6,8,9:20,23,21,24,22,26, 27,
29,30,32,33,36,34,35, 38,37,39),]$Median
    #predval = get_factor_scores(bvec, data.frame(obs.data), type = 3)
  }else if(class(model.obj) =='lavaan'){
    # predval = lavPredict(model.obj, method = 'regression')
    # Should I simulate the data with the data that modeled it?
    if(nfac==0){
      bvec =
c(coef(model.obj)[c(1,2,3,4,5,6,19,20,13,14,15,16,17,18,10,11,12,24,21,22,23,26,27,29,30,8,9)],
0,0,0,0,0,0)
    }else if(nfac==1){
      bvec =
c(coef(model.obj)[c(3,4,5,6,1,2,27,28,18,19,20,21,22,23,15,16,17,34,30,31,32,38,39,41,42,10,11
)], 0,0,0,0,coef(model.obj)[29],0)
    }else{
      bvec =
c(coef(model.obj)[c(1,2,3,4,5,6,49,50,28,29,30,31,32,33,25,26,27,63,54,55,56,71,72,74,75,14,15
)], 0,0,0, coef(model.obj)[c(52, 51, 53)])
    }
    #predval = lavPredict(model.obj, method = 'regression')
  }else if(class(model.obj) =='blavaan'){
    bvec =
c(coef(model.obj)[c(1,2,3,4,5,6,19,20,13,14,15,16,17,18,10,11,12,24,21,22,23,26,27,29,30,8,9)],
0,0,0,0,0,0)
    #predval = lavPredict(model.obj, method = 'regression')
  }else if(class(model.obj) =='emEst'){
    if(nfac==1){
    bvec = c(coef(model.obj)[1:(length(coef(model.obj))-1)], 0,
coef(model.obj)[length(coef(model.obj))],0)
```

```
    }else{
      bvec = coef(model.obj)
    }
    #predval = get_factor_scores(bvec, data.frame(obs.data), type = 3)
  }else{
    stop("Unrecognized class for model.obj")
  }

  names(bvec)=c('Lambda.x2' , 'Lambda.x3' , 'Lambda.x11' , 'Lambda.x12' ,
              'Lambda.y2' , 'Lambda.y3' , 'Gamma1' , 'Gamma2' , 'Theta.d1' ,
              'Theta.d8' , 'Theta.d15' , 'Theta.d22' , 'Theta.d29' ,
              'Theta.d36' , 'Theta.e1' , 'Theta.e5' , 'Theta.e9' , 'Psi' ,
              'Phi1' , 'Phi2' , 'Phi4' , 'nu.x2' , 'nu.x3' , 'nu.x5' , 'nu.x6' ,
              'nu.y2' , 'nu.y3' , 'alpha' , 'tau1' , 'tau2' , 'Omega1' ,
              'Omega3' , 'Omega4')
  return(bvec)
}

implicate <- function(bvec, model.obj=NULL, nfac, obs.data){
  gamma = bvec[grep("Gamma", names(bvec))]
  omega = matrix( c(bvec[grep("Omega", names(bvec))][1],0,
                  bvec[grep("Omega", names(bvec))][2],
                  bvec[grep("Omega", names(bvec))][3]), nrow=2)
  PsiPhi = c(bvec[grep("Psi", names(bvec))],  bvec[grep("Phi", names(bvec))])
  lamda =  bvec[grep("Lambda", names(bvec))]
  theta =  bvec[grep("Theta", names(bvec))]
  if(nfac==0){
    if(class(model.obj)=="lavaan"){
      implied=model.obj@implied$cov[[1]]
    }else{
      colnames(obs.data) = c(paste0("X", 1:6), paste0("Y",1:3))
    lambda = matrix(c(1,lamda[1:2], rep(0,9), 1,lamda[3:4], rep(0,9),1,lamda[5:6]), ncol=3)
    lambda.y = matrix(lambda[7:9,3])
    lambda.x = lambda[1:6,1:2]
    gamma = matrix(gamma, ncol = 2)
    phi = matrix(c(PsiPhi[2],PsiPhi[3],PsiPhi[3], PsiPhi[4]), nrow = 2)
    psi = PsiPhi[1]
    theta.e = diag(theta[7:9])
    theta.d = diag(theta[1:6])

    tl = lambda.y%*%(gamma%*%phi%*%t(gamma) + psi)%*%t(lambda.y) + theta.e
    tr = lambda.y%*%gamma%*%phi%*%t(lambda.x)
    bl = lambda.x%*%phi%*%t(gamma)%*%t(lambda.y)
    br = lambda.x%*%phi%*%t(lambda.x) + theta.d

    dimnames(tl) = list(paste0("Y", 1:3), paste0("Y", 1:3))
```

```
   dimnames(tr) = list(paste0("Y", 1:3), paste0("X", 1:6))
   dimnames(bl) = list(paste0("X", 1:6), paste0("Y", 1:3))
   dimnames(br) = list(paste0("X", 1:6), paste0("X", 1:6))

   implied = diag(rep(0,9))
   dimnames(implied) = list(c(paste0("X", 1:6), paste0("Y", 1:3)), c(paste0("X", 1:6),
paste0("Y", 1:3)))
   implied[match(rownames(br), rownames(implied)) , match(colnames(br), colnames(implied))]
= br
   implied[match(rownames(tr), rownames(implied)) , match(colnames(tr), colnames(implied))]
= tr
   implied[match(rownames(bl), rownames(implied)) , match(colnames(bl), colnames(implied))]
= bl
   implied[match(rownames(tl), rownames(implied)) , match(colnames(tl), colnames(implied))]
= tl
   }
  }else if(nfac==1){
   if(class(model.obj)=="lavaan"){
    implied=model.obj@implied$cov[[1]]
   }else{
    lambda = matrix(c(1,lamda[1:2], rep(0,9), 1,lamda[3:4],rep(0,9), rep(0,9),1,lamda[5:6]),
ncol=4)
    lambda.y = lambda[7:9,4]
    lambda.x = lambda[1:6,1:3]
    Igamma = matrix(c(gamma, omega[1,2]), ncol = 3)
    Iphi = cov(get_factor_scores(bvec, responses = obs.data[,grep("X", colnames(obs.data))],
type = 1)[-1])
    phi = matrix(c(PsiPhi[2],PsiPhi[3],PsiPhi[3], PsiPhi[4]), nrow = 2)
    Iphi[1:2, 1:2] = phi
    psi = PsiPhi[1]
    theta.e = diag(theta[7:9])
    theta.d = diag(theta[1:6])
    tl = lambda.y%*%(Igamma%*%Iphi%*%t(Igamma) + psi)%*%t(lambda.y) + theta.e
    tr = lambda.y%*%Igamma%*%Iphi%*%t(lambda.x)
    bl = lambda.x%*%Iphi%*%t(Igamma)%*%t(lambda.y)
    br = lambda.x%*%Iphi%*%t(lambda.x) + theta.d

    dimnames(tl) = list(paste0("Y", 1:3), paste0("Y", 1:3))
    dimnames(tr) = list(paste0("Y", 1:3), paste0("X", 1:6))
    dimnames(bl) = list(paste0("X", 1:6), paste0("Y", 1:3))
    dimnames(br) = list(paste0("X", 1:6), paste0("X", 1:6))

    implied = diag(rep(0,9))
    dimnames(implied) = list(c(paste0("X", 1:6), paste0("Y", 1:3)), c(paste0("X", 1:6),
paste0("Y", 1:3)))
```

```
      implied[match(rownames(br), rownames(implied)) , match(colnames(br),
colnames(implied))] = br
      implied[match(rownames(tr), rownames(implied)) , match(colnames(tr), colnames(implied))]
= tr
      implied[match(rownames(bl), rownames(implied)) , match(colnames(bl),
colnames(implied))] = bl
      implied[match(rownames(tl), rownames(implied)) , match(colnames(tl), colnames(implied))]
= tl
    }

  }else if(nfac==3){
    if(class(model.obj)=="lavaan"){
      implied=model.obj@implied$cov[[1]]
    }else{
      lambda = matrix(c(1,lamda[1:2], rep(0,9), 1,lamda[3:4],rep(0,9),rep(0,9),rep(0,9),
rep(0,9),1,lamda[5:6]), ncol=6)
      lambda.y = matrix(lambda[7:9,6])
      lambda.x = lambda[1:6,1:5]
      Fgamma = matrix(c(gamma, omega[1,1], omega[1,2], omega[2,2]), ncol = 5)
      Fphi = cov(get_factor_scores(bvec, responses = obs.data[,grep("X", colnames(obs.data))],
type = 3)[-1])
      phi = matrix(c(PsiPhi[2],PsiPhi[3],PsiPhi[3], PsiPhi[4]), nrow = 2)
      Fphi[1:2, 1:2] = phi
      psi = PsiPhi[1]
      theta.e = diag(theta[7:9])
      theta.d = diag(theta[1:6])

      tl = lambda.y%*%(Fgamma%*%Fphi%*%t(Fgamma) + psi)%*%t(lambda.y) + theta.e
      tr = lambda.y%*%Fgamma%*%Fphi%*%t(lambda.x)
      bl = lambda.x%*%Fphi%*%t(Fgamma)%*%t(lambda.y)
      br = lambda.x%*%Fphi%*%t(lambda.x) + theta.d

      dimnames(tl) = list(paste0("Y", 1:3), paste0("Y", 1:3))
      dimnames(tr) = list(paste0("Y", 1:3), paste0("X", 1:6))
      dimnames(bl) = list(paste0("X", 1:6), paste0("Y", 1:3))
      dimnames(br) = list(paste0("X", 1:6), paste0("X", 1:6))

      implied = diag(rep(0,9))
      dimnames(implied) = list(c(paste0("X", 1:6), paste0("Y", 1:3)), c(paste0("X", 1:6),
paste0("Y", 1:3)))
      implied[match(rownames(br), rownames(implied)) , match(colnames(br),
colnames(implied))] = br
      implied[match(rownames(tr), rownames(implied)) , match(colnames(tr), colnames(implied))]
= tr
      implied[match(rownames(bl), rownames(implied)) , match(colnames(bl),
colnames(implied))] = bl
```

```
      implied[match(rownames(tl), rownames(implied)) , match(colnames(tl), colnames(implied))]
= tl

    }
  }
  return(implied)

}


SRMR <- function(x=obs, y=imp){
  s = matrix(nrow=nrow(x), ncol=ncol(x))
  for(j in 1:nrow(x)){
    for(k in 1:nrow(x)){
      s[j,k] = x[j,k]/(sqrt(x[j,j])*sqrt(x[k,k]))
    }
  }

  sig = matrix(nrow=nrow(y), ncol=ncol(y))
  for(j in 1:nrow(y)){
    for(k in 1:nrow(y)){
      sig[j,k] = y[j,k]/(sqrt(y[j,j])*sqrt(y[k,k]))
    }
  }

  r = (s-sig)^2
  sums = apply(r, 2, sum)
  sumsums = sum(sums)
  p = nrow(s)
  e = p*(p+1)/2
  SRMR = sqrt(sumsums/e)
  return(SRMR)
}
get_factor_scores <-  function(res_qml, responses, type = 3, method = "regression"){

  psi <- res_qml[grep("Theta.d", names(res_qml))]
  psihat <- diag(psi[c(2:3,5:6,1,4)])

  omega1 <- res_qml[grep("Lambda.x", names(res_qml))]
  omega1hat <- matrix(0, nrow = 4, ncol = 2)

  omega1hat[1:2,1] <- omega1[1:2]
  omega1hat[3:4,2] <- omega1[3:4]

  omega0 <- res_qml[grep("nu.x", names(res_qml))]
```

```
 omega0hat <- matrix(0, nrow = 4, ncol = 1)
 omega0hat[1:2] <- omega0[1:2]
 omega0hat[3:4] <- omega0[3:4]

 b1.mod1 <- cbind(matrix(0, 2, 4), diag(2))
 b1.mod2 <- rbind(diag(4), -t(omega1hat))

 Gamma <- b1.mod1 %*% psihat %*% b1.mod2 %*% solve(t(b1.mod2) %*% psihat %*%
                                  (b1.mod2))

 n <- nrow(responses)

 fs <- matrix(0, n, 2)

 for(i in 1:n){
   fs[i,] <- tryCatch(cbind(-Gamma, (diag(2) + Gamma %*% omega1hat)) %*%
                  (t(t(as.matrix(responses[i, 1:6]))) - rbind(omega0hat, matrix(0, 2, 1))), error =
function(e) cbind(-Gamma, (diag(2) + Gamma %*% omega1hat)) %*%
                  (t(as.matrix(responses[i, 1:6])) - rbind(omega0hat, matrix(0, 2, 1))))
 }

 pars <- res_qml
 if(type == 3){
   read <- function(fs1, fs2) pars["alpha"] + pars["Gamma1"] * fs1 + pars["Gamma2"] * fs2 +
     pars["Omega1"] * fs1^2 + pars["Omega3"] * fs1 * fs2 +
     pars["Omega4"] * fs2^2
 }else if (type==1){
   read <- function(fs1, fs2) pars["alpha"] + pars["Gamma1"] * fs1 + pars["Gamma2"] * fs2 +
pars["Omega3"] * fs1 * fs2
 }else{
   read <- function(fs1, fs2) pars["alpha"] + pars["Gamma1"] * fs1 + pars["Gamma2"] * fs2
 }
 if(type == 3){
   scores = data.frame(ETA = read(fs[,1],fs[,2]), KSI1 = fs[,1], KSI2 = fs[,2], KSI1KSI2 =
fs[,1]*fs[,2], KSI1KSI1 = fs[,1]*fs[,1], KSI2KSI2 = fs[,2]*fs[,2])
 }else{
   scores = data.frame(ETA = read(fs[,1],fs[,2]), KSI1 = fs[,1], KSI2 = fs[,2], KSI1KSI2 =
fs[,1]*fs[,2])}
 if(!method=="regression"){
   if(nfac==0){
     lambda = matrix(c(1,lamda[1:2], rep(0,9), 1,lamda[3:4], rep(0,9),1,lamda[5:6]), ncol=3)
     scores$ETA = (data[,,1]%*%solve(cor(data[,,1]))%*%lambda)[,3]
     }else if(nfac==1){
      lambda = matrix(c(1,lamda[1:2], rep(0,9), 1,lamda[3:4],rep(0,9), rep(0,9),1,lamda[5:6]),
ncol=4)
      scores$ETA = (data[,,1]%*%solve(cor(data[,,1]))%*%lambda)[,4]
     }else{
      lambda = matrix(c(1,lamda[1:2], rep(0,9), 1,lamda[3:4],rep(0,9),rep(0,9),rep(0,9),
rep(0,9),1,lamda[5:6]), ncol=6)
      scores$ETA = (data[,,1]%*%solve(cor(data[,,1]))%*%lambda)[,6]
     }
   }
  return(scores)
}

simulate_nlsem <- function(N = 200, m = 1000, interaction_simulation = "eta1~xi1:xi2", gamma
= c(.50, .40), Omega = matrix(c(0,0,0.2,0), nrow=2),
                         covariance = .4, rel_x = c(0.64,0.64,0.49,0.64,0.64,0.49), rel_y =
c(0.64,0.49,0.49),
                         num.x_sim = 6, num.y_sim = 3, num.xi_sim = 2, xi_sim = "x1-x3,x4-x6",
eta_sim = "y1-y3"){

  # specify_sem is a function that comes in nlsem that generates the model syntax for an nlsem
function.
  model_simulation <- specify_sem(num.x=num.x_sim, num.y=num.y_sim, num.xi=num.xi_sim,
num.eta=1, xi=xi_sim,
                              eta=eta_sim,interaction = interaction_simulation)

  ### Parameters of the model

  # Lambda_x comes from the model simulation we have already specified. It has values of 1, 0,
and NA.
  #I am guessing that the NA values are the ones that are going to be estimated.
  Lambda_x <- model_simulation$matrices$class1$Lambda.x
  # This gives the row only for the Lambda_x values that are 1.
  fixed_x <- which(Lambda_x[Lambda_x == 1] == 1)
  #The NA values have been specified to be the square root of the rel_x values. The values that
were fixed at 1 were removed from
  # rel_x. I am not sure why yet.
  # Ms. Buchner said that this was
  Lambda_x[is.na(Lambda_x)] <- sqrt(rel_x[-c(fixed_x)])

  Lambda_y <- model_simulation$matrices$class1$Lambda.y
  fixed_y <- which(Lambda_y[Lambda_y == 1] == 1)
  Lambda_y[is.na(Lambda_y)] <- sqrt(rel_y[-c(fixed_y)])

  # Covariances of xis as vector for parameter vector
  Phi <- model_simulation$matrices$class1$Phi
  diag(Phi) <- 1
  Phi[is.na(Phi)] <- covariance

  Phi_vek <- c(1)
```

```
 if (num.xi_sim > 1){
   seq <- c(0, cumsum((num.xi_sim-1):1))
   for (i in 1:(num.xi_sim-1)){
     Phi_vek <- c(Phi_vek, covariance[(seq[i]+1):(seq[i+1])], 1)
   }
 }

 Theta_x <- c((Lambda_x^2 %*% diag(Phi) * (1-rel_x))/rel_x)
 Theta_y <- c((Lambda_y^2 * (1-rel_y))/rel_y)

 alpha    <- -sum(diag(Omega %*% (Phi + t(as.matrix(tril(Phi,-1))))))


 # Calculation of Psi
 Psi <- determine_Psi(Phi = Phi, gamma = gamma, Omega = Omega, var_eta = 1)
 if (Psi < 0) stop("Psi < 0")

 pars <- c(sqrt(rel_x[-c(fixed_x)]),
           sqrt(rel_y[-c(fixed_y)]),
           gamma,
           Theta_x,
           Theta_y,
           Psi,
           Phi_vek,
           rep(0,num.x_sim-num.xi_sim), # nu.x EW
           rep(0,num.y_sim-1), # nu.y2 EW
           alpha,
           rep(0,num.xi_sim), # tau, EW xi
           t(Omega)[t(Omega) != 0]
 )


 if(count_free_parameters(model_simulation) != length(pars)) warning("Length of pars not
correct.")

 ### generate data sets

 #model <- as.data.frame(fill_model(model_simulation,pars))

 data <- replicate(m, simulate(model_simulation, parameters=pars, n=N))  # array

 return(data)

}
```

```
determine_Psi <- function(Phi, gamma, Omega, var_eta){
  cov_i_q <- cov_int_quad(Phi = Phi)   ## Covariance matrix of all
  ## linear and nonlinear terms in the model
  coef <- gamma  # Omega is assumed to be upper triangular matrix
  if(dim(Phi)[1] > 1){
    for (i in 1:(dim(Phi)[1]-1)){
      coef <- c(coef, Omega[i,(i+1):dim(Phi)[1]])
    }
  }
  coef <- c(coef, diag(Omega))

  Psi <- var_eta - sum((coef %*% t(coef)) * cov_i_q)
  return(Psi)
}

cov_int_quad <- function(Phi){

  Phi <- as.matrix(tril(Phi,0))  # lower triangular matrix
  Phi_sym <- as.matrix(tril(Phi,0)) + t(as.matrix(tril(Phi,-1))) # as symmetric matrix

  # empty matrix
  num.xi <- dim(Phi)[1]
  num.int <- num.xi*(num.xi-1)/2
  Cov <- matrix(NA, nrow = 2*num.xi + num.int, ncol = 2*num.xi + num.int)

  if (num.xi == 1) {
    Cov[1,2] <- 0
    Cov[2,1] <- 0
    Cov[1,1] <- Phi
    Cov[2,2] <- 2*Phi^2
  } else {
    comb <- combn(num.xi, 2)  # Interactionens
    names_Cov <- c()
    for (i in 1:num.xi){
      names_Cov <- c(names_Cov, paste("xi", i, sep = ""))
    }
    for (i in 1:dim(comb)[2]){
      names_Cov <- c(names_Cov, paste("xi", comb[1,i], "*xi", comb[2,i],  sep = ""))
    }
    for (i in 1:num.xi){
      names_Cov <- c(names_Cov, paste("xi", i, "^2",  sep = ""))
    }

    rownames(Cov) <- names_Cov
    colnames(Cov) <- names_Cov
```

```
   # F?llen der unteren Dreiecksmatrix
   Cov[1:num.xi, 1:num.xi] <- Phi #+ t(tril(Phi, -1))            # Variances + Covariances
   Cov[(num.xi + 1): (2*num.xi + num.int), (1:num.xi)] <- 0   # linearer term * quadratic term or
linearer term * interaction term
   Cov[(num.xi + num.int+1):(2*num.xi + num.int), (num.xi + num.int+1):(2*num.xi + num.int)]
<- 2*Phi^2  # quadratic term * quadratic term


   # quadratic term * interaction term
   # all combinations in C1

   C1   <- matrix(NA, nrow = num.xi, ncol = num.int)
   for (k in 1:num.xi){
     for (l in 1:num.int){
       C1[k,l] <- 2*Phi_sym[k,comb[1,l]]*Phi_sym[k,comb[2,l]]
     }
   }

   Cov[(num.xi + num.int + 1):(2*num.xi + num.int), (num.xi + 1):(num.xi + num.int)] <- C1


   # interaction term * interaction term
   # all combinations ind C2

   C2   <- matrix(NA, nrow = num.int, ncol = num.int)
   for (k in 1:num.int){
     for (l in 1:num.int){
       C2[k,l] <- Phi_sym[comb[1,k],comb[1,l]]*Phi_sym[comb[2,k],comb[2,l]] +
Phi_sym[comb[1,k],comb[2,l]]*Phi_sym[comb[2,k],comb[1,l]]
     }
   }
       Cov[(num.xi + 1):(num.xi + num.int), (num.xi + 1):(num.xi + num.int)] <-
as.matrix(tril(C2,0))
    }
   return(as.matrix(tril(Cov,0)) + t(as.matrix(tril(Cov,-1))))

}
```